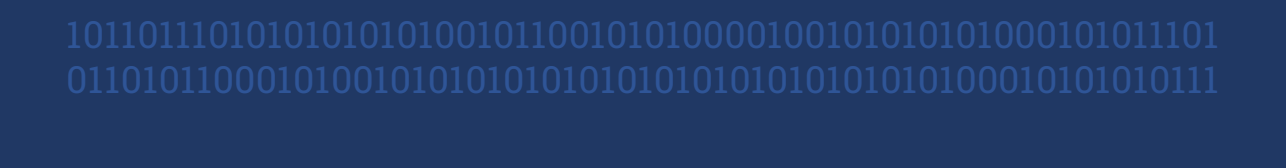
Arguing

with incomplete information

Formalisms, algorithms and applications in law enforcement

Daphne Odekerken



Arguing with incomplete information

Formalisms, algorithms and applications in law enforcement

Daphne Odekerken

© 2024 Daphne Odekerken
Printed by Proefschriftspecialist

ISBN: 978-90-393-7800-7
doi: 10.33540/2627

Arguing with incomplete information

Formalisms, algorithms and applications in law enforcement

Argumenteren met onvolledige informatie
Formalismen, algoritmes en toepassingen in de politiepraktijk
(met een samenvatting in het Nederlands)

Proefschrift

ter verkrijging van de graad van doctor aan de
Universiteit Utrecht
op gezag van de
rector magnificus, prof. dr. H.R.B.M. Kummeling,
ingevolge het besluit van het College voor Promoties
in het openbaar te verdedigen op
maandag 6 januari 2025 des middags te 4.15 uur

door

Daphne Odekerken

geboren op 25 augustus 1993
te Stein

Promotoren:

Prof. dr. F.J. Bex
Prof. dr. mr. H. Prakken

Copromotor:

Dr. A. Borg

Beoordelingscommissie:

Prof. dr. K.M. Atkinson
Prof. dr. M.M. Dastani
Prof. dr. C.M. Jonker
Dr. T. Rienstra
Prof. dr. M. Thimm

Dit proefschrift werd (mede) mogelijk gemaakt met financiële steun van de Nationale Politie.

Contents

1

Introduction

Every year, the Dutch police receives tens of thousands of complaints on online trade fraud. This involves cases such as fake web shops and malicious second-hand traders on trading platforms such as eBay (Schraagen et al., 2018). Two subtasks within the process of handling these complaints are the intake of fraud complaints and the investigation of potentially mala fide web shops.

For the intake task, the Dutch police collects complaints of online trade fraud using a web form on the police web site. Human analysts at the national centre for counteracting online trade fraud of the police process these forms and investigate whether the complaint concerns a case of fraud. Their decision is based on Article 326 of the Dutch Criminal Code, which defines fraud as the act of misleading through false contact details, deceptive tricks or an accumulation of lies. However, due to the inflexibility of the web form, a lot of information is missing in these initial complaints. As a consequence, the analysts often have to contact the complainant and ask questions in order to complete the complaint. This takes a lot of time and makes the intake process tedious for both the complainant and the analyst.

The second task is the classification of potentially mala fide web shops against which a complaint was submitted in the intake process. This task is required as not all of these shops have bad intentions: in many cases, the customer fell victim to malfunctioning delivery service rather than fraud. In contrast to the intake task, there are no legal rules that define under which conditions a web shop is mala fide. Instead, the decisions are made based on the experience of the analyst. It is important that the task is performed carefully, as the classification outcome has serious implications: web shops that are classified as mala fide will be taken offline, while web shops classified as bona fide can be placed on a list of trustworthy web shops that is used in the fraud intake process. The classification of mala fide web shops is challenging for multiple reasons. First, the makers of mala fide web shops adapt to procedures for taking web shops offline (Wabeke et al., 2020). Second,

not all information is available when the analysts start their classification task. In many situations, further investigation is required, such as making a call or testing the payment options.

Even though the Dutch police has a national centre for counteracting online trade fraud, it is challenging for human analysts to perform these tasks due to the high volume of complaints and web shops to be investigated and the necessity to act quickly (Testerink et al., 2019b). Given that a significant part of the tasks consists of routine work, the question arises whether this task could be automated by some artificial intelligence (AI) system. However, as stated before, some complaints and web shops require detailed manual investigation. Therefore, full automation is impossible. Since neither humans nor automatic procedures are able to perform these tasks on their own, it is interesting to study whether they can do it in collaboration. In this thesis, we develop and study AI systems that support humans in decision-making in law enforcement, with the tasks of fraud intake and web shop classification as the two main use cases.

1.1. Techniques for transparent and cautious AI

For decision-making tasks in law enforcement, it is essential that domain experts understand machine-made decisions or advice and have the possibility to correct possible mistakes. Therefore, AI systems used in this domain should be *transparent*. As an additional advantage, transparency has been shown to increase trust by citizens (Nieuwenhuizen, 2020). Furthermore, given that a wrong decision has serious consequences, the AI system should have the possibility to abstain from proposing a decision. In other words, it should be *cautious*. These two requirements should be taken into account for the selection of suitable AI techniques for decision support in the two use cases.

To optimise transparency, we focus on AI techniques that reason in a similar way as humans do when performing a given task. The two use cases related to online trade fraud require two different types of reasoning. In order to decide whether or not a citizen is a victim of fraud, the analyst checks whether a legal rule (and any of its exceptions) applies. This type of reasoning is *rule-based*. Conversely, when an analyst decides that a web shop is mala fide, they do this after a comparison with earlier (precedent) web shops for which they already made a decision. This is an example of *precedent-based* reasoning. In this thesis, we model rule-based reasoning with computational argumentation (Prakken, 2010; Dung, 1995) and capture precedent-based reasoning by models of precedential constraint (Horty, 2004; Horty, 2011). This section gives an informal introduction to computational argumentation and models of precedential constraint.

1.1.1. Argumentation for rule-based reasoning

Computational argumentation is a subfield of AI concerning reasoning with incomplete or inconsistent information (Atkinson et al., 2017). Two central concepts in

computational argumentation are abstract argumentation frameworks (Dung, 1995) and structured argumentation frameworks (Besnard et al., 2014).

In *abstract argumentation frameworks*, arguments are considered as abstract entities. Between arguments, there is an attack relation. An abstract argumentation *semantics* is a formal criterion that is used to determine which sets of arguments can be regarded as collectively acceptable (Baroni et al., 2011). These sets are called *extensions*. For example, semantics typically require that each extension is conflict-free, in that it does not contain two arguments that attack each other. Depending on the chosen semantics, it is possible to reason cautiously. In particular, for the grounded semantics (which will be defined formally in Chapter 2) the only extension consists of the minimal set of arguments such that all attackers are attacked by some accepted argument.

Structured argumentation frameworks model the internal structure of arguments. In contrast to abstract argumentation frameworks, they normally make the premises and claim of the argument explicit and formally define how those premises are related to the claim. The attack relation between these arguments is then derived from the conflicts between the arguments. For example, two arguments with contradicting conclusions may attack each other. Another type of attack, identified by the seminal paper on defeasible reasoning by Pollock (1987) is the so-called undercutter, which attacks the defeasible inference itself. Five modern examples of structured argumentation are assumption-based argumentation (ABA) (Bondarenko et al., 1997), ASPIC$^+$ (Prakken, 2010), Defeasible Logic Programming (DeLP) (García and Simari, 2004), Gorgias (Kakas and Moraitis, 2003) and deductive argumentation (Besnard and Hunter, 2008). In this thesis, we focus on ASPIC$^+$ for two reasons. First, the instantiated arguments and attacks in ASPIC$^+$ explicitly form abstract argumentation frameworks (in contrast to DeLP, Gorgias and deductive argumentation). Systems based on ASPIC$^+$ are thus maximally compatible with existing and forthcoming research in both structured and abstract approaches to argumentation. Second, it allows for reasoning with not only strict, but also defeasible rules (in contrast to ABA, Gorgias and deductive argumentation). Defeasible rules are interesting as they allow for a natural modelling of defeasible inference ("if the premises hold then *usually* the conclusion follows"). This is in particular useful in modelling legal reasoning (Prakken, 2020) and reasoning with evidence in law enforcement (Bex et al., 2003).

Argumentation-based AI fits our transparency and cautiousness requirements for AI in law enforcement very well. The transparency requirement is fulfilled as argumentation is naturally linked to explanation (Čyras et al., 2021; Vassiliades et al., 2021; Dietz et al., 2022), in particular in the legal domain (Atkinson et al., 2020). In his survey on relevant papers from philosophy, cognitive psychology, and social psychology, Miller (2019) argued that explainable AI can benefit from existing models of human explanation. For instance, he finds that referring to probabilities or statistical relationships in explanation is not very effective (Miller, 2019). Instead, arguments for a particular conclusion give basic (attributive) reasons of support (Atkinson et al., 2020; Dietz et al., 2022). Furthermore, human explanation is

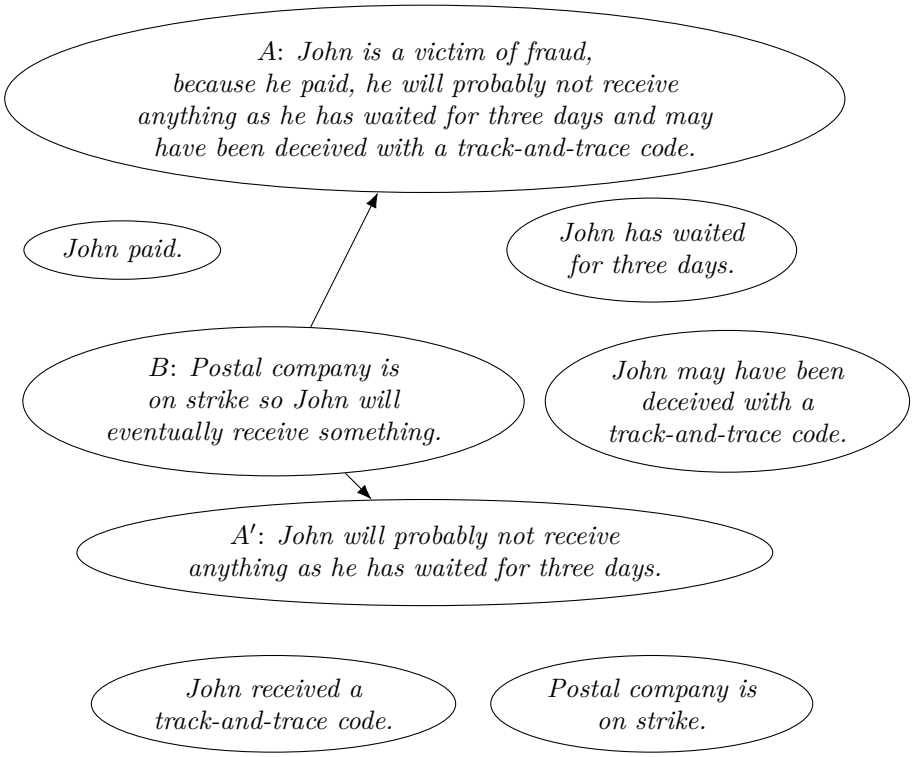


Figure 1.1: Example of an abstract argumentation framework in the domain of online trade fraud.

often contrastive: a good explanation will not only state why a particular conclusion is appropriate, but also why alternative conclusions are not (Miller, 2019). In argumentation, a contrastive explanation can be derived from the attack relation between arguments, for instance by providing counterarguments or arguments attacking counterarguments (Dietz et al., 2022; Borg and Bex, 2022). The cautiousness requirement is fulfilled by choosing the grounded semantics. Therefore, it seems promising to use argumentation-based techniques for developing AI in law enforcement. Whereas a formal specification of abstract and structured approaches of argumentation will follow in Chapters 2 and 3 respectively, the example below already gives some intuition.

Example 1.1 (Argumentation)**.** John wants to buy a bike and sees an advertisement on Fred's web shop. They agree on the transaction. Fred claims that he has delivered the bike to the post office and sends a track-and-trace code to John. After receiving this code, John transfers the money for the bike. However, having waited for three days, John still has not received the bicycle. He suspects that Fred is a fraudster and submits a complaint to the police. The police officer Paul acknowledges that Fred paid, he did not receive the bike and he has waited, so there is an

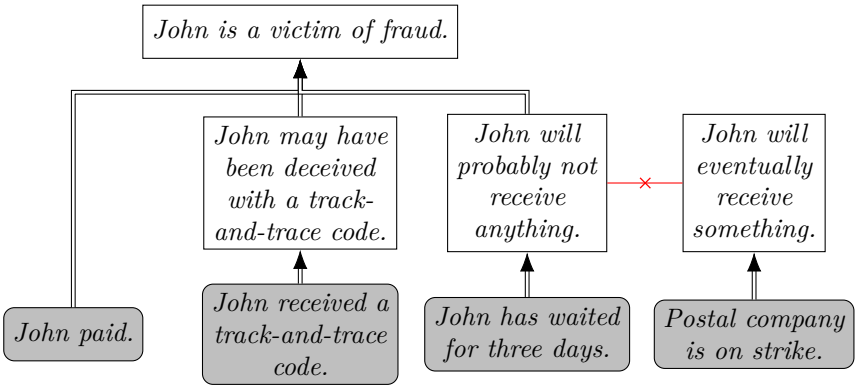


Figure 1.2: Illustration of a structured argumentation framework. It shows that the argument A is based on some defeasible rule, stating that if John paid, he may have been deceived and will probably not receive anything then he *probably* is a victim of fraud.

argument for fraud. However, he remarks that this argument is not sufficient: the postal company is on strike, hence John can still expect to receive the package.

From an *abstract* view on argumentation, there is John's argument for fraud (A), which is attacked by Paul's argument (B), see Figure 1.1. Following the theory of semantics in abstract argumentation, one can decide which argument(s) should be accepted. For example, given that argument B is not attacked, we could say that it should be accepted. On the contrary, argument A should not be accepted as it is attacked by an accepted argument. Note that the structure of the arguments is not considered when deciding which arguments should be accepted: arguments are viewed as abstract entities between which attack relations exist.

From a *structured* view on argumentation, the structure of arguments does play a role in arguments and attacks, see Figure 1.2. Let us consider John's argument A more closely. John claims to be a victim of fraud as he paid for a product, does not expect to receive it and it seems that he has been deceived in some way. The reason that John does not expect that he will still receive the bike is argument A': he has already waited for three days. Argument A' is a subargument of the argument A for fraud. In fact, Paul's argument B attacks argument A'. Given that A' is a subargument of A, this implies that B attacks A as well.

1.1.2. Precedent-based reasoning

A second type of reasoning that is similar to the way humans reason in law enforcement is *precedent-based reasoning*. In particular, this type of reasoning mimics how judges decide a case based on earlier cases: if a new case is at least as strong towards an outcome as an earlier (precedent) case with that outcome, then the new case is constrained to have the same outcome (Horty, 2004; Horty, 2011). Chapter 4 addresses the formal definitions, but the example below is intended to provide some intuition.

Web shop	Terms	Safe payment	Complaints	Outcome
X	No	Yes	Some	Mala fide
Y	No	No	Many	?

Table 1.1: Characteristics of potentially mala fide web shops. Given that web shop Y has characteristics that are even more mala fide than web shop X, *a fortiori* Y is mala fide as well.

Example 1.2 (Precedent-based reasoning)**.** Paul is analysing web shops. For this task, he takes into account the characteristics presented in Table 1.1. Yesterday, he decided that web shop X, which did not have a page with terms and conditions, provided a safe payment possibility and against which some complaints were submitted, should be taken offline as it is probably fraudulent. Today, he analysed web shop Y, which does not have a terms and conditions page either, does not have a safe payment possibility and there are many complaints submitted against this web shop. Given that X was probably fraudulent and that Y is "even more" mala fide based on its properties, Paul reasons that web shop Y should be taken offline as well, based on the precedent X.

1.1.3. Static reasoning with arguments and precedents

Sections 1.1.1 and 1.1.2 described three formalisms that are promising for explaining machine-made decisions to humans: abstract argumentation frameworks, structured argumentation frameworks (for argumentation-based reasoning) and models of precedential constraint (for precedent-based reasoning). Although each of the three formalisms has been studied before, up until this point most research has been devoted to the "static" problems in these formalisms, such as: given the current information regarding arguments or precedents, which conclusions can we draw *at this moment*? In this thesis, we will refer to this problem as the task of determining the *justification* status. Informally, this can be seen as giving an initial advice regarding a specific topic, thereby only considering information that is currently available. A topic can be an argument (in an abstract argumentation framework), a statement (in structured approaches to argumentation) or an outcome (in the precedential model of constraint).

1.2. Dynamic reasoning with incomplete information

By determining the justification status, it is possible to transparently decide whether a topic should be accepted given current information. However, solving this static problem is insufficient for solving the problems arising in police practice. These problems are dynamic in that new information may lead to new arguments or precedents. Police officers need to reason with incomplete information: typically, not all information is known, but they do know beforehand which potential new information they would like to take into account. In case this new information may still be relevant, they further inquire into this before drawing a conclusion. This is illustrated by the following example.

Example 1.3. Paul is handling the intake of fraud complaints by citizens. Ann enters the police station and states that she has not received a product, although she has waited for three months. At this point, Paul cannot yet decide whether Ann should file a complaint, because some information is still missing. In particular, Ann did not state if she paid for the product. If she did not, she is certainly not a victim of fraud; if she did pay, she might be. Therefore, Paul first inquires upon the payment (and possibly has to ask additional information) before he can decide whether Ann should submit her complaint.

In the field of computational argumentation, dynamic argumentation (Doutre and Mailly, 2018) is a research direction that assumes that the argumentation framework is not static but can be altered by, for instance, the addition or removal of arguments and/or attacks, possibly resulting in a changed acceptability status of some argument(s). Research on dynamic argumentation includes work on the impact of a change operation (Cayrol et al., 2010; Alfano et al., 2021a), enforcement (Baumann and Brewka, 2010; Doutre and Mailly, 2018; Borg and Bex, 2021b), resolution (Modgil and Prakken, 2012) and the relation with belief revision (Falappa et al., 2009; Snaith and Reed, 2016). However, none of these approaches is designed for the types of reasoning with incomplete information that were sketched before. In order to handle situations similar to the previous example, it is therefore required to extend the aforementioned formalisms so that they can reason with incomplete information.

Research question 1. How can argumentation- and precedent-based formalisms be extended to enable reasoning with incomplete information?

- A How can incomplete information be represented?
- B In which situations does new information not update the justification status?
- C Which new information can still change the justification status?

Representing incomplete information As a first step towards enabling reasoning with incomplete information, research question 1A must be answered. In other words, argumentation- and precedent-based formalisms must be extended to represent incomplete information. For abstract approaches to argumentation, such an extension already exists in the notion of Incomplete Argumentation Frameworks (IAFs). IAFs explicitly include uncertain arguments and attacks and have received considerable attention in recent research (Cayrol et al., 2007; Baumeister et al., 2018; Baumeister et al., 2021; Mailly and Rossit, 2020). For structured approaches to argumentation, an extension to represent incomplete information only exists in a restricted fragment of ASPIC$^+$. Testerink et al. (2019a) introduced a formalism that adds the concept of queryable literals to ASPIC$^+$ argumentation theories. Such queryable literals can be seen as uncertain knowledge. In the field of precedent-based reasoning, no such extension exists. Current formalisms assume that all information is certain in that all characteristics of precedents and the new (focus) case are known (Horty, 2004; Horty, 2011).

In this thesis we therefore extend (a more general fragment of) ASPIC$^+$, as well as the model of precedential constraint so that they explicitly represent qualitative uncertainty. Each of these extended formalisms encodes incomplete information and thus effectively represents a potentially large set of possible ways in which the uncertainty can be resolved.

In order to answer research questions 1B and 1C, we introduce the (theoretical) notions of *stability* and *relevance* that are defined for all three extended formalisms.

Stability The goal of the notion of stability is to identify situations in which a decision can be made, even though there is still some uncertainty about the information. For example, the premises of arguments or the exact characteristics of a focus case may not be known in advance. Informally, the topic satisfies *stability* if for all possible ways of resolving this uncertainty, the justification status is the same. In an application, this means that the corresponding advice will not change, regardless of information updates.

Example 1.4. Suppose that Ann answered Paul that she did not pay. In that case, the topic claim that she has been a victim of fraud cannot be justified, regardless of additional information that Ann may have. We then say that the topic is stable. This implies that there is no reason for Paul to ask additional questions.

Relevance In situations where the topic is not stable, it is not clear which decision should be made as some decisive information is missing. It is therefore necessary to identify *relevant* information updates, in that investigation into the presence of this information possibly leads to a stable topic.

Example 1.5. Just after Ann told her story, the topic claim was not yet stable. In order to reach a stable situation, Paul identifies information that is still relevant and therefore asks Ann whether she did pay.

Reasoning with incomplete information In order to make AI systems that, just like Paul, are able to identify stability and relevance, we will define these notions in the three aforementioned formalisms (abstract and structured approaches to argumentation, as well as precedent-based reasoning) in Chapters 2, 3 and 4 respectively. One of the contributions in Chapter 3 is a comparison between the notions of stability and relevance in abstract approaches to these notions in structured approaches. This comparison demonstrates that the definitions of stability and relevance on a structured level are more suitable for rule-based reasoning with incomplete information.

Human-in-the-loop decision support These notions will then be used in a human-in-the-loop decision-making procedure, as illustrated in Figure 1.3. The upper part of this figure shows a system that is only able to compute the justification status based on certain information and simply informs the user. The lower part of the figure illustrates the human-in-the-loop procedure for decision-making with

incomplete argumentation. Such a procedure requires algorithms for stability and relevance identification. Given the certain and uncertain information, the stability algorithm first decides whether the current situation is stable. If this is the case, a final advice can be presented to the human analyst. Otherwise, the algorithm for relevance identifies those uncertainties that should be investigated by the analyst. After investigation, the analyst can return findings to the system. This process is repeated until a final advice is found (or the analyst decides not to investigate any further).

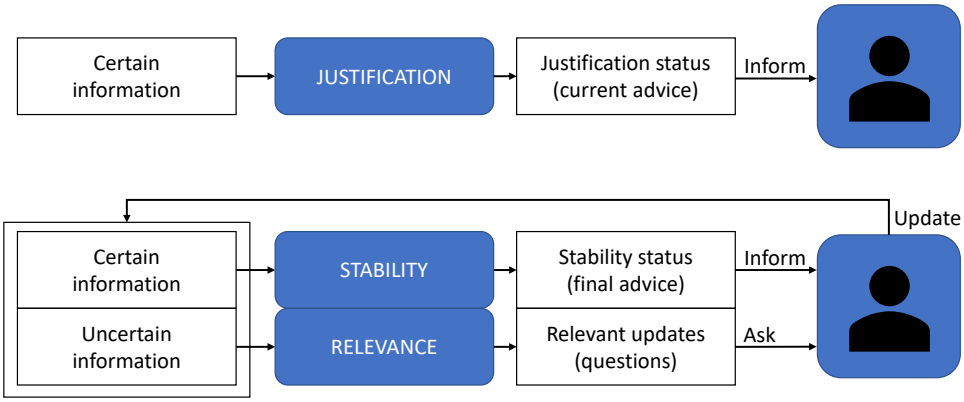


Figure 1.3: High-level overview of the proposed human-in-the-loop decision-making process, involving algorithms for justification, stability and relevance identification.

1.3. Computational aspects

The previous section informally sketched how the notions of justification, stability and relevance contribute to human-in-the-loop systems that support decision-making with incomplete information. In order to develop efficient algorithms, it is important to know the complexity class in which a problem is situated.

Even though the idea of finding the justification status is not new, it has not been studied from a computational perspective for the formalisms of ASPIC$^+$ or precedential constraint. For abstract notions of argumentation, the complexity of the acceptance of an argument (which is closely related to justification) has been studied in earlier work (Dimopoulos and Torres, 1996; Dunne and Bench-Capon, 2002; Coste-Marquis et al., 2005). Furthermore, for stability in IAFs only loose complexity boundaries have been established by Mailly and Rossit (2020). The notion of relevance has not been studied in any of the three formalisms. Therefore, the following question needs to be answered to enable development of efficient algorithms.

Research question 2. In which complexity classes are the problems of justification, stability and relevance in the context of argumentation- and precedent-based formalisms situated?

Chapters 2, 3 and 4 will answer this question for each of the three formalisms. Many of these problems will turn out to be rather complex, being situated in NP, CoNP or even higher complexity classes. These answers are interesting from a theoretical perspective, as they give insight into the complexity of reasoning with incomplete information. From a practical perspective, they are important as well: their situation in high complexity classes implies that it is unlikely (under the common assumption that P $\neq$ NP even impossible) that there exist exact algorithms that solve these problems in polynomial time. We can thus either use inexact polynomial approaches or exact exponential algorithms. We will try both approaches in answering the following research question, which is answered for each of the three formalisms in Chapters 2, 3 and 4.

Research question 3. How can the problems of justification, stability and relevance in the context of argumentation- and precedent-based formalisms be solved efficiently?

1.4. Application in law enforcement

Finally, the algorithms for solving justification, stability and relevance can be used in the decision support systems described above. Still, for practical applications in specific domains there are some additional questions that need to be answered. For example, in order to automatically obtain information from user input, we need some procedure for information extraction from natural language. Which methods are practical for this purpose? Another question, which may differ between various applications, would be: how to specify which information could be added in the future? In general, in order to apply the human-in-the-loop decision support system to solve actual police problems, we need to answer the following research question:

Research question 4. How can the algorithms for stability and relevance be applied in practical decision support systems at the police?

We will answer this last research question in Chapter 5, by discussing two instances of the decision support system at the Dutch police, for fraud intake and for web shop classification.

1.5. Thesis outline

This thesis is structured as follows. The following three chapters will answer the first three research questions for each of the aforementioned formalisms. Chapter 2 describes a complexity analysis and exact algorithms for stability and relevance for IAFs, thus answering research questions 1, 2 and 3 for an abstract approach to argumentation. For ASPIC$^+$, a structured approach to argumentation, research questions 1, 2 and 3 are answered in Chapter 3, which presents the complexity analysis and various algorithms for justification, stability and relevance. Answers to research questions 1, 2 and 3 for precedent-based reasoning are covered in Chapter 4. After covering the theoretical aspects in these three chapters, Chapter 5 describes the applications, answering research question 4. Chapter 6 concludes this thesis.

2

Stability & relevance in abstract approaches to argumentation

This chapter is based on the following papers:

1. In Odekerken et al. (2022b) the problems of stability and relevance were introduced for incomplete argumentation frameworks and the complexity of part of these problems was established. This conference paper, presented at COMMA 2022, was written under supervision of AnneMarie Borg and Floris Bex.
2. Odekerken et al. (2023e) is the journal version of the aforementioned conference paper, published in Argument & Computation. Compared to the conference version, the journal version contains full proofs and a complexity analysis of all variants of the stability and relevance problem in four semantics. This paper was written under supervision of AnneMarie Borg and Floris Bex.
3. The algorithms for computing relevance in incomplete argumentation frameworks were proposed in Odekerken (2024). This paper is accepted for COMMA 2024.

2.1. Introduction

A central concept in computational argumentation is that of argumentation frameworks (AFs), in which arguments and the attack relation between them are represented as a directed graph where nodes correspond to arguments and edges correspond to attack relations between arguments (Dung, 1995). One of the assumptions in such Dung-style argumentation frameworks is that all arguments and attacks are known. However, in practice, argumentation is a dynamic process in which not all arguments and attacks may be known in advance. For example, not all the evidence on which arguments are based might already have been observed, making the presence of a specific argument uncertain. AFs are not able to represent qualitative uncertainty on the existence of specific arguments and attacks. For this reason, incomplete argumentation frameworks (IAFs) have been proposed. Initially introduced as partial AFs in Cayrol et al. (2007), IAFs have received considerable attention in recent research (Baumeister et al., 2018; Mailly and Rossit, 2020; Skiba et al., 2020; Fazzinga et al., 2020; Baumeister et al., 2021; Alfano et al., 2022; Mailly, 2023). IAFs are an extension to AFs in which not only certain arguments are specified, but also arguments and attacks for which it is uncertain whether they are present. By deciding for every uncertain argument and attack whether it is present or absent, it is possible to "complete" an IAF, turning it into an AF. Thus, an IAF represents a set of possible AFs, its *completions*.

Since every completion of an IAF is an AF, standard argumentation semantics can be applied to determine the extensions of any completion (Dung, 1995). An extension is a collectively acceptable set of arguments. Based on these extensions, the *justification* status of every argument in every completion of the IAF can be determined, stating whether the argument should be accepted, rejected or otherwise. Note that the notion of justification status is only defined on the AFs that make up the completions of an IAF, but not on the IAF itself.

Now suppose that we are interested in the justification status of a particular argument, but are faced with an IAF containing uncertainties concerning the presence of (other) arguments and attacks. Then we need to know whether it is required to resolve these uncertainties. Note that this is not required in situations in which, no matter which uncertain arguments we add or remove, the justification status of the argument of interest will always be the same. In other words, if this argument has the same justification status in each completion of the IAF, there is no need for further investigation into the uncertain arguments and attacks. In such a situation, we say that the argument is stable with respect to the IAF and justification status. The detection of such *stability* has practical applications. For instance, as discussed in Chapter 1, it serves as a termination criterion for argumentative human-in-the-loop decision support systems: stability detection prevents the system from asking unnecessary questions to the user. Another example, proposed by Mailly and Rossit (2020), is the application of stability detection in negotiating agents to recognise situations in which an agent should stop negotiating and accept its opponent's offer.

In the case that the argument of interest is not stable in the given IAF, we know that the system or agent should collect more information. However, not all information that is currently unknown can influence the justification status of the argument of interest. It is therefore interesting to determine which uncertainties should be resolved in order to reach a point where the argument is stable. In other words: which uncertain arguments or attacks are still *relevant* for the justification status? Adding relevance to an inquiry or negotiation process ensures that the questions that are asked contribute to reaching stability.

One contribution of this chapter is the definition of stability and relevance for abstract approaches to argumentation. Our definition of stability in this context is more fine-grained than the definition proposed earlier by Mailly and Rossit (2020). The notion of relevance has not been studied before in this context. The second contribution is the extensive study of the computational aspects of justification, stability and relevance in the context of IAFs. Specifically, it presents precise complexity results for each of these three problems under grounded, complete, stable and preferred semantics. As a third contribution, we propose algorithms for identifying stability, as well as the first algorithms for finding relevant updates for grounded and complete semantics in the context of IAFs. A common approach for handling complex problems (on the first level of the polynomial hierarchy) in computational argumentation is by reduction to other formalisms, employing SAT- or ASP-solvers (Cerutti et al., 2018). In this line of research, we propose ASP-based algorithms for enumerating relevant updates.[1] Our empirical evaluation shows that the algorithms are promising, even on inputs with many (uncertain) arguments and attacks. These three contributions correspond to answers to the first three research questions identified in Chapter 1.

The chapter is structured as follows. Section 2.2 provides the necessary preliminaries. In Section 2.3, we introduce the problems of justification for AFs and stability and relevance for IAFs. We then provide complexity results for all variants of these problems in Section 2.4. In Section 2.5, we propose exact algorithms for solving the stability and relevance problems in the context of IAFs under grounded and complete semantics and evaluate their runtime. Related work is discussed in Section 2.6; we conclude in Section 2.7.

2.2. Preliminaries

In this section, we recall the most important notions from abstract argumentation and the associated semantics (Dung, 1995), as well as incomplete argumentation frameworks and their completions (Cayrol et al., 2007; Baumeister et al., 2018; Baumeister et al., 2021; Mailly and Rossit, 2020). We also provide a brief introduction to the polynomial hierarchy, which is required for the complexity study, and to answer set programming, which is necessary for understanding the proposed algorithms.

[1] All algorithms are available on GitHub at `https://github.com/DaphneOdekerken/asp_relevance` and can be tested in an interactive visualisation at `https://pyarg.npai.science.uu.nl/24-visualise-iafs`.

2.2.1. Argumentation frameworks and semantics

An argumentation framework (AF) $\langle \mathcal{A}, \mathcal{C} \rangle$, as introduced in Dung (1995), consists of a set $\mathcal{A}$ of arguments and a binary attack relation $\mathcal{C} \subseteq \mathcal{A} \times \mathcal{A}$ on them, where $(A, B) \in \mathcal{C}$ indicates that argument A attacks argument B. In this chapter, $\mathcal{A}$ is assumed to be finite. The evaluation of arguments is done using the following semantics (Dung, 1995).

Definition 2.1 (Extension-based semantics)**.** Let $AF = \langle \mathcal{A}, \mathcal{C} \rangle$ be an AF with arguments $\mathcal{A}$ and attack relation $\mathcal{C}$ and let $S \subseteq \mathcal{A}$. Then:

- S is *conflict-free* iff for each $X, Y \in S : (X, Y) \notin \mathcal{C}$;
- S *defends* $X \in \mathcal{A}$ iff for each $Y \in \mathcal{A}$ such that $(Y, X) \in \mathcal{C}$, there is a $Z \in S$ such that $(Z, Y) \in \mathcal{C}$;
- S is an *admissible set* iff S is conflict-free and S defends each $X \in S$;
- S is a *complete extension* (CP) iff S is admissible and each $X \in \mathcal{A}$ that is defended by S is in S;
- S is a *preferred extension* (PR) iff it is a set inclusion maximal admissible set;
- S is the *grounded extension* (GR) iff it is the set inclusion minimal complete extension; and
- S is a *stable extension* (ST) iff it is complete and attacks all the arguments in $\mathcal{A} \setminus S$.

Example 2.1. Figure 2.1 shows an example of an argumentation framework $AF = \langle \mathcal{A}, \mathcal{C} \rangle$ where $\mathcal{A} = \{A, B, C, D, E\}$ and $\mathcal{C} = \{(A, B), (B, C), (C, B), (D, C), (D, E), (E, D)\}$. The grounded extension of AF is $\{A\}$. This is also a complete extension. Additionally, there are two complete extensions that are also preferred and stable: $\{A, C, E\}$ and $\{A, D\}$.

In order to decide if an argument should be accepted w.r.t. a given AF and semantics, different strategies can be chosen. We will refer to these as acceptability strategies, following Borg and Bex (2021a, Definition 1). Dependent on the strategy, the argument is accepted if and only if it occurs in one and/or all extensions.

Definition 2.2 (Acceptability strategies)**.** Let $AF = \langle \mathcal{A}, \mathcal{C} \rangle$ be an argumentation framework and σ some semantics in $\{\text{GR}, \text{CP}, \text{PR}, \text{ST}\}$ and let A be some argument in $\mathcal{A}$.

- A is sceptically accepted w.r.t. σ semantics iff A belongs to *each* σ-extension of AF;
- A is credulously accepted w.r.t. σ semantics iff A belongs to *some* σ-extension of AF; and
- A is sceptically-existent accepted w.r.t. σ semantics iff AF has at least one σ-extension and A belongs to *each* σ-extension of AF.

We refer to *sceptical*, *credulous* and *sceptical-existent* as acceptability strategies.

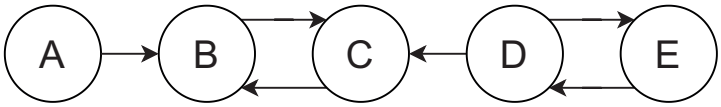


Figure 2.1: An example of an argumentation framework. Arguments are depicted as circles, whereas attacks are depicted as arrows.

Note that the sceptical-existent acceptability strategy only differs from the sceptical strategy for ST semantics: for GR, CP and PR semantics, each AF has at least one extension, while it is possible for an AF to have no ST extension (Dung, 1995).

Example 2.2. In the AF illustrated in Figure 2.1, only the argument A is sceptically accepted w.r.t. CP semantics. The arguments C, D and E are all credulously accepted. Argument B is not accepted for any acceptability strategy.

2.2.2. Incomplete argumentation frameworks

Incomplete argumentation frameworks (IAFs) are an extension to AFs, initially proposed as partial AFs in Cayrol et al. (2007). In an IAF, the set of arguments and attacks is split into two disjoint parts: a certain part ($\mathcal{A}$ and $\mathcal{C}$) and an uncertain part ($\mathcal{A}^?$ and $\mathcal{C}^?$). For the uncertain elements, it is unknown whether they are part of the argumentation framework or not. They may be added in the future, for example, because more information is acquired, or removed, for example, because after investigation, this element turned out not to be present in the given setting.

Definition 2.3 (Incomplete argumentation framework)**.** An incomplete argumentation framework is a tuple $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$, where $\mathcal{A} \cap \mathcal{A}^? = \emptyset$, $\mathcal{C} \cap \mathcal{C}^? = \emptyset$ and:

- $\mathcal{A}$ is the set of certain arguments;
- $\mathcal{A}^?$ is the set of uncertain arguments;
- $\mathcal{C} \subseteq (\mathcal{A} \cup \mathcal{A}^?) \times (\mathcal{A} \cup \mathcal{A}^?)$ is the certain attack relation; and
- $\mathcal{C}^? \subseteq (\mathcal{A} \cup \mathcal{A}^?) \times (\mathcal{A} \cup \mathcal{A}^?)$ is the uncertain attack relation.

Example 2.3 (IAF)**.** Figure 2.2 shows an example of an incomplete argumentation framework $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ where $\mathcal{A} = \{B, C, E\}$, $\mathcal{A}^? = \{A, D\}$, $\mathcal{C} = \{(A, B), (B, C), (D, C), (D, E), (E, D)\}$ and $\mathcal{C}^? = \{(C, B)\}$. Arguments A and D are currently absent but may be added in the future. The attack from A to B is certainly present if A is present. Similarly, the attacks (D, C), (D, E) and (E, D) are certainly present if D is present. The attack from C to B, on the other hand, is uncertain: although the arguments B and C are certainly present, the attack itself is currently absent but may still be added.

An incomplete argumentation framework can be *completed* by deciding for all uncertain arguments and attacks whether or not they are present, as defined below.

Definition 2.4 (Completions)**.** Given an IAF $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$, a completion is any AF $\langle \mathcal{A}', \mathcal{C}' \rangle$ that satisfies $\mathcal{A} \subseteq \mathcal{A}' \subseteq \mathcal{A} \cup \mathcal{A}^?$ and $\mathcal{C}|_{\mathcal{A}'} \subseteq \mathcal{C}' \subseteq (\mathcal{C} \cup \mathcal{C}^?)|_{\mathcal{A}'}$ where

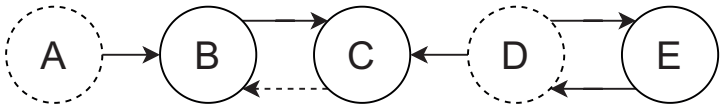


Figure 2.2: An example of an incomplete argumentation framework. Certain arguments are depicted as circles with solid borders, whereas uncertain arguments are circles with dashed borders. Attacks are depicted as arrows, which have a solid line if they represent certain attacks and a dashed line if they represent uncertain attacks.

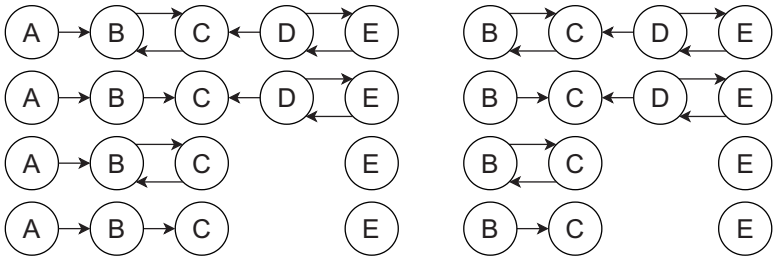


Figure 2.3: The eight completions of our incomplete argumentation framework.

the restriction $\mathcal{C}|_{\mathcal{A}'}$ of a set of attacks $\mathcal{C}$ to a set of arguments $\mathcal{A}'$ is defined as $\mathcal{C}|_{\mathcal{A}'} = \{(A, B) \in \mathcal{C} \mid A \in \mathcal{A}' \text{ and } B \in \mathcal{A}'\}$.

Example 2.4 (Completions)**.** The incomplete argumentation framework from our previous example has eight completions. These are illustrated in Figure 2.3.

Since completions are abstract argumentation frameworks, the semantics from Section 2.2.1 can be used to evaluate arguments in the completions of an incomplete argumentation framework. This leads to two ways of defining acceptance for incomplete argumentation frameworks, proposed in Baumeister et al. (2021, pages 6-7): necessary and possible acceptance. Informally, some argument is necessarily accepted if it is accepted in each completion, whereas the argument is possibly accepted if this holds for some completion. The definition below make a distinction between the different acceptability strategies.

Definition 2.5 (Possible and necessary acceptance)**.** Let $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ be an incomplete argumentation framework, let $A \in \mathcal{A}$ be a certain argument, let σ in $\{\text{GR}, \text{CP}, \text{PR}, \text{ST}\}$ be some semantics and let $\alpha \in \{$*sceptical*, *credulous*, *sceptical-existent*$\}$ be some acceptability strategy.

- A is *possibly* α-σ accepted w.r.t. $\mathcal{I}$ if and only if A is α-σ accepted in *some* completion of $\mathcal{I}$, and
- A is *necessarily* α-σ accepted w.r.t. $\mathcal{I}$ if and only if A is α-σ accepted in *each* completion of $\mathcal{I}$.

Example 2.5 (Necessary and possible acceptance)**.** Figure 2.4 displays the CP extensions of each of the eight completions of $\mathcal{I}$ from Example 2.4, where the completion is repeated for each CP extension. None of the certain arguments in $\mathcal{I}$ is

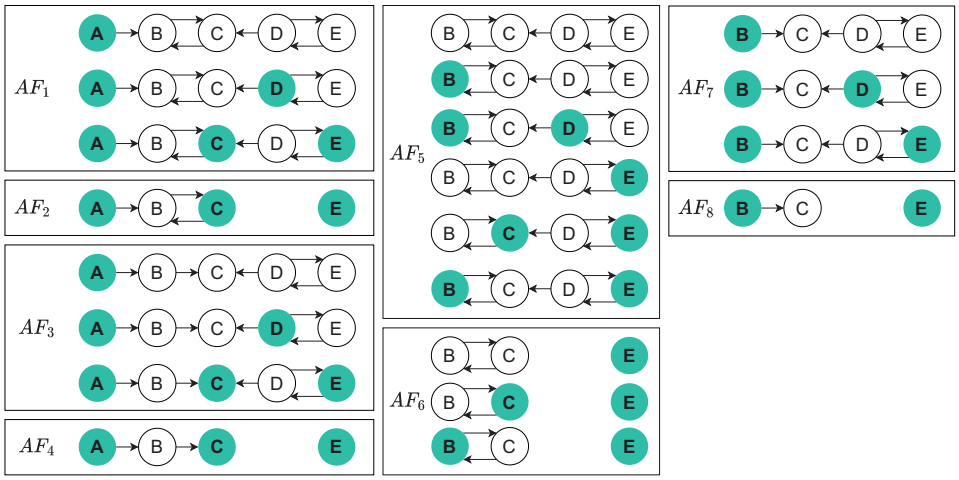


Figure 2.4: Visualisation of the CP extensions of each of the eight completions of $\mathcal{I}$ from Example 2.3, where the completion is repeated for each CP extension and argument that are present in that extension are coloured green.

necessarily sceptically accepted w.r.t. CP semantics, because no argument is in all extensions of all completions. Note that A, although it is present in all extensions of all completions of $\mathcal{I}$ in which it occurs, is not necessarily sceptically accepted w.r.t. CP semantics: A is not a certain argument in $\mathcal{I}$. The only argument that is necessarily credulously accepted w.r.t. CP semantics is E. B is possibly sceptically accepted w.r.t. CP semantics, because the argument is in each extension of the completion AF_7 (as well as AF_8). C and E are possibly sceptically accepted w.r.t. CP semantics as well, thanks to their presence in the only extension of AF_2. Finally, all certain arguments of $\mathcal{I}$ (B, C and E) are possibly credulously accepted w.r.t. CP semantics.

The definitions of necessary and possible acceptance are particularly interesting for our work, as they are strongly related to the notion of stability, to be defined formally in Section 2.3.2. Before we do so, we first explain the polynomial hierarchy in the next section.

2.2.3. The polynomial hierarchy

In this section, we give a brief introduction to the polynomial hierarchy – for a more detailed explanation, we refer to Dvorák and Dunne (2018). This is required for the complexity study presented in this chapter as the problems we study are situated on various levels in the polynomial hierarchy.

The polynomial hierarchy (Papadimitriou, 1994) is a hierarchy of complexity classes defined using oracle machines, i.e., Turing machines that are allowed to call a subroutine (oracle), deciding some fixed problem in constant time. For a class of decision problems $\mathbf{C}$ and a class $\mathcal{X}$ defined by resource bounds, $\mathcal{X}^{\mathbf{C}}$ denotes the class of

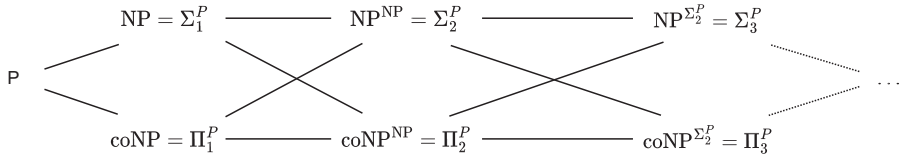


Figure 2.5: The relation between complexity classes in the polynomial hierarchy. The lines between classes denote a subset relationship: all problems in the class of the left side are also contained in the class on the right side.

problems decidable on a Turing machine with a resource bound given by $\mathcal{X}$ and an oracle for a problem in $\mathbf{C}$. For example, problems in NP are decidable with a resource bound given by NP and an oracle for a problem in P, therefore $\text{NP} = \text{NP}^{\text{P}}$.

Based on these notions, the sets Σ_k^p and Π_k^p are defined as follows: $\Sigma_0^p = \Pi_0^p = P$, $\Sigma_{k+1}^p = \text{NP}^{\Sigma_k^p}$ and $\Pi_{k+1}^p = \text{CoNP}^{\Sigma_k^p}$. So problems in Σ_2^p are decidable with a resource bound given NP and an oracle for a problem in $\Sigma_1^p = \text{NP}$. The polynomial hierarchy (PH) is then defined as the union of these complexity classes: $\text{PH} = \bigcup_{k=0}^{\infty} \Sigma_k^p = \bigcup_{k=0}^{\infty} \Pi_k^p$. Finally, note that the definition over classes in the polynomial hierarchy imply a subset relation, illustrated in Figure 2.5. By this relation, for each $i \in \mathbb{N}$, each problem in Σ_i^p is also in Σ_{i+1}^p, as well as in Π_{i+1}^p.

A problem that is Σ_2^p-complete is Σ_2-SAT (Stockmeyer, 1976); in this chapter we will use the CNF formulation that is also used in, e.g. Baumeister et al. (2018) and Baumeister et al. (2021). An instance of Σ_2-SAT would be (Φ, X, Y), where Φ is an input formula in CNF over the pairwise disjoint sets of propositional variables X and Y. Then the Σ_2-SAT problem is to decide if there exists a truth value assignment τ_X to variables of X such that for each truth value assignment τ_Y: $\Phi[\tau_X, \tau_Y] = \text{False}$, where $\Phi[\tau_X, \tau_Y]$ is the truth value that Φ evaluates to when applying the assignment τ_X to X and τ_Y to Y.

2.2.4. Answer set programming

This section gives a brief introduction to answer set programming. A normal ASP program π is a set of rules of the form $\mathbf{b_0} \leftarrow \mathbf{b_1}, \ldots, \mathbf{b_k}, \mathbf{not}\ \mathbf{b_{k+1}}, \ldots, \mathbf{not}\ \mathbf{b_m}.$, where each $\mathbf{b_i}$ is an atom. A rule is positive if $k = m$. A rule without head $\mathbf{b_0}$ is a constraint and a shorthand for $\mathbf{a} \leftarrow \mathbf{b_1}, \ldots, \mathbf{b_k}, \mathbf{not}\ \mathbf{b_{k+1}}, \ldots, \mathbf{not}\ \mathbf{b_m}, \mathbf{not}\ \mathbf{a}.$ for a fresh $\mathbf{a}$. An atom $\mathbf{b_i}$ has the form $\mathbf{p}(\mathbf{t_1}, \ldots, \mathbf{t_n})$, where each $\mathbf{t_j}$ is either a constant or a variable. An answer set program is ground if it is free of variables. For a non-ground program, the grounded version is the set of rules obtained by applying all possible substitutions from the variables to the set of constants appearing in the program. An interpretation I, i.e., a subset of all the ground atoms, satisfies a positive rule $\mathbf{r} = \mathbf{b_0} \leftarrow \mathbf{b_1}, \ldots, \mathbf{b_k}.$ if and only if all positive body elements $\mathbf{b_1}, \ldots, \mathbf{b_k}$ being in I implies that the head atom is in I. For a program π consisting only of positive rules, let $Cl(\pi)$ be the uniquely determined interpretation I that satisfies all rules in π and no subset of I satisfies all rules in π. An interpretation I is an answer set of a ground program π if $I = Cl(\pi^I)$ where π^I is the reduct, defined as follows.

$$\pi^I = \{\mathbf{b_0} \leftarrow \mathbf{b_1}, \ldots, \mathbf{b_k}. \mid \mathbf{b_0} \leftarrow \mathbf{b_1}, \ldots, \mathbf{b_k}, \mathbf{not}\ \mathbf{b_{k+1}}, \ldots, \mathbf{not}\ \mathbf{b_m}. \in \pi \text{ and } \{\mathbf{b_{k+1}}, \ldots, \mathbf{b_m}\} \cap I = \emptyset\}$$

For a non-ground program π, an interpretation I is an answer set if I is an answer set of the grounded version of π.

2.3. Justification, stability and relevance

In this section, we define the notions of justification status, stability and relevance in the context of IAFs. The *justification* status of an argument is a notion of acceptance for (certain) arguments in abstract argumentation frameworks (including completions of IAFs). It is defined in Section 2.3.1. For a given IAF and certain argument, the justification status of that argument need not be the same in all completions. This is the issue of *stability*, defined in Section 2.3.2, where an argument is stable if its justification status is the same in all completions. For arguments that are not stable in an IAF, the *relevance* problem is of interest: which uncertain arguments or attacks should be investigated for the argument to become stable? This problem is defined in Section 2.3.3.

2.3.1. Justification status

We first define the notion of justification status. The justification status is a notion of acceptance for arguments in abstract argumentation frameworks that is more fine-grained than only considering the presence or absence in extensions as it combines acceptability strategies with the labels IN, OUT and UNDEC. Given an AF $\langle \mathcal{A}, \mathcal{C} \rangle$, an argument A and a semantics σ, A's justification status can be determined by either considering all σ-extensions (sceptical and sceptical-existent) or at least one σ-extension of the AF (credulous). In this context, an argument can be IN (part of all/some σ-extensions), OUT (attacked by all/some σ-extensions) or UNDEC (otherwise).[2]

Definition 2.6 (Argument justification status)**.** Let $AF = \langle \mathcal{A}, \mathcal{C} \rangle$ be an argumentation framework and σ some semantics in $\{\text{GR}, \text{CP}, \text{PR}, \text{ST}\}$. Let A be some argument in $\mathcal{A}$.

- The justification statuses for the IN label are:
  - A is σ-sceptical-IN iff A belongs to each σ-extension of AF;
  - A is σ-credulous-IN iff A belongs to some σ-extension of AF; and
  - A is σ-sceptical-existent-IN iff AF has a σ-extension and A belongs to each σ-extension of AF.
- The justification statuses for the OUT label are:

[2]Note that IN, OUT and UNDEC labels are also used as reinstatement labellings by Caminada (2006). The approach used in this chapter is different as it defines justification statuses based on extensions, rather than defining them directly on the argumentation framework.

  - A is σ-sceptical-OUT iff for each σ-extension S of AF, A is attacked by some argument in S;
  - A is σ-credulous-OUT iff for some σ-extension S of AF, A is attacked by some argument in S; and
  - A is σ-sceptical-existent-OUT iff AF has a σ-extension and for each σ-extension S of AF, A is attacked by some argument in S.

- The justification statuses for the UNDEC label are:
  - A is σ-sceptical-UNDEC iff for each σ-extension S of AF, A is not in S and not attacked by any argument in S;
  - A is σ-credulous-UNDEC iff for some σ-extension S of AF, A is not in S and not attacked by any argument in S; and
  - A is σ-sceptical-existent-UNDEC iff AF has a σ-extension and for each σ-extension S of AF, A is not in S and not attacked by any argument in S.

The justification statuses that we consider in this chapter are $\{\text{GR}, \text{CP}, \text{PR}, \text{ST}\} \times \{\text{sceptical}, \text{credulous}, \text{sceptical-existent}\} \times \{\text{IN}, \text{OUT}, \text{UNDEC}\}$. In the remainder of this chapter, we refer to IN-, OUT or UNDEC-justification in case the semantics and acceptability strategy is obvious or irrelevant.

Example 2.6 (Argument justification status)**.** Consider the argumentation framework AF_1, as illustrated in Figure 2.4. For $\sigma \in \{\text{GR}, \text{CP}, \text{PR}, \text{ST}\}$, A is σ-sceptical-IN, while B is σ-sceptical-OUT. For $\sigma \in \{\text{CP}, \text{PR}, \text{ST}\}$, the arguments C, D and E are σ-credulous-IN, σ-credulous-OUT as well as σ-credulous-UNDEC.

2.3.2. Stability

Stability can be seen as a dynamic variant of justification, defined on incomplete argumentation frameworks: whereas the notion of justification only takes certain arguments and attacks into account, the notion of stability also considers arguments and attacks for which their presence is still uncertain. Whereas justification status is defined on arguments in an abstract argumentation framework, stability status is defined on certain arguments in an incomplete argumentation framework. Informally, a certain argument is *stable* if its justification is the same in all completions of the IAF. In the definition below, we define j-stability based on j-justification, where j can be any justification status considered in the previous section: $j \in \{\text{GR}, \text{CP}, \text{PR}, \text{ST}\} \times \{\text{sceptical}, \text{credulous}, \text{sceptical-existent}\} \times \{\text{IN}, \text{OUT}, \text{UNDEC}\}$.

Definition 2.7 (Stability on IAFs)**.** Given an IAF $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$, a certain argument $A \in \mathcal{A}$ and some justification status j, A is *stable-j* w.r.t. $\mathcal{I}$ iff A is j in each completion of $\mathcal{I}$.

Example 2.7 (Stability)**.** We reconsider the incomplete argumentation framework $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ from Example 2.3. The arguments in $\mathcal{A}$ are B, C and E. Figure 2.6 illustrates the complete IN/OUT/UNDEC-labellings of each of the completions

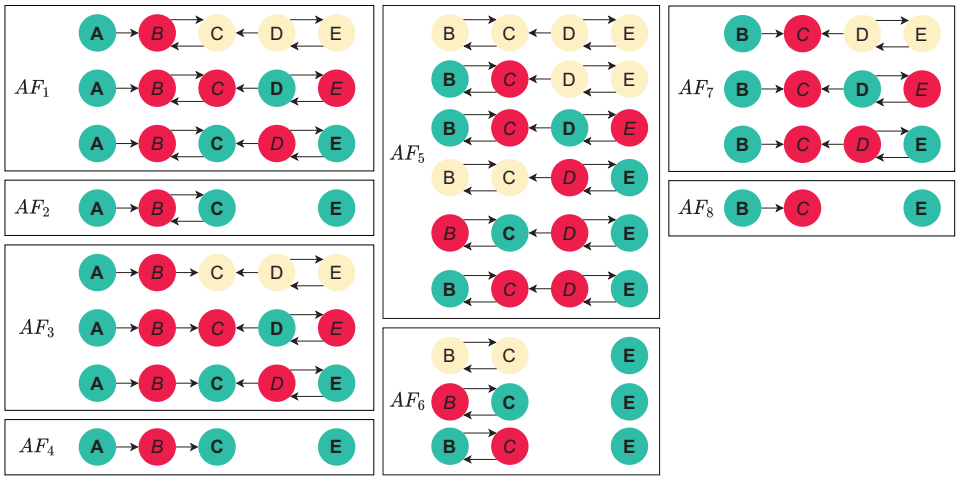


Figure 2.6: Visualisation of the complete IN/OUT/UNDEC-labellings of each of the eight unique completions of $\mathcal{I}$, where the argumentation framework is repeated for each CP extension. Arguments that are in that extension are coloured green and with boldface font; arguments attacked by some argument in that extension are red and with italic font; and all other arguments are yellow and with regular font.

$(AF_1, \ldots, AF_8)$ of $\mathcal{I}$. For each of these eight AFs, each complete extension is represented by colouring the argument nodes: nodes corresponding to arguments in the extension are coloured green and with boldface font; arguments attacked by an argument in the extension are coloured red and with italic font; all other arguments are yellow and with regular font. Note that for each completion of $\mathcal{I}$, there is at least one complete extension containing E. In other words: E is stable-CP-credulous-IN. Similarly, for each completion of $\mathcal{I}$, there is at least one preferred and stable extension containing E, so E is stable-PR-credulous-IN and stable-ST-credulous-IN as well. Under grounded semantics, E is not stable-IN, since there are completions (such as AF_1) for which E is not in the grounded extension.

For each $\sigma \in \{\text{GR}, \text{CP}, \text{PR}, \text{ST}\}$, there is no argument that is stable-σ-sceptical-IN, -OUT or -UNDEC. In practice, this means that a sceptical reasoner interested in one of the arguments in $\mathcal{A}$ would require more information.

Finally, recall that stability is not defined for A and D, since they are in $\mathcal{A}^?$ rather than $\mathcal{A}$. So although the argument A is in each GR, CP, PR and ST extension in each completion in which the argument exists, it is not stable-GR/CP/PR/ST-sceptical/credulous-IN because there are completions that do not contain A.[3]

[3]One may also be interested in an alternative notion of stability for uncertain arguments, which we call existent-stability here: an uncertain argument $A \in \mathcal{A}^?$ is j-existent-stable w.r.t. some IAF $\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ if it has justification status j in all completions $\langle \mathcal{A}', \mathcal{C}' \rangle$ such that $A \in \mathcal{A}'$. Note that the set of completions $\langle \mathcal{A}', \mathcal{C}' \rangle$ of $\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ such that $A \in \mathcal{A}'$ equals the set of completions of $\langle \mathcal{A} \cup \{A\}, \mathcal{A}^? \setminus \{A\}, \mathcal{C}, \mathcal{C}^? \rangle$. Therefore, $A \in \mathcal{A}^?$ is j-existent-stable w.r.t. some $\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ iff A is j-stable w.r.t. $\langle \mathcal{A} \cup \{A\}, \mathcal{A}^? \setminus \{A\}, \mathcal{C}, \mathcal{C}^? \rangle$. Analogously, $B \in \mathcal{A}$ is j-stable w.r.t. some $\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ iff B is j-existent-stable w.r.t. $\langle \mathcal{A} \setminus \{B\}, \mathcal{A}^? \cup \{B\}, \mathcal{C}, \mathcal{C}^? \rangle$. This implies that the problem of deciding

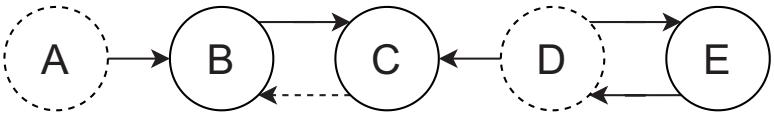


Figure 2.7: Illustration of the IAF $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ from Example 2.3.

Note that the notion of stability is strongly related to the notion of necessary acceptance, defined in Section 2.2. In fact, for any semantics σ, certain arguments are stable-σ-sceptical-IN if and only if they are necessarily sceptically accepted (i.e. in each extension of each completion); similarly, certain arguments are stable-σ-credulous-IN if and only if they are necessarily credulously accepted (i.e. in some extension of each completion). However, stability provides a more fine-grained notion of (non-)acceptance in IAFs than necessary acceptance. Using (OUT-)stability, it is, for example, also possible to express that, in any completion, some certain argument is attacked by an argument (not necessarily the same) in a σ-extension of that completion.

2.3.3. Relevance

For IAFs in which a given argument is not stable, a natural follow-up question is: which uncertainties should be resolved in order to reach a point where the argument is stable? These uncertainties are *relevant* to investigate in the given IAF. In this section, we will define the problem of relevance. First, we give some intuition on the notion of relevance in the context of IAFs.

Example 2.8. We return to the IAF $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ from Example 2.3, which is shown in Figure 2.7 (which is a repetition of Figure 2.2). Suppose that we want to know how we can make the certain argument C stable-GR-sceptical-IN. In order to do so, we should make sure that argument B is stable-GR-sceptical-OUT, which can only be the case if it is attacked by argument A. So in order to make sure that C is stable-GR-sceptical-IN, we need to make sure that argument A is present. In addition, argument C is attacked by the uncertain argument D. If argument D turns out to be present, then C cannot be stable-GR-sceptical-IN as there is no suitable argument to defend C from D under grounded semantics. Therefore it is relevant to make sure that argument D is absent. To conclude, the two relevant operations in this case are adding argument A and removing argument D. Note that there is still some uncertainty left in the resulting IAF: it is still unknown if the attack (C, B) should be present or absent. However, adding or removing this attack does not influence the GR-sceptical-IN-stability status of C. Therefore, these operations are not relevant.

The example shows that being certain about the existence of the attack (C, B) does not contribute anything to the stability status of C in a situation where we already know that A is present and D is absent: both with and without (C, B), C is stable-GR-sceptical-IN. In general, adding or removing an argument or attack

j-existent-stability is in the same complexity class as the problem of deciding j-stability.

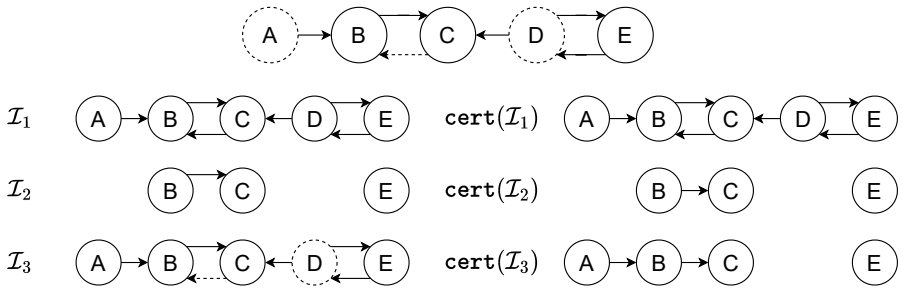


Figure 2.8: Three partial completions of our incomplete argumentation framework and their certain projections.

is only relevant if there is a situation in which this argument or attack is really necessary to obtain stability. In order to define which uncertainties are relevant to be resolved for obtaining some stability status, we therefore need some notion of "partial" completions, in which only those uncertainties are resolved that are required for stability. A partial completion of an IAF $\mathcal{I}$ is an IAF $\mathcal{I}'$ such that a (possibly empty) part of the uncertain elements of $\mathcal{I}$ is resolved in $\mathcal{I}'$, while another (possibly empty) part of the uncertain elements is still uncertain.[4] Next, we formally define such partial completions.

Definition 2.8 (Partial completion)**.** Given an IAF $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$, a partial completion is an IAF $\mathcal{I}' = \langle \mathcal{A}', \mathcal{A}^{?\prime}, \mathcal{C}', \mathcal{C}^{?\prime} \rangle$, where:

- $\mathcal{A} \subseteq \mathcal{A}' \subseteq \mathcal{A} \cup \mathcal{A}^?$;
- $\mathcal{C}|_{(\mathcal{A}' \cup \mathcal{A}^{?\prime})} \subseteq \mathcal{C}' \subseteq (\mathcal{C} \cup \mathcal{C}^?)|_{(\mathcal{A}' \cup \mathcal{A}^{?\prime})}$;
- $\mathcal{A}^{?\prime} \subseteq \mathcal{A}^?$;
- $\mathcal{C}^{?\prime} \subseteq \mathcal{C}^?$.

Note that, since $\mathcal{I}'$ is an IAF, it must still hold that $\mathcal{A}' \cap \mathcal{A}^{?\prime} = \emptyset$; $\mathcal{C}' \cap \mathcal{C}^{?\prime} = \emptyset$; $\mathcal{C}' \subseteq (\mathcal{A}' \cup \mathcal{A}^{?\prime}) \times (\mathcal{A}' \cup \mathcal{A}^{?\prime})$ and $\mathcal{C}^{?\prime} \subseteq (\mathcal{A}' \cup \mathcal{A}^{?\prime}) \times (\mathcal{A}' \cup \mathcal{A}^{?\prime})$. We denote all possible partial completions for $\mathcal{I}$ by $\texttt{part}(\mathcal{I})$.

In order to be able to apply the semantics of Dung (1995), which are defined on AFs, to IAFs, we define the certain projection of an IAF. This is an AF consisting of only the IAF's certain arguments and the attacks between them.

Definition 2.9 (Certain projection)**.** Given an IAF $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$, the certain projection $\texttt{cert}(\mathcal{I})$ is the argumentation framework $AF = \langle \mathcal{A}, \mathcal{C}|_{\mathcal{A}} \rangle$.

Note that there exists a relation between completions and partial completions via certain projections: for any IAF $\mathcal{I}$ and for each completion AF of $\mathcal{I}$, there is some partial completion $\mathcal{I}'$ in $\texttt{part}(\mathcal{I})$ such that AF is the certain projection of $\mathcal{I}'$.

[4] Partial completions in this chapter are a corrected version of specifications in Odekerken et al. (2022b).

Example 2.9 (Partial completions and certain projections)**.** Returning to the IAF $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$, given in Example 2.3 and illustrated in Figure 2.7, the following IAFs are some (but not all) examples of partial completions in $\texttt{part}(\mathcal{I})$:

- $\mathcal{I}_1 = \langle \mathcal{A} \cup \mathcal{A}^?, \emptyset, \mathcal{C} \cup \mathcal{C}^?, \emptyset \rangle$: all uncertain arguments and attacks have become certain.
- $\mathcal{I}_2 = \langle \mathcal{A}, \emptyset, \{(B, C)\}, \emptyset \rangle$: all uncertain arguments and attacks are removed, as well as all attacks that are not in the restriction of $\mathcal{C}$ to $\mathcal{A}$.
- $\mathcal{I}_3 = \langle \mathcal{A} \cup \{A\}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$: the argument A is moved from the uncertain to the certain part. The argument D and the attack (C, B) are still uncertain.

The partial completions $\mathcal{I}_1$, $\mathcal{I}_2$ and $\mathcal{I}_3$ are illustrated on the left side of Figure 2.8. The certain projection of $\mathcal{I}_1$ is $\langle \mathcal{A} \cup \mathcal{A}^?, \mathcal{C} \cup \mathcal{C}^? \rangle$. For $\mathcal{I}_2$, the certain projection is $\langle \mathcal{A}, \{(B, C)\} \rangle$. Finally, $\texttt{cert}(\mathcal{I}_3) = \langle \{A, B, C, E\}, \{(A, B), (B, C))\} \rangle$. These AFs are illustrated on the right side of Figure 2.8.

Before proceeding to a formal definition of relevance that matches the intuitions in the example above, we define the notion of minimal stable partial completions, where j refers to the justification status: $j \in \{\text{GR}, \text{CP}, \text{PR}, \text{ST}\} \times \{\text{sceptical}, \text{credulous}, \text{sceptical-existent}\} \times \{\text{IN}, \text{OUT}, \text{UNDEC}\}$.

Definition 2.10 (Minimal stable-j partial completion)**.** Given an incomplete argumentation framework $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$, a certain argument $A \in \mathcal{A}$ and a justification status j, a *minimal stable-j partial completion* for A w.r.t. $\mathcal{I}$ is a partial completion $\mathcal{I}'$ in $\texttt{part}(\mathcal{I})$ such that A is stable-j in $\mathcal{I}'$ and there is no partial completion $\mathcal{I}''$ in $\texttt{part}(\mathcal{I})$ such that A is stable-j in $\mathcal{I}''$, $\mathcal{I}'' \neq \mathcal{I}'$ and $\mathcal{I}' \in \texttt{part}(\mathcal{I}'')$.

Intuitively, the minimal stable-j partial completion for A is a partial completion in which A is stable-j, while A is not stable-j in any partial completion with more uncertain elements.

Example 2.10 (Minimal stable-j partial completion)**.** Recall the incomplete argumentation framework $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ from Example 2.3. Suppose that we are interested to know if argument C is stable-GR-sceptical-IN. $\mathcal{I}$ has one minimal stable-GR-sceptical-IN partial completion for C, which is $\mathcal{I}_4 = \langle \{A, B, C, E\}, \emptyset, \{(A, B), (B, C)\}, \mathcal{C}^? \rangle$. Given that $\mathcal{C}^?$ contains the attack (C, B) as its only uncertain element, $\mathcal{I}_4$ has three partial completions, resulting in two unique certain projections. These are $\langle \{A, B, C, E\}, \{(A, B), (B, C), (C, B)\} \rangle$ (depicted as AF_2 in Figure 2.6) and $\langle \{A, B, C, E\}, \{(A, B), (B, C)\} \rangle$ (depicted as AF_4 in the same figure). Since AF_2 and AF_4 each have a grounded extension – in both cases $\{A, C, E\}$ – that contains C, C is stable-GR-sceptical-IN in $\mathcal{I}_4$. In addition, note that $\mathcal{I}_4$ is minimal in that C would not be stable-GR-sceptical-IN in partial completions with more uncertain elements:

- Suppose that the presence of argument A is unknown as in $\mathcal{I}_5 = \langle \{B, C, E\}, \{A\}, \{(A, B), (B, C)\}, \{(C, B)\} \rangle$. Then AF_6 and AF_8 are certain projections of partial completions of $\mathcal{I}_5$. Since C is not in the grounded extension of each of these AFs, C is not stable-GR-sceptical-IN w.r.t. $\mathcal{I}_5$.

Figure 2.9: Partial completion $\mathcal{I}_4$, which is the only minimal stable-GR-sceptical-IN partial completion for C w.r.t. $\mathcal{I}$ where $\mathcal{I} = \langle\{B, C, E\}, \{A, E\}, \mathcal{C}, \mathcal{C}^?\rangle$ and $\mathcal{I}_4 = \langle\{A, B, C, E\}, \emptyset, \{(A, B), (B, C)\}, \mathcal{C}^?\rangle$.

- Alternatively, suppose that the absence of argument D is yet unknown, as in $\mathcal{I}_3$, defined earlier as $\langle\{A, B, C, E\}, \{D\}, \mathcal{C}, \{(C, B)\}\rangle$. Then AF_1 and AF_3 are also certain projections of partial completions of $\mathcal{I}_3$, where these AFs do have C in their grounded extensions. Therefore, C is not stable-GR-sceptical-IN w.r.t. $\mathcal{I}_3$.

This shows that $\mathcal{I}_4$ is a minimal stable-GR-sceptical-IN partial completion for C w.r.t. $\mathcal{I}$. Note that, although in this case there is a *single* minimal stable-GR-sceptical-IN partial completion for C w.r.t. $\mathcal{I}$, in general there can be multiple minimal stable partial completions.

For example, there are three minimal stable-CP-credulous-UNDEC partial completions for C w.r.t. $\mathcal{I}$:

1. $\langle\mathcal{A} \cup \{A, D\}, \emptyset, \mathcal{C}, \{(C, B)\}\rangle$ (having AF_1 and AF_3 as certain projections of partial completions);
2. $\langle\mathcal{A} \cup \{D\}, \{A\}, \mathcal{C} \cup \{(C, B)\}, \emptyset\rangle$ (for AF_1 and AF_5); and
3. $\langle\mathcal{A}, \{D\}, \mathcal{C} \setminus \{(A, B)\} \cup \{(C, B)\}, \emptyset\rangle$ (for AF_5 and AF_6).

Using the notion of minimal stable-j partial completions, we can now define j-RELEVANCE.

Definition 2.11 (j-RELEVANCE)**.** Given an IAF $\mathcal{I} = \langle\mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^?\rangle$, an argument $A \in \mathcal{A}$, an uncertain argument or attack $U \in \mathcal{A}^? \cup \mathcal{C}^?$ and a justification status j,

- *Addition of* U is *j-relevant* for A w.r.t. $\mathcal{I}$ iff there is a minimal stable-j partial completion $\mathcal{I}' = \langle\mathcal{A}', \mathcal{A}^{?\prime}, \mathcal{C}', \mathcal{C}^{?\prime}\rangle$ for A w.r.t. $\mathcal{I}$ such that $U \in \mathcal{A}' \cup \mathcal{C}'$; and
- *Removal of* U is *j-relevant* for A w.r.t. $\mathcal{I}$ iff there is a minimal stable-j partial completion $\mathcal{I}' = \langle\mathcal{A}', \mathcal{A}^{?\prime}, \mathcal{C}', \mathcal{C}^{?\prime}\rangle$ for A w.r.t. $\mathcal{I}$ such that $U \notin \mathcal{A}' \cup \mathcal{A}^{?\prime} \cup \mathcal{C}' \cup \mathcal{C}^{?\prime}$.

In other words, addition of an uncertain element U is j-relevant if a minimal stable-j partial completion can be reached by moving U from the uncertain to the certain part of the IAF $\mathcal{I}$; and removal of U is j-relevant if completely removing U from $\mathcal{I}$, possibly in combination with other actions, leads to a minimal stable-j partial completion.

Example 2.11 (j-RELEVANCE)**.** To illustrate j-relevance, we build on the minimal stable-j partial completions from Example 2.10. Recall that $\mathcal{I}_4$, illustrated in Figure 2.9, is the only minimal stable-GR-sceptical-IN partial completion for C w.r.t. $\mathcal{I}$ where $\mathcal{I} = \langle\{B, C, E\}, \{A, D\}, \mathcal{C}, \mathcal{C}^?\rangle$ and $\mathcal{I}_4 = \langle\{A, B, C, E\}, \emptyset, \{(A, B), (B, C)\}, \mathcal{C}^?\rangle$. Given that A was an uncertain argument in $\mathcal{I}$ and is a certain argument in (the

minimal stable-GR-sceptical-IN partial completion) $\mathcal{I}_4$, addition of A is GR-sceptical-IN-relevant for C w.r.t. $\mathcal{I}$. Furthermore, as D was an uncertain argument in $\mathcal{I}$ and is no longer present in $\mathcal{I}_4$, removal of D is GR-sceptical-IN-relevant for C w.r.t. $\mathcal{I}$.

Considering the justification status CP-credulous-UNDEC, there are three minimal stable-CP-credulous-UNDEC partial completions for C (see Example 2.10). The CP-credulous-UNDEC-relevant operations are addition of A, addition of D, addition of (C, B) and removal of A.

Note that this example shows the possibility that both the addition and removal of some uncertain argument or attack are relevant. This example also demonstrates that performing a relevant action does not necessarily lead to a stable situation, but may be just the first step in becoming stable. For instance, the addition of A to $\mathcal{I}$ does not yet result in an IAF that is stable-CP-credulous-UNDEC as this IAF still has at least one completion (to be precise, the completions AF_2 and AF_4 in Figure 2.6) in which C is not CP-credulous-UNDEC. In order to become stable, an additional relevant action (in this case, the addition of D) is required.

2.4. Complexity results

As we have seen in Section 2.2, the extension from AFs to IAFs for modeling incomplete information basically represents a space of multiple possible situations. Consequently, the number of completions grows exponentially with the amount of uncertain arguments and attacks. Given that the notions of stability and relevance were defined in terms of these completions (Section 2.3), a naive approach for deciding on stability or relevance by enumerating all completions has a runtime that is exponential compared to the runtime of justification algorithms. In order to gain more insight in the possibility of more efficient algorithms, we perform a complexity analysis for the problems of determining the justification and stability status, as well as identifying relevant updates. An overview of all complexity results presented in this section is given in Table 2.1.

2.4.1. Justification

In this section, we study the complexity of determining the justification status for a given argument in an incomplete argumentation framework.

For any justification status j where $j \in \{\text{GR}, \text{CP}, \text{PR}, \text{ST}\} \times \{\text{sceptical}, \text{credulous}, \text{sceptical-existent}\} \times \{\text{IN}, \text{OUT}, \text{UNDEC}\}$, we formulate the identification of justification status as a decision problem j-JUSTIFICATION.

j-JUSTIFICATION	
Given:	An argumentation framework $\langle \mathcal{A}, \mathcal{C} \rangle$, a justification status j and an argument $A \in \mathcal{A}$
Question:	Does A's justification status in $\langle \mathcal{A}, \mathcal{C} \rangle$ equal j?

Whereas the complexity of IN-JUSTIFICATION has been studied before (as acceptance problems, see Dimopoulos and Torres (1996), Coste-Marquis et al. (2005), Dung (1995), and Dunne and Bench-Capon (2002)), this is not the case for OUT-

Semantics	Acceptance strategy	Label	JUSTIFICATION	STABILITY	RELEVANCE
ST	credulous	IN/OUT	NP-c [1,2]	Π_2^p-c [5]	$\mathbf{\Sigma_2^p}$**-c** P2.17
ST	credulous	UNDEC	**Trivial (no)** P2.4	**Trivial (no)** P2.10	**Trivial (no)** P2.18
ST	sceptical	IN/OUT	CoNP-c [1]	CoNP-c [5]	$\mathbf{\Sigma_2^p}$**-c** P2.17
ST	sceptical	UNDEC	**CoNP-c** P2.5	**CoNP-c** P2.9	$\mathbf{\Sigma_2^p}$**-c** P2.19
ST	sceptical-existent	IN/OUT	DP-c [4]	Π_2^p-c [5]	$\mathbf{\Sigma_2^p}$**-c** P2.20
ST	sceptical-existent	UNDEC	**Trivial (no)** P2.4	**Trivial (no)** P2.10	**Trivial (no)** P2.18
CP	credulous	IN/OUT	NP-c [1,2]	Π_2^p-c [5]	$\mathbf{\Sigma_2^p}$**-c** P2.13
CP	credulous	UNDEC	**P-c** P2.1	**CoNP-c** P2.6	**NP-c** P2.12
CP	sceptical	IN/OUT	P-c [0]	CoNP-c [5]	**NP-c** P2.12
CP	sceptical	UNDEC	**CoNP-c** P2.2	**CoNP-c** P2.7	$\mathbf{\Sigma_2^p}$**-c** P2.15
GR	credulous	IN/OUT	P-c [0]	CoNP-c [5]	**NP-c** P2.12
GR	credulous	UNDEC	**P-c** P2.1	**CoNP-c** P2.6	**NP-c** P2.12
GR	sceptical	IN/OUT	P-c [0]	CoNP-c [5]	**NP-c** P2.12
GR	sceptical	UNDEC	**P-c** P2.1	**CoNP-c** P2.6	**NP-c** P2.12
PR	credulous	IN/OUT	NP-c [1,2]	Π_2^p-c [5]	$\mathbf{\Sigma_2^p}$**-c** P2.13
PR	credulous	UNDEC	$\mathbf{\Sigma_2^p}$**-c** P2.3	$\mathbf{\Pi_3^p}$**-c** P2.8	$\mathbf{\Sigma_3^p}$**-c** P2.14
PR	sceptical	IN/OUT	Π_2^p-c [3]	Π_2^p-c [5]	$\mathbf{\Sigma_3^p}$**-c** P2.16
PR	sceptical	UNDEC	**CoNP-c** P2.2	**CoNP-c** P2.7	$\mathbf{\Sigma_2^p}$**-c** P2.15

Table 2.1: Overview of all complexity results discussed in this chapter. We considered four different semantics: stable (ST), complete (CP), grounded (GR) or preferred (PR) (see Section 2.2.1). If a reference is specified, this complexity result is trivial from an earlier result in the literature, where [0] refers to Dung (1995); [1] refers to Dimopoulos and Torres (1996); [2] refers to Coste-Marquis et al. (2005); [3] to Dunne and Bench-Capon (2002); [4] to Dunne and Wooldridge (2009) and [5] to Baumeister et al. (2021). New results are printed bold; we refer to the corresponding proposition by "P" and the proposition number.

and UNDEC-JUSTIFICATION. This is unfortunate, as detecting OUT- and UNDEC-JUSTIFICATION has interesting applications as well: since arguments that are UNDEC are "more acceptable" than arguments that are OUT, these justification statuses provide a more fine-grained notion of acceptability than the distinction between accepted (IN) and not accepted (not IN). In addition, more insight in the complexity of detecting these justification statuses could, for example, be helpful in developing fast algorithms. Furthermore, we will use the complexities of OUT- and UNDEC-JUSTIFICATION later for finding the complexities of OUT- and UNDEC-STABILITY (Section 2.4.2) and OUT- and UNDEC-RELEVANCE (Section 2.4.3). We will therefore provide complexity proofs for these types of JUSTIFICATION under GR, CP, ST and PR semantics. Our strategy in proving these complexities is to relate them to the complexities of IN-JUSTIFICATION and using existing results from the complexity of acceptance problems. Short proofs will be given directly after the lemmas and propositions, whereas we provide sketches for longer proofs; the full proofs can be found in Appendix A.1. We start with a lemma for JUSTIFICATION that shows that IN-JUSTIFICATION and OUT-JUSTIFICATION are in the same complexity class:

Lemma 2.1 (Justification status IN and OUT)**.** *For any given $\sigma \in \{GR, CP, PR, ST\}$ and $c \in \{sceptical, credulous\}$, the complexity of σ-c-OUT-JUSTIFICATION equals the complexity of σ-c-IN-JUSTIFICATION.*

Proof. Let $AF = \langle \mathcal{A}, \mathcal{C} \rangle$ be an AF and $A \in \mathcal{A}$ an argument. Now construct $\mathcal{A}'$ as $\mathcal{A} \cup \{B\}$ (where $B \notin \mathcal{A}$) and $\mathcal{C}' = \mathcal{C} \cup \{(A, B)\}$; let $AF' = \langle \mathcal{A}', \mathcal{C}' \rangle$. Then A is σ-c-OUT in AF iff B is σ-c-IN in AF'; in addition, A is σ-c-IN in AF iff B is σ-c-OUT in AF'. □

Contrary to OUT-JUSTIFICATION, UNDEC-JUSTIFICATION is not necessarily in the same complexity class as IN-JUSTIFICATION. However, under GR, CP and PR semantics, there is another relation, as we will show in Lemma 2.3.[5] Before we can do so, we prove relations between credulous-IN- and sceptical-UNDEC-JUSTIFICATION, as well as between credulous-UNDEC- and sceptical-IN-JUSTIFICATION, in the following lemma. The transformations used in this lemma are illustrated in Figure 2.10.

Lemma 2.2 (Complementary relation IN- and UNDEC-JUSTIFICATION)**.** *For any given $\sigma \in \{GR, CP, PR\}$, for each argumentation theory $AF = \langle \mathcal{A}, \mathcal{C} \rangle$ and argument $A \in \mathcal{A}$, each of the following holds:*

1. *A is σ-credulous-IN in AF iff A' is **not** σ-sceptical-UNDEC in $\langle \mathcal{A} \cup \{A'\}, \mathcal{C} \cup \{(A, A'), (A', A')\} \rangle$;*
2. *A is σ-sceptical-IN in AF iff A' is **not** σ-credulous-UNDEC in $\langle \mathcal{A} \cup \{A'\}, \mathcal{C} \cup \{(A, A'), (A', A')\} \rangle$;*
3. *A is σ-credulous-UNDEC in AF iff A' is **not** σ-sceptical-IN in $\langle \mathcal{A} \cup \{A', B, C\}, \mathcal{C} \cup \{(A, B), (A, C), (B, C), (C, A')\} \rangle$; and*

[5] This relation does not exist for ST semantics as arguments cannot be ST-credulous-UNDEC. We will prove the complexity of ST-credulous-, ST-sceptical-existent- and ST-sceptical-JUSTIFICATION later in this section.

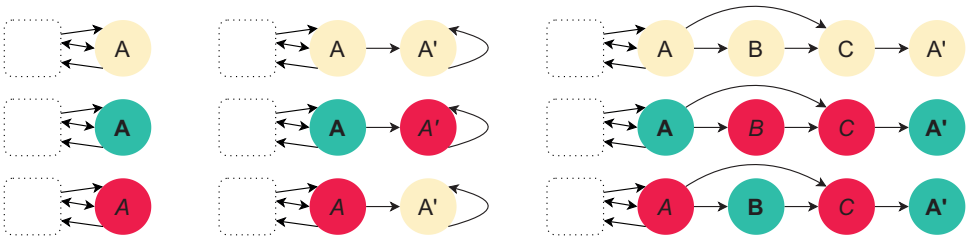


Figure 2.10: Illustration of the argumentation frameworks that are used in transformations between justification problem instances used for proving Lemma 2.2. The figure illustrates three argumentation frameworks: $AF = \langle \mathcal{A}, \mathcal{C} \rangle$ (first column), $AF' = \langle \mathcal{A} \cup \{A'\}, \mathcal{C} \cup \{(A, A'), (A', A')\} \rangle$ (second column) and $AF'' = \langle \mathcal{A} \cup \{A', B, C\}, \mathcal{C} \cup \{(A, B), (A, C), (B, C), (C, A')\} \rangle$ (third column). The dotted, rounded rectangles represent all arguments in $\mathcal{A}$ except A. Each of the argumentation frameworks is displayed three times, corresponding to different extensions of AF: the first row, where A is coloured yellow, represents extensions that do not contain A or any attacker of A ("A is UNDEC"). The second row, where A is green and with boldface font, represents extensions containing A ("A is IN"). Finally, the third row, where A is red and with italic font, represents extensions containing some attacker of A ("A is OUT"). The colours and typesetting refer to the justification statuses of arguments, where green and boldface font stands for IN; yellow and regular font for UNDEC and red and italic font for OUT.

4. *A is σ-sceptical-UNDEC in AF iff A' is* ***not*** *σ-credulous-IN in $\langle \mathcal{A} \cup \{A', B, C\}, \mathcal{C} \cup \{(A, B), (A, C), (B, C), (C, A')\} \rangle$.*

Proof sketch. We give the proof for the first item here. For full proofs of all items, we refer to Appendix A.1. Consider an arbitrary semantics $\sigma \in \{\text{GR}, \text{CP}, \text{PR}\}$, argumentation theory $AF = \langle \mathcal{A}, \mathcal{C} \rangle$ and argument $A \in \mathcal{A}$. Construct $AF' = \langle \mathcal{A} \cup \{A'\}, \mathcal{C} \cup \{(A, A'), (A', A')\} \rangle$; for an illustration, see the first and second columns of Figure 2.10.

$\Rightarrow$ Suppose that A is σ-credulous-IN in AF: then there is some σ-extension S of AF containing A. Note that S also must be a σ-extension of AF': all arguments in $\mathcal{A}$ attacking attackers of A are still in $\mathcal{A} \cup \{A'\}$ and $S \cup \{A'\}$ is not a σ-extension as it is not conflict-free. Then there exists some σ-extension (i.e. S) of AF' in which A' is attacked by S, so A' is not σ-sceptical-UNDEC in AF'.

$\Leftarrow$ Suppose that A' is not σ-sceptical-UNDEC in AF'; then there exists some σ-extension S of AF' such that either $A' \in S$ or some argument attacking A' is in S. Given that A' is self-attacking, $A' \notin S$, so A' is attacked by some argument in S, which can only be A. Furthermore note that S is also a σ-extension of AF, since the arguments that are defended by S in AF are exactly the same as the arguments that are defended by S in AF'. To conclude, there exists some σ-extension (i.e. S) of AF in which $A \in S$, so A is σ-credulous-IN in AF. □

Lemma 2.2 is used in the following lemma to show that credulous-IN-JUSTIFICATION and sceptical-UNDEC-JUSTIFICATION are in complementary complexity classes, as

well as credulous-UNDEC-JUSTIFICATION and sceptical-IN-JUSTIFICATION, under GR, CP and PR semantics.

Lemma 2.3 (Complexities UNDEC-JUSTIFICATION)**.** *For any given semantics σ in $\{GR, CP, PR\}$:*

1. *If the complexity of σ-credulous-IN-JUSTIFICATION is $\boldsymbol{C}$, then the complexity of σ-sceptical-UNDEC-JUSTIFICATION is co-$\boldsymbol{C}$; and*
2. *If the complexity of σ-sceptical-IN-JUSTIFICATION is $\boldsymbol{C}$, then the complexity of σ-credulous-UNDEC-JUSTIFICATION is co-$\boldsymbol{C}$.*

Proof sketch. We give a proof sketch for the first item here and refer to the full proofs for both items to Appendix A.1. The first item can be proved by two reductions:

- Each instance $I_1 = (\langle \mathcal{A}, \mathcal{C} \rangle, A)$ of σ-credulous-IN-JUSTIFICATION can, in polynomial time, be converted to an instance $I_2 = (\langle \mathcal{A} \cup \{A'\}, \mathcal{C} \cup \{(A, A'), (A', A')\} \rangle, A')$ of σ-sceptical-UNDEC-JUSTIFICATION where I_1 is a positive instance iff I_2 is a negative instance by item 1 of Lemma 2.2.
- Similarly, each instance $I_1 = (\langle \mathcal{A}, \mathcal{C} \rangle, A)$ of σ-sceptical-UNDEC-JUSTIFICATION can, in polynomial time, be converted to an instance $I_2 = (\langle \mathcal{A} \cup \{A', B, C\}, \mathcal{C} \cup \{(A, B), (A, C), (B, C), (C, A')\} \rangle, A')$ of σ-credulous-IN-JUSTIFICATION where, by Lemma 2.2 item 4, I_1 is a positive instance iff I_2 is a negative instance. □

Using Lemma 2.3, we can now directly derive justification statuses for UNDEC-JUSTIFICATION under CP, GR and PR semantics in Propositions 2.1, 2.2 and 2.3.

Proposition 2.1. *The decision problems CP-credulous-UNDEC-JUSTIFICATION, GR-credulous-UNDEC-JUSTIFICATION and GR-sceptical-UNDEC-JUSTIFICATION are all P-complete.*

Proof. This follows directly from Lemma 2.3 in combination with P-completeness of CP-sceptical-IN-JUSTIFICATION, GR-sceptical-IN-JUSTIFICATION and GR-credulous-IN-JUSTIFICATION (Dung, 1995) and the fact that P = CoP. □

Proposition 2.2. *The decision problems CP-sceptical-UNDEC-JUSTIFICATION and PR-sceptical-UNDEC-JUSTIFICATION are CoNP-complete.*

Proof. This follows directly from Lemma 2.3 and the fact that CP-credulous-IN-JUSTIFICATION and PR-credulous-IN-JUSTIFICATION are NP-complete (Dimopoulos and Torres, 1996; Coste-Marquis et al., 2005). □

Proposition 2.3. *PR-credulous-UNDEC-JUSTIFICATION is Σ_2^p-complete.*

Proof. This follows directly from Lemma 2.3 and the fact that PR-sceptical-IN-JUSTIFICATION is Π_2^p-complete (Dunne and Bench-Capon, 2002). □

For ST semantics, the situation is different, since for each ST extension S, each argument is either in S or attacked by some argument in S. In the remainder of this section, we give the complexity results of UNDEC-JUSTIFICATION for ST semantics.

Proposition 2.4. *The decision problems ST-credulous-UNDEC-JUSTIFICATION and ST-sceptical-existent-UNDEC-JUSTIFICATION are trivial.*

Proof. In each ST extension S, each argument is either in S or attacked by some argument in S. Consequently, each instance of ST-credulous-UNDEC-JUSTIFICATION or ST-sceptical-existent-UNDEC-JUSTIFICATION is False. □

In addition, an argument can only be ST-sceptical-UNDEC in a given argumentation framework AF if AF does not have any ST extension.

Proposition 2.5. *ST-sceptical-UNDEC-JUSTIFICATION is CoNP-complete.*

Proof. For each AF $\langle \mathcal{A}, \mathcal{C} \rangle$ and argument $A \in \mathcal{A}$, A is ST-sceptical-UNDEC in $\langle \mathcal{A}, \mathcal{C} \rangle$ iff no stable extension exists for $\langle \mathcal{A}, \mathcal{C} \rangle$. The problem of deciding if a given AF has a stable extension is NP-complete (Dimopoulos and Torres, 1996), so the complementary problem of deciding if an AF has no stable extension is CoNP-complete. □

At this point, we have studied the complexity of the JUSTIFICATION problem for GR, CP, PR and ST semantics, for sceptical and credulous (and sceptical-existent of ST semantics) acceptance and labels IN, OUT and UNDEC. These results are summarised in Table 2.2. In the following sections, we will see that these definitions and complexity results can be used for defining and studying the complexity of the stability and relevance problems.

2.4.2. Stability

This section presents complexity results for the task of identifying a stability status. The following formulates this task as a decision problem.

j-STABILITY	
Given:	An incomplete argumentation framework $\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$, a justification status j and an argument $A \in \mathcal{A}$
Question:	Does A's stability status w.r.t. $\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ equal stable-j?

Next, we give complexity results for variants of the STABILITY problem. We start with relating IN-STABILITY to necessary (sceptical and credulous) acceptance, as defined in Definition 2.5 (which was adapted from Baumeister et al. (2021)).

Lemma 2.4. *For any given $\sigma \in \{GR, CP, PR, ST\}$ and $c \in \{sceptical, credulous\}$, the complexity of σ-c-IN-STABILITY equals the complexity of necessary c acceptance w.r.t. semantics σ.*

Proof. This is trivial from Definition 2.7 of stability and Definition 2.5 of necessary acceptance. □

Justification status j	Complexity j-JUSTIFICATION
ST-credulous-IN/OUT	NP-c [1,2]
ST-credulous-UNDEC	**Trivial (no)** P2.4
ST-sceptical-IN/OUT	CoNP-c [1]
ST-sceptical-UNDEC	**CoNP-c** P2.5
ST-sceptical-existent-IN/OUT	DP-c [4]
ST-sceptical-existent-UNDEC	**Trivial (no)** P2.4
CP-credulous-IN/OUT	NP-c [1,2]
CP-credulous-UNDEC	**P-c** P2.1
CP-sceptical-IN/OUT	P-c [0]
CP-sceptical-UNDEC	**CoNP-c** P2.2
GR-credulous-IN/OUT	P-c [0]
GR-credulous-UNDEC	**P-c** P2.1
GR-sceptical-IN/OUT	P-c [0]
GR-sceptical-UNDEC	**P-c** P2.1
PR-credulous-IN/OUT	NP-c [1,2]
PR-credulous-UNDEC	$\mathbf{\Sigma_2^p}$**-c** P2.3
PR-sceptical-IN/OUT	Π_2^p-c [3]
PR-sceptical-UNDEC	**CoNP-c** P2.2

Table 2.2: Overview of all complexity results related to justification. If a reference is specified, this complexity result is trivial from an earlier result in the literature, where [0] refers to Dung (1995); [1] refers to Dimopoulos and Torres (1996); [2] refers to Coste-Marquis et al. (2005); [3] to Dunne and Bench-Capon (2002) and [4] to Dunne and Wooldridge (2009). New results are printed bold; we refer to the corresponding proposition by "P" and the proposition number. Full proofs for each of the propositions are presented in the appendix. The complexities for IN- and OUT-JUSTIFICATION are the same; this follows from Lemma 2.1.

We continue with relating IN- and OUT-STABILITY. Similar to what we did for IN- and OUT-JUSTIFICATION in Lemma 2.1, we show that the complexity of IN-STABILITY is the same as the complexity of OUT-STABILITY.

Lemma 2.5. *For any given $\sigma \in \{GR, CP, PR, ST\}$ and $c \in \{sceptical, credulous\}$, the complexity of σ-c-OUT-STABILITY equals the complexity of σ-c-IN-STABILITY.*

Proof sketch. We prove this by a reduction from the σ-c-OUT-STABILITY problem to σ-c-IN-STABILITY and by a reduction in the other direction. Below, we give proof sketch; the full proof can be found in Appendix A.2.

- For each instance $(\mathcal{I}, A)$ of σ-c-OUT-STABILITY where $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ and $A \in \mathcal{A}$, one can construct $\mathcal{I}^* = \langle \mathcal{A} \cup \{B\}, \mathcal{A}^?, \mathcal{C} \cup \{(A, B)\}, \mathcal{C}^? \rangle$, where $B \notin \mathcal{A} \cup \mathcal{A}^?$. Then $(\mathcal{I}, A)$ is a positive instance of σ-c-OUT-STABILITY iff $(\mathcal{I}^*, B)$ is a positive instance of σ-c-IN-STABILITY.
- Similarly, each instance $(\mathcal{I}, A)$ of the σ-c-IN-STABILITY problem, where $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ and $A \in \mathcal{A}$, can be transformed into $(\mathcal{I}^*, B)$ where $\mathcal{I}^* = \langle \mathcal{A} \cup \{B\}, \mathcal{A}^?, \mathcal{C} \cup \{(A, B)\}, \mathcal{C}^? \rangle$ with $B \notin \mathcal{A} \cup \mathcal{A}^?$. The instance $(\mathcal{I}, A)$ is positive for σ-c-IN-STABILITY iff $(\mathcal{I}^*, B)$ is a positive instance for σ-c-OUT-STABILITY.

From these two reductions, it follows that σ-c-IN-STABILITY and σ-c-OUT-STABILITY have the same complexity. □

Using Lemma 2.5, it can be shown that for any given $\sigma \in \{\text{GR}, \text{CP}, \text{PR}, \text{ST}\}$ and $c \in \{\text{sceptical}, \text{credulous}\}$, the complexity of σ-c-OUT-STABILITY can be derived directly from the complexity of σ-c-IN-STABILITY. In the following lemma, we relate UNDEC-STABILITY for specific semantics with possible sceptical and credulous acceptance (Definition 2.5). Similar to Lemma 2.2, we prove this for GR, CP and PR semantics but not for ST semantics: the relation does not hold for ST semantics because arguments cannot be ST-credulous-UNDEC.

Lemma 2.6 (Complexities UNDEC-STABILITY)**.** *For any given $\sigma \in \{GR, CP, PR\}$:*

1. *If possible credulous acceptance w.r.t. σ semantics is in the complexity class $\boldsymbol{C}$, then σ-sceptical-UNDEC-STABILITY is in the complexity class co-$\boldsymbol{C}$; and*
2. *If possible sceptical acceptance w.r.t. σ semantics is in the complexity class $\boldsymbol{C}$, then σ-credulous-UNDEC-STABILITY is in the complexity class co-$\boldsymbol{C}$.*

Proof sketch. We only give a proof sketch for the first item here; full proofs of both items are in Appendix A.2. This proof consists of two reductions: we show that possible credulous acceptance w.r.t. σ semantics reduces to the complementary problem of σ-sceptical-UNDEC-STABILITY and the other way around. The proof strategy is similar to the proof of Lemma 2.2, illustrated in Figure 2.10.

- Let $I_1 = (\mathcal{I}, A)$ where $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ and $A \in \mathcal{A}$. Construct $I_2 = (\mathcal{I}^*, A')$ where $\mathcal{I}^* = \langle \mathcal{A} \cup \{A'\}, \mathcal{A}^?, \mathcal{C} \cup \{(A, A'), (A', A')\}, \mathcal{C}^? \rangle$ and $A' \notin \mathcal{A} \cup \mathcal{A}^?$. Then I_1 is a positive instance of possible credulous acceptance w.r.t. σ semantics iff I_2 is a negative instance of σ-sceptical-UNDEC-STABILITY.

- Let $I_1 = (\mathcal{I}, A)$ where $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ and $A \in \mathcal{A}$. Now let $I_2 = (\mathcal{I}^*, A')$ where $\mathcal{I}^* = \langle \mathcal{A} \cup \{A', B, C\}, \mathcal{A}^?, \mathcal{C} \cup \{(A, B), (A, C), (B, C), (C, A')\}, \mathcal{C}^? \rangle$ and none of A', B and C is in $\mathcal{A} \cup \mathcal{A}^?$. I_1 is a positive instance of σ-sceptical-UNDEC-STABILITY iff I_2 is a negative instance of possible credulous acceptance w.r.t. σ semantics. □

The results of Lemma 2.6 can be used to derive the complexity classes of UNDEC-STABILITY based on existing results for possible acceptance from Baumeister et al. (2021) for CP, GR and PR semantics.

Proposition 2.6. *GR-sceptical-UNDEC-STABILITY, GR-credulous-UNDEC-STABILITY and CP-credulous-UNDEC-STABILITY are CoNP-complete.*

Proof. This follows directly from Lemma 2.6 and the fact that possible sceptical acceptance under GR and CP semantics and possible credulous acceptance under GR semantics are NP-complete (Baumeister et al., 2021). □

Proposition 2.7. *The problems CP-sceptical-UNDEC-STABILITY and PR-sceptical-UNDEC-STABILITY are CoNP-complete.*

Proof. This follows directly from Lemma 2.6 and the fact that possible credulous acceptance under CP and PR semantics are NP-complete (Baumeister et al., 2021). □

Proposition 2.8. *PR-credulous-UNDEC-STABILITY is Π_3^p-complete.*

Proof. This follows directly from Lemma 2.6 and the fact that possible sceptical acceptance under PR semantics is Σ_3^p-complete (Baumeister et al., 2021). □

Finally, we turn to ST semantics. Our strategy for proving these complexities is similar to our approach for the ST-UNDEC-JUSTIFICATION complexity proofs in the previous section.

Proposition 2.9. *ST-sceptical-UNDEC-STABILITY is CoNP-complete.*

Proof. The problem is in CoNP, as a negative instance $(\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle, A)$ can be verified in polynomial time given a certificate (AF', S) such that AF' is a completion of $\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$, $A \in \mathcal{A}$ and S is a ST extension of AF'. If S is a ST extension then each argument in $\mathcal{A}$, including A, is either in S or attacked by S; therefore A cannot be stable-ST-sceptical-UNDEC w.r.t. $\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$.

For hardness, we can reduce from the CoNP-complete problem ST-sceptical-UNDEC-JUSTIFICATION: any instance $I_1 = (\langle \mathcal{A}, \mathcal{C} \rangle, A)$ of ST-sceptical-UNDEC-JUSTIFICATION can be solved by solving ST-sceptical-UNDEC-STABILITY for $I_2 = (\langle \mathcal{A}, \emptyset, \mathcal{C}, \emptyset \rangle, A)$. Given that $\langle \mathcal{A}, \mathcal{C} \rangle$ is the only completion of $\langle \mathcal{A}, \emptyset, \mathcal{C}, \emptyset \rangle$, I_1 is positive iff I_2 is positive. □

Proposition 2.10. *The problems* ST*-credulous-*UNDEC*-*STABILITY *and* ST*-sceptical-existent-*UNDEC*-*STABILITY *are trivial.*

Proof. For each argumentation framework $AF = \langle \mathcal{A}, \mathcal{C} \rangle$ such that a ST extension S exists, each argument in $\mathcal{A}$ is either in S or attacked by an argument in S. Therefore, A cannot be ST-credulous-UNDEC or ST-sceptical-existent-UNDEC in AF. This applies for each AF, including all completions of each possible IAF, so each instance of ST-credulous-UNDEC-STABILITY and ST-sceptical-existent-UNDEC-STABILITY must be negative. □

To conclude this section, we have studied the complexity of the STABILITY problem for GR, CP, PR and ST semantics, for sceptical and credulous (and sceptical-existent for ST semantics) acceptance and labels IN, OUT and UNDEC. For an overview of the results, we refer to Table 2.3. In the following section, we consider the problem of identifying relevance.

Justification status j	Complexity j-STABILITY
ST-credulous-IN/OUT	Π_2^p-c [5]
ST-credulous-UNDEC	**Trivial (no)** P2.10
ST-sceptical-IN/OUT	CoNP-c [5]
ST-sceptical-UNDEC	**CoNP-c** P2.9
ST-sceptical-existent-IN/OUT	Π_2^p-c [5]
ST-sceptical-existent-UNDEC	**Trivial (no)** P2.10
CP-credulous-IN/OUT	Π_2^p-c [5]
CP-credulous-UNDEC	**CoNP-c** P2.6
CP-sceptical-IN/OUT	CoNP-c [5]
CP-sceptical-UNDEC	**CoNP-c** P2.7
GR-credulous-IN/OUT	CoNP-c [5]
GR-credulous-UNDEC	**CoNP-c** P2.6
GR-sceptical-IN/OUT	CoNP-c [5]
GR-sceptical-UNDEC	**CoNP-c** P2.6
PR-credulous-IN/OUT	Π_2^p-c [5]
PR-credulous-UNDEC	$\mathbf{\Pi_3^p}$**-c** P2.8
PR-sceptical-IN/OUT	Π_2^p-c [5]
PR-sceptical-UNDEC	**CoNP-c** P2.7

Table 2.3: Overview of all complexity results related to stability. If a reference is specified, this complexity result is trivial from an earlier result in the literature, where [5] refers to Baumeister et al. (2021). New results are printed bold; we refer to the corresponding proposition by "P" and the proposition number. Full proofs for each of the propositions are presented in Appendix A.2. The complexities for IN- and OUT-STABILITY are the same; this follows from Lemma 2.5.

2.4.3. Relevance

In this section, we study the complexity of relevance for complete, grounded, preferred and stable semantics. Like for justification and stability status, we formulate the identification of j-relevant actions as a decision problem:

j-RELEVANCE of action **a**	
Given:	An incomplete argumentation framework $\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$, a justification status j, an action $\mathbf{a} \in \{addition, removal\}$, an argument $A \in \mathcal{A}$ and an uncertain argument or attack $U \in \mathcal{A}^? \cup \mathcal{C}^?$.
Question:	Is **a** of U j-relevant for A w.r.t. $\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$?

In order to prove the complexity of j-RELEVANCE for each justification status j, we first give proofs for the upper bounds, that is: the membership in a given complexity class. Subsequently, we provide lower bounds, that is: hardness results, for RELEVANCE under grounded, complete and preferred semantics. Finally, we prove the remaining hardness results for stable semantics.

Upper bounds

First, we will prove a general upper bound on j-RELEVANCE. In order to do so, we first prove Lemma 2.7. This lemma shows that the relevance of adding or removing an uncertain argument or attack can be validated by checking the justification status of the certain projections of two particular future partial completions.

Lemma 2.7. *Given an IAF $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$, a certain argument $A \in \mathcal{A}$ and a justification status j:*

1. *For each $U \in \mathcal{A}^?$, addition of U is j-relevant for A w.r.t. $\mathcal{I}$ iff there exists some $\mathcal{I}' = \langle \mathcal{A}', \{U\}, \mathcal{C}', \emptyset \rangle \in \mathtt{part}(\mathcal{I})$ such that A is **not** j in the certain projection of $\mathcal{I}'$, while A is j in the certain projection of $\langle \mathcal{A}' \cup \{U\}, \emptyset, \mathcal{C}', \emptyset \rangle$.*
2. *For each $U \in \mathcal{C}^?$, addition of U is j-relevant for A w.r.t. $\mathcal{I}$ iff there exists some $\mathcal{I}' = \langle \mathcal{A}', \emptyset, \mathcal{C}', \{U\} \rangle \in \mathtt{part}(\mathcal{I})$ such that A is **not** j in the certain projection of $\mathcal{I}'$, while A is j in the certain projection of $\langle \mathcal{A}', \emptyset, \mathcal{C}' \cup \{U\}, \emptyset \rangle$.*
3. *For each $U \in \mathcal{A}^?$, removal of U is j-relevant for A w.r.t. $\mathcal{I}$ iff there exists some $\mathcal{I}' = \langle \mathcal{A}', \{U\}, \mathcal{C}', \emptyset \rangle \in \mathtt{part}(\mathcal{I})$ such that A is j in the certain projection of $\mathcal{I}'$, while A is **not** j in the certain projection of $\langle \mathcal{A}' \cup \{U\}, \emptyset, \mathcal{C}', \emptyset \rangle$.*
4. *For each $U \in \mathcal{C}^?$, removal of U is j-relevant for A w.r.t. $\mathcal{I}$ iff there exists some $\mathcal{I}' = \langle \mathcal{A}', \{U\}, \mathcal{C}', \emptyset \rangle \in \mathtt{part}(\mathcal{I})$ such that A is j in the certain projection of $\mathcal{I}'$, while A is **not** j in the certain projection of $\langle \mathcal{A}', \emptyset, \mathcal{C}' \cup \{U\}, \emptyset \rangle$.*

Proof sketch. Let $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ be an incomplete argumentation framework, $A \in \mathcal{A}$ a certain argument and j a justification status. We prove both directions of the first item here. For proofs for the other items, we refer to Appendix A.3.

$\Rightarrow$ See Figure 2.11 for an illustration of the steps in this proof.

1. Suppose that addition of $U \in A^?$ is j-relevant for A w.r.t. $\mathcal{I}$.
2. Then by Definition 2.11 there is a minimal stable-j partial completion $\mathcal{I}^* = \langle \mathcal{A}^*, \mathcal{A}^{?*}, \mathcal{C}^*, \mathcal{C}^{?*} \rangle$ for A w.r.t. $\mathcal{I}$ such that $U \in \mathcal{A}^*$.
3. Now construct the IAF $\mathcal{I}'$ from $\mathcal{I}^*$ by moving U from the certain to the uncertain part: $\mathcal{I}' = \langle \mathcal{A}^* \setminus \{U\}, \mathcal{A}^{?*} \cup \{U\}, \mathcal{C}^*, \mathcal{C}^{?*} \rangle$.

4. Given that $\mathcal{I}^*$ was minimal and $\mathcal{I}^* \in \texttt{part}(\mathcal{I}')$, A cannot be stable-j w.r.t. $\mathcal{I}'$. So there must be some $\mathcal{I}'' = \langle \mathcal{A}'', \mathcal{A}^{?\prime\prime}, \mathcal{C}'', \mathcal{C}^{?\prime\prime} \rangle$ in $\texttt{part}(\mathcal{I}')$ such that A's justification status in the certain projection of $\mathcal{I}''$ is not j – note that this means that U is not in $\mathcal{A}''$ (since A was stable-j in $\mathcal{I}^*$).
5. Then A's justification status in the certain projection of $\mathcal{I}''' = \langle \mathcal{A}'', \{U\}, \mathcal{C}'', \emptyset \rangle$ is not j (because this is the same as the certain projection of $\mathcal{I}''$, i.e. $\langle \mathcal{A}'', \mathcal{C}''|_{\mathcal{A}''} \rangle$).
6. Next, construct $\mathcal{I}^{*\prime} = \langle \mathcal{A}'' \cup \{U\}, \emptyset, \mathcal{C}'', \emptyset \rangle$ from $\mathcal{I}'''$ by moving U from the uncertain part to the certain part. Since $\mathcal{I}^{*\prime}$ is in $\texttt{part}(\mathcal{I}^*)$ and A is stable-j in $\mathcal{I}^*$, A must be j in the certain projection of $\mathcal{I}^{*\prime}$.

$\Leftarrow$ Suppose that there exists some $\mathcal{I}' = \langle \mathcal{A}', \{U\}, \mathcal{C}', \emptyset \rangle \in \texttt{part}(\mathcal{I})$ such that A is not j in $\texttt{cert}(\mathcal{I}')$ and A is j in $\texttt{cert}(\langle \mathcal{A}' \cup \{U\}, \emptyset, \mathcal{C}', \emptyset \rangle)$. Given that $\langle \mathcal{A}' \cup \{U\}, \emptyset, \mathcal{C}', \emptyset \rangle$ has only one completion (i.e., its certain projection), A must be stable-j w.r.t. $\langle \mathcal{A}' \cup \{U\}, \emptyset, \mathcal{C}', \emptyset \rangle$. Consequently, there must be some minimal stable-j partial completion $\mathcal{I}'' = \langle \mathcal{A}'', \mathcal{A}^{?\prime\prime}, \mathcal{C}'', \mathcal{C}^{?\prime\prime} \rangle$ for A w.r.t. $\mathcal{I}$ such that $\langle \mathcal{A}' \cup \{U\}, \emptyset, \mathcal{C}', \emptyset \rangle \in \texttt{part}(\mathcal{I}'')$. Note that $U \in \mathcal{A}''$: otherwise $\langle \mathcal{A}', \emptyset, \mathcal{C}', \emptyset \rangle$ would also be in $\texttt{part}(\mathcal{I}'')$, which contradicts the assumption that A is not j in $\texttt{cert}(\langle \mathcal{A}', \{U\}, \mathcal{C}', \emptyset \rangle)$. To conclude, addition of U is j-relevant for A w.r.t. $\mathcal{I}$. □

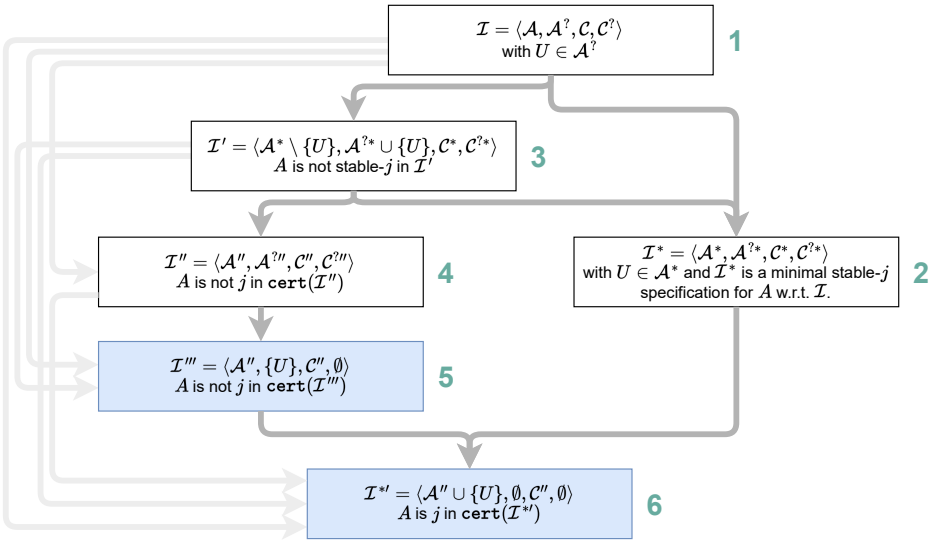


Figure 2.11: Illustration of the steps in the proof of Lemma 2.7 item 1 (from left to right). The IAFs used in the proof are depicted as rectangles; grey arrows between these rectangles represent partial completions – note that not all partial completions of all IAFs are shown in the figure, but only those that we refer to in the proof. Rectangles corresponding to $\mathcal{I}'''$ and $\mathcal{I}^{*\prime}$ are coloured blue, as these are the partial completions of $\mathcal{I}$ for which Lemma 2.7 shows some properties.

In the following proposition, we use the results from Lemma 2.7 to prove a general upper bound on the complexity of j-RELEVANCE.

Proposition 2.11 (Upper bound j-RELEVANCE)**.** *Given an IAF $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$, a certain argument $A \in \mathcal{A}$, an uncertain argument or attack $U \in \mathcal{A}^? \cup \mathcal{C}^?$ and a justification status j, if the problem of deciding j's justification status in a given completion of $\mathcal{I}$ is in the complexity class $\mathbf{C}$, then the problem of deciding if addition or removal of U is j-relevant for A w.r.t. $\mathcal{I}$ is in the complexity class $NP^{\mathbf{C}}$.*

Proof. In order to validate that a given $U \in \mathcal{A}^? \cup \mathcal{C}^?$ is j-relevant for a given $A \in \mathcal{A}$, a suitable polynomial-sized certificate would be some $\mathcal{I}' = \langle \mathcal{A}', \mathcal{A}^{?\prime}, \mathcal{C}', \mathcal{C}^{?\prime} \rangle$ as specified in Lemma 2.7 (so $\mathcal{A}^{?\prime} \cup \mathcal{C}^{?\prime} = \{U\}$). The following procedure can be used to validate that U is j-relevant for A w.r.t. $\mathcal{I}$:

1. Check in polynomial time if $\mathcal{I}' \in \texttt{part}(\mathcal{I})$ and store the result in r_1;
2. Call the $\mathbf{C}$ oracle for finding justification status j to check if A is j w.r.t. $\langle \mathcal{A}', \mathcal{C}'|_{\mathcal{A}'} \rangle$ and store the result in r_2;
3. Let $AF' = \langle \mathcal{A}' \cup \{U\}, \mathcal{C}'|_{\mathcal{A}' \cup \{U\}} \rangle$ if $U \in A^?$ and $AF' = \langle \mathcal{A}', (\mathcal{C}' \cup \{U\})|_{\mathcal{A}'} \rangle$ otherwise. Then call the $\mathbf{C}$ oracle to check if A is j w.r.t. AF' and store the result in r_3.

Then by Lemma 2.7, addition of U is j-relevant for A w.r.t. $\mathcal{I}$ iff $r_1 \wedge \neg r_2 \wedge r_3$. Removal of U is j-relevant for A w.r.t. $\mathcal{I}$ iff $r_1 \wedge r_2 \wedge \neg r_3$. Checking that $r_1 \wedge \neg r_2 \wedge r_3$ (for addition) or $\mathcal{I}$ iff $r_1 \wedge r_2 \wedge \neg r_3$ (for removal) can be done in polynomial time. To conclude, the problem of deciding if addition or removal of U is j-relevant for A w.r.t. $\mathcal{I}$ is in $NP^{\mathbf{C}}$. □

Proposition 2.11 gives a general upper bound that can be exploited to obtain an upper bound for j-RELEVANCE for all justification statuses for which we know the complexity of j-JUSTIFICATION. For example, given that ST-credulous-IN-JUSTIFICATION is in NP, both the addition and the removal variants of ST-credulous-IN-RELEVANCE must be in NP^{NP}, so in Σ_2^p. For justification statuses for which the JUSTIFICATION problem is in P, like GR-credulous-IN, the RELEVANCE problem is in NP. If the JUSTIFICATION problem is on the second level of the polynomial hierarchy, like PR-sceptical-OUT-JUSTIFICATION, then the RELEVANCE variant is in Σ_3^p. Having proved the upper bounds for all variants of the RELEVANCE problem, we turn to the lower bounds in the following two sections.

Lower bounds for grounded, complete and preferred semantics

In order to prove lower bounds for RELEVANCE under GR, CP and PR semantics, we give reductions from the complementary problem of STABILITY, to which we will refer as the INSTABILITY problem.[6] More formally, for every IAF $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$, justification status j and certain argument $A \in \mathcal{A}$, the instance $(\mathcal{I}, A)$ is negative for j-STABILITY iff it is positive for j-INSTABILITY. In the following lemma, we provide

[6]The complexity of RELEVANCE for ST semantics will be covered in Section 2.4.3.

some relations between INSTABILITY and RELEVANCE that will turn out to be useful for reductions from INSTABILITY to RELEVANCE for specific justification statuses.

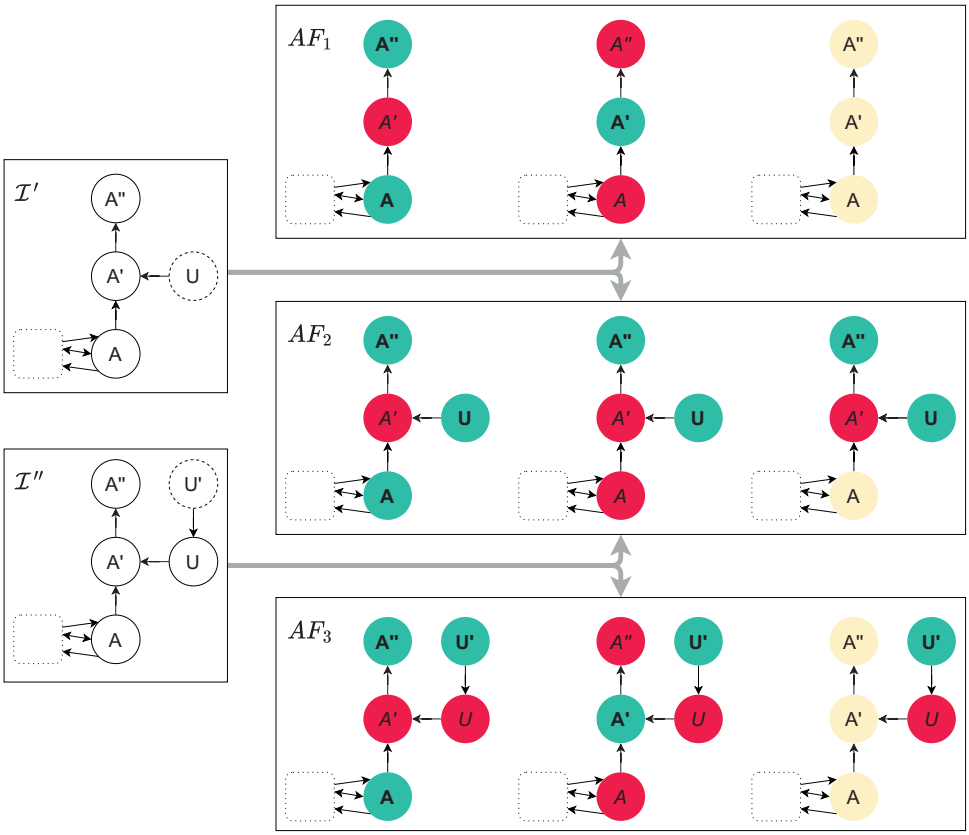


Figure 2.12: Illustration of the IAFs that are used to show Lemma 2.8 item 1. The IAFs given on the left are $\mathcal{I}'$ (upper) and $\mathcal{I}''$ (lower). The rounded rectangle with dotted borders represents the original IAF $\mathcal{I}$ (without A and in- and outgoing attacks). The grey arrows point to certain projections AF_1, AF_2 and AF_3 of partial completions. For each of these AFs, the possible justification statuses are colour-coded: green arguments with boldface font are IN, yellow arguments are UNDEC and red arguments with italic font are OUT. Note that, for a given justification status of A, there is only one possible justification status for each of the additional arguments in $\{A', A'', U, U'\}$.

Lemma 2.8. *[Reduction INSTABILITY to RELEVANCE] Given an incomplete argumentation framework $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$, a certain argument $A \in \mathcal{A}$, semantics $\sigma \in \{GR, CP, PR\}$ and $c \in \{sceptical, credulous\}$:*

1. *Construct $\mathcal{I}'$ and $\mathcal{I}''$ as follows (see Figure 2.12), where A', A'', U and U' are not in $\mathcal{A} \cup \mathcal{A}^?$:*
   - $\mathcal{A}' = \mathcal{A} \cup \{A', A''\}$;
   - $\mathcal{C}' = \mathcal{C} \cup \{(A, A'), (A', A''), (U, A')\}$;

- $\mathcal{I}' = \langle \mathcal{A}', \mathcal{A}^? \cup \{U\}, \mathcal{C}', \mathcal{C}^? \rangle$; *and*
- $\mathcal{I}'' = \langle \mathcal{A}' \cup \{U\}, \mathcal{A}^? \cup \{U'\}, \mathcal{C}' \cup \{(U', U)\}, \mathcal{C}^? \rangle$.

The following three items are equivalent:

(a) A *is* ***not*** *stable-σ-c-IN w.r.t.* $\mathcal{I}$*; and*

(b) addition of U *is* σ*-c-IN-relevant for* A'' *w.r.t.* $\mathcal{I}'$*; and*

(c) removal of U' *is* σ*-c-IN-relevant for* A'' *w.r.t.* $\mathcal{I}''$*.*

2. *Let* $\mathcal{I}' = \langle \mathcal{A}, \mathcal{A}^? \cup \{U\}, \mathcal{C} \cup \{(U, A)\}, \mathcal{C}^? \rangle$ *and* $\mathcal{I}'' = \langle \mathcal{A} \cup \{U\}, \mathcal{A}^? \cup \{U'\}, \mathcal{C} \cup \{(U, A), (U', U)\}, \mathcal{C}^? \rangle$ *where* U *and* U' *are not in* $\mathcal{A} \cup \mathcal{A}^?$*. The following three items are equivalent:*

(a) A *is* ***not*** *stable-σ-c-OUT w.r.t.* $\mathcal{I}$*; and*

(b) addition of U *is* σ*-c-OUT-relevant for* A *w.r.t.* $\mathcal{I}'$*; and*

(c) removal of U' *is* σ*-c-OUT-relevant for* A *w.r.t.* $\mathcal{I}''$*.*

3. *Construct* $\mathcal{I}'$ *and* $\mathcal{I}''$ *as follows, where* A'*,* A''*,* U *and* U' *are not in* $\mathcal{A} \cup \mathcal{A}^?$*:*

- $\mathcal{A}' = \mathcal{A} \cup \{A', A''\}$*;*
- $\mathcal{C}' = \mathcal{C} \cup \{(A, A'), (A', A''), (U, A'), (U, A''), (U, U)\}$*;*
- $\mathcal{I}' = \langle \mathcal{A}', \mathcal{A}^? \cup \{U\}, \mathcal{C}', \mathcal{C}^? \rangle$*; and*
- $\mathcal{I}'' = \langle \mathcal{A}' \cup \{U\}, \mathcal{A}^? \cup \{U'\}, \mathcal{C}' \cup \{(U', U)\}, \mathcal{C}^? \rangle$.

The following three items are equivalent:

(a) A *is* ***not*** *stable-σ-c-UNDEC w.r.t.* $\mathcal{I}$*; and*

(b) addition of U *is* σ*-c-UNDEC-relevant for* A'' *w.r.t.* $\mathcal{I}'$*; and*

(c) removal of U' *is* σ*-c-UNDEC-relevant for* A'' *w.r.t.* $\mathcal{I}''$*.*

Proof sketch. We only prove the first item here, related to IN justification statuses. For the other items, see Appendix A.3. Let $\mathcal{I}' = \langle \mathcal{A} \cup \{A', A''\}, \mathcal{A}^? \cup \{U\}, \mathcal{C} \cup \{(A, A'), (A', A''), (U, A')\}, \mathcal{C}^? \rangle$ and let $\mathcal{I}'' = \langle \mathcal{A} \cup \{A', A'', U\}, \mathcal{A}^? \cup \{U'\}, \mathcal{C} \cup \{(A, A'), (A', A''), (U, A'), (U', U)\}, \mathcal{C}^? \rangle$ (see Figure 2.12).

(a) ⇒ (b) and (c) If A is not stable-σ-c-IN w.r.t. $\mathcal{I}$ (**a**) then by Definition 2.7 of stability there is some completion $AF^* = \langle \mathcal{A}^*, \mathcal{C}^* \rangle$ of $\mathcal{I}$ in which A is not σ-c-IN. Next, we construct three argumentation frameworks based on AF, containing the argument A'', and discuss its status.

- First, construct $AF_1 = \langle \mathcal{A}^* \cup \{A', A''\}, \mathcal{C}^* \cup \{(A, A'), (A', A'')\} \rangle$. Given that A'' is attacked by A', which is only attacked by A in AF_1, A'' cannot be σ-c-IN in AF_1.
- Next, construct $AF_2 = \langle \mathcal{A}^* \cup \{A', A'', U\}, \mathcal{C}^* \cup \{(A, A'), (A', A''), (U, A')\} \rangle$. A'' is σ-c-IN in AF_2, since the unattacked argument U attacks the only attacker of A'' (i.e. A').

- Let $AF_3 = \langle \mathcal{A}^* \cup \{A', A'', U, U'\}, \mathcal{C}^* \cup \{(A, A'), (A', A''), (U, A'), (U', U)\}\rangle$. Given that A is not σ-c-IN in AF^*, A cannot be σ-c-IN in AF_3 either. Since the argument A'' in AF_3 is attacked by A', which is only attacked by A, A'' cannot be σ-c-IN in AF_3.

Now item **(b)** (addition of U is σ-c-IN-relevant for A'' w.r.t. $\mathcal{I}'$) follows from Lemma 2.7, the fact that the IAF $\langle \mathcal{A}^* \cup \{A', A''\}, \{U\}, \mathcal{C}^* \cup \{(A, A'), (A', A''), (U, A')\}, \emptyset\rangle$ is in $\texttt{part}(\mathcal{I}')$ and the status of A'' in AF_1 and AF_2.

Similarly, item **(c)** (removal of U' is σ-c-IN-relevant for A'' w.r.t. $\mathcal{I}''$) follows from Lemma 2.7, the fact that the IAF $\langle \mathcal{A}^* \cup \{A', A'', U\}, \{U'\}, \mathcal{C}^* \cup \{(A, A'), (A', A''), (U, A'), (U', U)\}, \emptyset\rangle$ is in $\texttt{part}(\mathcal{I}'')$ and the status of A'' in AF_2 and AF_3.

(b) $\Rightarrow$ **(a)** If addition of U is σ-c-IN-relevant for A'' w.r.t. $\mathcal{I}'$ then there is some $\mathcal{I}^{*\prime}$ in $\texttt{part}(\mathcal{I}')$ such that A'' is **not** σ-c-IN in $\texttt{cert}(\mathcal{I}^{*\prime}) = \langle \mathcal{A}^{*\prime}, \mathcal{C}^{*\prime}\rangle$. Then A cannot be σ-c-IN in $\texttt{cert}(\mathcal{I}^{*\prime})$ either (since A attacks A', which is the only attacker of A''). Now construct $AF' = \langle \mathcal{A}', \mathcal{C}'\rangle$ where $\mathcal{A}' = \mathcal{A}^{*\prime} \setminus \{A', A'', U\}$ and $\mathcal{C}' = \mathcal{C}^{*\prime} \setminus \{(A, A'), (A', A''), (U, A')\}$. Since A is not σ-c-IN in $\texttt{cert}(\mathcal{I}^{*\prime})$, it cannot be σ-c-IN in AF'. Given that AF' is a completion of $\mathcal{I}$, A cannot be stable-σ-c-IN w.r.t. $\mathcal{I}$.

(c) $\Rightarrow$ **(a)** If removal of U' is σ-c-IN-relevant for A'' w.r.t. $\mathcal{I}''$ then by Lemma 2.7, there is some $\mathcal{I}^{*\prime\prime}$ in $\texttt{part}(\mathcal{I}'')$ such that A'' is **not** σ-c-IN in the certain projection of $\mathcal{I}^{*\prime\prime}$ – without loss of generality, let this certain projection be $AF^{*\prime\prime} = \langle \mathcal{A}^{*\prime\prime}, \mathcal{C}^{*\prime\prime}\rangle$. Now construct $AF' = \langle \mathcal{A}', \mathcal{C}'\rangle$ where $\mathcal{A}' = \mathcal{A}^{*\prime\prime} \setminus \{A', A'', U, U'\}$ and $\mathcal{C}' = \mathcal{C}^{*\prime\prime} \setminus \{(A, A'), (A', A''), (U, A'), (U', U)\}$; since A'' is not σ-c-IN in $AF^{*\prime}$, it cannot be that A is σ-c-IN in AF'. Given that AF' is a completion of $\mathcal{I}$, A cannot be stable-σ-c-IN w.r.t. $\mathcal{I}$. □

Using Lemma 2.8 and the complexity results for STABILITY from Section 2.4.2, we obtain lower bounds for both the addition and removal variants of RELEVANCE under complete, grounded and preferred semantics. For some variants of the RELEVANCE problem, combining the upper bounds identified in Section 2.4.3 with these lower bounds already yields tight complexity results. We present these results in Propositions 2.12, 2.13 and 2.14. The "easiest" of these problems are NP-complete, as we show in Proposition 2.12.

Proposition 2.12. *The following problems are NP-complete:*

1. *CP-credulous-UNDEC-RELEVANCE;*
2. *CP-sceptical-IN-RELEVANCE;*
3. *CP-sceptical-OUT-RELEVANCE;*
4. *GR-credulous-IN-RELEVANCE;*
5. *GR-credulous-OUT-RELEVANCE;*
6. *GR-credulous-UNDEC-RELEVANCE;*
7. *GR-sceptical-IN-RELEVANCE;*

8. *GR-sceptical-OUT-RELEVANCE; and*
9. *GR-sceptical-UNDEC-RELEVANCE.*

Proof sketch. For each of these problems, membership in NP follows from membership in P of the corresponding JUSTIFICATION problems (Dung, 1995), listed in Table 2.1, and Proposition 2.11 (since $\mathrm{NP} = \mathrm{NP}^{\mathrm{P}}$).

For NP-hardness, we can give a reduction from the complementary of the corresponding STABILITY problem, using Lemma 2.8. Here we only consider item 1; the other items are proved in Appendix A.3. By Proposition 2.6, CP-CREDULOUS-UNDEC-STABILITY is CoNP-complete, which means that the complementary problem CP-CREDULOUS-UNDEC-INSTABILITY is NP-complete. By Lemma 2.8 item 3, each instance $I = \langle \mathcal{I}, A \rangle$ of CP-CREDULOUS-UNDEC-INSTABILITY can be transformed into an instance I' such that I is a positive instance of CP-CREDULOUS-UNDEC-INSTABILITY iff I' is a positive instance of CP-CREDULOUS-UNDEC-RELEVANCE. □

For each of the four justification statuses j for which we discuss RELEVANCE in Proposition 2.13, the j-STABILITY problem is Π_2^p-complete, hence j-INSTABILITY must be Σ_2^p-complete and j-RELEVANCE (both addition and removal) must be Σ_2^p-hard. In combination with the upper bound identified in Section 2.4.3, this implies that the corresponding RELEVANCE problems are Σ_2^p-complete.

Proposition 2.13. *The following problems are Σ_2^p-complete:*

1. *CP-credulous-IN-RELEVANCE;*
2. *CP-credulous-OUT-RELEVANCE;*
3. *PR-credulous-IN-RELEVANCE; and*
4. *PR-credulous-OUT-RELEVANCE.*

Proof. For each of these problems, membership in Σ_2^p follows from NP-completeness of the corresponding JUSTIFICATION problems (Dimopoulos and Torres, 1996; Coste-Marquis et al., 2005) in combination with Proposition 2.11. For Σ_2^p-hardness, we can give a reduction from the corresponding INSTABILITY problem, using Lemma 2.8.

1. Since CP-credulous-IN-STABILITY is Π_2^p-complete (by Lemma 2.4 and Baumeister et al. (2021, Theorem 24)), the complementary problem (CP-credulous-IN-INSTABILITY) is Σ_2^p-complete. By Lemma 2.8 item 1, each instance $I = \langle \mathcal{I}, A \rangle$ of CP-credulous-IN-INSTABILITY can be transformed into an instance $I' = \langle \mathcal{I}', A'', U \rangle$ (for addition) or $I' = \langle \mathcal{I}'', A'', U' \rangle$ (for removal) such that I is a positive instance of CP-credulous-IN-INSTABILITY iff I' is a positive instance of CP-credulous-IN-RELEVANCE.
2. CP-credulous-OUT-STABILITY is Π_2^p-complete (by Lemma 2.4 and Baumeister et al. (2021, Theorem 24) in combination with Lemma 2.5), which means that CP-credulous-IN-INSTABILITY is Σ_2^p-complete. By Lemma 2.8 item 2, each instance $I = \langle \mathcal{I}, A \rangle$ of CP-credulous-OUT-INSTABILITY can be transformed into an instance $I' = \langle \mathcal{I}', A, U \rangle$ (for addition) or $I' = \langle \mathcal{I}'', A, U' \rangle$ (for removal) such

that I is a positive instance of CP-credulous-OUT-INSTABILITY iff I' is a positive instance of CP-credulous-OUT-RELEVANCE.

3. Since PR-credulous-IN-STABILITY is Π_2^p-complete (by Lemma 2.4 and Baumeister et al. (2021, Theorem 24)), the complementary problem (PR-credulous-IN-INSTABILITY) is Σ_2^p-complete. By Lemma 2.8 item 1, each instance $I = \langle \mathcal{I}, A \rangle$ of PR-credulous-IN-INSTABILITY can be transformed into an instance $I' = \langle \mathcal{I}', A'', U \rangle$ (for addition) or $I' = \langle \mathcal{I}'', A'', U' \rangle$ (for removal) such that I is a positive instance of PR-credulous-IN-INSTABILITY iff I' is a positive instance of PR-credulous-IN-RELEVANCE.
4. PR-credulous-OUT-STABILITY is Π_2^p-complete (by Lemma 2.4 and Baumeister et al. (2021, Theorem 24) in combination with Lemma 2.5), which means that PR-credulous-IN-INSTABILITY is Σ_2^p-complete. By Lemma 2.8 item 2, each instance $I = \langle \mathcal{I}, A \rangle$ of PR-credulous-OUT-INSTABILITY can be transformed into an instance $I' = \langle \mathcal{I}', A, U \rangle$ (for addition) or $I' = \langle \mathcal{I}'', A, U' \rangle$ (for removal) such that I is a positive instance of PR-credulous-OUT-INSTABILITY iff I' is a positive instance of PR-credulous-OUT-RELEVANCE. □

Similarly, in Proposition 2.14 we show that PR-credulous-UNDEC-RELEVANCE must be Σ_3^p-complete, as PR-credulous-UNDEC-INSTABILITY is Σ_3^p-hard and PR-credulous-UNDEC-JUSTIFICATION is on the second level of the polynomial hierarchy.

Proposition 2.14. *PR-credulous-UNDEC-RELEVANCE is Σ_3^p-complete.*

Proof. Membership in Σ_3^p follows from Σ_2^p-completeness of the corresponding JUSTIFICATION problem (Proposition 2.3) in combination with Proposition 2.11.

For Σ_3^p-hardness, we reduce from PR-credulous-UNDEC-INSTABILITY, which is Σ_3^p-complete since the co-problem PR-credulous-UNDEC-STABILITY is Π_3^p-complete (Proposition 2.8). By Lemma 2.8 item 3, each instance $I = \langle \mathcal{I}, A \rangle$ of PR-credulous-UNDEC-INSTABILITY can be transformed into an instance $I' = \langle \mathcal{I}', A'', U \rangle$ (for addition) or $I' = \langle \mathcal{I}'', A'', U' \rangle$ (for removal) such that I is a positive instance of PR-credulous-UNDEC-INSTABILITY iff I' is a positive instance of PR-credulous-UNDEC-RELEVANCE. □

For some other variants of the RELEVANCE problem, the strategy used in Propositions 2.12, 2.13 and 2.14 does not yield tight complexity results. For instance, for CP-sceptical-UNDEC-RELEVANCE, using this strategy we could only derive that the problem is in Σ_2^p (because CP-sceptical-UNDEC-JUSTIFICATION is CoNP-complete by Proposition 2.2) and that it is NP-hard (because CP-sceptical-UNDEC-STABILITY is CoNP-complete by Proposition 2.7). For these variants, we use another approach, based on an alternative reduction: for the sceptical IN- and OUT-RELEVANCE variants, we reduce from credulous UNDEC-INSTABILITY; for the sceptical UNDEC-RELEVANCE variants, we reduce from credulous IN-INSTABILITY. We will use these reductions to prove the remaining complexities for RELEVANCE under complete, grounded and preferred semantics in Propositions 2.15 and 2.16. In order to use these reductions, we need the following lemma.

Lemma 2.9. *Given an incomplete argumentation framework $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$, a certain argument $A \in \mathcal{A}$, semantics $\sigma \in \{GR, CP, PR\}$:*

1. *Construct $\mathcal{I}'$ and $\mathcal{I}''$ as follows, where A', U and U' are fresh arguments not in $\mathcal{A} \cup \mathcal{A}^?$:*
   - $\mathcal{A}' = \mathcal{A} \cup \{A'\}$;
   - $\mathcal{C}' = \mathcal{C} \cup \{(A, A'), (A', A'), (U, A')\}$;
   - $\mathcal{I}' = \langle \mathcal{A}', \mathcal{A}^? \cup \{U\}, \mathcal{C}', \mathcal{C}^? \rangle$; *and*
   - $\mathcal{I}'' = \langle \mathcal{A}' \cup \{U\}, \mathcal{A}^? \cup \{U'\}, \mathcal{C}' \cup \{(U', U)\}, \mathcal{C}^? \rangle$.

   The following items are equivalent:
   (a) *A is **not** stable-σ-credulous-IN w.r.t. $\mathcal{I}$; and*
   (b) *addition of U' is σ-sceptical-UNDEC-relevant for A' w.r.t. $\mathcal{I}''$; and*
   (c) *removal of U is σ-sceptical-UNDEC-relevant for A' w.r.t. $\mathcal{I}'$.*
2. *Construct $\mathcal{I}'$ and $\mathcal{I}''$ as follows, where A_1, A_2, A_3, A_4, U and U' are fresh arguments not in $\mathcal{A} \cup \mathcal{A}^?$:*
   - $\mathcal{A}' = \mathcal{A} \cup \{A_1, A_2, A_3, A_4\}$;
   - $\mathcal{C}' = \mathcal{C} \cup \{(A, A_1), (A, A_2), (A_1, A_2), (A_2, A_3), (A_3, A_4), (U, U), (U, A_3)\}$;
   - $\mathcal{I}' = \langle \mathcal{A}', \mathcal{A}^? \cup \{U\}, \mathcal{C}', \mathcal{C}^? \rangle$; *and*
   - $\mathcal{I}'' = \langle \mathcal{A}' \cup \{U\}, \mathcal{A}^? \cup \{U'\}, \mathcal{C}' \cup \{(U', U)\}, \mathcal{C}^? \rangle$.

   The following items are equivalent:
   (a) *A is **not** stable-σ-credulous-UNDEC w.r.t. $\mathcal{I}$; and*
   (b) *addition of U' is σ-sceptical-IN-relevant for A_3 w.r.t. $\mathcal{I}''$; and*
   (c) *removal of U is σ-sceptical-IN-relevant for A_3 w.r.t. $\mathcal{I}'$; and*
   (d) *addition of U' is σ-sceptical-OUT-relevant for A_4 w.r.t. $\mathcal{I}''$; and*
   (e) *removal of U is σ-sceptical-OUT-relevant for A_4 w.r.t. $\mathcal{I}'$.*

Proof idea. The proof for this lemma, given in Appendix A.3, is structured in the same way as the proof for Lemma 2.8. □

In the following two propositions, we use Lemma 2.9 to prove the remaining complexities for RELEVANCE under CP and PR semantics. First, we prove that CP-sceptical-UNDEC-RELEVANCE and PR-sceptical-UNDEC-RELEVANCE are Σ_2^p-complete.

Proposition 2.15. *The decision problems CP-sceptical-UNDEC-RELEVANCE and PR-sceptical-UNDEC-RELEVANCE are Σ_2^p-complete.*

Proof. Membership in Σ_2^p follows from CoNP-completeness of the corresponding JUSTIFICATION problems (Proposition 2.2) in combination with Proposition 2.11.

For Σ_2^p-hardness and $\sigma \in \{\text{CP}, \text{PR}\}$, we reduce from σ-credulous-IN-INSTABILITY, which is Σ_2^p-complete since the co-problem σ-credulous-IN-STABILITY is Π_2^p-complete (Lemma 2.4 and the results for necessary credulous acceptance from Baumeister et al. (2021, Theorem 24)). By Lemma 2.9 item 1, each instance $I = \langle \mathcal{I}, A \rangle$ of σ-credulous-IN-INSTABILITY can be transformed into an instance $I' = \langle \mathcal{I}', A'', U \rangle$ (for addition) or $I' = \langle \mathcal{I}'', A'', U' \rangle$ (for removal) such that I is a positive instance of σ-credulous-UNDEC-INSTABILITY iff I' is a positive instance of σ-sceptical-UNDEC-RELEVANCE. □

In Proposition 2.16 we use Lemma 2.9 to show that PR-sceptical-IN- and -OUT-RELEVANCE are Σ_3^p-hard. Together with upper bounds from Section 2.4.3, this implies that PR-sceptical-IN- and -OUT-RELEVANCE are Σ_3^p-complete.

Proposition 2.16. *PR-sceptical-IN-RELEVANCE and PR-sceptical-OUT-RELEVANCE are Σ_3^p-complete.*

Proof. Membership in Σ_3^p follows from Σ_2^p-completeness of the corresponding JUSTIFICATION problems (Dunne and Bench-Capon (2002) and Lemma 2.1) in combination with Proposition 2.11.

For Σ_3^p-hardness, we reduce from PR-credulous-UNDEC-INSTABILITY, which is Σ_3^p-complete since the co-problem PR-credulous-UNDEC-STABILITY is Π_3^p-complete (by Proposition 2.8 and Lemma 2.5). By Lemma 2.9 item 2, each instance $I = \langle \mathcal{I}, A \rangle$ of PR-credulous-UNDEC-INSTABILITY can be transformed into:

1. an instance $I' = \langle \mathcal{I}'', A''', U' \rangle$ such that I is a positive instance of PR-credulous-UNDEC-INSTABILITY iff I' is a positive instance of PR-sceptical-IN-RELEVANCE (for addition); or
2. an instance $I' = \langle \mathcal{I}', A''', U \rangle$ such that I is a positive instance of PR-credulous-UNDEC-INSTABILITY iff I' is a positive instance of PR-sceptical-IN-RELEVANCE (for removal); or
3. an instance $I' = \langle \mathcal{I}'', A'', U' \rangle$ such that I is a positive instance of PR-credulous-UNDEC-INSTABILITY iff I' is a positive instance of PR-sceptical-OUT-RELEVANCE (for addition); or
4. an instance $I' = \langle \mathcal{I}', A'', U \rangle$ such that I is a positive instance of PR-credulous-UNDEC-INSTABILITY iff I' is a positive instance of PR-sceptical-OUT-RELEVANCE (for removal). □

At this point, we have proven all complexity results for RELEVANCE under complete, grounded and preferred semantics as summarised in Table 2.4.

Lower bounds for stable semantics

In this section, we consider the RELEVANCE problems under ST semantics. We will prove that the IN and OUT variants of these problems are Σ_2^p-complete in Proposition 2.17. For the hardness-proof in this proposition, we require two transformations

from Σ_2-SAT instances into IAFs. Recall from Section 2.2.3 that the Σ_2-SAT problem is to decide if there is some truth value assignment τ_X to variables of X such that for each truth value assignment τ_Y: $\Phi[\tau_X, \tau_Y] = \text{False}$ where Φ is a formula in CNF. The transformations are given in the following definition.

Definition 2.12 (Transformations)**.** Let (ϕ, X, Y) be an instance of Σ_2-SAT and let $\Phi = \bigwedge_i c_i$ and $c_i = \bigvee_j \alpha_j$ for each clause c_i in Φ, where α_j are the literals over $X \cup Y$ that occur in clause c_i. We define two transformations for this instance. Let:

$$
\begin{aligned}
\mathcal{A} &= \{x_i, \overline{x_i} \mid x_i \in X\} \cup \{y_i, \overline{y_i} \mid y_i \in Y\} \cup \{\overline{c_i} \mid c_i \in \phi\} \cup \{\phi', \overline{\phi}, \phi\} \\
\mathcal{A}^? &= \{g_i \mid x_i \in X\} \\
\mathcal{C} &= \{(\overline{x_i}, x_i), (g_i, \overline{x_i}) \mid x_i \in X\} \cup \{(\overline{y_i}, y_i), (y_i, \overline{y_i}) \mid y_i \in Y\} \cup \\
&\quad \{(x_k, \overline{c_i}) \mid x_k \in c_i\} \cup \{(\overline{x_k}, \overline{c_i}) \mid \neg x_k \in c_i\} \cup \\
&\quad \{(y_k, \overline{c_i}) \mid y_k \in c_i\} \cup \{(\overline{y_k}, \overline{c_i}) \mid \neg y_k \in c_i\} \cup \\
&\quad \{(\overline{c_i}, \phi') \mid c_i \in \Phi\} \cup \{(\phi', \overline{\phi}), (\chi, \overline{\phi}), (\overline{\phi}, \phi)\}
\end{aligned}
$$

A first transformation into an IAF can be constructed as: $T_1(\phi, X, Y) = \langle \mathcal{A}, \mathcal{A}^? \cup \{\chi\}, \mathcal{C}, \emptyset \rangle$, where χ is a fresh uncertain argument not in $\mathcal{A} \cup \mathcal{A}^?$. A second transformation is $T_2(\phi, X, Y) = \langle \mathcal{A} \cup \{\chi\}, \mathcal{A}^? \cup \{\overline{\chi}\}, \mathcal{C} \cup \{(\overline{\chi}, \chi)\}, \emptyset \rangle$ where χ and $\overline{\chi}$ are not in $\mathcal{A} \cup \mathcal{A}^?$.

An example of transformation T_1 is illustrated in Figure 2.13 for the instance (Φ, X, Y) where the formula $\Phi = (x_1 \vee \neg y_1) \wedge (y_1 \vee y_2)$, $X = \{x_1\}$ and $Y = \{y_1, y_2\}$. Figure 2.14 shows transformation T_2 for the same Σ_2-SAT instance (Φ, X, Y).

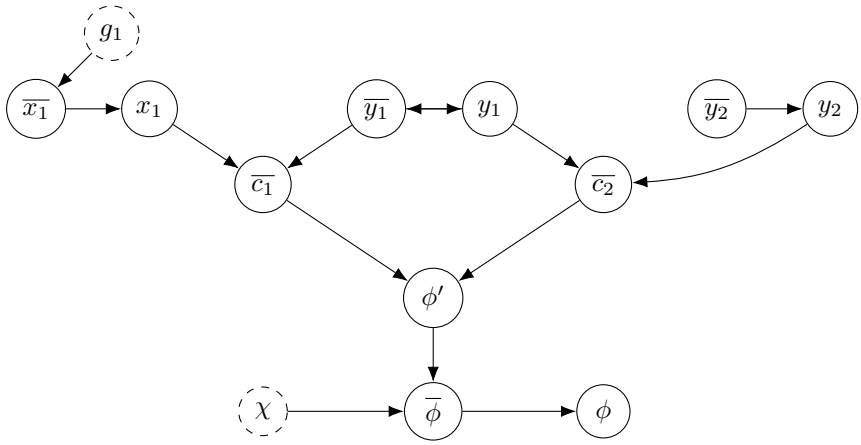


Figure 2.13: Visualisation of the IAF created for the clauses $c_1 = x_1 \vee \neg y_1$ and $c_2 = y_1 \vee y_2$ using transformation T_1 of Definition 2.12.

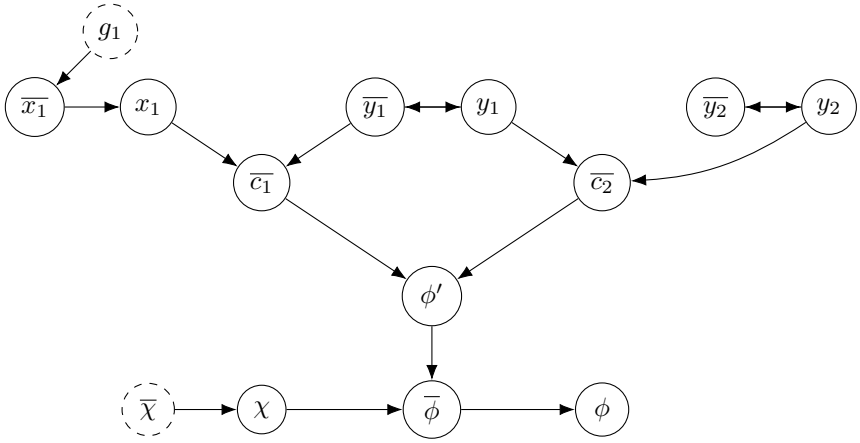


Figure 2.14: Visualisation of the IAF created for the clauses $c_1 = x_1 \vee \neg y_1$ and $c_2 = y_1 \vee y_2$ using transformation T_2 of Definition 2.12.

In the following lemma, we use the transformations T_1 and T_2 to identify equivalences between instances of Σ_2-SAT and RELEVANCE.

Lemma 2.10. *Let (ϕ, X, Y) be an instance of Σ_2-SAT and let $\phi = \bigwedge_i c_i$ and $c_i = \bigvee_j \alpha_j$ for each clause c_i in ϕ, where α_j are the literals over $X \cup Y$ that occur in clause c_i. Now let $\mathcal{I}_1 = T_1(\phi, X, Y)$ and let $\mathcal{I}_2 = T_2(\phi, X, Y)$, using the transformations T_1 and T_2 specified in Definition 2.12. The following items are equivalent:*

1. *(ϕ, X, Y) is a positive instance of Σ_2-SAT;*
2. *Removal of χ is ST-sceptical-IN-relevant for $\overline{\phi}$ w.r.t. $\mathcal{I}_1$;*
3. *Addition of χ is ST-credulous-IN-relevant for ϕ w.r.t. $\mathcal{I}_1$;*
4. *Addition of $\overline{\chi}$ is ST-sceptical-IN-relevant for $\overline{\phi}$ w.r.t. $\mathcal{I}_2$;*
5. *Removal of $\overline{\chi}$ is ST-credulous-IN-relevant for ϕ w.r.t. $\mathcal{I}_2$;*
6. *Removal of χ is ST-sceptical-OUT-relevant for ϕ w.r.t. $\mathcal{I}_1$;*
7. *Addition of χ is ST-credulous-OUT-relevant for $\overline{\phi}$ w.r.t. $\mathcal{I}_1$;*
8. *Addition of $\overline{\chi}$ is ST-sceptical-OUT-relevant for ϕ w.r.t. $\mathcal{I}_2$; and*
9. *Removal of $\overline{\chi}$ is ST-credulous-OUT-relevant for $\overline{\phi}$ w.r.t. $\mathcal{I}_2$.*

Proof sketch. We introduce an auxiliary statement, for which we prove that it equals all of the items above:

(0) There is some $\mathcal{I}^* \in \mathtt{part}(\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \emptyset \rangle)$ such that $\overline{\phi}$ is ST-sceptical-IN in its certain projection AF^* (where $\mathcal{A}$, $\mathcal{A}^?$ and $\mathcal{C}$ are chosen as in Definition 2.12).

Using this additional item, we prove these items separately. For brevity, we only show the equivalence between items 0 and 1 and 0 and 2 here; for full proofs for all equivalences, we refer to Appendix A.3.

(0) ⇒ (1) Suppose that there is some $\mathcal{I}^* \in \texttt{part}(\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \emptyset \rangle)$ such that $\overline{\phi}$ is ST-sceptical-IN in the certain projection $AF^* = \langle \mathcal{A}^*, \mathcal{C}^* \rangle$ of $\mathcal{I}^*$. Let τ_X be an assignment to variables in X such that it assigns True to all $x_i \in X$ such that $g_i \in \mathcal{A}^*$ and False otherwise. Let τ_Y be an arbitrary assignment to all variables in Y. Given that $\overline{\phi}$ is ST-sceptical-IN in AF^*, for each ST extension S of AF^*, at least one of the arguments $\overline{c_i}$ must have been in S, so there is at least one clause in the formula Φ for which none of the variables was assigned True by τ_X and τ_Y. Since we chose τ_Y arbitrarily, we have that (ϕ, X, Y) is a positive instance of Σ_2-SAT.

(1) ⇒ (0) Let (ϕ, X, Y) be a positive instance of Σ_2-SAT. Then there is some assignment τ_X to all variables of X such that for each assignment τ_Y to the variables of Y, $\Phi[\tau_X, \tau_Y]$ is False. Let $G = \{g_i \mid x_i \in X \text{ and } x_i \text{ is assigned True by } \tau_X\}$. Construct $\mathcal{I}^* = \langle \mathcal{A} \cup G, \emptyset, \mathcal{C}|_{\mathcal{A} \cup G}, \emptyset \rangle$ and let AF^* be its certain projection. Note that $\mathcal{I}^* \in \texttt{part}(\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \emptyset \rangle)$. Given that all arguments in G are unattacked, each ST extension of AF^* contains all arguments in G. Furthermore, for each argument $x \in X$, each ST extension of AF^* contains either x (if x is assigned True by τ_X) or $\overline{y}$ (if x is assigned False by τ_Y). Additionally, for each argument $y \in Y$, each ST extension of AF^* contains either y or $\overline{y}$. Given that for each assignment τ_Y to the variables of Y, $\phi[\tau_X, \tau_Y]$ is False, it must be that for each ST extension S of this AF, at least one of the clause arguments $\overline{c_i}$ is in S, so ϕ' is attacked by an argument in S; therefore $\overline{\phi} \in S$. Thus, $\overline{\phi}$ is ST-sceptical-IN in AF^*.

(0) ⇒ (2) Suppose that there is some $\langle \mathcal{A}^*, \mathcal{A}^{?*}, \mathcal{C}|_{\mathcal{A}^* \cup \mathcal{A}^{?*}}, \emptyset \rangle \in \texttt{part}(\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \emptyset \rangle)$ such that $\overline{\phi}$ is ST-sceptical-IN in its certain projection AF^*. This implies that for $\mathcal{I}' = \langle \mathcal{A}^*, \{\chi\}, \mathcal{C}|_{\mathcal{A}^* \cup \{\chi\}}, \emptyset \rangle$ it also holds that $\overline{\phi}$ is ST-sceptical-IN in its certain projection $AF' = AF^* = \langle \mathcal{A}^*, \mathcal{C}|_{\mathcal{A}^*} \rangle$. Note that $\mathcal{I}' \in \texttt{part}(\mathcal{I}_1)$. Now consider $\mathcal{I}'' = \langle \mathcal{A}^* \cup \{\chi\}, \emptyset, \mathcal{C}|_{\mathcal{A}^* \cup \{\chi\}}, \emptyset \rangle$ and its certain projection AF''. In AF'', the argument $\overline{\phi}$ is attacked by the unattacked argument χ, so $\overline{\phi}$ is not ST-sceptical-IN in AF''. Then by Lemma 2.7 item 3, removal of χ is ST-sceptical-IN-relevant for $\overline{\phi}$ w.r.t. $\mathcal{I}_1$.

(2) ⇒ (0) If removal of χ is ST-sceptical-IN-relevant for $\overline{\phi}$ w.r.t. $\mathcal{I}_1$ then by item 3 of Lemma 2.7 there is some $\mathcal{I}' = \langle \mathcal{A}^*, \emptyset, \mathcal{C}|_{\mathcal{A}^*}, \emptyset \rangle$ in $\mathcal{I}_1$ (where $\chi \notin \mathcal{A}^*$, so $\mathcal{I}' \in \texttt{part}(\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \emptyset \rangle)$) such that $\overline{\phi}$ is ST-sceptical-IN in $\texttt{cert}(\mathcal{I}')$. □

The equivalences proven in Lemma 2.10 are exploited in Proposition 2.17 to show Σ_2^p-completeness of some of the RELEVANCE instances under ST semantics.

Proposition 2.17. *The following problems are Σ_2^p-complete:*

- *ST-credulous-IN-RELEVANCE;*
- *ST-sceptical-IN-RELEVANCE;*

- *ST-credulous-OUT-RELEVANCE; and*
- *ST-sceptical-OUT-RELEVANCE.*

Proof. From Proposition 2.11 and Dimopoulos and Torres (1996) and Coste-Marquis et al. (2005) it follows that these problems are in Σ_2^p.

For the hardness proofs, we use Lemma 2.10.

1. ST-credulous-IN-RELEVANCE is Σ_2^p-hard because Σ_2-SAT can be reduced to this problem:
   - For the *addition* variant, convert a given Σ_2-SAT instance (ϕ, X, Y) to the IAF $\mathcal{I}_1 = T_1(\phi, X, Y)$ and check that addition of χ is ST-credulous-IN-relevant for ϕ w.r.t. $\mathcal{I}_1$; this is the case iff (ϕ, X, Y) is a positive Σ_2-SAT instance (by the equality between item 1 and item 3 of Lemma 2.10).
   - For the *removal* variant, convert a given Σ_2-SAT instance (ϕ, X, Y) to the IAF $\mathcal{I}_2 = T_2(\phi, X, Y)$ and check that removal of $\overline{\chi}$ is ST-credulous-IN-relevant for ϕ w.r.t. $\mathcal{I}_2$; this is the case iff (ϕ, X, Y) is a positive Σ_2-SAT instance (by the equality between item 1 and item 5 of Lemma 2.10).
2. ST-sceptical-IN-RELEVANCE is Σ_2^p-hard because Σ_2-SAT can be reduced to this problem: for *addition*, convert a given Σ_2-SAT instance (ϕ, X, Y) to the IAF $\mathcal{I}_2 = T_2(\phi, X, Y)$ and check that addition of $\overline{\chi}$ is ST-sceptical-IN-relevant for $\overline{\phi}$ w.r.t. $\mathcal{I}_2$; this is the case iff (ϕ, X, Y) is a positive Σ_2-SAT instance (by the equality between item 1 and item 4 of Lemma 2.10). For *removal*, the proof is similar but we use the equality between item 1 and item 2.
3. ST-credulous-OUT-RELEVANCE is Σ_2^p-hard because Σ_2-SAT can be reduced to this problem: for *addition*, convert a given Σ_2-SAT instance (ϕ, X, Y) to the IAF $\mathcal{I}_1 = T_1(\phi, X, Y)$ and check that addition of χ is ST-credulous-OUT-relevant for $\overline{\phi}$ w.r.t. $\mathcal{I}_1$; this is the case iff (ϕ, X, Y) is a positive Σ_2-SAT instance (by the equality between item 1 and item 7 of Lemma 2.10). For *removal*, the proof is similar but we use the equality between item 1 and item 9.
4. ST-sceptical-OUT-RELEVANCE is Σ_2^p-hard because Σ_2-SAT can be reduced to this problem: for *addition*, convert a given Σ_2-SAT instance (ϕ, X, Y) to the IAF $\mathcal{I}_2 = T_2(\phi, X, Y)$ and check that addition of $\overline{\chi}$ is ST-credulous-IN-relevant for ϕ w.r.t. $\mathcal{I}_2$; this is the case iff (ϕ, X, Y) is a positive Σ_2-SAT instance (by the equality between item 1 and item 8 of Lemma 2.10). For *removal*, the proof is similar but we use the equality between item 1 and item 6. □

Proposition 2.18 shows that the problems ST-credulous-UNDEC-RELEVANCE and ST-sceptical-existent-UNDEC-RELEVANCE are easy, because these problems only have negative instances.

Proposition 2.18. *The decision problems ST-credulous-UNDEC-RELEVANCE and ST-sceptical-existent-UNDEC-RELEVANCE are trivial.*

Proof. For each argumentation framework $AF = \langle \mathcal{A}, \mathcal{C} \rangle$ such that a ST extension S exists, each argument in $\mathcal{A}$ is either in S or attacked by an argument in S. Therefore, A cannot be ST-credulous-UNDEC or ST-sceptical-existent-UNDEC in AF. This applies for each AF, including all certain projections of all partial completions of each possible IAF, so each instance of ST-credulous-UNDEC-RELEVANCE and ST-sceptical-existent-UNDEC-RELEVANCE must be negative. □

Next, we consider the ST-sceptical-UNDEC-RELEVANCE problem.

Proposition 2.19. *ST-sceptical-UNDEC-RELEVANCE is Σ_2^p-complete.*

Proof sketch. First, we will show that ST-sceptical-UNDEC-RELEVANCE is in Σ_2^p. By Proposition 2.5, ST-sceptical-UNDEC-JUSTIFICATION is CoNP-complete. By Proposition 2.11, this implies that ST-sceptical-UNDEC-RELEVANCE is in Σ_2^p.

For the hardness proof, we use an existing result on the problem of necessary nonempty existence under ST semantics from Skiba et al. (2020), which is defined as follows: given an IAF $\mathcal{I}$, does each completion AF' of $\mathcal{I}$ have a nonempty ST extension? It is shown in Skiba et al. (2020, Theorem 21) that this problem is Π_2^p-hard.

Let $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ be an arbitrary instance of the necessary nonempty existence problem under ST semantics. If $\mathcal{A} = \emptyset$ then $\mathcal{I}$ is a negative instance, because there is a completion $AF' = \langle \mathcal{A}', \mathcal{C}' \rangle$ of $\mathcal{I}$ where $\mathcal{A}' = \emptyset$, which means that AF' has no nonempty ST extension. Alternatively, assume that $\mathcal{A}$ contains at least one argument and let A be an arbitrary argument in $\mathcal{A}$. Then we transform $\mathcal{I}$ into an instance $(\mathcal{I}', A, U)$ of the argument removal variant of the ST-sceptical-UNDEC-RELEVANCE problem, where U is a fresh uncertain argument, not occurring in $\mathcal{A} \cup \mathcal{A}^?$; and $\mathcal{I}' = \langle \mathcal{A}, \mathcal{A}^? \cup \{U\}, \mathcal{C} \cup \{(U, B) \mid B \in \mathcal{A} \cup \mathcal{A}^?\}, \mathcal{C}^? \rangle$.

Then $\mathcal{I}$ is a *positive* instance of necessary nonempty existence under ST semantics iff $(\mathcal{I}', A, U)$ is a *negative* instance of ST-sceptical-UNDEC-RELEVANCE – we prove this in Appendix A.3. This appendix also contains proofs for the argument addition, attack addition and attack removal variants of the ST-sceptical-UNDEC-RELEVANCE problem.

Given that the necessary nonempty existence problem under ST semantics is Π_2^p-hard, the complementary problem of ST-sceptical-UNDEC-RELEVANCE must be Σ_2^p-hard. Together with the membership result from the beginning of this proof, this implies that ST-sceptical-UNDEC-RELEVANCE is Σ_2^p-complete. □

The final variants of the RELEVANCE problem are ST-sceptical-existent-IN-RELEVANCE and ST-sceptical-existent-OUT-RELEVANCE. These are on the second level of the polynomial hierarchy, as we prove in Proposition 2.20.

Proposition 2.20. *ST-sceptical-existent-IN-RELEVANCE and ST-sceptical-existent-OUT-RELEVANCE are Σ_2^p-complete.*

Proof sketch. Membership in Σ_2^p directly follows from the complexity of ST-sceptical-existent-IN- and -OUT-JUSTIFICATION and Proposition 2.11, in the following way: ST-sceptical-existent-IN-JUSTIFICATION is DP-complete by Dunne and Wooldridge (2009, page 92). By Lemma 2.1, ST-sceptical-existent-OUT-JUSTIFICATION is DP-complete as well. Then by Proposition 2.11 the problems of ST-sceptical-existent-IN-RELEVANCE and ST-sceptical-existent-OUT-RELEVANCE are in $NP^{DP} = \Sigma_2^p$; note that $NP^{DP} \subseteq \Sigma_2^p$ as any DP query can be answered by two (adaptive) SAT queries.

In order to prove Σ_2^p-hardness, we reduce from possible sceptical-existent acceptance under ST semantics (Definition 2.5), which was proven to be Σ_2^p-hard in Baumeister et al. (2021, Theorem 25). Let $(\mathcal{I}, A)$ be an arbitrary instance of possible sceptical-existent acceptance under ST semantics where $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$. We transform this into an instance $(\mathcal{I}', A, U)$ of ST-sceptical-existent-IN-RELEVANCE where U is a fresh uncertain argument that is not in $\mathcal{A} \cup \mathcal{A}^?$ and $\mathcal{I}' = \langle \mathcal{A}, \mathcal{A}^? \cup \{U\}, \mathcal{C} \cup \{(U, U)\}, \mathcal{C}^? \rangle$. Then $(\mathcal{I}, A)$ is a positive instance of possible sceptical-existent acceptance under ST semantics iff $(\mathcal{I}', A, U)$ is a positive instance of ST-sceptical-existent-IN-RELEVANCE; this is proven in Appendix A.3.

For ST-sceptical-existent-OUT-RELEVANCE, we transform an arbitrary instance of possible sceptical-existent acceptance under ST semantics $(\mathcal{I}, A)$ to an instance $(\mathcal{I}'', A, U)$ of ST-sceptical-existent-OUT-RELEVANCE, where U is a fresh uncertain argument that is not in $\mathcal{A} \cup \mathcal{A}^?$, B is a fresh argument that is not in $\mathcal{A} \cup \mathcal{A}^?$ and $\mathcal{I}'' = \langle \mathcal{A} \cup \{B\}, \mathcal{A}^? \cup \{U\}, \mathcal{C} \cup \{(B, A), (U, U)\}, \mathcal{C}^? \rangle$. Then $(\mathcal{I}, A)$ is a positive instance of possible sceptical-existent acceptance under ST semantics iff $(\mathcal{I}'', A, U)$ is a positive instance of ST-sceptical-existent-OUT-RELEVANCE. □

This proposition completes our study on complexity for j-RELEVANCE for all justification statuses j within the scope of this chapter. These results, proven in Propositions 2.12–2.20, can be found in Table 2.4.

2.5. Algorithms for stability and relevance

Having identified the complexity classes in which the STABILITY and RELEVANCE problems are situated, in this section we focus on efficient algorithms for solving these problems. To the best of our knowledge no algorithms for stability or relevance in IAFs have been proposed, although algorithms for other problems in the context of IAFs have been proposed in earlier work (in particular, algorithms for necessary and possible acceptance were developed by Baumeister et al. (2021)). In this section, we propose the first algorithms for determining stability and finding relevant updates in the context of IAFs for grounded and complete semantics. A common approach for handling complex problems (on the first level of the polynomial hierarchy) in computational argumentation is by reduction to problems in other formalisms, such as verifying the satisfiability of a propositional formula or finding the stable model (answer set) in a logic program (Cerutti et al., 2018). In line with this approach, our algorithms are based on answer set programming (ASP).

Justification status j	Complexity j-RELEVANCE
ST-credulous-IN/OUT	$\mathbf{\Sigma_2^p}$**-c** P2.17
ST-credulous-UNDEC	**Trivial (no)** P2.18
ST-sceptical-IN/OUT	$\mathbf{\Sigma_2^p}$**-c** P2.17
ST-sceptical-UNDEC	$\mathbf{\Sigma_2^p}$**-c** P2.19
ST-sceptical-existent-IN/OUT	$\mathbf{\Sigma_2^p}$**-c** P2.20
ST-sceptical-existent-UNDEC	**Trivial (no)** P2.18
CP-credulous-IN/OUT	$\mathbf{\Sigma_2^p}$**-c** P2.13
CP-credulous-UNDEC	**NP-c** P2.12
CP-sceptical-IN/OUT	**NP-c** P2.12
CP-sceptical-UNDEC	$\mathbf{\Sigma_2^p}$**-c** P2.15
GR-credulous-IN/OUT	**NP-c** P2.12
GR-credulous-UNDEC	**NP-c** P2.12
GR-sceptical-IN/OUT	**NP-c** P2.12
GR-sceptical-UNDEC	**NP-c** P2.12
PR-credulous-IN/OUT	$\mathbf{\Sigma_2^p}$**-c** P2.13
PR-credulous-UNDEC	$\mathbf{\Sigma_3^p}$**-c** P2.14
PR-sceptical-IN/OUT	$\mathbf{\Sigma_3^p}$**-c** P2.16
PR-sceptical-UNDEC	$\mathbf{\Sigma_2^p}$**-c** P2.15

Table 2.4: Overview of all complexity results related to relevance. We refer to the corresponding proposition by "P" and the proposition number. Full proofs for each of the propositions are presented in the appendix.

ASP-based encodings for AFs

We first discuss ASP-based encodings for complete and grounded semantics in abstract argumentation frameworks that were proposed in earlier work. The program $\pi_{\langle \mathcal{A},\mathcal{C}\rangle}$ encodes an AF, π_{lab} encodes the OUT and UNDEC labels and π_{cp} encodes complete semantics. The π_{cp} program was proposed and proven to be correct by Egly et al. (2010). The program π_{gr} encodes grounded semantics, using the conditional literal **lab(out,Y) : attack(Y,X)**, which can be read as a conjunction of atoms **lab(out,y)** where also **attack(y,x)** is True (Dvořák et al., 2020).

$$
\begin{aligned}
\pi_{\langle \mathcal{A},\mathcal{C}\rangle} &= \{\mathbf{arg(a)} \mid a \in \mathcal{A}\} \cup \{\mathbf{attack(a,b)} \mid (a,b) \in \mathcal{C}\}.\\
\pi_{lab} &= \{\mathbf{lab(out,Y)} \leftarrow \mathbf{lab(in,X)},\ \mathbf{attack(X,Y)}.\}\\
&\quad \cup \{\mathbf{lab(undec,X)} \leftarrow \mathbf{arg(X)},\ \mathbf{not\ lab(in,X)},\ \mathbf{not\ lab(out,X)}.\}\\
\pi_{cp} &= \{\mathbf{lab(in,X)} \leftarrow \mathbf{not\ not\_in(X)},\ \mathbf{arg(X)}.\}\\
&\quad \cup \{\mathbf{not\_in(X)} \leftarrow \mathbf{not\ lab(in,X)},\ \mathbf{arg(X)}.\}\\
&\quad \cup \{\leftarrow \mathbf{lab(in,X)},\ \mathbf{lab(in,Y)},\ \mathbf{attack(X,Y)}.\}\\
&\quad \cup \{\mathbf{undefended(Y)} \leftarrow \mathbf{attack(X,Y)},\ \mathbf{not\ lab(out,X)}.\}\\
&\quad \cup\ \{\leftarrow \mathbf{lab(in,X)},\ \mathbf{undefended(X)}.\}\\
&\quad \cup \{\leftarrow \mathbf{not\_in(X)},\ \mathbf{not\ undefended(X)}.\}\\
\pi_{gr} &= \{\mathbf{lab(in,X)} \leftarrow \mathbf{arg(X)},\ \mathbf{lab(out,Y)}\ :\ \mathbf{attack(Y,X)}.\}
\end{aligned}
$$

Example 2.12. Consider the AF $\langle \mathcal{A}, \mathcal{C} \rangle$ where $\mathcal{A} = \{A, B, C, D, E\}$, and $\mathcal{C} = \{(A, B), (B, C), (C, B), (D, C), (D, E), (E, D)\}$ (a completion of $\mathcal{I}$ in the running example). Then $\pi_{\langle \mathcal{A},\mathcal{C} \rangle} \cup \pi_{lab} \cup \pi_{cp}$ has three answer sets. One of these answer sets contains, amongst others, **lab(in,a), lab(out,b), lab(undec,c), lab(undec,d), lab(undec,e)**. This answer set corresponds to the set $\{A\}$, which is a CP extension of $\mathcal{I}$. Note that $\{A\}$ is also the GR extension. Indeed, the program $\pi_{\langle \mathcal{A},\mathcal{C} \rangle} \cup \pi_{lab} \cup \pi_{gr}$ has one answer set, containing **lab(in,a), lab(out,b), lab(undec,c), lab(undec,d)** and **lab(undec,e)**.

ASP-based encodings for IAFs

Having recalled ASP-based encodings for AFs from earlier work, we now extend them to ASP-based encodings for IAFs. The program $\pi_{\langle \mathcal{A},\mathcal{A}^?,\mathcal{C},\mathcal{C}^? \rangle}$ models the IAF in a similar way as the program $\pi_{\langle \mathcal{A},\mathcal{C} \rangle}$ models an AF. The program π_{guess} "guesses" a completion from $\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ by selecting a subset of $\mathcal{A}^?$ in the first rule (informally, the choice rule **{arg(X)}** means: choose which of the uncertain arguments to include in the model). Similarly, the second rule selects a subset of $\mathcal{C}^?$. Finally, π_{val_comp} ensures that attacks are only included if both the incoming and the outgoing argument are present.

$$\begin{aligned}
\pi_{\langle \mathcal{A},\mathcal{A}^?,\mathcal{C},\mathcal{C}^? \rangle} &= \{\mathbf{argument}(a). \mid a \in \mathcal{A}\} \cup \{\mathbf{uarg}(a). \mid a \in \mathcal{A}^?\} \\
&\quad \cup \{\mathbf{att}(a, b). \mid (a, b) \in \mathcal{C}\} \cup \{\mathbf{uatt}(a, b). \mid (a, b) \in \mathcal{C}^?\} \\
\pi_{guess} &= \{\{\mathbf{arg(X)}\} \leftarrow \mathbf{uarg(X)}.\} \cup \{\{\mathbf{att(X, Y)}\} \leftarrow \mathbf{uatt(X, Y)}.\} \\
\pi_{val_comp} &= \{\mathbf{attack(X, Y)} \leftarrow \mathbf{att(X, Y)}, \mathbf{arg(X)}, \mathbf{arg(Y)}.\}
\end{aligned}$$

Example 2.13. For our example IAF $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ where $\mathcal{A} = \{B, C, E\}$, $\mathcal{A}^? = \{A, D\}$, $\mathcal{C} = \{(A, B), (B, C), (D, C), (D, E), (E, D)\}$ and $\mathcal{C}^? = \{(C, B)\}$, the program $\pi = \pi_{\mathcal{I}} \cup \pi_{guess} \cup \pi_{val\_comp}$ has eight answer sets, corresponding to the eight completions of $\mathcal{I}$. The program $\pi \cup \pi_{lab} \cup \pi_{cp}$ has 21 answer sets, corresponding to all CP extensions of all eight completions of $\mathcal{I}$. If we add the constraint $\{\leftarrow \mathbf{not\ lab(in, b)}.\}$ to $\pi \cup \pi_{lab} \cup \pi_{cp}$, only eight answer sets remain. These correspond to completions with CP extensions that contain B. If, alternatively, we add the constraint $\{\leftarrow \mathbf{lab(in, b)}.\}$, we obtain 13 answer sets, corresponding to CP extensions of completions where B is not IN.

Example 2.13 indicates a correspondence between answer sets and the possible justification status in the IAF's completions. Proposition 2.21 states this correspondence formally.

Proposition 2.21. *Let $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ be an IAF. Consider a query argument $a \in \mathcal{A}$, a semantics $\sigma \in \{CP, GR\}$ and a label $l \in \{IN, OUT, UNDEC\}$.*

- *$\pi_{\mathcal{I}} \cup \pi_{guess} \cup \pi_{lab} \cup \pi_{\sigma} \cup \{\leftarrow \mathbf{not\ lab(l, a)}.\}$ has an answer set iff there is a completion AF of $\mathcal{I}$ such that a is σ-credulous-l w.r.t. AF.*
- *$\pi_{\mathcal{I}} \cup \pi_{guess} \cup \pi_{lab} \cup \pi_{\sigma} \cup \{\leftarrow \mathbf{lab(l, a)}.\}$ has no answer set iff for each completion AF of $\mathcal{I}$, a is σ-sceptical-l w.r.t. AF.*

2.5.1. Encodings for stability under grounded semantics

The correspondence between answer sets and justification statuses in the IAF's completions that was identified in the second part of Proposition 2.21 can directly be used to determine stability under grounded semantics. All variants of the stability problem for grounded semantics are CoNP-complete (Section 2.4.2) and can therefore be solved in a single call of an ASP-solver, using the program $\pi_{\mathcal{I}} \cup \pi_{guess} \cup \pi_{lab} \cup \pi_{gr} \cup \{\leftarrow \mathbf{lab}(\mathbf{l}, \mathbf{a}).\}$ for a query argument a and label l. If this program does not have an answer set, then the argument a is stable-l; if some answer set can be found, then a is not stable-l.

Proposition 2.22. *Let $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ be an IAF. Consider a query argument $a \in \mathcal{A}$ and a label $l \in \{IN, OUT, UNDEC\}$. $\pi_{\mathcal{I}} \cup \pi_{guess} \cup \pi_{lab} \cup \pi_{gr} \cup \{\leftarrow \mathbf{lab}(\mathbf{l}, \mathbf{a}).\}$ has no answer set if and only if a is stable-GR-sceptical/credulous-l w.r.t. $\mathcal{I}$.*

Given that some variants of stability in complete semantics correspond to stability under grounded semantics, this encoding can also be used to solve some of the stability problems for complete semantics. In particular, GR-sceptical-IN-stability corresponds to CP-sceptical-IN-stability, GR-sceptical-OUT-stability corresponds to CP-sceptical-OUT-stability and GR-credulous-UNDEC-stability corresponds to CP-credulous-UNDEC-stability.

2.5.2. Encodings for relevance under grounded semantics

In this section, we provide encodings for relevance under grounded semantics. As we have seen in Section 2.4.3, for grounded semantics all variants of the relevance problem are NP-complete. This implies that they can be solved in a single call of an ASP-solver. The crux of the complexity proof is the following property, formally stated in Lemma 2.7: for a justification status j, addition of an uncertain argument or attack U is j-relevant for an argument A if and only if there is some completion such that A does not have the justification status j, while A is j in the completion resulting from adding U. This property is also useful for developing encodings for relevance.

The program π_{arg_rel} in Listing 1 uses the property from Lemma 2.7 to find the uncertain arguments for which adding or removing a "query" uncertain argument is relevant for some topic argument. In Lines 1–3, an uncertain argument (**query**) is added to the completion. Lines 4–6 handle the definition of the GR labels of the resulting completion. Lines 7–8 check for relevance using Lemma 2.7 based on the label of the topic and the **topic**'s label in the original completion.

Example 2.14. In Example 2.11, we observed that addition of A is GR-sceptical-IN-relevant for C w.r.t. $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$. Indeed, there is a partial completion $\langle \mathcal{A}', \{A\}, \mathcal{C}', \emptyset \rangle$ of $\mathcal{I}$ where $\mathcal{A}' = \mathcal{A}$ and $\mathcal{C}' = \mathcal{C}$, such that C is not GR-sceptical-IN in $\langle \mathcal{A}', \mathcal{C}'|_{\mathcal{A}'} \rangle$, while C is GR-sceptical-IN in $\langle \mathcal{A}' \cup \{A\}, \mathcal{C}'|_{\mathcal{A}' \cup \{A\}} \rangle$. When solving the program $\pi_{\mathcal{I}} \cup \pi_{gr} \cup \pi_{lab} \cup \pi_{guess} \cup \pi_{arg\_rel} \cup \{\mathbf{query}(\mathbf{a}).\} \cup \{\mathbf{topic}(\mathbf{c}).\} \cup \{\leftarrow \mathbf{not\ add\_query\_relevant\_for}(\mathbf{in}, \mathbf{c}, \mathbf{a}).\}$, we find an answer set containing $\mathbf{add\_query\_relevant\_for}(\mathbf{in}, \mathbf{a}, \mathbf{c})$.

Listing 1 Module π_{arg_rel}

```
1  arg_with_q(X,Q) :- arg(X), query(Q).
2  arg_with_q(Q,Q) :- query(Q).
3  attack_with_q(X,Y,Q) :- att (X,Y), arg_with_q(X,Q), arg_with_q(Y,Q).
4  lab_with_q(in,X,Q) :- arg_with_q(X,Q), lab_with_q(out,Y,Q) :
       attack_with_q(Y,X,Q).
5  lab_with_q(out,X,Q) :- attack_with_q(Y,X,Q), lab_with_q(in,Y,Q).
6  lab_with_q(undec,X,Q) :- arg_with_q(X,Q), not lab_with_q(in,X,Q), not
       lab_with_q(out,X,Q).
7  add_relevant_for(J,Q,T) :- query(Q), topic(T), not lab(J,T),
       lab_with_q(J,T,Q).
8  remove_relevant_for(J,Q,T) :- query(Q), topic(T), lab(J,T), not
       lab_with_q(J,T,Q).
```

The program π_{att_rel} in Listing 2 is similar to π_{arg_rel} and finds the uncertain attacks that are relevant to add or remove for some topic argument.

Listing 2 Module π_{att_rel}

```
1 attack_with_q_att(X,Y,Q1,Q2) :- attack(X,Y), q_att(Q1,Q2).
2 attack_with_q_att(Q1,Q2,Q1,Q2) :- q_att(Q1,Q2).
3 lab_with_q_att(in,X,Q1,Q2) :- arg(X), q_att(Q1,Q2),
      lab_with_q_att(out,Y,Q1,Q2) : attack_with_q_att(Y,X,Q1,Q2).
4 lab_with_q_att(out,X,Q1,Q2) :- attack_with_q_att(Y,X,Q1,Q2),
      lab_with_q_att(in,Y,Q1,Q2).
5 lab_with_q_att(undec,X,Q1,Q2) :- arg(X), q_att(Q1,Q2), not
      lab_with_q_att(in,X,Q1,Q2), not lab_with_q_att(out,X,Q1,Q2).
6 add_att_relevant_for(J,Q1,Q2,T) :- q_att(Q1,Q2), topic(T), not lab(J,T),
      lab_with_q_att(J,T,Q1,Q2).
7 remove_att_relevant_for(J,Q1,Q2,T) :- q_att(Q1,Q2), topic(T), lab(J,T),
      not lab_with_q_att(J,T,Q1,Q2).
```

Proposition 2.23. *Let $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ be an IAF. Consider a (topic) argument $t \in \mathcal{A}$ and a label $l \in \{IN, OUT, UNDEC\}$. Let $\pi = \pi_{\mathcal{I}} \cup \pi_{gr} \cup \pi_{lab} \cup \pi_{guess} \cup \pi_{val\_comp} \cup \pi_{arg\_rel}$. For any uncertain (query) argument $q \in \mathcal{A}^?$ and for any uncertain (query) attack $(a, b) \in \mathcal{C}^?$:*

- $\pi \cup \{\mathbf{query}(\mathbf{q}).\, \mathbf{topic}(\mathbf{t}).\, \leftarrow \mathbf{not\ add\_relevant\_for}(\mathbf{l}, \mathbf{q}, \mathbf{t}).\}$ *has an answer set iff addition of q is GR-l-relevant for t w.r.t. $\mathcal{I}$.*
- $\pi \cup \{\mathbf{query}(\mathbf{q}).\, \mathbf{topic}(\mathbf{t}).\, \leftarrow \mathbf{not\ remove\_relevant\_for}(\mathbf{l}, \mathbf{q}, \mathbf{t}).\}$ *has an answer set iff removal of q is GR-l-relevant for t w.r.t. $\mathcal{I}$.*
- $\pi \cup \{\mathbf{q\_att}(\mathbf{a}, \mathbf{b}).\, \mathbf{topic}(\mathbf{t}).\, \leftarrow \mathbf{not\ add\_att\_relevant\_for}(\mathbf{l}, \mathbf{a}, \mathbf{b}, \mathbf{t}).\}$ *has an answer set iff addition of (a, b) is GR-l-relevant for t w.r.t. $\mathcal{I}$.*
- $\pi \cup \{\mathbf{q\_att}(\mathbf{a}, \mathbf{b}).\, \mathbf{topic}(\mathbf{t}).\, \leftarrow \mathbf{not\ remove\_att\_relevant\_for}(\mathbf{l}, \mathbf{a}, \mathbf{b}, \mathbf{t}).\}$ *has an answer set iff removal of (a, b) is GR-l-relevant for t w.r.t. $\mathcal{I}$.*

From Proposition 2.23 it follows that the proposed programs using π_{arg_rel} and π_{att_rel} solve the decision problems of finding relevant updates. However, in a practical setting, we would like to enumerate all relevant updates for specific topics, without specifying the query in the input. Listing 3 therefore guesses one query from the uncertain arguments. Line 1 in Listing 3 chooses a subset of uncertain arguments as query argument. Line 2 then makes sure that there is at most one query (using the operator $<$ for comparison between symbolic constants), while Line 3 adds the constraint that at least one query should exist, provided that there is at least one uncertain argument. Similarly, Listing 4 guesses an uncertain attack query.

Listing 3 Module π_{query_arg}

```
1 {query(X)} 1 :- uarg(X).
2 :- query(X), query(Y), X < Y.
3 :- uarg(Y), not query(X) : uarg(X).
```

Listing 4 Module π_{query_att}

```
1 {q_att(X,Y)} 1 :- uatt(X,Y).
2 :- q_att(A,B), q_att(C,D), A < C.
3 :- q_att(A,B), q_att(C,D), B < D.
4 :- uatt(A,B), not q_att(X,Y) : uatt(X,Y).
```

Proposition 2.24. *Let $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ be an IAF. Consider a (topic) argument $t \in \mathcal{A}$ and a label $l \in \{IN, OUT, UNDEC\}$. Let $\pi = \pi_{\mathcal{I}} \cup \pi_{gr} \cup \pi_{lab} \cup \pi_{guess} \cup \pi_{val\_comp}$. For any uncertain (query) argument $q \in \mathcal{A}^?$ and for any uncertain (query) attack $(a, b) \in \mathcal{C}^?$:*

- *Addition of q is GR-l-relevant for t w.r.t. $\mathcal{I}$ iff $\pi \cup \pi_{arg\_rel} \cup \pi_{query\_arg} \cup \{\mathbf{topic(t)}.\}$ has an answer set with* **add_query_relevant_for(l,q,t)**.
- *Removal of q is GR-l-relevant for t w.r.t. $\mathcal{I}$ iff $\pi \cup \pi_{arg\_rel} \cup \pi_{query\_arg} \cup \{\mathbf{topic(t)}.\}$ has an answer set with* **remove_query_relevant_for(l,q,t)**.
- *Addition of (a, b) is GR-l-relevant for t w.r.t. $\mathcal{I}$ iff $\pi \cup \pi_{att\_rel} \cup \pi_{query\_att} \cup \{\mathbf{topic(t)}.\}$ has an answer set with* **add_att_relevant_for(l,a,b,t)**.
- *Removal of (a, b) is GR-l-relevant for t w.r.t. $\mathcal{I}$ iff $\pi \cup \pi_{att\_rel} \cup \pi_{query\_att} \cup \{\mathbf{topic(t)}.\}$ has an answer set with* **remove_att_relevant_for(l,a,b,t)**.

All relevant operations for t with respect to $\mathcal{I}$ can be obtained by running $\pi \cup \pi_{arg\_rel} \cup \pi_{query\_arg} \cup \pi_{att\_rel} \cup \pi_{query\_att} \cup \{\mathbf{topic(t)}.\}$ in the "brave" enumeration mode of the ASP solver clingo, which takes the union of all answer sets (Gebser et al., 2019).

2.5.3. CEGAR-based approach for stability in complete semantics

While deciding stability for GR is CoNP-complete (as shown in Section 2.4) and can therefore be solved by a single ASP call, this does not hold in general for

other semantics. For complete semantics, deciding CP-credulous-IN-, CP-credulous-OUT- and CP-sceptical-UNDEC-stability is Π_2^P-complete. We therefore propose algorithms that make iterative ASP calls, based on multi-shot solving in which the solver is not restarted between solver calls but retains learned information and avoids repeated grounding (Gebser et al., 2019). Specifically, we propose ASP-based counterexample-guided abstraction refinement (CEGAR) algorithms (Clarke et al., 2003). In CEGAR, an initial abstract model (which is an overapproximation of the solution space) is iteratively refined by drawing candidates from this space and verifying if the candidate is an actual solution, or otherwise refining the abstraction and continuing the search procedure. The procedure for credulous-CP-IN-stability is given in Algorithm 1.

The idea of this algorithm is that it first looks for a counterexample to credulous-CP-IN-stability. By definition of stability, an argument t is not stable-credulous-CP-IN w.r.t. an IAF $\mathcal{I}$ if and only if $\mathcal{I}$ has a completion that does not not have any CP extension with the (topic) argument t. As a first step towards finding such a completion, the encoding in Line 3 searches for a completion that has *at least one* extension without t. If no such completion can be found (Line 15), then in each completion, all extensions contain t. Given that, under CP semantics, each completion contains at least one extension (Dung, 1995), each completion has an extension containing t. Consequently, t must be stable-credulous-CP-IN. Therefore, Line 16 returns True. Alternatively, there is a completion that has *at least one* extension without t. Then it is required to test whether some other CP extension contains t. In order to verify this, the algorithm extracts the completion from the model (Lines 7 and 8) and uses subprocedures COMPLETE (Line 9) to specify one completion and REFINE (Line 11) to exclude a completion in further search. These are defined as follows.

Definition 2.13. Consider an IAF $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ and tuple $\langle \mathcal{A}^*, \mathcal{C}^* \rangle$ with $\mathcal{A}^* \subseteq \mathcal{A} \cup \mathcal{A}^?$ and $\mathcal{C}^* \subseteq \mathcal{C} \cup \mathcal{C}^?$.

$$
\begin{aligned}
\textsc{Parts}(\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle, \langle \mathcal{A}^*, \mathcal{C}^* \rangle) = &\{\mathbf{arg(a_i)} \mid a_i \in \mathcal{A}^? \cap \mathcal{A}^*\} \\
&\cup \{\mathbf{not\ arg(a'_i)} \mid a'_i \in \mathcal{A}^? \setminus \mathcal{A}^*\} \\
&\cup \{\mathbf{att(c_i)} \mid c_i \in \mathcal{C}^? \cap \mathcal{C}^*\} \\
&\cup \{\mathbf{not\ att(c'_i)} \mid c'_i \in \mathcal{C}^? \setminus \mathcal{C}^*\} \\
\textsc{Complete}(\mathcal{I}, AF) = &\{\mathbf{r.} \mid r \in \textsc{Parts}(\mathcal{I}, AF)\} \\
\textsc{Refine}(\mathcal{I}, AF) = &\{\leftarrow\ \mathbf{r_1}, \ldots, \mathbf{r_n}. \mid \{r_1, \ldots, r_n\} = \textsc{Parts}(\mathcal{I}, AF)\}
\end{aligned}
$$

In Line 10, the algorithm tries to find a CP extension of the completion that contains t. If it is possible to find such an extension, then this completion is not a suitable counterexample for disproving stability. Therefore, the algorithm marks that it has completed search for this completion using REFINE and then continues searching for another counterexample. If, alternatively, it is not possible to find an extension of the completion that contains t, then t cannot be stable-credulous-CP-IN. In this case, Algorithm 1 returns False in Line 14.

Algorithm 1 Algorithm to determine whether t is stable-credulous-CP-IN

```
1: procedure GET-CP-CREDULOUS-STABILITY(⟨A, A?, C, C?⟩, l, t)
2:     π_c = π_⟨A,A?,C,C?⟩ ∪ π_guess ∪ π_val_comp ∪ π_lab ∪ π_cp ∪ {← lab(l, t).}
3:     π_v = π_⟨A,A?,C,C?⟩ ∪ π_val_comp ∪ π_lab ∪ π_cp ∪ {← not lab(l, t).}
4:     while True do
5:         (sat, model) = SOLVE(π_c)
6:         if sat then
7:             A* = {a ∈ A ∪ A? | arg(a) ∈ model}
8:             C* = {(a, b) ∈ C ∪ C? | att(a, b) ∈ model}
9:             π_completion = π_v ∪ COMPLETE(⟨A*, C*⟩)
10:            (sat', model') = SOLVE(π_completion)
11:            if sat' then
12:                π_c = π_c ∪ REFINE(⟨A, A?, C, C?⟩, ⟨A*, C*⟩)
13:            else
14:                return False
15:        else
16:            return True
```

2.5.4. CEGAR-based approach for relevance in complete semantics

Just like for stability, for relevance it also holds that many variants of the problem under complete semantics are on the second level of the polynomial hierarchy. In particular, for CP semantics, deciding CP-credulous-IN-, CP-credulous-OUT- and CP-sceptical-UNDEC-relevance is Σ_2^P-complete. The procedure for retrieving all queryables that are credulous-CP-IN-relevant to add and/or remove is given in Algorithm 2. This algorithm reuses subprocedures COMPLETE (Lines 13 and 19) to specify one completion and REFINE (Line 23) from Definition 2.13 to exclude a completion in further search.

The program π_c in Line 3 has an answer set if and only if $\mathcal{I}$ has a completion AF such that t is CP-credulous-l with respect to AF (Proposition 2.21). Line 6 then runs the solver to search for an answer set. If such an answer set exists, an initial abstract model has been found. The algorithm then extracts the corresponding completion from the model in Line 8–9 and verifies for each "query" uncertain argument and attack if removing (Line 11–16) or adding (Line 17–22) the query results in a completion where t is **not** CP-credulous-l. If so, the query is added to the relevant updates R^+ or R^-. Then, REFINE adapts the original program, instructing the solver to find another completion in which t is CP-credulous-l. If no such completion is found, Line 25 returns all relevant updates.

Proposition 2.25. *Let $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ be an IAF, $t \in \mathcal{A}$ an argument and $l \in \{\text{IN}, \text{OUT}, \text{UNDEC}\}$ a label. Let $(R^+, R^-) = \text{GET-CP-CREDULOUS-RELEVANT}(\mathcal{I}, l, t)$. For any uncertain argument or attack $q \in \mathcal{A}^? \cup \mathcal{C}^?$,*

- *addition of q is CP-credulous-l-relevant for t w.r.t. $\mathcal{I}$ iff $q \in R^+$ and*
- *removal of q is CP-credulous-l-relevant for t w.r.t. $\mathcal{I}$ iff $q \in R^-$.*

Algorithm 2 Listing queryables that are credulous-relevant for t

procedure GET-CP-CREDULOUS-RELEVANT($\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle, l, t$)
  $R^+ = \emptyset$, $R^- = \emptyset$
  $\pi_c = \pi_{\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle} \cup \pi_{guess} \cup \pi_{val\_comp} \cup \pi_{lab} \cup \pi_{cp} \cup \{\leftarrow \textbf{not lab}(\mathbf{l}, \mathbf{t}).\}$
  $\pi_v = \pi_{\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle} \cup \pi_{val\_comp} \cup \pi_{lab} \cup \pi_{cp} \cup \{\leftarrow \textbf{not lab}(\mathbf{l}, \mathbf{t}).\}$
  while True **do**
    $(sat, model) = \text{SOLVE}(\pi_c)$
    if *sat* **then**
      $\mathcal{A}^* = \{a \in \mathcal{A} \cup \mathcal{A}^? \mid \textbf{arg}(\mathbf{a}) \in model\}$
      $\mathcal{C}^* = \{(a, b) \in \mathcal{C} \cup \mathcal{C}^? \mid \textbf{att}(\mathbf{a}, \mathbf{b}) \in model\}$
      for $q \in \mathcal{A}^? \cup \mathcal{C}^?$ **do**
        if $q \in \mathcal{A}^* \cup \mathcal{C}^*$ **then**
          if $q \in \mathcal{A}^*$ **then** $F^- = \langle \mathcal{A}^* \setminus \{q\}, \mathcal{C}^* \rangle$ **else** $F^- = \langle \mathcal{A}^*, \mathcal{C}^* \setminus \{q\} \rangle$
          $\pi_{completion} = \pi_v \cup \text{COMPLETE}(F^-)$
          $(sat', model') = \text{SOLVE}(\pi_{completion})$
          if not *sat'* **then**
            $R^+ = R^+ \cup \{q\}$
        else
          if $q \in \mathcal{A}^*$ **then** $F^+ = \langle \mathcal{A}^* \cup \{q\}, \mathcal{C}^* \rangle$ **else** $F^+ = \langle \mathcal{A}^*, \mathcal{C}^* \cup \{q\} \rangle$
          $\pi_{completion} = \pi_v \cup \text{COMPLETE}(F^+)$
          $(sat', model') = \text{SOLVE}(\pi_{completion})$
          if not *sat'* **then**
            $R^- = R^- \cup \{q\}$
      $\pi_c = \pi_c \cup \text{REFINE}(\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle, \langle \mathcal{A}^*, \mathcal{C}^* \rangle)$
    else
      return $(R^+,\ R^-)$

2.5.5. Reducing the input IAF in a preprocessing step

Each of the algorithms can be extended by a preprocessing step that prunes all (certain or uncertain) arguments and attacks that cannot reach the topic argument via a path of (certain or uncertain) attacks. For grounded and complete semantics, this does not influence the stability status or relevant updates, as these semantics satisfy directionality (Baroni and Giacomin, 2007).

Definition 2.14. Let $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ be an IAF and $t \in \mathcal{A}$ be a (topic) argument. Then $\text{REDUCE}(\mathcal{I}, t) = \langle \mathcal{A} \cap R, \mathcal{A}^? \cap R, \mathcal{C}|_R, \mathcal{C}^?|_R \rangle$ is the IAF reduced to $R = \{a \in \mathcal{A} \cup \mathcal{A}^? \mid \langle (a, a_i), (a_i, a_{i+1}), \ldots, (a_{j-1}, a_j), (a_j, t) \rangle \text{ in } \mathcal{C} \cup \mathcal{C}^?\}$.

Proposition 2.26. *For any IAF $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$, justification status j under* GR *or* CP *semantics and argument $t \in \mathcal{A}$,*

- *t is stable-j w.r.t. $\mathcal{I}$ iff t is stable-j w.r.t.* REDUCE*$(\mathcal{I}, t)$ and*
- *for each uncertain argument or rule $q \in \mathcal{A}^? \cup \mathcal{C}^?$, addition (resp. removal) of q is j-relevant for t w.r.t. $\mathcal{I}$ iff addition (resp. removal) of q is j-relevant for t w.r.t.* REDUCE*$(\mathcal{I}, t)$.*

Proof sketch. In order to prove this, we use the following property. For any justification status j under GR or CP semantics (or any other semantics satisfying directionality) and for any completion $AF = \langle \mathcal{A}', \mathcal{C}' \rangle$ of $\mathcal{I}$, t has the justification status j w.r.t. AF iff t has the justification status j w.r.t. REDUCE-COMPLETION$(AF, t) = \langle \mathcal{A}' \cap R', \mathcal{C}'|_{R'} \rangle$ where $R' = \{a \in \mathcal{A}' \mid \langle (a, a_i), (a_i, a_{i+1}), \ldots, (a_{j-1}, a_j), (a_j, t) \rangle \text{ in } \mathcal{C}'\}$. This directly follows from the fact that GR and CP are directional (Baroni and Giacomin, 2007, Proposition 55), that the directionality criterion requires that the extensions of any unattacked set are unaffected by the remaining part of the argumentation framework (Baroni and Giacomin, 2007, Definition 19) and that the set R' is unattacked (Baroni and Giacomin, 2007, Definition 18).

Next, we prove the stability item.

(Left to right) First suppose that t is *not* stable-j w.r.t. REDUCE$(\mathcal{I}, t)$. So there is some completion $\langle \mathcal{A}'', \mathcal{C}'' \rangle$ of REDUCE$(\mathcal{I}, t)$ such that t is *not* j. Then each argument in $\mathcal{A}''$ must be in R as well as in $\mathcal{A} \cup \mathcal{A}^?$, while each attack in $\mathcal{C}''$ must be in $(\mathcal{C} \cup \mathcal{C}^?)|_R$. Thus there must be some completion $\langle \mathcal{A}^*, \mathcal{C}^* \rangle$ of $\mathcal{I}$ such that REDUCE-COMPLETION$(\langle \mathcal{A}^*, \mathcal{C}^* \rangle) = \langle \mathcal{A}'', \mathcal{C}'' \rangle$. By the property above, t cannot be j in $\langle \mathcal{A}^*, \mathcal{C}^* \rangle$. Thus t cannot be stable-j w.r.t. $\mathcal{I}$.

(Right to left) Suppose that t is *not* stable-j w.r.t. $\mathcal{I}$. So there is some completion $\langle \mathcal{A}'', \mathcal{C}'' \rangle$ of $\mathcal{I}$ such that t is *not* j. By the property above, t cannot be j in REDUCE-COMPLETION$(\langle \mathcal{A}'', \mathcal{C}'' \rangle, t)$, which is a completion of REDUCE$(\mathcal{I}, t)$. Therefore t cannot be stable-j w.r.t. REDUCE$(\mathcal{I}, t)$.

The proof for the relevance item is similar. □

2.5.6. Empirical evaluation

Finally, we empirically evaluate the runtime of the proposed algorithms for stability and relevance on synthetically generated benchmarks, using Clingo 5.4.1 for ASP solving (Gebser et al., 2019) on a Intel(R) Core(TM) i7-7820HQ CPU 2.90 GHz 16 GB RAM machine. For generating the benchmarks for the grounded semantics, we randomly generated IAFs $\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ such that $|\mathcal{A} \cup \mathcal{A}^?| \in \{50, 100, 150, 200, 250\}$, $|\mathcal{C} \cup \mathcal{C}^?| = 1.5 \cdot |\mathcal{A} \cup \mathcal{A}^?|$ with a probability of uncertainty $p \in \{0.1, 0.2, 0.3, 0.4\}$, where $|\mathcal{A}^?| = \lfloor p * |\mathcal{A} \cup \mathcal{A}^?| \rfloor$ and $|\mathcal{C}^?| = \lfloor p * |\mathcal{C} \cup \mathcal{C}^?| \rfloor$. For each of these settings, we generated 25 IAFs, where the first argument of $\mathcal{A}$ is the topic argument. This resulted in a total of 500 IAFs. For each of these IAFs, we measured the runtime of the stability algorithm. This algorithm is very fast for all instances, taking 7 milliseconds on

average. In addition, we measured the runtime of the relevance algorithm for enumerating all grounded-relevant operations, in two variations: without preprocessing step and with preprocessing step – see the Table 2.5. The first number refers to the number of instances (out of 100) that were computed successfully within a time limit of 60 seconds. The number between brackets refers to the average runtime in seconds for instances that were computed within this time limit. Table 2.5 shows improved runtimes of the variation of the algorithm with preprocessing step, where first the IAF is reduced to the elements reaching the topic (Definition 2.14) and the relevance procedure is applied on the reduced IAF. The reduction in runtime is present in particular for large IAFs with high p.

$\lvert\mathcal{A} \cup \mathcal{A}?\rvert$	$\lvert\mathcal{C} \cup \mathcal{C}^?\rvert$	**No preprocessing**	**With preprocessing**
50	75	100/100 (0.47)	100/100 (0.11)
100	150	96/100 (1.25)	98/100 (0.37)
150	225	91/100 (3.84)	93/100 (1.48)
200	300	84/100 (3.57)	89/100 (2.47)
250	375	84/100 (6.80)	89/100 (2.03)

Table 2.5: Number of finished instances and average runtime in seconds for the algorithms for grounded relevance (without and with preprocessing).

To evaluate the runtime of Algorithm 2 for the (more complex) relevance problem for complete semantics we applied a similar benchmark generation algorithm but used smaller instances: we now generated IAFs with $|\mathcal{A} \cup \mathcal{A}^?| \in \{5, 10, 15, 20, 25\}$. We evaluated the algorithms with preprocessing step for the tasks of deciding CP-credulous-IN-stability and for enumerating all CP-credulous-IN-relevant updates. The stability algorithm is very fast, taking 8 milliseconds on average for all instances. As shown in Table 2.6, the relevance algorithm is very fast for the instances that can be solved within the time limit of 60 seconds. For larger instances (in particular those with high p) the algorithm is not able to finish in time.

$\lvert\mathcal{A} \cup \mathcal{A}?\rvert$	$\lvert\mathcal{C} \cup \mathcal{C}^?\rvert$	**With preprocessing**
5	7	100/100 (0.01)
10	15	100/100 (0.06)
15	22	100/100 (0.23)
20	30	96/100 (1.29)
25	37	89/100 (1.10)

Table 2.6: Number of finished instances and average runtime in seconds for the algorithms for complete-credulous-IN-relevance with preprocessing.

2.6. Related work

The computational complexity of various problems defined on argumentation frameworks is well-studied; see Dvorák and Dunne (2018) for an overview. Most studies only identify accepted arguments and do not distinguish other justification statuses. Notable exceptions are Dvořák (2012) and Alfano et al. (2022), but neither of these works give complexity results for separate statuses, as we do. In Dvořák (2012), the complexity of justification is studied for the eight justification statuses proposed in Wu and Caminada (2010). These are related to, but not the same as the justification statuses that we study: in their work, every subset of $\{\text{IN}, \text{UNDEC}, \text{OUT}\}$ is a justification status, which should be interpreted as "there is at least one extension where the argument has this label". Accordingly, the justification status $\{\text{IN}\}$ with respect to some semantics σ from Wu and Caminada (2010) corresponds to our σ-sceptical-IN justification status; their $\{\text{OUT}\}$ corresponds to our σ-sceptical-OUT and their $\{\text{UNDEC}\}$ corresponds to our σ-sceptical-UNDEC justification status. There is, however, no direct mapping between the other statuses. As an additional difference, Dvořák (2012) gives an aggregated result for all statuses, whereas we prove the complexity for each justification status separately. Alfano et al. (2022) consider the same three justification statuses that we do, but give complexity results for an aggregation of these statuses: they introduce the notions of determinism, totality and functionality and provide complexity results for determining these for a given argument. They define an argument as deterministic if has the same label (IN, OUT or UNDEC) in all extensions. An argument is total if it is IN or OUT in all extensions; arguments are functional if they are both deterministic and total.

Complexity studies on problems defined on IAFs have emerged more recently. For example, variants of the verification problem on IAFs are studied in Baumeister et al. (2018). The problems of stability and relevance differ from the verification problem as they are defined on arguments rather than on sets of arguments. More related is Baumeister et al. (2021): the authors study potential and necessary credulous and sceptical acceptance in IAFs, where necessary sceptical acceptance of a given argument A, for example, means that in each partial completion's certain projection, each extension (under a given semantics) contains A. The notions of necessary credulous and sceptical acceptance are very similar to specific stability problems: in fact, we used results regarding their complexity for proving the complexity of stable-IN statuses. Finally, the notion of stability, which was originally defined on structured argumentation frameworks in Testerink et al. (2019a), is transposed to the context of IAFs in Mailly and Rossit (2020) and preliminary complexity results for stability under four semantics are provided. In our work, we define a more fine-grained notion of stability and provide more precise complexity characterisations.

Our notion of relevance has not been introduced or studied before. Relevance is related to the notion of influenced sets in Alfano et al. (2021b), which intuitively are sets of arguments whose justification status may change after an update. However, this notion is less strict than relevance: there are situations in which some argument A is in the influenced set of adding an uncertain attack (B, C), while addition of (B, C) is not relevant for A. Other work related to relevance is Rienstra et al. (2020)

on the notion of independence in abstract argumentation. Building on the graph-theoretical criterion of d-separation, the authors introduce independence between argument sets, where the evaluation of one set of arguments can be independent of the evaluation of another set of arguments if the status of a third set of arguments is already known. This seems to be related to our notion of relevance, which can also be seen as a kind of dependence, but in contrast to (ir)relevance, their notion of independence is conditional. Finally, our notion of relevance is related to work on repairing abstract argumentation frameworks (Ulbricht and Baumann, 2019). An AF can be repaired if it is possible to remove (a subset-minimal set of) arguments such that some argument becomes accepted. It is therefore related to our notion of IN-relevance, but a difference is that relevance is defined on incomplete argumentation frameworks rather than normal AFs, and therefore puts a constraint on the arguments that can be removed.

Although no algorithms for detecting stability and finding relevant updates have been proposed in earlier work, our approach is in line with research on algorithms in computational argumentation. A common approach for handling complex problems (on the first level of the polynomial hierarchy) in computational argumentation is by reduction to other formalisms, employing SAT- or ASP-solvers (Cerutti et al., 2018). An example of this approach are the SAT-based algorithms provided in Baumeister et al. (2021) for solving necessary and possible acceptance in IAFs. These are related to our work, as the notion of possible acceptance is strongly related to IN-relevance. The ASP-based algorithms that we have presented can be seen as a variation to this approach. Furthermore, our algorithms for the *dynamic* problem of finding relevant updates are based on the ASP-encodings for *static* problems in abstract argumentation in Egly et al. (2010) and Dvořák et al. (2020). Finally, similar to the algorithms proposed in Niskanen and Järvisalo (2020) and Baumeister et al. (2021), our algorithms for problems in higher levels of the polynomial hierarchy are based on CEGAR (Clarke et al., 2004; Clarke et al., 2003).

2.7. Conclusion

In this chapter, we have studied the problems of detecting stability and relevance in incomplete argumentation frameworks. First, we redefined stability (Testerink et al., 2019a; Mailly and Rossit, 2020) on IAFs. Our definition is a more fine-grained notion than the existing definition on IAFs (Mailly and Rossit, 2020), since it distinguishes between IN, OUT and UNDEC justification statuses. This distinction is appropriate in applications where a dialogue discussing a given argument should be terminated if more information cannot change the argument's (exact) justification status. In addition, we introduced the notion of relevance for IAFs. The identification of relevant actions can be used to select the next uncertain arguments and/or attacks to be added or removed in order to reach a stable situation in an efficient way.

As second main contribution of this chapter, we performed a complexity analysis for the stability and relevance problems. This complexity analysis revealed that the nontrivial variants of the relevance and stability problems have a complexity ranging from the first to the third level of the polynomial hierarchy (cf. Table 2.1).

Interestingly, even within the same semantics, there are differences in the complexity of UNDEC-STABILITY problems and the corresponding IN-STABILITY problems – we consider this to be an additional reason to study a fine-grained notion of stability and relevance.

The third main contribution consists of providing the first algorithms for solving stability and relevance problems under grounded and complete semantics. In our empirical evaluation, we have shown that the algorithms are sufficiently fast for instances with hundreds of (uncertain) arguments under grounded semantics and tens of (uncertain) arguments under complete semantics.

These three contributions answer to research questions 1, 2 and 3 for abstract approaches to argumentation. In the next chapter, these questions are answered for structured approaches to argumentation.

3

Stability & relevance in ASPIC+

This chapter is based on the following papers:

1. In Odekerken et al. (2020) we introduced the problem of stability for ASPIC$^+$, established the complexity of the problem, proposed an initial version of the heuristic algorithm for estimating stability and proved that the algorithm is sound. This conference paper, presented at COMMA 2020, was written under supervision of AnneMarie Borg and Floris Bex.
2. Odekerken et al. (2022a) is the journal version of the aforementioned conference paper, published in Intelligent Systems with Applications. Compared to the conference version, the journal version also contains full proofs, a generalisation of the notion of conflict from classical negation to the use of a contradiction function, a more precise analysis of its performance and a description of use cases (which will be discussed in Chapter 5). This paper was supervised by Floris Bex and AnneMarie Borg. Bas Testerink contributed to the description of the applications.
3. In Odekerken et al. (2023f), presented at KR 2023, we introduced the notion of relevance for ASPIC$^+$, proved its computational complexity and presented ASP-based algorithms for deciding stability and relevance. This paper was written in collaboration with Tuomo Lehtonen, AnneMarie Borg, Johannes Wallner and Matti Järvisalo. My contributions to this paper include the definition of the relevance problem motivated by practical applications, complexity proofs, proofs of properties used for refinement of the ASP solver, collection of the benchmarks and writing of the manuscript.
4. In a manuscript submitted to the Journal of AI Research, we extend upon the aforementioned KR paper. The two main additional contributions are a comparison of stability and relevance in ASPIC+ and the analogous no-

tions in IAFs; and an extension of the algorithm for computing all pieces of relevant information. This manuscript was written in collaboration with Tuomo Lehtonen, Johannes Wallner and Matti Järvisalo. My contributions include the comparison between ASPIC+ and IAFs, as well as a revision of the presentation and proofs.

3.1. Introduction

In the previous chapter, we studied the problems of stability and relevance in the context of incomplete abstract information frameworks (IAFs), which model qualitative uncertainty on the presence of information by distinguishing certain and uncertain arguments and attacks. These (certain and uncertain) arguments are considered to be abstract entities. However, in practice, arguments are not abstract but have an underlying structure. In law enforcement, they are based on legal rules, which are applied to some knowledge base. These rules are generally known in advance, but the information to which they are applied (the knowledge base) may still be incomplete. By adding new knowledge, multiple new arguments may be arise, while other arguments may be excluded. This chapter therefore studies the notions of stability and relevance from a structured perspective.

Five modern examples of structured argumentation are assumption-based argumentation (ABA) (Bondarenko et al., 1997), ASPIC$^+$ (Prakken, 2010; Modgil and Prakken, 2013), Defeasible Logic Programming (DeLP) (García and Simari, 2004), Gorgias (Kakas and Moraitis, 2003) and deductive argumentation (Besnard and Hunter, 2008). In this chapter, we focus on ASPIC$^+$. ASPIC$^+$ enables representation of argument structure as a so-called argumentation theory, which contains a knowledge base and a set of rules, which may be strict or defeasible (Prakken, 2010; Modgil and Prakken, 2013). This argumentation theory then gives rise to arguments and attacks and thus instantiates an abstract argumentation framework. As we motivated in Chapter 1, these two properties of ASPIC$^+$ constitute the two reasons why we focus on this particular argumentation framework.

In the original ASPIC$^+$ definitions it is impossible to model qualitative uncertainty on the presence of specific items in the knowledge base (Prakken, 2010; Modgil and Prakken, 2013). Testerink et al. (2019a) introduced a formalism that adds the concept of queryable literals to argumentation theories for a restricted fragment of ASPIC$^+$. These queryable literals represent uncertain knowledge, which may be added to the knowledge base after further investigation. In this chapter, we extend a more general fragment of ASPIC$^+$ to represent uncertain knowledge. In addition, we redefine the notion of stability and define relevance in the context of these extended "incomplete" ASPIC$^+$ argumentation theories. If no additional information can change the acceptability status of the topic claim, then the claim is *stable* and it does not make sense to inquire into additional arguments related to this claim. Otherwise, one should ask for information that is still *relevant*, in the sense that it can lead towards a stable topic claim. Consequently, the notion of stability is a natural termination criterion for argument-based decision support systems whereas the notion of relevance can be used for efficient collection of information.

As a next step, we compare the notions of stability and relevance in ASPIC$^+$ proposed in this chapter to their IAF counterparts as proposed in Chapter 2. It will turn out that, although the notions are related, stability and relevance in incomplete ASPIC$^+$ cannot be derived directly from stability and relevance in IAFs.

In order to use the notions of stability in practical decision support systems, as sketched in Chapter 1, efficient algorithms should be developed. The first step towards developing such algorithms is obtaining insight in the computational complexity of the problems of deciding stability and relevance. We therefore study the computational complexity of deciding stability and relevance in the considered variant of ASPIC$^+$ under grounded semantics.

To establish an upper bound for these problems, we need to know the complexity of the justification problem of deciding the status of a given (topic) claim in the "complete" ASPIC$^+$ argumentation theory. The complexity class of this problem has not yet been established for grounded semantics. One approach to solving the justification problem consists of first explicitly constructing the abstract argumentation framework that is instantiated by the argumentation theory, then computing the grounded extension and finally deriving the justification status from the presence of arguments concluding the topic claim in the grounded extension. However, this approach is problematic as the first step of argument construction may give rise to an AF that is exponentially larger than the ASPIC$^+$ argumentation theory (Strass et al., 2019). Recent work on assumption-based argumentation (Lehtonen et al., 2021) and specific fragments of ASPIC$^+$ for various semantics (Lehtonen et al., 2020; Lehtonen et al., 2022) indicates that the explicit construction of arguments can be avoided by rephrasing argumentation semantics based on defeasible elements and directly drawing conclusions on the level of these rephrasings. As an additional contribution of this chapter, we show that a similar rephrasing can be applied to develop a polynomial algorithm for justification under grounded semantics in the considered ASPIC$^+$ variant. The existence of such a polynomial algorithm proves that the justification problem is in the complexity class P.

This complexity-theoretic result for the justification problem is then used to prove that stability is CoNP-complete, while deciding relevance is Σ_2^P-complete. Problems in these complexity classes are generally considered intractable: under the common assumption that P $\neq$ NP, there is no exact polynomial-time algorithm for any CoNP-complete problem. Problems in Σ_2^P are considered to be even harder to solve. This means that each algorithm that accurately detects stability or relevance for every input needs exponential time. A naive but accurate stability detection algorithm would, for example, generate all argumentation theories that can be obtained by adding information and subsequently compare the acceptability statuses of topic claims. We will experimentally show that such an algorithm has an impractically large running time, even for very small inputs.

Since practical applications require fast computation for arbitrary inputs, we experiment with alternative solutions. Given that it is impossible to solve all instances of the stability and relevance problems in polynomial time (assuming P $\neq$ NP), there are basically two choices: either we use heuristic algorithms, for which correctness is not guaranteed, or we use algorithms that require worst-case exponential time. In this chapter, we examine both approaches for solving the stability problem.

First, we propose a polynomial-time heuristic algorithm for estimating stability. By proving the algorithm's worst-case time complexity, we guarantee an upper limit on the running time in relation to the size of the input. Having a polynomial algorithm paves the road to efficient stability detection that is scalable to large argumentation theories. The efficiency of our algorithm comes at a cost: it is not exact, which is an inevitable property of applying polynomial algorithms to solve CoNP-complete problems. We perform a theoretical analysis, to give more general guarantees on the algorithm's performance. First, we will show that the algorithm is sound: if the algorithm determines that a claim is stable, then indeed no additional information can change its acceptability. In addition, we will show that the algorithm is complete given specific conditions on the input. If these conditions are met, the algorithm is able to recognize all stable situations and consequently makes sure that the collection of additional information is terminated exactly in time.

As an alternative that is guaranteed to be sound and complete (but not polynomial), we also propose an exact algorithm for stability based on the declarative paradigm of answer set programming (ASP) (Gelfond and Lifschitz, 1988; Niemelä, 1999). This algorithm is inspired by earlier work by Lehtonen et al. (2020), Lehtonen et al. (2021), and Lehtonen et al. (2022). An extensive empirical evaluation of both algorithms on both real-world and synthetic data shows that the ASP-based algorithms are even faster than the heuristic algorithm. Given these promising results, we also develop an ASP-based algorithm for relevance. These algorithms are also able to handle inputs where a preference relation between the (defeasible) reasoning rules is specified in the input. The empirical results show promising scalability, which makes the proposed algorithms usable for real-world applications.[1]

This chapter is structured as follows. Section 3.2 provides the necessary preliminaries. In Section 3.3 we introduce the problems of justification, stability and relevance in the context of ASPIC$^+$. In Section 3.5 it is shown that the justification status can be decided in polynomial time. This result is used for the complexity analysis of stability and relevance in Section 3.6. Subsequently, the algorithms for stability and relevance are presented in Sections 3.7 and 3.8. Then runtime of these algorithms is evaluated in Section 3.9. Section 3.10 discusses related work and Section 3.11 concludes the chapter.

3.2. ASPIC+ definitions

Before formally defining stability and relevance, we first define the instantiation of ASPIC$^+$ that we discuss in this chapter, following the definitions by (Prakken, 2010; Modgil and Prakken, 2013). We will motivate this instantiation at the end of this section. A basic notion of ASPIC$^+$ is that of an argumentation system.

[1]The algorithms are available on GitHub at `https://github.com/DaphneOdekerken/asp_relevance`.

Definition 3.1 (Argumentation system)**.** An *argumentation system* is a quadruple $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ where

- $\mathcal{L}$ is a finite set of literals,
- $\bar{}$ is a contradiction function from $\mathcal{L}$ to $2^{\mathcal{L}}$ such that:
  - for each ϕ, ψ: if $\phi \in \overline{\psi}$ then $\psi \in \overline{\phi}$ and
  - there is no $\phi \in \mathcal{L}$ such that $\phi \in \overline{\phi}$
- $\mathcal{R}$ is a finite set of defeasible rules of the form $a_1, \ldots, a_m \Rightarrow c$ such that $\{a_1, \ldots, a_m, c\} \subseteq \mathcal{L}$, and
- $\leq$ is a partial preorder (i.e., a reflexive and transitive binary relation) on $\mathcal{R}$.

We say that l is a *contradictory* of m iff $m \in \bar{l}$ and $l \in \overline{m}$. In examples we will use classical negation as the contradiction function: for each $x \in \mathcal{L}$, $\overline{x} = \{\neg x\}$ and $\overline{\neg x} = \{x\}$. For a defeasible rule $r : a_1, \ldots, a_m \Rightarrow c$, $\texttt{ants}(r) = \{a_1, \ldots, a_m\}$ are the *antecedents* and $\texttt{cons}(r) = c$ is the *consequent* of r.

An argumentation system gives rise to arguments with respect to a knowledge base. We assume that knowledge bases are consistent sets of axioms.

Definition 3.2 (Knowledge base)**.** A knowledge base $\mathcal{K} \subseteq \mathcal{L}$ over an argumentation system $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ is a consistent set of literals, i.e., for any $l, m \in \mathcal{K}$ we have $l \notin \overline{m}$.

Definition 3.3 (Argumentation theory)**.** An argumentation theory (AT) $T = (AS, \mathcal{K})$ consists of an argumentation system AS and a knowledge base $\mathcal{K}$ over AS.

An argumentation theory gives rise to arguments as follows.

Definition 3.4 (Arguments)**.** Let $T = (AS, \mathcal{K})$ be an argumentation theory. An *argument* A on the basis of the argumentation theory T is a structure obtainable by applying one or more of the following steps finitely many times:

- c is an *observation-based* argument if $c \in \mathcal{K}$.
  The set of premises $\texttt{prem}(A)$ of A is $\{c\}$.
  The set of defeasible rules is $\texttt{defrules}(c) = \emptyset$.
  The conclusion $\texttt{conc}(A)$ of A is c.
  The set of subarguments $\texttt{sub}(A)$ of A is $\{c\}$.
- $A_1, \ldots, A_m \Rightarrow c$ is a *rule-based* argument if for each $i \in [1 \,..\, m]$: there is an argument A_i on the basis of T with conclusion c_i and there is a rule $r : c_1, \ldots, c_m \Rightarrow c$ in $\mathcal{R}$.
  The set of premises $\texttt{prem}(A)$ of A is $\texttt{prem}(A_1) \cup \ldots \cup \texttt{prem}(A_m)$.
  The set of defeasible rules is $\texttt{defrules}(A) = \{r\} \cup \texttt{defrules}(A_1) \cup \cdots \cup \texttt{defrules}(A_m)$.
  The conclusion $\texttt{conc}(A)$ of A is c.
  The set of subarguments $\texttt{sub}(A)$ of A is $\texttt{sub}(A_1) \cup \ldots \cup \texttt{sub}(A_m) \cup \{A\}$.
  The top rule $\texttt{top-rule}(A)$ is r.

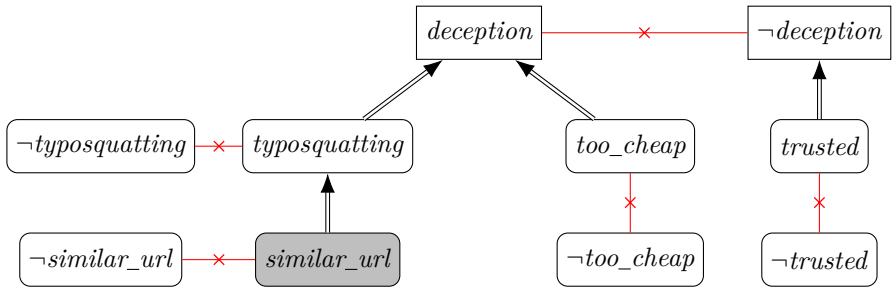


Figure 3.1: Example AT $T = (AS, \mathcal{K})$ from Example 3.1. Each rectangle is a literal in $\mathcal{L}$. Rounded rectangles are queryable literals (defined in Section 3.6.3), literals in $\mathcal{K}$ are shaded, double-lined arrows are defeasible rules and red single lines with a cross correspond to contradictory literals.

We denote by Arg_T the set of *arguments* on the basis of T. An argument with conclusion c is referred to as "an argument for c" and an argument with $\texttt{defrules}(A) \subseteq R \subseteq \mathcal{R}$ by "an argument based on R". Given $R \subseteq \mathcal{R}$, we define $Arg_T(R) = \{A \in Arg_T \mid \texttt{defrules}(A) \subseteq R\}$, so $Arg_T(R)$ is the set of arguments that are based on R.

Example 3.1. We consider an excerpt from the domain of online trade fraud, visualised in Figure 3.1. Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and $\mathcal{L}$ consists of the following literals and their negations.

- *deception*: the web shop used some deceptive tricks.
- *typosquatting*: the url is a misspelling of a popular web shop (rather than an original company name).
- *similar_url*: the url is similar to the url of a popular web shop.
- *too_cheap*: the prices displayed are unrealistically low.
- *trusted*: the url is registered at a trustmark company.

We use classical negation as the contradiction function: for all positive $x \in \mathcal{L}$ we define $\overline{x} = \{\neg x\}$ and $\overline{\neg x} = \{x\}$. The set of rules $\mathcal{R}$ consists of the following rules: ($similar\_url \Rightarrow typosquatting$), ($typosquatting \Rightarrow deception$), ($too\_cheap \Rightarrow deception$), and ($trusted \Rightarrow \neg deception$). For now, assume that rules are equally preferred.

Suppose that we are faced with a url that is similar to that of a popular web shop: $\mathcal{K} = \{similar\_url\}$. Then the set of arguments Arg_T consists of the observation-based argument *similar_url*, as well as the two rule-based arguments $similar\_url \Rightarrow typosquatting$ and $[similar\_url \Rightarrow typosquatting] \Rightarrow deception$.

In ASPIC$^+$, attacks between arguments are based on the elements in the arguments. We consider rebuttal attacks, where arguments attack each other on the conclusion of a defeasible inference.

Definition 3.5 (Rebuttal attack)**.** Argument A *rebuts* argument B (on B') iff $\texttt{conc}(A) \in \overline{\texttt{conc}(B')}$ for some rule-based argument $B' \in \texttt{sub}(B)$.

Example 3.2. Continuing Example 3.1, none of the arguments in Arg_T attack any other argument in Arg_T. However, in the AT $T' = (AS, \mathcal{K} \cup \{\neg typosquatting\})$, there is an argument for $\neg typosquatting$ that attacks the argument for *deception* (on the argument for *typosquatting*). Being observation-based, this argument for $\neg typosquatting$ is not attacked by any argument in $Arg_{T'}$.

As an alternative example, the AT $T'' = (AS, \mathcal{K} \cup \{trusted\})$ contains two additional arguments compared to Arg_T: *trusted* and $trusted \Rightarrow \neg deception$. The argument for $\neg deception$ attacks the argument for *deception* and vice versa.

Not all rebuttals succeed as defeats. In ASPIC$^+$ this depends on a preference relation, denoted by $\preceq$, between arguments. As usual, the strict variant is given by $A \prec B$ iff $A \preceq B$ and $B \not\preceq A$.

Definition 3.6 (Defeat)**.** Argument A *defeats* argument B iff A rebuts B on B' and $A \not\prec B'$.

That is, an argument A might rebut B, but not succeed (because $A \prec B'$ holds). The idea is to obtain $\prec$ from the given partial order $\leq$ on defeasible rules $\mathcal{R}$. The "lifting" of $\leq$ to $\prec$ is defined using two steps. First, the ordering on defeasible rules is lifted to sets of defeasible rules and, subsequently, this order on sets is lifted to $\prec$, comparing arguments. Modgil and Prakken (2013) define four orderings, based on combinations of the so-called elitist or democratic comparisons on sets (of defeasible rules) and the weakest and last-link principles. Here we focus on the last-link ordering, which looks at the top-most defeasible rule(s). In the fragment that we consider, there is at most one defeasibe top rule per argument (none for observation-based arguments). Therefore, two arguments can be compared by examination of at most two defeasible elements. This property motivates our choice for the last-link ordering rather than the weakest-link ordering, as it has multiple advantages. First, when modelling domain knowledge, it is convenient to establish a partial preference ordering on the defeasible rules as only the preference relation between rules with contradicting consequents needs to be considered. Second, from a complexity-theoretic perspective it is advantageous that the number of comparisons between defeasible elements is bounded. This is not necessarily the case for weakest-link ordering (Lehtonen et al., 2022). Third, in this ASPIC$^+$ fragment, the elitist and democratic liftings coincide under last-link ordering: if one compares empty sets or sets containing at most one defeasible rule, both elitist and democratic deliver the same ranking.

Definition 3.7 (Last-link principle)**.** Let A and B be two arguments on the basis of an AT. Under the last-link principle it holds that $B \preceq A$ iff A is observation-based, or both A and B are rule-based and $\texttt{top-rule}(B) \leq \texttt{top-rule}(A)$.

In words, an observation-based argument cannot be strictly less preferred to another argument. For rule-based arguments, we compare the top rules of the arguments.

Example 3.3. Suppose that the rule for $\neg$*deception* given that the website is *trusted* is considered stronger than the rule for *deception* given *typosquatting*. We slightly adapt the AT $T'' = (AS, \mathcal{K} \cup \{trusted\})$ with $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ from Example 3.2. Let $T^* = (AS^*, \mathcal{K} \cup \{trusted\})$, $AS^* = (\mathcal{L}, \bar{}, \mathcal{R}, \leq^*)$ and $\leq^* = \{(typosquatting \Rightarrow deception), (trusted \Rightarrow \neg deception)\}$. This AT contains an argument for *deception* ($A = [similar\_url \Rightarrow typosquatting] \Rightarrow deception$) and an argument for $\neg$*deception* ($B = trusted \Rightarrow \neg deception$). Given that A and B are rule-based and $\texttt{top-rule}(A) \leq \texttt{top-rule}(B)$, it holds that $A \preceq B$. As $\texttt{top-rule}(B) \not\leq \texttt{top-rule}(A)$, $B \not\preceq A$. So argument B for $\neg$*deception* is strictly stronger than argument A for *deception*.

At this point, we have defined our instantiation of the ASPIC$^+$ framework (Prakken, 2010; Modgil and Prakken, 2013). Readers familiar with ASPIC$^+$ will notice that our instantiation is simplified in that we only consider a finite set of literals as axioms, a finite set of defeasible rules and rebuttal attacks. The simplifications provide a minimal number of formal concepts that maintains the theoretical underpinnings of computational argumentation whilst being sufficiently expressive for real-world applications, such as the application at the Dutch police, as we motivate below.

First, full ASPIC$^+$ makes a distinction between ordinary premises (which are defeasible and therefore can be attacked) and axiom premises that cannot be attacked. We only consider axioms in the knowledge base, because in our applications the reasoning is always based on observations – or evidence – that can be considered to have been established with certainty in the context under consideration. For instance, in the context of online trade fraud, the reasoning is based on basic observations provided by citizens or gathered from a police database, and we assume that the fact that such observations have been made can be established with certainty. Of course, it can be argued that we can never be completely certain about an observation: for example, the observation may have been automatically extracted from (noisy) text by a less-than-perfect classifier (cf. Section 5.2.1). However, we solve such issues with establishing observations as part of the information extraction step, which differs per use case. For example, in the fraud intake application, we included an observation validation step in which the citizen can check and correct the extracted observations, as we will explain in Section 5.2.1.

Second, we only consider defeasible rules. If we considered strict rules as well, it would be necessary to train police employees to construct argumentation theories in such a way that the rationality postulates, i.e. desirable properties for structured argumentation, are satisfied. Caminada and Amgoud (2007) proposed the rationality postulates of sub-argument closure, closure under strict rules, direct consistency and indirect consistency. These four rationality postulates are satisfied if the argumentation theory is well-formed (Prakken, 2010). Since we do not consider strict rules between arguments and the knowledge base is required to be consistent (which is not a prerequisite in general by ASPIC$^+$ definitions of Prakken (2010) and Modgil and Prakken (2013)), each argumentation theory as defined in Definition 3.3 is well-formed. This makes it more feasible for police employees without background in computational argumentation to adapt or create rule sets.

Third, we define the language and rules of the argumentation system as finite sets. This enables us to iterate over each literal and rule in the algorithms proposed later in this chapter. For the application in police practice this is not restricting, since it is quite natural to use a limited number of rules and literals are used to capture domain-specific information.

Fourth, we chose to restrict the attack types to rebuttal attacks. In full ASPIC$^+$, arguments can attack each other in three ways: rebuttal, undermining and undercutting attacks (Prakken, 2010). Since our knowledge base only contains axiom premises, undermining attacks on premises cannot occur and therefore do not need to be considered. Furthermore, all notions of conflict in our practical use-cases can be modelled using rebuttal attacks, as we illustrate with examples in Chapter 5.

To conclude, our instantiation of ASPIC$^+$, in which we only consider a finite set of axiom premises, a finite set of defeasible rules and rebuttal attacks, makes it more feasible for police employees to construct or edit argumentation systems, but still suffices to create meaningful decision support systems for our use cases at the police. For future use cases, it might be interesting study more general instantiations of ASPIC$^+$. However, these would complicate the definitions, solutions and proofs; therefore, we leave this for future work.

An argumentation theory gives rise to an abstract argument framework (AF) (Dung, 1995), consisting of the arguments inferred from the argumentation theory and the defeat relation between them. Semantics for argumentation theories are defined through the semantics for AFs, which were introduced in Definition 2.1 (where "attack" should be replaced by "defeat").

Definition 3.8 (AFs corresponding to ATs)**.** An *abstract argumentation framework* (AF) defined by an AT $T = (AS, \mathcal{K})$ is a pair $\langle \mathcal{A}, \mathcal{C} \rangle$ where $\mathcal{A} = Arg_T$ and $\mathcal{C}$ is the defeat relation on $\mathcal{A}$ determined by T.

Example 3.4. In the AT $T = (AS, \mathcal{K})$ of Example 3.1, all arguments in Arg_T are undefeated and therefore in the grounded extension by Definition 2.1. Adding $\neg$*typosquatting* to the knowledge base results in $T' = (AS, \mathcal{K} \cup \{\neg typosquatting\})$: in this AT, the arguments for *similar_url* and $\neg$*typosquatting* are undefeated and therefore in the grounded extension, while the arguments for *typosquatting* and *deception* are defeated by an argument in the grounded extension.

As for $T'' = (AS, \mathcal{K} \cup \{trusted\})$, neither the argument for *deception*, nor the argument for $\neg$*deception* is in the grounded extension.

As we have seen in Chapter 2, multiple semantics exist. In this chapter, we concentrate on the grounded semantics, for two reasons. First, the grounded semantics is the most sceptical complete semantics, which implies that as few arguments as possible are accepted, while still being a complete extension. This fits the application in police investigation: in our case studies, we only accept a statement if, given current information, for every reasonable position (extension) that one can take, an argument for the statement is in that extension. This requirement is only met by the grounded extension, which is the intersection of all complete extensions of the

argumentation framework (Dung, 1995). Second, Cerutti et al. (2020) show that using a grounded reasoner is already an almost perfect heuristic algorithm for many semantics.

3.3. Justification, stability and relevance

In this section, we define justification, stability and relevance in the context of ASPIC$^+$. Whereas the notion of justification gives a static position on the acceptability of a claim, the notions of stability and relevance capture forms of argumentation dynamics: they provide perspectives on reasoning about justifiability of conclusions under incomplete information. Specifically, stability refers to checking whether the justification status of a conclusion cannot be altered by adding new information. Stability hence provides a key point of view, for example, to the decision support systems mentioned in Chapter 1, where the goal is to gather information on a possible conclusion: once a conclusion is stable, gathering additional information is no longer necessary. Relevance provides a point of view to reasoning about not (yet) stable conclusions. In such cases, the choice of which additional information is gathered can play an important role in the efficiency of a decision support system: to determine which yet unknown information should be investigated to ensure stability of a conclusion, only information that can change the stability status of a conclusion is relevant.

3.3.1. Justification

For our applications in decision support, we are interested in the justification status of statements rather than arguments. In ASPIC$^+$, a statement is justified under grounded semantics if and only if there exists an argument for that statement that is justified; otherwise, the statement is not justified (Modgil and Prakken, 2013, Definition 15). Our application in police practice, however, demands a distinction between more than two statuses, which gives a more fine-grained notion of justification in the argumentation theory. This enables us, for example, to differentiate between exculpatory evidence and a lack of evidence in police inquiry.

In earlier work, Prakken and Vreeswijk (2001, Definition 4.28) distinguish three possible statuses that can be assigned to conclusions of arguments; Wu and Caminada (2010) distinguish six statuses to be assigned to literals; and Hecham et al. (2018) propose six statement labels for different variations of defeasible reasoning.

We distinguish four justification statuses, in which we have a special status *unsatisfiable* for literals for which there is no argument, in addition to the definitions of Prakken and Vreeswijk (2001) and Wu and Caminada (2010). Our four justification statuses are similar to the UNSUP, IN_{def}, OUT_{def} and AMBIG statement labels by Hecham et al. (2018).

Definition 3.9 (Multi-valued statement justification status)**.** Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and let $AF = \langle \mathcal{A}, \mathcal{C} \rangle$ be the argumentation framework defined by T. We refer to the grounded extension of AF by $G(T)$. The justification status of $l \in \mathcal{L}$ in T is:

- *unsatisfiable* iff there is no argument for l in $\mathcal{A}$;
- *defended* iff there exists an argument for l in $G(T)$;
- *out* iff there exists an argument for l in $\mathcal{A}$, but each argument for l in $\mathcal{A}$ is defeated by an argument in the grounded extension $G(T)$;
- *blocked* iff there exists an argument for l in $\mathcal{A}$, but no argument for l is in the grounded extension $G(T)$ and at least one argument for l is not defeated by an argument in the grounded extension $G(T)$.

Example 3.5 (Justification status)**.** We reconsider the AT $T = (AS, \mathcal{K})$ from Example 3.1. Let $T''' = (AS, \{similar\_url, too\_cheap, \neg typosquatting, trusted\})$. There are no arguments for $\neg similar\_url$, $\neg too\_cheap$ and $\neg trusted$. These literals are unsatisfiable. There are undefeated arguments for *similar_url*, *too_cheap* and *trusted* in $G(T''')$, so these literals must be defended w.r.t. T''' and $\mathcal{Q}$. The argument for *typosquatting* is defeated by the undefeated argument for $\neg typosquatting$, so *typosquatting* is out w.r.t. T''' and $\mathcal{Q}$. Finally, there are arguments for *deception* and $\neg deception$ that defeat each other, but are not in $G(T''')$ or defeated by any argument in $G(T''')$. Consequently, these literals must be blocked w.r.t. T''' and $\mathcal{Q}$.

The *defended* status that we defined here corresponds to the justified status of conclusions of arguments in Modgil and Prakken (2013, Definition 15). Conclusions of arguments that are not justified can be either *out* or *blocked*, where intuitively a literal that is *blocked* can be accepted by a credulous reasoner, under different semantics (Baroni et al., 2011).

Note that the four statement justification statuses are mutually exclusive and complementary, so each literal in each argumentation theory has exactly one justification status.

Lemma 3.1. *The statement justification statuses unsatisfiable, defended, out and blocked from Definition 3.9 are mutually exclusive and complementary.*

Proof. Consider an arbitrary argumentation theory $T = (AS, \mathcal{K})$ where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and let $l \in \mathcal{L}$ be an arbitrary literal. We consider all possibilities.

- If there is an argument for l in Arg_T, then l cannot be unsatisfiable in T.
  - If there is an argument for l in $G(T)$, then l is defended and cannot be blocked in T. This also means that l cannot be out in T: that would require that each argument for l in $\mathcal{A}$ is defeated by an argument in $G(T)$, which would contradict conflict-freeness of $G(T)$.
  - Alternatively, there is an argument for l in Arg_T, but not in $G(T)$. Then l cannot be defended in T.
    - If there is an argument for l in Arg_T but each argument for l in Arg_T is defeated by an argument in the grounded extension $G(T)$, then l is out and not blocked in T.

  - Alternatively, there is an argument for l in Arg_T, but not in $G(T)$ and at least one argument for l in Arg_T is not defeated by any argument in $G(T)$. Then l is blocked and cannot be out in T.

- If there is no argument for l in Arg_T, then l is unsatisfiable and not defended, out or blocked in T. □

3.3.2. Stability

Stability can be seen as a dynamic variant of the justification status defined in the previous section. The justification status determines if a literal l is justified *given current information*. However, in practice, more information can be added by, for instance, querying a citizen, which possibly results in a change of l's justification status. If additional information cannot influence l's justification status, then we say that l is *stable*. We impose some restrictions on the allowed additions on the knowledge base, for two reasons. The first reason is that, in any dynamic argumentation setting, allowing any change to the argumentation framework effectively makes the problem trivial; a similar problem was identified by Baumann and Brewka (2010). In our case, allowing any literal to be added to the knowledge base makes the stability problem trivial, as then any literal can be made stable and defended simply by adding it to the knowledge base $\mathcal{K}$ (provided that none of its contradictories is already in $\mathcal{K}$). The second reason that prevents us from simply adding any literal to the knowledge base is more practical. Since our use cases are within the law enforcement domain, we base our argumentation theories on laws and literals that represent certain legal concepts. In our running example on online trade fraud, 'fraud' is such a concept, which is defined in Article 326 of the Dutch Criminal Code. Because we cannot expect citizens to know the exact legal definition of 'fraud', we do not want to directly ask them if they have been a victim of such fraud. Instead, we want to ask citizens whether they observed certain basic, non-legal facts, such as whether they sent money or received a product, and then use legal rules captured in the rule base $\mathcal{R}$ to derive legal conclusions about fraud.[2] We therefore distinguish between queryable and non-queryable literals, where queryables are a specific set of literals that can be obtained (i.e. added to the knowledge base) by querying the environment (e.g. the citizen).

Definition 3.10 (Queryables)**.** Given an argumentation theory $T = (AS, \mathcal{K})$ with $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$, a set of queryables $\mathcal{Q}$ is a set of literals $\mathcal{L}$ such that $\mathcal{K} \subseteq \mathcal{Q} \subseteq \mathcal{L}$ and if $q \in \mathcal{Q}$ then for each $q' \in \overline{q} : q' \in \mathcal{Q}$.

The set of queryables restricts the literals that can be added to the knowledge base. Note that Definition 3.10 requires that all contradictories of each literal in $\mathcal{Q}$ are also in $\mathcal{Q}$. The reason for this is that contradictories can be seen as alternative answers to a given query: when querying the environment (e.g., q?), both the literal (q) and its contradictory ($\in \overline{q}$, e.g., $\neg q$) can be given as an answer and added to the knowledge base. Adding a queryable literal q to the knowledge base of an argumentation

[2] See e.g. Bex and Verheij (2013) for the interplay between facts and law in a formal argumentation setting.

theory $T = (AS, \mathcal{K})$ (where $\overline{q} \cap \mathcal{K} = \emptyset$) results in a new argumentation theory $T' = (AS, \mathcal{K} \cup \{q\})$. The set of all argumentation theories that can be obtained by adding queryables to the knowledge base is the set of future argumentation theories.

Definition 3.11 (Future argumentation theories)**.** Let $T = (AS, \mathcal{K})$ be an AT and $\mathcal{Q}$ a set of queryables. We say that AT $T' = (AS, \mathcal{K}')$ is a future argumentation theory of T, denoted by $T \sqsubseteq_Q T'$, if $\mathcal{K} \subseteq \mathcal{K}' \subseteq \mathcal{Q}$.

We define a strict variant $T \sqsubset_Q T'$ by $T \sqsubseteq_Q T'$ and $T' \not\sqsubseteq_Q T$. By definition, $T \sqsubseteq_Q T$. Since all future argumentation theories are argumentation theories (Definition 3.3), their knowledge base must be consistent.

Next, we formally define stability based on the notions of future argumentation theories and the justification status of statements. We distinguish four types of stability, relative to the four justification statuses from Definition 3.9.

Definition 3.12 (j-stability)**.** Let $T = (AS, \mathcal{K})$ be an AT and $\mathcal{Q}$ is a set of queryables. Given literal $l \in \mathcal{L}$ and justification status j in {unsatisfiable, defended, out, blocked}, l is stable-j in T w.r.t. $\mathcal{Q}$ iff l is j in T' for each T' with $T \sqsubseteq_Q T'$.

Example 3.6 (Stability statuses)**.** Consider the argumentation system AS from Example 3.1, illustrated in Figure 3.1. The rounded rectangles in this figure correspond to queryables: $\mathcal{Q} = \{$*typosquatting*, *similar_url*, *too_cheap*, *trusted*, $\neg$*typosquatting*, $\neg$*similar_url*, $\neg$*too_cheap*, $\neg$*trusted*$\}$.

- Let $T_1 = (AS, \{$*similar_url*, $\neg$*trusted*$\})$. There is no future AT of T_1 having *trusted* in its knowledge base, as $\neg$*trusted* is already in the knowledge base of T_1 and knowledge bases must be consistent. Thus $\neg$*deception* is stable-unsatisfiable w.r.t. T_1 and $\mathcal{Q}$.
- Let $T_2 = (AS, \{$*similar_url*, *typosquatting*, $\neg$*trusted*$\})$. There are arguments for *deception* in T_2, having subarguments for *similar_url* and *typosquatting*. These arguments will still exist in every future AT of T_2. Furthermore, there is no future AT containing an argument defeating any of these arguments, because of the presence of $\neg$*trusted* and *typosquatting* in the knowledge base. Therefore, every future AT of T_2 contains at least one undefeated argument for *deception*, which must be in the grounded extension. This implies that *deception* is stable-defended w.r.t. T_2 and $\mathcal{Q}$.
- Let $T_3 = (AS, \{$*similar_url*, $\neg$*too_cheap*, $\neg$*typosquatting*$\})$. There is an argument for *deception* in T_3, but it is defeated by the observation-based argument for $\neg$*typosquatting*, which is in the grounded extension of T_3 and every future AT. In addition, there is no alternative argument for *deception* in any future AT of T_3. Consequently, *deception* must be stable-out w.r.t. T_3 and $\mathcal{Q}$.
- Let $T_4 = (AS, \{$*similar_url*, *typosquatting*, *trusted*$\})$. Just like in T_2, there are arguments for *deception* in T_4, having subarguments for *similar_url* and *typosquatting*. These arguments will still exist in every future AT of T_4. Because of the presence of *typosquatting* in the knowledge base of T_4, there is no future AT containing an observation-based argument for $\neg$*typosquatting*.

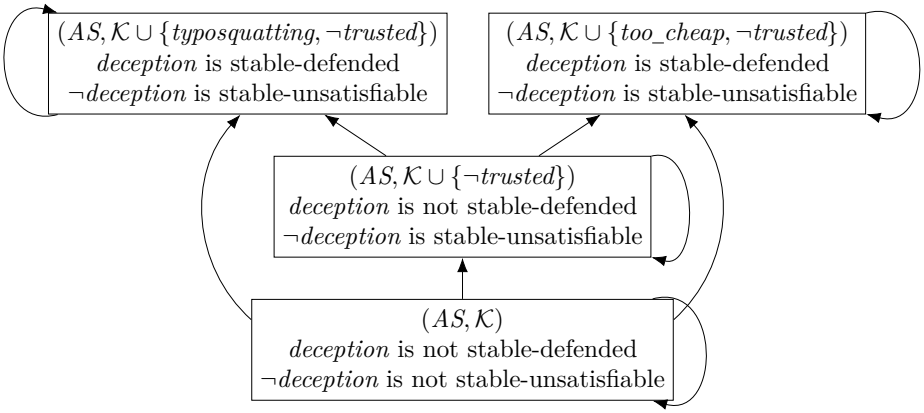


Figure 3.2: Illustration of the future argumentation theories of T from Example 3.1 with $\mathcal{Q}$ as specified in Example 3.6. Argumentation theories are depicted as rectangular nodes. For any pair of ATs T_i and T_j, if there is an arrow from the node corresponding to T_i to the node corresponding to T_j then $T_i \sqsubseteq_Q T_j$. Not all future ATs w.r.t. T and $\mathcal{Q}$ are illustrated in this figure.

As a result, the arguments for *deception* cannot be defeated by an argument in the grounded extension. However, they are defeated by the argument for $\neg$*defeated* with the subargument for *trusted*, which will remain present in every future AT of T_4. To conclude, *deception* is stable-blocked w.r.t. T_4.

3.3.3. Relevance

When a literal does not have a stable status, that is, there is a future AT that changes the justification status of the literal, a natural question to ask is which queryables are *relevant* for making the literal stable. In other words, which queryables should be added to the knowledge base in order to obtain an AT where this literal is stable? This is reminiscent of the notion of relevance, introduced in the context of incomplete (abstract) argumentation frameworks in Section 2.3.3. Here we propose an analogous definition of relevance for ASPIC$^+$, based on the notion of minimal stable future ATs, i.e., future ATs where the knowledge base is minimally expanded and the considered literal is stable.

Definition 3.13 (Minimal stable-j future theory)**.** Let $T = (AS, \mathcal{K})$ be an AT, $\mathcal{Q}$ be a set of queryables, and j be a justification status. Given a literal $l \in \mathcal{L}$, a *minimal stable-j future theory* for l w.r.t. T and $\mathcal{Q}$ is an AT T' with $T \sqsubseteq_Q T'$ such that:

1. l is stable-j in T', and
2. there is no T'' such that l is stable-j in T'' and $T \sqsubseteq_Q T'' \sqsubset_Q T'$.

Example 3.7 (Minimal stable-j future theory)**.** Consider again the AT $T = (AS, \mathcal{K})$ from Example 3.1 with $\mathcal{Q}$ and $\mathcal{K} = \{similar\_url\}$ as specified in Example 3.6. This

AT and some of its future ATs are illustrated in Figure 3.2. We have that $\neg$*deception* is stable-unsatisfiable w.r.t. $T' = (AS, \{$*similar_url*, *too_cheap*, $\neg$*trusted*$\})$ and $\mathcal{Q}$, but T' is not a minimal stable-unsatisfiable future theory for $\neg$*deception* w.r.t. T and $\mathcal{Q}$, since $\neg$*deception* would also be stable-unsatisfiable without *too_cheap*. In contrast, the future AT $(AS, \{$*similar_url*, $\neg$*trusted*$\})$ is minimal stable-unsatisfiable for $\neg$*deception* as removing $\neg$*trusted* from the knowledge base would result in an AT where $\neg$*deception* is not stable-unsatisfiable.

There are two minimal stable-defended future theories for *deception* w.r.t. T and $\mathcal{Q}$. The first one is $(AS, \{$*similar_url*, *typosquatting*, $\neg$*trusted*$\})$ and the other one is $(AS, \{$*similar_url*, *too_cheap*, $\neg$*trusted*$\})$. The future AT $(AS, \{$*similar_url*, $\neg$*trusted*$\})$ is not minimal stable-defended for *deception*, because *deception* is not stable-defended w.r.t. this AT.

Literals in the knowledge base of a minimal stable-j future theory that do not occur in the original knowledge base are considered relevant.

Definition 3.14 (j-relevance)**.** Let $T = (AS, \mathcal{K})$ be an AT with $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$, let $\mathcal{Q}$ be a set of queryables and let j be a justification status. Given $l \in \mathcal{L}$ and $q \in Q$ with $q \notin \mathcal{K}$ and $\overline{q} \cap \mathcal{K} = \emptyset$, we say that q is j-relevant for l w.r.t. T and $\mathcal{Q}$ iff there is a minimal stable-j future theory $T' = (AS, \mathcal{K}')$ for l w.r.t. T and $\mathcal{Q}$ such that $q \in \mathcal{K}'$.

Example 3.8 (j-relevance)**.** Continuing Example 3.7 illustrated in Figure 3.2, for the AT $T = (AS, \mathcal{K})$ and queryables $\mathcal{Q}$ there is a single minimal stable-unsatisfiable future theory for $\neg$*deception* w.r.t. T and $\mathcal{Q}$. The only queryable in the knowledge base of this theory that is not in $\mathcal{K}$ is $\neg$*trusted*. Consequently, $\neg$*trusted* is the only literal that is unsatisfiable-relevant for $\neg$*deception* w.r.t. T and $\mathcal{Q}$.

For *deception*, there are two minimal stable-defended future theories w.r.t. T and $\mathcal{Q}$. Their knowledge bases contain (combinations of) the queryables *too_cheap*, *typosquatting* and $\neg$*trusted*. Therefore, the literals that are defended-relevant for *deception* w.r.t. T and $\mathcal{Q}$ are *too_cheap*, *typosquatting* and $\neg$*trusted*.

Note that it is possible that a queryable and its negation are both relevant for a given topic literal.

Example 3.9. Consider the AT $T = (AS, \mathcal{K})$ where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$, $\mathcal{L} = \{q, \neg q, l, \neg l\}$, the contradiction function corresponds to classical negation, $\mathcal{R} = \{(q \Rightarrow l), (\neg q \Rightarrow l)\}$, $\leq = \emptyset$ and $\mathcal{K} = \emptyset$. Suppose that $\mathcal{Q} = \{q, \neg q\}$. Then both q and $\neg q$ are defended-relevant for l w.r.t. T since l is unsatisfiable in T and defended in both $(AS, \{q\})$ and $(AS, \{\neg q\})$.

3.4. Comparison to stability and relevance in abstract argumentation

In the previous section, we defined the notions of stability and relevance for an incomplete version of ASPIC$^+$. A natural question to ask then is whether one can express our ASPIC$^+$-version of stability in terms of stability on Incomplete

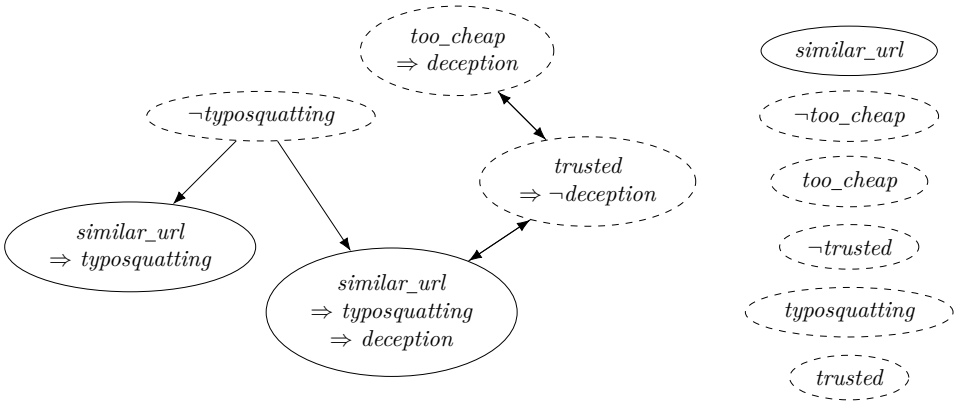


Figure 3.3: The IAF corresponding to the argumentation theory and set of queryables from our running example in the domain of online trade fraud. Arguments are presented as ellipses, where those corresponding to certain arguments have solid borders and those corresponding to uncertain arguments have dashed borders. Defeats are represented by arrows.

Argumentation Frameworks (IAFs), as defined in Chapter 2. And, analogously, can ASPIC$^+$-relevance be expressed in terms of IAF-relevance? In this section, we compare our notions of stability and relevance to the existing notions of stability and relevance as defined on IAFs in Chapter 2.

A natural way of instantiating an IAF for a given argumentation theory and set of queryables is to include all arguments that can be inferred from a future argumentation theory but not from the current one into the set of uncertain arguments. For the attack relation in the IAF, we include all defeats that occur in some future argumentation theory.

Definition 3.15 (IAF corresponding to AT and $\mathcal{Q}$)**.** Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and let $\mathcal{Q}$ be a set of queryables. The corresponding IAF is a tuple $IAF(T, \mathcal{Q}) = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \emptyset \rangle$, where

- $\mathcal{A} = Arg_T$,
- $\mathcal{A}^? = \{A \mid \exists T' \text{ s.t. } T \sqsubseteq_Q T' \text{ and } A \in Arg_{T'}\} \setminus \mathcal{A}$, and
- $\mathcal{C} = \{(A, B) \mid \exists T' \text{ s.t. } T \sqsubseteq_Q T', A \in Arg_{T'}, B \in Arg_{T'} \text{ and } A \text{ defeats } B \text{ in } T'\}$.

In the following example, we present the corresponding IAF for our example argumentation theory in the domain of online trade fraud.

Example 3.10. In Figure 3.3 we illustrate the IAF $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \emptyset \rangle = IAF(T, Q)$ corresponding to the AT T and set of queryables $\mathcal{Q}$ from our running example on online trade fraud. Recall from Example 3.1 that the set of arguments Arg_T consists of an observation-based argument for *similar_url* and rule-based arguments for *typosquatting* and *deception*. In addition, all queryables (identified in Example 3.6) that are not contradictory to some literal in the knowledge base can still become an observation-based argument in some future argumentation theory. These cor-

respond to six of the eight uncertain arguments, depicted with dashed borders in Figure 3.3. Note that there is no uncertain argument for $\neg$*similar_url* in $A^?$ in $IAF(T, Q)$: given that *similar_url* is in the knowledge base, there is no future argumentation theory of T that gives rise to an (observation-based) argument for $\neg$*similar_url*. In addition, $\mathcal{A}^?$ contains uncertain arguments for *deception* and $\neg$*deception*. Between these certain and uncertain arguments, we identify six defeats, represented by arrows. On first sight, it may be surprising that there are no defeats between, for example, $\neg$*too_cheap* and *too_cheap* $\Rightarrow$ *deception*. The reason for this is that there is no future argumentation theory giving rise to both of these arguments.

An IAF can be *completed* by deciding for all uncertain arguments and defeats whether or not they are present, as defined in Definition 2.4 from the previous chapter. Recall from Definitions 2.6, 2.7 and 2.11 that the stability status and relevant uncertain arguments are defined based on the justification statuses of all completions.

Example 3.11. Continuing the IAF $\mathcal{I}$ from Example 3.10, the argument *similar_url* is stable-GR-sceptical-IN w.r.t. $\mathcal{I}$ as it is in the grounded extension of every completion of $\mathcal{I}$. The other arguments are not stable-GR-sceptical-IN as they can be defeated by undefeated arguments in completions of $\mathcal{I}$.

For the argument *similar_url* $\Rightarrow$ *typosquatting* $\Rightarrow$ *deception*, removal of $\neg$*typosquatting* is GR-sceptical-IN-relevant w.r.t. $\mathcal{I}$. Similarly, the removal of *trusted* $\Rightarrow \neg$*deception* is GR-sceptical-IN-relevant w.r.t. $\mathcal{I}$.

3.4.1. Structured and abstract stability

Having recalled stability and relevance on IAFs, we can now investigate relations between these notions on structured argumentation frameworks and on the corresponding IAFs. We first give an example of an AT T and set of queryables $\mathcal{Q}$ where an argument that is stable-GR-sceptical-IN w.r.t. $IAF(T, \mathcal{Q})$ (on the abstract level) has as its conclusion a literal that is stable-defended w.r.t. T and $\mathcal{Q}$ (on the structured level).

Example 3.12. Recall the IAF $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \emptyset \rangle = IAF(T, \mathcal{Q})$ corresponding to the AT T and set of queryables $\mathcal{Q}$ from our running example on online trade fraud, illustrated in Figure 3.3. In Example 3.11 we have seen that the argument *similar_url* was stable-GR-sceptical-IN w.r.t. $\mathcal{I}$ as it was in the grounded extension of every completion of $\mathcal{I}$. Furthermore, the conclusion of this argument (the literal *similar_url*) is stable-defended w.r.t. T and $\mathcal{Q}$.

Next, we prove that in general, GR-sceptical-IN-stability on the abstract level implies defended-stability on the structured level.

Proposition 3.1. *Let $T = (AS, \mathcal{K})$ be an argumentation theory and $\mathcal{Q}$ a set of queryables and let $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \emptyset \rangle = IAF(T, \mathcal{Q})$. If some $A \in \mathcal{A}$ is stable-GR-sceptical-IN w.r.t. $\mathcal{I}$ then* `conc`(A) *is stable-defended w.r.t. T and $\mathcal{Q}$.*

Proof. Suppose that $A \in \mathcal{A}$ is stable-GR-sceptical-IN w.r.t. $\mathcal{I}$. Now let $T' = (AS, \mathcal{K}')$ be an arbitrary future argumentation theory of T. We consider the AF defined by T' (Definition 3.8): let $\mathcal{A}' = Arg_{T'}$ and let $\mathcal{C}'$ be all defeats between arguments from $\mathcal{A}'$: $\mathcal{C}' = \{(A, B) \mid A \in \mathcal{A}', B \in \mathcal{A}' \text{ and } A \text{ defeats } B \text{ in } T'\}$. By Definition 3.8, $\mathcal{A} \cup \mathcal{A}^? = \{A \mid \text{there is a } T'' \text{ s.t. } T \sqsubseteq_Q T'' \text{ and } A \in Arg_{T''}\}$. Consequently, $\mathcal{A}' \subseteq \mathcal{A} \cup \mathcal{A}^?$. In addition, since $\mathcal{K} \subseteq \mathcal{K}'$, each argument in Arg_T must also be in $Arg_{T'}$, so $\mathcal{A} \subseteq \mathcal{A}'$. Further note that $\mathcal{C}' = \mathcal{C}|_{\mathcal{A}'}$. Then by Definition 2.4, $\langle \mathcal{A}', \mathcal{C}' \rangle$ is a completion of $\mathcal{I}$. Given that $A \in \mathcal{A}$ is stable-GR-sceptical-IN w.r.t. $\mathcal{I}$, by Definition 2.7, A must be in the grounded extension of $\langle \mathcal{A}', \mathcal{C}' \rangle$ as well: $A \in G(T')$. Then by Definition 3.9 of justification, $\texttt{conc}(A)$ is defended w.r.t. $\langle \mathcal{A}', \mathcal{C}' \rangle$. Since T' was chosen arbitrarily from the future argumentation theories of T, by Definition 3.12 $\texttt{conc}(A)$ is stable-defended w.r.t. T and $\mathcal{Q}$. □

Given that GR-sceptical-IN-stability w.r.t. the IAF implies defended-stability w.r.t. the AT, as we have seen in Proposition 3.1, it is interesting to know whether this also holds the other way around. In other words: if there is a literal l that is stable-defended w.r.t. some argumentation theory T and set of queryables $\mathcal{Q}$, is there always some argument for l that is stable-GR-sceptical-IN w.r.t. $IAF(T, \mathcal{Q})$? We will prove that this is not the case by showing a counterexample.

Counterexample 3.1. Consider the argumentation theory $T = (AS, \mathcal{K})$ and set of queryables $\mathcal{Q}$ illustrated in Figure 3.4. This is another example in the domain of online trade fraud, which models (possible) reasons for a web shop being suspect. In this toy example, a web shop is suspect if it pretends to be old (*pretends_old*) while it is recently registered (*new_registration*) or if it does not have many positive reviews ($\neg$*many_reviews*) while this would be expected (*reviews_expected*). One would expect reviews if the web shop has a page on Trustpilot (a platform where customers can share reviews), unless the web shop has been registered recently.

In a situation where we have a webshop that pretends to be old, has a page on Trustpilot and does not have many reviews, as shown in Figure 3.4, there are three future argumentation theories: $T_1 = T$, $T_2 = T \cup \{$*new_registration*$\}$ and $T_3 = T \cup \{\neg$*new_registration*$\}$. In each of these future AT's, there is an undefeated argument for suspect: for the future AT's T_1 and T_3, the argument [*trustpilot* $\Rightarrow$ *reviews_expected*], $\neg$*many_reviews* $\Rightarrow$ *suspect* is undefeated (as the argument for $\neg$*reviews_expected* does not exist); for T_2 the argument *pretends_old*, *new_registration* $\Rightarrow$ *suspect* is undefeated. Consequently, there is an argument for *suspect* in the grounded extension of every future AT w.r.t. T. Then *suspect* is stable-defended w.r.t. T and $\mathcal{Q}$ by Definition 3.12 of stability.

Now consider the IAF that corresponds to T and $\mathcal{Q}$: $\mathcal{I} = IAF(T, Q) = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \emptyset \rangle$. This IAF is illustrated in Figure 3.5. The only argument for *suspect* in $\mathcal{A}$ is [*trustpilot* $\Rightarrow$ *reviews_expected*], $\neg$*many_reviews* $\Rightarrow$ *suspect*. This argument can still be defeated by the uncertain argument *new_registration* $\Rightarrow \neg$*reviews_expected*. In a completion where this argument is added, the argument for *suspect* ([*trustpilot* $\Rightarrow$ *reviews_expected*], $\neg$*many_reviews* $\Rightarrow$ *suspect*) is no longer in the grounded extension. Given that $\mathcal{I}$ has a completion in which the argument for *suspect* is not in the

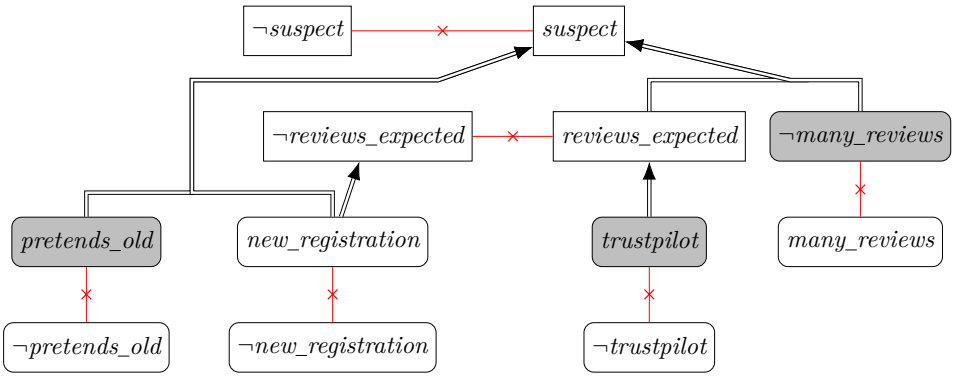


Figure 3.4: The literal *suspect* is stable-defended, so no queryable is defended-relevant for this literal.

grounded extension, by Definition 2.7 this argument cannot be stable-GR-sceptical-IN. Finally note that there is an uncertain argument for *suspect*, but this argument is not stable-GR-sceptical-IN either as it is uncertain.

To conclude, if there is a literal l that is stable-defended w.r.t. some argumentation theory T and set of queryables $\mathcal{Q}$, there is not always some argument for l that is stable-GR-sceptical-IN w.r.t. $IAF(T, \mathcal{Q})$. The issue in this case is that IAFs do not model the phenomenon in incomplete ASPIC$^+$ that the addition of an uncertain argument A may require the addition or removal of supplementary arguments, in particular those arguments that can be constructed from the premises of A and the premises of arguments that were already in $\mathcal{A}$, as well as the removal of uncertain arguments with inconsistent premises.

3.4.2. Structured and abstract relevance

In this section we will show that GR-sceptical-IN-relevance on the abstract level does not imply defended-relevance on the structured level.

Counterexample 3.2. Consider the argumentation theory $T = (AS, \mathcal{K})$ and set of queryables $\mathcal{Q}$ illustrated in Figure 3.6. This AT models a dilemma on traveling. Suppose that you are planning some journey to a location far away, on a weekday in April. You can go by train or by bike, but not at the same time: $by\_train \in \overline{by\_bike}$ and $by\_bike \in \overline{by\_train}$. Usually, you take the train for traveling far away, provided that the trains are running. On the other hand, you generally take the bike as long as there is no snow. You do not know yet whether there will be snow during your journey. But you do know that if it snows, usually all trains are canceled. At this point, there is an argument for traveling by train and there is an equally strong argument for traveling by bike. Therefore, the claim *by_train* is blocked w.r.t. T. You want to investigate whether *by_train* can still become stable-defended (so you can buy a train ticket) and wonder whether it makes sense to wait for the weather forecast.

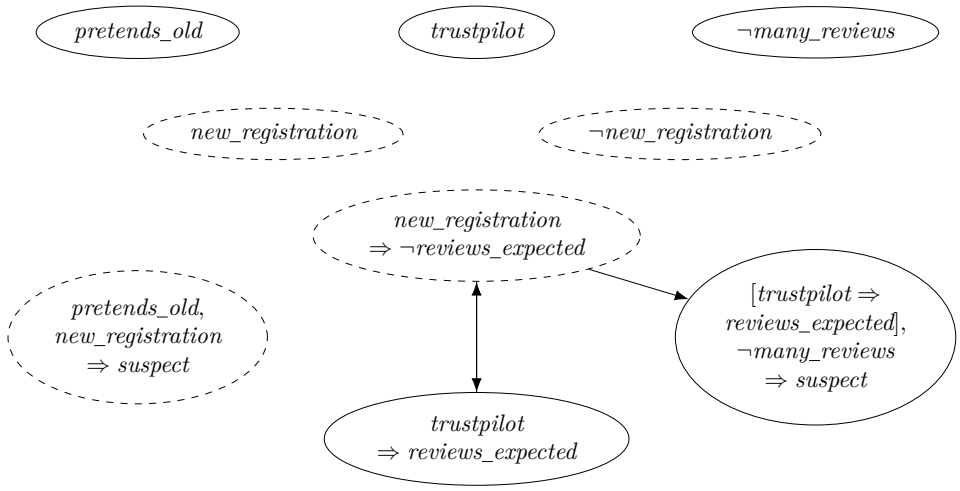


Figure 3.5: An incomplete argumentation framework, where *suspect* is not stable-GR-sceptical-IN. The removal of *new_registration* $\Rightarrow \neg$*reviews_expected* is GR-sceptical-IN-relevant.

The AT T has three future argumentation theories: $T_1 = T$, $T_2 = (AS, \mathcal{K} \cup \{snow\})$ and $T_3 = (AS, \mathcal{K} \cup \{\neg snow\})$. In none of these argumentation theories, *by_train* is defended: in T_1 and T_3, the only argument for *by_train* (i.e. $[weekday \Rightarrow trains\_running], far \Rightarrow by\_train$) is defeated by an argument for *by_bike* ($[april \Rightarrow \neg snow] \Rightarrow by\_bike$, and in case of T_3 also $\neg snow \Rightarrow by\_bike$); in T_2, the only argument for *by_train* is defeated by the argument $snow \Rightarrow \neg trains\_running$. This implies that none of the queryables is defended-relevant for *by_train* w.r.t. T and $\mathcal{Q}$.

Now let $\mathcal{I} = IAF(T, \mathcal{Q}) = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \emptyset \rangle$, as illustrated in Figure 3.7. Note that there is no defeat depicted between observation-based uncertain arguments such as *snow* and $\neg$*snow* as there is no T' such that $T \sqsubseteq_Q T'$ and these arguments are present in T'. It holds that $\mathcal{I}$ has many different partial completions, including $\mathcal{I}' = \langle \mathcal{A} \cup \{snow\}, \emptyset, \mathcal{C}', \emptyset \rangle$. In fact, $\mathcal{I}'$ is a minimal stable-GR-sceptical-IN partial completion for $[weekday \Rightarrow trains\_running], far \Rightarrow by\_train$ w.r.t. $\mathcal{I}$. Then by Definition 2.11, the addition of *snow* is GR-sceptical-IN-relevant for $[weekday \Rightarrow trains\_running], far \Rightarrow by\_train$ w.r.t. $\mathcal{I}$. This shows that addition of $snow \in \mathcal{A}^?$ is GR-sceptical-IN-relevant for $[weekday \Rightarrow trains\_running], far \Rightarrow by\_train$, but none of the premises of *snow* is defended-relevant for the conclusion literal *by_train* w.r.t. T and $\mathcal{Q}$. The reason for this is that there is no T' such that $T \sqsubseteq_Q T'$ and $IAF(T', \mathcal{Q}) = \mathcal{I}'$: *snow* should only be an argument in argumentation frameworks that also include $snow \Rightarrow \neg trains\_running$, as they have exactly the same premises.

To conclude this section, we observed that it can be problematic to study the notions of stability and relevance only on the abstract level. This observation is in line with earlier studies on the relation between abstract and structured approaches in argumentation (Prakken and Winter, 2018), in particular in the resolution of attacks (Modgil and Prakken, 2012), strength of arguments (Prakken, 2022), en-

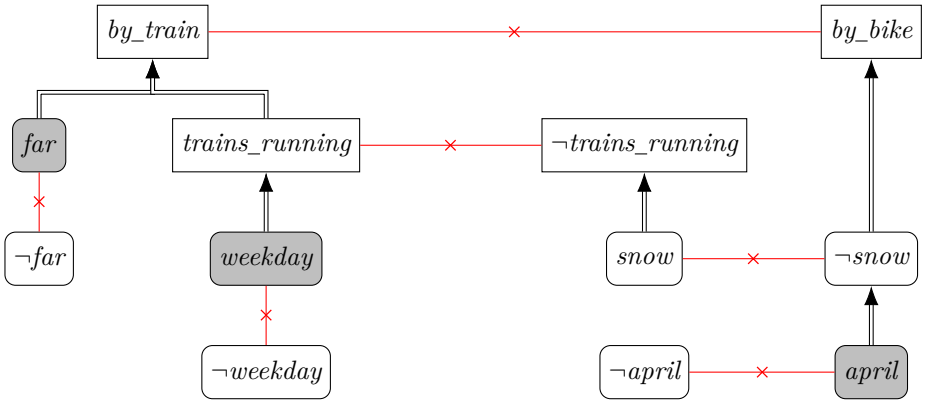


Figure 3.6: The claim *by_train* is stable-blocked, so no queryable is defended-relevant for *by_train*.

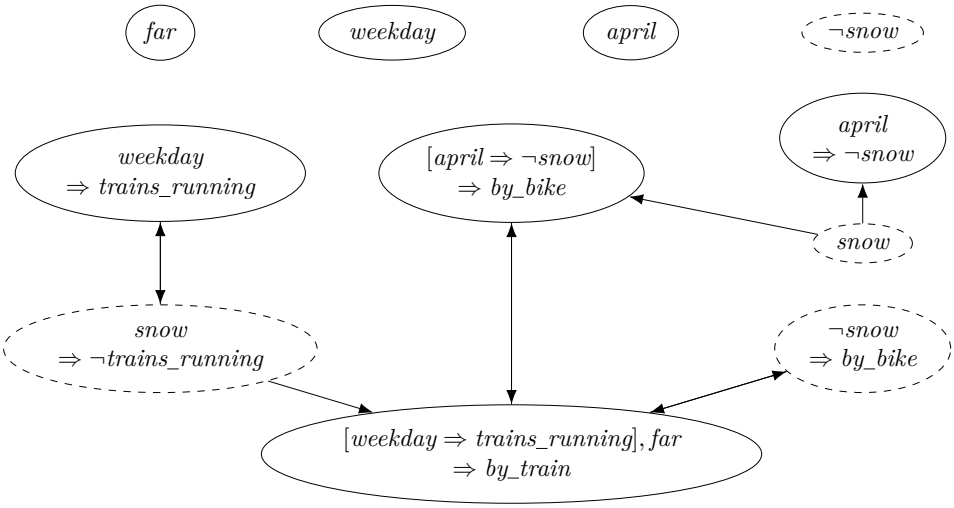


Figure 3.7: Adding *snow* is defended-relevant for *by_train* in the IAF.

forcement (Borg and Bex, 2021b) and strong equivalence (Rapberger and Ulbricht, 2023), specification of expansions (Prakken, 2023) and forgetting arguments from a knowledge base (Berthold et al., 2023b). We therefore argue that the notions of stability and relevance should be studied under the assumption that completions can only be induced by updates on the structured level (that is, argumentation frameworks corresponding to future argumentation theories).

3.5. Computing justification in polynomial time

Due to the incompatibilities between abstract and structured notions of stability and relevance that we identified in the previous section, we cannot directly reuse existing complexity results for the problems on the abstract level from Chapter 2 to determine the complexity classes of the problems on the structured level. Instead, we first study the problem of determining the justification status of a literal. In this section, we show that the justification status of a literal is decidable in polynomial time, when assuming grounded semantics and last-link ordering for the ASPIC$^+$ fragment discussed in this chapter.

3.5.1. Rephrasing grounded semantics

As a first step, we prove that the justification problem can be computed in polynomial time. Note that this is not immediately clear from the definitions, since Definition 2.1 specifies the grounded extension in terms of arguments and an AT can have a number of arguments that is not bounded polynomially by the AT size. An example that exhibits an exponential number of arguments was earlier provided by (Strass et al., 2019). To establish polynomial-time decidability, an alternative approach is required.

Recent work on assumption-based argumentation (Lehtonen et al., 2021) and specific fragments of ASPIC$^+$ for various semantics (Lehtonen et al., 2020; Lehtonen et al., 2022) indicates that the explicit construction of arguments can be avoided by rephrasing argumentation semantics based on defeasible elements and directly drawing conclusions on the level of these rephrasings. In this section, we propose a similar rephrasing for grounded semantics. We reformulate the grounded extension in terms of sets of *rules* rather than sets of *arguments*.

As a first step towards this goal, we introduce the notions of applicability, defeats and defense with respect to sets of rules. A rule r is applicable by D if there is an argument that is based on $D \cup \{r\}$ and uses r.

Definition 3.16 (Applicable by rule set)**.** Given an AT $T = (AS, \mathcal{K})$ and a set of defeasible rules $D \subseteq \mathcal{R}$. A rule $r \in \mathcal{R}$ is applicable by D if and only if there is an argument A based on $D \cup \{r\}$ with $r \in \texttt{defrules}(A)$.

Example 3.13. Recall the AT $T = (AS, \mathcal{K})$ in the web shop domain from Example 3.1 with the argumentation system $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$. The set $\mathcal{R}$ contains four defeasible rules: (*similar_url* $\Rightarrow$ *typosquatting*), (*typosquatting* $\Rightarrow$ *deception*), (*too_cheap* $\Rightarrow$ *deception*) and (*trusted* $\Rightarrow \neg$*deception*). Given that $\mathcal{K}$ contains only

similar_url, $Arg_T(\mathcal{R})$ consists of three arguments (for *similar_url*, for *typosquatting* and for *deception*). Hence only *similar_url* $\Rightarrow$ *typosquatting* and *typosquatting* $\Rightarrow$ *deception* are applicable by $\mathcal{R}$.

Now consider $D = \{(\textit{similar\_url} \Rightarrow \textit{typosquatting})\} \subseteq \mathcal{R}$. Then $Arg_T(D)$ only contains the arguments for *similar_url* and for *typosquatting*. The rules that are applicable by D are *similar_url* $\Rightarrow$ *typosquatting* and *typosquatting* $\Rightarrow$ *deception*.

For $D' = \{(\textit{typosquatting} \Rightarrow \textit{deception}), (\textit{too\_cheap} \Rightarrow \textit{deception})\} \subseteq \mathcal{R}$, $Arg_T(D')$ only contains the observation-based argument for *similar_url*. Consequently, the only rule that is applicable by D' is *similar_url* $\Rightarrow$ *typosquatting*.

Turning to the general case with preferences, we define defeat in terms of rule sets (in analogy with Definition 3.6 for defeat of arguments).

Definition 3.17 (Defeat by rule set)**.** Given an AT $T = (AS, \mathcal{K})$, a set of defeasible rules $D \subseteq \mathcal{R}$ and a rule $r \in \mathcal{R}$, we say that D defeats r if and only if

- there is a l in $\mathcal{K}$ such that $l \in \overline{\texttt{cons}(r)}$ or
- there is a r' in D such that $\texttt{cons}(r') \in \overline{\texttt{cons}(r)}$, r' is applicable by D and $r' \not\leq r$.

In words, a set of defeasible rules D defeats a single rule r if (i) the knowledge base contains an axiom that is contradictory to the consequent of r, or (ii) there is a rule $r' \in D$ applicable by D that has as consequent a contradictory of the consequent of r and is not strictly less preferred to r. Intuitively, in the former case, an observation-based argument directly defeats any argument with r as its top rule, while in the latter case, a rule-based argument with r as top rule is defeated by a rule-based argument that has as its defeasible rules only rules in D and r' as its top rule. Then the rebut succeeds as a defeat.

Example 3.14. Continuing Example 3.4, consider the AT $T'' = (AS, \mathcal{K} \cup \{\textit{trusted}\})$. Then $D = \{(\textit{similar\_url} \Rightarrow \textit{typosquatting}), (\textit{typosquatting} \Rightarrow \textit{deception})\}$ defeats the rule *trusted* $\Rightarrow \neg$*deception*, since the consequence of one of the rules in D and the latter rule are contradictory to each other. The set $D' = \{(\textit{trusted} \Rightarrow \neg\textit{deception})\}$ defeats *typosquatting* $\Rightarrow$ *deception*. Oh the other hand, if *typosquatting* $\Rightarrow$ *deception* is strictly preferred to *trusted* $\Rightarrow \neg$*deception* then D' does not defeat *typosquatting* $\Rightarrow$ *deception*.

The preceding example suggests a correspondence between defeats by rule sets and defeats by arguments: we have seen that the rule set $D = \{(\textit{similar\_url} \Rightarrow \textit{typosquatting}), (\textit{typosquatting} \Rightarrow \textit{deception})\}$ defeats the rule *trusted* $\Rightarrow \neg$*deception* and, correspondingly, the argument [*similar_url* $\Rightarrow$ *typosquatting*] $\Rightarrow$ *deception* (which is in $Arg_T(D)$) defeats the argument *trusted* $\Rightarrow \neg$*deception* (which has the rule *trusted* $\Rightarrow \neg$*deception* in its defeasible rules). This correspondence holds in general, as we prove next.

Proposition 3.2. *Given an AT $T = (AS, \mathcal{K})$, the corresponding AF $\langle \mathcal{A}, \mathcal{C} \rangle$, and a set of defeasible rules D and an argument $A \in \mathcal{A}$, it holds that at least one $B \in Arg_T(D)$ defeats A if and only if D defeats a rule $r \in \texttt{defrules}(A)$.*

Proof. We prove this in both directions:

- *From left to right*: suppose that there is some argument B in $Arg_T(D)$ that defeats A on $A' \in \texttt{sub}(A)$. Since A' is defeated, it cannot be observation-based and therefore must be rule-based. Let r be the top rule of A'. Then B is either observation-based or rule-based, having some $r' \in D$ as its top rule. If B is observation-based then $\texttt{conc}(B) \in \mathcal{K}$ and $\texttt{conc}(B) \in \overline{\texttt{cons}(r)}$. This implies that D defeats r. Otherwise B has some rule $r' \in D$ as its top rule. In that case, there must be an argument with top rule r' in $Arg_T(D) = Arg_T(D \cup \{r'\})$, so r' is applicable by D. Since B defeats A', we have $\texttt{cons}(r') \in \texttt{cons}(r)$ and $r' \not\leq r$. This implies that D defeats r. Finally, since r is the top rule of A' and A' is a subargument of A, we have $r \in \texttt{defrules}(A)$.
- *From right to left*: assume that $D \subseteq \mathcal{R}$ defeats some $r \in \texttt{defrules}(A)$. Since $r \in \texttt{defrules}(A)$, there must be some subargument of A with top rule r. Let $A' \in \texttt{sub}(A)$ be such a subargument. Given that D defeats r, either (i) there is some $l \in \overline{\texttt{cons}(r)}$ in $\mathcal{K}$ or (ii) there is a $r' \in D$ whose consequent is a contradictory of r, r' is applicable by D and $r' \not\leq r$. In case (i), there is an observation-based argument in $Arg_T(D)$ that defeats A on A'. In case (ii), since r' is applicable by D, there is an argument B in $Arg_T(D)$ with top rule r'. The argument B is not less preferred than A under the last-link principle. Hence B defeats A' and thus also A. □

Analogously to defeat, we introduce a notion of defense in terms of rule sets. Recall from Definition 2.1 that an argument A is defended by a set of arguments S if each argument B defeating A is defeated by some argument in S. In other words, each argument that is not defeated by any argument in S must not defeat A. We rephrase this aspect into defense on rule sets.

Definition 3.18 (Defense by rule set)**.** Given an $T = (AS, \mathcal{K})$, a set of defeasible rules $D \subseteq \mathcal{R}$ and a rule $r \in \mathcal{R}$, let U be the set of rules in $\mathcal{R}$ that are not defeated by D. Then r is defended by D if and only if U does not defeat r.

Example 3.15. Consider the AT $T = (AS, \mathcal{K})$ where $AS = (\mathcal{L}, \overline{\ }, \mathcal{R}, \leq)$ and:

- $\mathcal{L} = \{q_1, \neg q_2, q_2, \neg q_2, q_3, \neg q_3, a, b, c\}$;
- For each $q \in \{q_1, q_2, q_3\}$: $q \in \overline{\neg q}$ and $\neg q \in \overline{q}$; $\overline{a} = \{b\}$, $\overline{b} = \{a, c\}$ and $\overline{c} = \{b\}$.
- $\mathcal{R} = \{r_1 : (q_1 \Rightarrow a), r_2 : (q_2 \Rightarrow b), r_3 : (q_3 \Rightarrow c)\}$.
- $q_1 \Rightarrow a > q_2 \Rightarrow b$.
- $\mathcal{K} = \{q_1, q_2, q_3\}$.

This AT and the corresponding argumentation framework are illustrated in Figure 3.8.

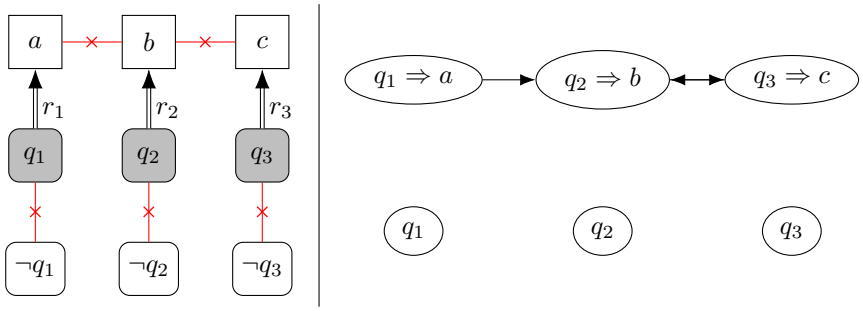


Figure 3.8: Illustration of the argumentation theory used in Example 3.15 and the corresponding argumentation framework. The rule preferences are such that $r_1 > r_2$.

- As a first example, let $D = \emptyset$. Then the set U of rules not defeated by D is $\{r_1, r_2, r_3\}$. Since $\{r_1, r_2, r_3\}$ does not defeat r_1, the rule r_1 is defended by D. The other rules are not defended by D.
- Now let $D' = \{r_1\}$. The set of rules not defeated by D' is $U' = \{r_1, r_3\}$. U' does not defeat r_1 and r_3. Thus r_1 and r_3 are defended by D'.
- Similarly, for $D'' = \{r_3\}$, the set of rules undefeated by D'' is $U'' = \{r_1, r_3\}$. As U'' does not defeat r_1 and r_3, both r_1 and r_3 are defended by D''.

Note that there is a correspondence between defense by arguments and defense by rule sets. For example, D' defends r_1 and r_3. Also the set of all arguments in $Arg_T(D')$ defends all arguments that can be constructed from r_1 and r_3 (i.e. $q_1 \Rightarrow a$ and $q_3 \Rightarrow c$).

Next, we formally prove that this correspondence between defense by arguments (Definition 2.1) and defense by rule sets (Definition 3.18) holds in general.

Proposition 3.3. *Given an $T = (AS, \mathcal{K})$, the corresponding AF $\langle \mathcal{A}, \mathcal{C} \rangle$, and a set of defeasible rules D, and an argument $A \in \mathcal{A}$, it holds that $Arg_T(D)$ defends A if and only if D defends every rule $r \in$ `defrules`(A).*

Proof. We prove this in both directions:

- *From left to right*: suppose that $Arg_T(D)$ defends A and let r be an arbitrary rule in `defrules`(A). Suppose, towards a contradiction, that r is not defended by D. Then by Definition 3.18 the set U, consisting of all rules in $\mathcal{R}$ that are not defeated by D, defeats r. Then by Proposition 3.2 there is some $B \in Arg_T(U)$ that defeats A. Given that $B \in Arg_T(U)$, the argument B can be constructed using $\mathcal{K}$ and U. Due to the way U is constructed (consisting only of rules not defeated by D), by Proposition 3.2 there is no argument $C \in Arg_T(D)$ that defeats B. Then by Definition 2.1, $Arg_T(D)$ does not defend A. From this contradiction it follows that r is defended by D.
- *From right to left*: assume that D defends all $r \in$ `defrules`(A). Suppose, towards a contradiction, that $Arg_T(D)$ does not defend A. Then by Defini-

tion 2.1 there is some $B \in \mathcal{A}$ that defeats A, while no argument in $Arg_T(D)$ defeats B. By Proposition 3.2, this implies that each rule $r' \in \texttt{defrules}(B)$ is not defeated by D. Hence $\texttt{defrules}(B) \subseteq U$, where U is the set of all rules in $\mathcal{R}$ not defeated by D. This implies that $B \in Arg_T(U)$. As $B \in Arg_T(U)$ defeats A, by Proposition 3.2 U defeats a rule $r \in \texttt{defrules}(A)$. By Definition 3.18, r is not defended by D, which contradicts our assumption. Therefore $Arg_T(D)$ must defend A. □

Based on the notion of defense for rule sets, we define a counterpart of Dung's fundamental lemma (Dung, 1995, Lemma 10) for rule sets (instead of argument sets). Before we do so in Proposition 3.4, we first prove that the defense relation is monotonous in the following lemma.

Lemma 3.2 (Monotonicity of defense)**.** *Let $T = (AS, \mathcal{K})$ be an AT where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$. For each $R \subseteq \mathcal{R}$ and $r \in \mathcal{R}$, we have that if R defends r, then each R' such that $R \subseteq R' \subseteq \mathcal{R}$ defends r.*

Proof. If R defends r, then by Definition 3.18 the set of rules U in $\mathcal{R}$ that is not defeated by R does not defeat r. Let R' be an arbitrary rule set such that $R \subseteq R' \subseteq \mathcal{R}$ and let U' be the set of all rules in $\mathcal{R}$ that is not defeated by R'. Then $U' \subseteq U$. Since U does not defeat r, U' cannot defeat r either. □

Proposition 3.4. *Let $T = (AS, \mathcal{K})$ be an AT where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and let $R \subseteq \mathcal{R}$ be a set of defeasible rules such that (i) each rule in R is applicable by R and (ii) $Arg_T(R)$ is admissible. Furthermore, let r and r' be rules in $\mathcal{R}$ defended by R. Then (1) $Arg_T(R \cup \{r\})$ is admissible and (2) $R \cup \{r\}$ defends r'.*

Proof. Suppose that each rule in R is applicable by R, $Arg_T(R)$ is admissible and r and r' are rules in $\mathcal{R}$ defended by R.

1. First, we show that $Arg_T(R \cup \{r\})$ is admissible: it defends itself and is conflict-free.

   - By Definition 2.1 of admissibility, $Arg_T(R)$ defends each argument in $Arg_T(R)$. Then by Proposition 3.3, R defends each rule in R. Given that r is defended by R as well, R defends each rule in $R \cup \{r\}$. By monotonicity of defense (Lemma 3.2), $R \cup \{r\}$ defends each rule in $R \cup \{r\}$. Then for each argument A in $Arg_T(R \cup \{r\})$ it holds that each rule in $\texttt{defrules}(A)$ is defended by $R \cup \{r\}$, which by Proposition 3.3 implies that $Arg_T(R \cup \{r\})$ defends itself.
   - To prove admissibility, what remains to be shown is conflict-freeness. Towards a contradiction, suppose that $Arg_T(R \cup \{r\})$ is not conflict-free. Then there are A, B in $Arg_T(R \cup \{r\})$ such that A defeats B. Given that R defends each rule in $R \cup \{r\}$, by Proposition 3.3 $Arg_T(R)$ it holds that defends $Arg_T(R \cup \{r\})$. As $B \in Arg_T(R \cup \{r\})$ is defeated (by A), there is an argument C in $Arg_T(R)$ that defeats A. However, $Arg_T(R)$ defends

$Arg_T(R \cup \{r\})$ and thence there is an argument D in $Arg_T(R)$ defeating C. That implies that C and D in $Arg_T(R)$ defeat each other, which contradicts conflict-freeness of the admissible set $Arg_T(R)$. It follows that $Arg_T(R \cup \{r\})$ is conflict-free.

Therefore $Arg_T(R \cup \{r\})$ is admissible.

2. From the assumption that R defends r' and Lemma 3.2 it directly follows that $R \cup \{r\}$ defends r'. □

Towards defining the grounded extension without computing arguments, we define a characteristic function for rule sets, in analogy with the classical version of the characteristic function for AFs (Dung, 1995, Definition 16).

Definition 3.19. Let $T = (AS, \mathcal{K})$ be an AT where $AS = (\mathcal{L}, ^-, \mathcal{R}, \leq)$ and $D \subseteq \mathcal{R}$ a rule set. Then $def_T(D) = \{r \in \mathcal{R} \mid r \text{ is applicable and defended by } D\}$.

For $i > 0$, we denote i applications of def_T on $\emptyset$ by $def_T^i(\emptyset)$ and define $def_T^0(\emptyset) = \emptyset$.

By Proposition 3.5, iterating the characteristic function starting from the empty set gives the grounded extension.

Proposition 3.5. *Given an AT $T = (AS, \mathcal{K})$ where $AS = (\mathcal{L}, ^-, \mathcal{R}, \leq)$, let C be the least fixed point of def_T. Then $G(T) = Arg_T(C)$.*

Proof. First we show that def_T has a unique least fixed point. The def_T function is based on rule applicability and defense by Definition 3.19. Since defense is $\subseteq$-monotonic (recall Lemma 3.2) and the notion of applicability (i.e., for any $D \subseteq D' \subseteq \mathcal{R}$, if a rule is applicable by D, then the rule is also applicable by D'), we have that def_T is $\subseteq$-monotonic. Hence def_T has a unique least fixed point. The least fixed point is obtained by iterating def_T, starting with the empty set.

Next, we show by induction that $Arg_T(def_T^i(\emptyset))$ is admissible for any $i \in \mathbb{N}$ and thus $Arg_T(C)$ is admissible. For the base case $i = 0$, we have $def_T^0(\emptyset) = \emptyset$. Then $Arg_T(\emptyset)$ consists of knowledge-based arguments which cannot be defeated. Hence $Arg_T(def_T^0(\emptyset))$ is admissible by Definition 2.1. Now, we assume as the induction hypothesis that $Arg_T(def_T^k(\emptyset))$ is admissible for some $k \in \mathbb{N}$. By Definition 3.19, $def_T^{k+1}(\emptyset)$ consists of all rules in $\mathcal{R}$ that are applicable and defended by $def_T^k(\emptyset)$. By Proposition 3.4, $def_T^{k+1}(\emptyset)$ is admissible. Since $\mathcal{R}$ is finite, we reach a fixed point C at some $i \in \mathbb{N}$. This implies that $Arg_T(C)$ is admissible.

Next, we show that $Arg_T(C)$ is complete. We will prove this by induction on the height of defended arguments, where the height $h(A)$ of any argument $A \in \mathcal{A}$ is defined as follows. If A is observation-based, then $h(A) = 0$. Otherwise, A has the form $A_1, \ldots, A_m \Rightarrow \texttt{conc}(A)$ and $h(A) = 1 + \max(h(A_1), \ldots, h(A_m))$.

For the base case, each argument $A \in \mathcal{A}$ with $h(A) = 0$ defended by $Arg_T(C)$ must be in $Arg_T(C)$; as $\texttt{defrules}(A) = \emptyset$ and A cannot be defeated, we have $A \in Arg_T(\emptyset)$ and hence $A \in Arg_T(C)$. Now assume as the induction hypothesis that, for each $k \leq n$, every argument $A \in \mathcal{A}$ with $h(A) = k$ defended by $Arg_T(C)$ is in $Arg_T(C)$.

Let $A = A_1, \ldots, A_m \Rightarrow \texttt{conc}(A)$ be an arbitrary argument in $\mathcal{A}$ with $h(A) = n + 1$ defended by $Arg_T(C)$. As $Arg_T(C)$ defends A, by Proposition 3.3 C defends every rule in $\texttt{defrules}(A)$. Furthermore, each rule in $\texttt{defrules}(A)$ is applicable by C, since for each $A' \in \{A_1, \ldots, A_m\}$, A' is in $Arg_T(C)$ by the induction hypothesis. By Definition 3.19 of def_T, each rule in $\texttt{defrules}(A')$ is applicable by C. By Definition 3.16 of applicability, $\texttt{top-rule}(A)$ is also applicable by C. As all rules in $\texttt{defrules}(A)$ are applicable and defended by C, $def_T(C) = C$ contains all rules in $\texttt{defrules}(A)$, and hence $A \in Arg_T(C)$. Since every argument in $\mathcal{A}$ has finite height (by Definition 3.4), $Arg_T(C)$ is complete.

Finally, suppose towards a contradiction that $Arg_T(C)$ is not grounded. As $Arg_T(C)$ is complete, we have $G(T) \subset Arg_T(C)$ and hence there is an argument in $Arg_T(C)$ that is not in $G(T)$. Let j be the first iteration of def_T for which there is an argument $B \in Arg_T(def_T^j(\emptyset))$, $B \notin G(T)$, and each argument in $Arg_T(def_T^{j-1}(\emptyset))$ is in $G(T)$. By definition of def_T, every rule in $def_T^j(\emptyset)$ is applicable and defended by $def_T^{j-1}(\emptyset)$. Hence every rule in $\texttt{defrules}(B)$ is applicable and defended by $def_T^{j-1}(\emptyset)$. Then, by Proposition 3.3, $Arg_T(def_T^{j-1}(\emptyset))$ defends B. As each argument in $Arg_T(def_T^{j-1}(\emptyset))$ is in $G(T)$, we have that $G(T)$ defends B, which is a contradiction. We conclude that $Arg_T(C) = G(T)$. □

Example 3.16. Consider AS from Example 3.1, AT $T = (AS, \{$*similar_url*, *trusted*, *too_cheap*$\})$ and let $($*trusted* $\Rightarrow \neg$*deception*$) <$ *(similar_url* $\Rightarrow$ *deception*$)$. Then the least fixed point of def_T is $\{($*similar_url* $\Rightarrow$ *typosquatting*$), ($*too_cheap* $\Rightarrow$ *deception*$), ($*typosquatting* $\Rightarrow$ *deception*$)\}$.

1. $i = 1$: no rules are defeated by $\emptyset$, and hence the set of undefeated rules is $U = \mathcal{R}$. Firstly, $($*similar_url* $\Rightarrow$ *typosquatting*$)$ is applicable by U and, as $\neg$*typosquatting* is not derivable by U, defended by $\emptyset$. Secondly, $\neg$*deception* is derivable from U only by $($*trusted* $\Rightarrow \neg$*deception*$)$, which is less preferred than *(too_cheap* $\Rightarrow$ *deception*$)$, the latter of which is thus defended (and applicable) by $\emptyset$. Thus we have $def_T^1(\emptyset) = def_T(\emptyset) = \{($*similar_url* $\Rightarrow$ *typosquatting*$), ($*too_cheap* $\Rightarrow$ *deception*$)\}$.
2. $i = 2$: $U = \mathcal{R} \setminus \{($*trusted* $\Rightarrow \neg$*deception*$)\}$ and thus $\neg$*deception* is not derivable from U and hence $($*typosquatting* $\Rightarrow$ *deception*$)$ is not defeated by U. Thus we have $def_T^2(\emptyset) = def_T^1(\emptyset) \cup \{($*typosquatting* $\Rightarrow$ *deception*$)\}$.
3. $def_T^3(\emptyset) = def_T^2(\emptyset)$; terminate.

The arguments that can be constructed based on these rules (deriving *deception*, *typosquatting*, *too_cheap* and *similar_url*) constitute the grounded extension.

3.5.2. Complexity of justification

We are ready to establish that justification is polynomial-time computable. We show that the least fixed point of def_T can be computed in $|\mathcal{R}|/2$ iterations, starting with the empty set of rules. At the fixed point we conclude a rule to be defended or defeated.

Proposition 3.6. *Given an AT $T = (AS, \mathcal{K})$ with $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$, the least fixed point of def_T is reached in at most $|\mathcal{R}|/2$ iterations.*

Proof. Consider for each $i \in \mathbb{N}$ the sets $S^i = def_T^i(\emptyset)$ and the set of rules $D^i = \{r \in \mathcal{R} \mid r \text{ is defeated by } S^i\}$ defeated by S^i. Note that $S^i \subseteq S^{i+1}$ and $D^i \subseteq D^{i+1}$. Furthermore, S^i and D^j are disjoint for all i, j, since $Arg(S^i)$ is conflict-free by Proposition 3.4. Moreover, S^i defending a rule that S^{i-1} does not defend requires that S^i defeats some rule that S^{i-1} does not defeat. This implies that if $D^i = D^{i-1}$, then S^i is the least fixed point of $def_T^i(\emptyset)$. Thus on every iteration either at least one element is added to both S^i and D^i or a least fixed point is reached. Therefore, when $i = |\mathcal{R}|/2$ is reached, every $r \in \mathcal{R}$ is in either S^i or D^i, and hence a least fixed point has been reached. □

The least fixed point of def_T allows for directly inferring the justification status of a literal.

Proposition 3.7. *Assume an AT $T = (AS, \mathcal{K})$ with $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and let C be the least fixed point of def_T. A literal $l \in \mathcal{L}$ is*

(i) unsatisfiable iff there is no argument A for l in Arg_T,

(ii) defended iff there is an argument for l in $Arg_T(C)$,

(iii) out iff there is an argument for l in Arg_T and no argument for l in $Arg_T(U)$, where U is the set of rules not defeated by C and

(iv) blocked otherwise.

Proof. We prove each of these items individually.

(i) The *unsatisfiable* case corresponds directly to Definition 3.9 of justification.

(ii) A literal l is *defended* iff there is an argument for l in the grounded extension $G(T)$ (Definition 3.9). By Proposition 3.5, $Arg_T(C) = G(T)$, so l is defended iff there is an argument for l based on C. This is the case iff there is an argument for l in $Arg_T(C)$.

(iii) A literal l is *out* iff (a) there is an argument for l in Arg_T and (b) each argument for l in Arg_T is defeated by some argument in $G(T)$ (Definition 3.9), where $G(T) = Arg_T(C)$ (Proposition 3.5). For any argument A for l in Arg_T, A is defeated by $Arg_T(C)$ iff C defeats some rule r in $\texttt{defrules}(A)$ (Proposition 3.2). Hence all arguments for l in Arg_T are defeated by $Arg_T(C)$ iff it is not possible to construct an argument for l based on only rules not defeated by C (i.e., based on U), which holds if and only if there is no argument for l in $Arg_T(U)$.

(iv) The justification statuses in Definition 3.9 are mutually exclusive and complementary, so a literal is *blocked* iff it is not unsatisfiable, defended or out. By cases (i)–(iii), this is the case iff there is some argument for l in $Arg_T(U)$ and there is no argument for l in $Arg_T(C)$. □

In summary, it holds that we can infer the justification status of a literal in polynomial time for each of the four justification statuses.

Theorem 3.1. *Let $T = (AS, \mathcal{K})$ be an AT with $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and let j be the unsatisfiable, defended, out, or blocked justification status. The problem of deciding whether a literal $l \in \mathcal{L}$ has justification status j w.r.t. T is in P.*

Proof. The following operations can be done in polynomial time.

1. Given $R \subseteq \mathcal{R}$ and $l \in \mathcal{L}$, decide if there is some argument for l in $Arg_T(R)$. This is achieved by starting from the set $S = \mathcal{K}$ and adding conclusions of rules for which all antecedents are in S, until a fixed point is reached.
2. Compute the least fixed point of def_T. This procedure takes polynomial time as the def_T operator is applied no more than $|\mathcal{R}|/2$ times before reaching a fixed point (Proposition 3.6).
3. Given a set of rules $R' \subseteq \mathcal{R}$, compute the set of rules that are not defeated by R'.

These observations and Proposition 3.7 together imply that deciding whether a given $l \in \mathcal{L}$ has justification status j is achieved as follows.

- j = unsatisfiable: Decide in polynomial time (by Item 1) whether there is some argument for l in $Arg_T(\mathcal{R})$. If this is not the case, then l is unsatisfiable w.r.t. T.
- j = defended: Compute the least fixed point of def_T in polynomial time (by Item 2) and let this be C. Then decide in polynomial time (by Item 1) whether there is some argument for l in $Arg_T(C)$. It holds that l is defended w.r.t. T if only if such an argument exists (Proposition 3.7 Item 2).
- j = out: Compute the least fixed point of def_T in polynomial time (by Item 2) and let this be C. Compute $U = \{r \in \mathcal{R} \mid r \text{ is not defeated by } C\}$, which can be done in polynomial time (by Item 3). Then decide in polynomial time (by Item 1) whether there is an argument for l in $Arg_T(\mathcal{R})$ and whether there is no argument for l in $Arg_T(U)$. By Item 3 of Proposition 3.7, this holds if and only if l is out w.r.t. T.
- If j = blocked: By the above, check in polynomial time if j cannot be unsatisfiable, defended or out. This is the case if and only if j is blocked w.r.t. T (Item 4 of Proposition 3.7). □

3.6. Complexity of stability and relevance

In the previous section, we showed that the justification problem can be solved in polynomial time. This result lays the ground for the complexity results presented in this section. In Section 3.6.1, we will show that the stability problems for the four justification statuses are CoNP-complete. We then establish Σ_2^P-completeness for deciding relevance in Section 3.6.2.

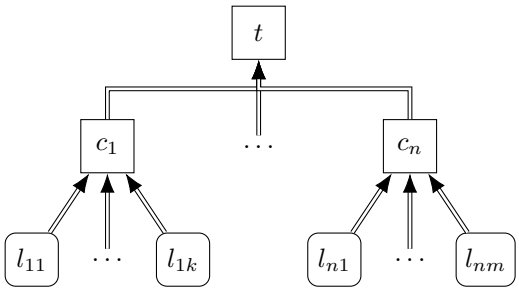


Figure 3.9: Reduction UNSAT.

3.6.1. Stability complexity

Using results from justification complexity, proven in Theorem 3.1, in this section, we discuss the complexity of the stability problem.

Theorem 3.2 (Complexity of deciding stability)**.** *Deciding whether a literal is stable-j in an AT is CoNP-complete for each justification status j in $\{$unsatisfiable, defended, out, blocked$\}$. Hardness holds even without preferences.*

Proof sketch. CoNP-hardness of unsatisfiable-stability can be shown by a polynomial-time reduction from the CoNP-complete problem UNSAT. Given a CNF formula $\phi = (l_{11} \vee \ldots \vee l_{1k}) \wedge \ldots \wedge (l_{n1} \vee \ldots \vee l_{nm})$, we could transform ϕ into the argumentation theory $T = (AS, \mathcal{K})$ and the set of queryables $\mathcal{Q}$ shown in Figure 3.9. In this reduction, ϕ is unsatisfiable iff t is stable-unsatisfiable in T w.r.t. $\mathcal{Q}$.

Furthermore, deciding stability is in CoNP: a certificate would be an argumentation theory T' such that $T \sqsubseteq_Q T'$ and the justification status of t in T' differs from t's justification status in T. We can check in polynomial time if $T \sqsubseteq_Q T'$; furthermore, the justification status of a literal in a given argumentation theory can be checked in polynomial time. □

The full proof can be found in Appendix B.1.1.

3.6.2. Relevance complexity

We turn to the problem of deciding whether a given queryable is j-relevant for a given literal for a justification status $j \in \{\text{unsatisfiable}, \text{defended}, \text{out}, \text{blocked}\}$. This problem turns out to be Σ_2^P-complete for each of the four justification statuses.

We first establish an auxiliary result that characterizes relevance of literals in terms of checking (non-)stability. Intuitively, it can be verified that a queryable q is j-relevant for a literal l if it is possible to find an AT for which the literal is not stable-j, but by adding q to the axioms, the literal becomes stable.

Lemma 3.3. *Let $T = (AS, \mathcal{K})$ be an AT, let $\mathcal{Q}$ be a set of queryables and let j be a justification status in $\{$unsatisfiable, defended, out, blocked$\}$. Given a literal $l \in \mathcal{L}$*

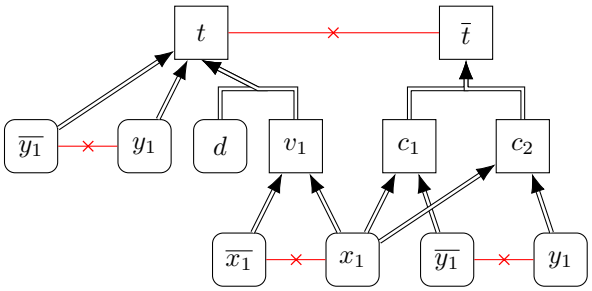


Figure 3.10: Illustration of the reduction used in Theorem 3.3 for the defended status for the formula $\phi = (x_1 \vee y_1) \wedge (x_i \vee \neg y_1)$. The queryables $\overline{y_1}$ and y_1 are displayed twice for readability.

and a queryable literal $q \in Q$ where $q \notin \mathcal{K}$ and $\overline{q} \cap \mathcal{K} = \emptyset$, q is j-relevant for l w.r.t. T and $\mathcal{Q}$ iff

- *there is an AT $T' = (AS, \mathcal{K}')$ with $T \sqsubseteq_Q T'$ such that l is not stable-j w.r.t. T' and*
- *l is stable-j w.r.t. $(AS, \mathcal{K}' \cup \{q\})$.*

Proof. From left to right: if q is j-relevant for l w.r.t. T and $\mathcal{Q}$ then there is some minimal stable-j future theory $(AS, \mathcal{K}^*)$ such that $q \in \mathcal{K}^*$; by minimality, l cannot be stable-j in $(AS, \mathcal{K}^* \setminus \{q\})$.

From right to left: suppose that an $(AS, \mathcal{K}' \cup \{q\})$ exists such that l is stable-j, while l is not stable-j w.r.t. $(AS, \mathcal{K}')$. If $(AS, \mathcal{K}' \cup \{q\})$ is minimal stable-j then the claim follows. Otherwise there is some $\mathcal{K}'' \subset \mathcal{K}' \cup \{q\}$ such that l is stable-j w.r.t. $(AS, \mathcal{K}'')$. If $q \notin \mathcal{K}''$, then $(AS, \mathcal{K}'') \sqsubseteq_Q (AS, \mathcal{K}')$, contradicting that $(AS, \mathcal{K}'')$ is minimally stable-j for l. Thus, $q \in \mathcal{K}''$, and the claim follows. □

Lemma 3.3 provides the basis for showing Σ_2^P-membership for relevance for all justification statuses considered in this chapter. We establish hardness via a reduction from quantified Boolean formulas with one quantified alternation (2-QBFs). Below, we give the hardness proof for the defended status. The hardness proofs for the justification statuses for unsatisfiable, out, and blocked can be found in Appendix B.1.2.

Theorem 3.3. *Deciding whether a queryable is j-relevant for a literal in an AT w.r.t. a set of queryables is Σ_2^P-complete for any $j \in \{$unsatisfiable, defended, out, blocked$\}$. Hardness holds even without preferences.*

Proof of Theorem 3.3 membership and hardness for the defended status. For an AT $T = (AS, \mathcal{K})$, justification status j, set of queryables Q, a specific queryable $q \in Q$ and a literal l, we show that deciding whether q is j-relevant for l is in Σ_2^P. Consider a non-deterministic construction of a future AT $T' = (AS, \mathcal{K}')$ of T. Check that l is not stable-j for T' and l is stable-j w.r.t. $(AS, \mathcal{K}' \cup \{q\})$. Checking whether

stability holds is in CoNP by Theorem 3.2. If these conditions hold for some non-deterministically constructed future AT, then q is j-relevant for l in T by Lemma 3.3. If no such future AT can be found, then q is not j-relevant for l in T. Thus the decision problem is in Σ_2^P.

For establishing Σ_2^P-hardness, we give a polynomial-time reduction from deciding whether a given 2-QBF formula $\Phi = \exists X \forall Y \neg\phi$ is True, where ϕ is a propositional formula in conjunctive normal form (CNF), and $X = \{x_1, \ldots, x_n\}$ and $Y = \{y_1, \ldots, y_m\}$ are pairwise disjoint sets of variables. This problem is Σ_2^P-complete (Stockmeyer, 1976; Wrathall, 1976).

For a given Φ, construct an AT T and queryables $\mathcal{Q}$ defined via

$$
\begin{aligned}
\mathcal{Q} &= X \cup \overline{X} \cup Y \cup \overline{Y} \cup \{d, \overline{d}\},\\
\mathcal{L} &= \mathcal{Q} \cup C \cup \overline{C} \cup V \cup \overline{V} \cup \{t, \overline{t}\},\\
\overline{\ } &= \{(x, \overline{x}), (\overline{x}, x) \mid x \in X \cup Y \cup V \cup C \cup \{d, t\}\},\\
\mathcal{R} &= \{(d, v_1, \ldots, v_n \Rightarrow t)\} \cup\\
&\quad \{(x \Rightarrow c) \mid x \in c\} \cup \{(\overline{x} \Rightarrow c) \mid \neg x \in c\} \cup\\
&\quad \{(y \Rightarrow c) \mid y \in c\} \cup \{(\overline{y} \Rightarrow c) \mid \neg y \in c\} \cup\\
&\quad \{(c_1, \ldots, c_p \Rightarrow \overline{t})\} \cup\\
&\quad \{(x_i \Rightarrow v_i), (\overline{x_i} \Rightarrow v_i) \mid x_i \in X\}\\
&\quad \{(y \Rightarrow t), (\overline{y} \Rightarrow t) \mid y \in Y\},\\
\mathcal{K} &= \emptyset,\\
\leq &= \emptyset,
\end{aligned}
$$

with $C = \{c_1, \ldots, c_p\}$ the set of clauses in ϕ, $\overline{X} = \{\overline{x} \mid x \in X\}$, $\overline{Y} = \{\overline{y} \mid y \in Y\}$, $\overline{C} = \{\overline{c} \mid c \in C\}$, and $V = \{v_i \mid x_i \in X\}$ and $\overline{V} = \{\overline{v_i} \mid x_i \in X\}$. The reduction is illustrated by an example in Figure 3.10.

We assume without loss of generality that d, $\overline{d}$, t, and $\overline{t}$ are fresh variables not occurring in Φ. Thereby $T = (AS, \mathcal{K})$ with $AS = (\mathcal{L}, \overline{\ }, \mathcal{R}, \leq)$ and $\mathcal{Q}$ can be constructed in polynomial time w.r.t. Φ.

Next, we argue that Φ is True if and only if d is defended-relevant for t w.r.t. T.

- *From left to right.* Assume that Φ is True. Then there is an assignment τ'_X to variables of X such that for each assignment τ'_Y to variables of Y, $\phi[\tau'_X, \tau'_Y]$ is False. Let τ_X be such an assignment. Construct a knowledge base $\mathcal{K}' = \{x \in X \mid \tau_X[x] = \text{True}\} \cup \{\overline{x} \in X \mid \tau_X[x] = \text{False}\}$. Note that $\mathcal{K}'$ must be consistent as no $x \in X$ can be assigned both True and False by τ_X. Hence $T \sqsubseteq_Q (AS, \mathcal{K}')$. We make the following observations.
  - t is not stable-defended w.r.t. $(AS, \mathcal{K}')$ and $\mathcal{Q}$, because t is not defended w.r.t. $(AS, \mathcal{K}')$: given that $d \notin \mathcal{K}'$ and for each $y \in Y$ both $y \notin \mathcal{K}'$ and $\overline{y} \notin \mathcal{K}'$, there is no argument for t in $Arg_{(AS,\mathcal{K}')}$.

– t is stable-defended w.r.t. $(AS, \mathcal{K}' \cup \{d\})$ and $\mathcal{Q}$. To see this, let $T'' = (AS, \mathcal{K}' \cup \{d\} \cup \mathcal{K}'')$ be an arbitrary AT such that $(AS, \mathcal{K}' \cup \{d\}) \sqsubseteq_Q T''$. Note that $\mathcal{K}'' \subseteq Y \cup \overline{Y}$. As there is no assignment τ_Y to variables in Y such that $\phi[\tau_X, \tau_Y]$ is True, there is no argument for $\bar{t}$ in $Arg_{T''}$. On the other hand, there is at least one argument for t with top rule $(d, v_1, \ldots, v_n \Rightarrow t)$ in $Arg_{T''}$. Since the argument for t is undefeated, t is defended w.r.t. T''. As T'' was chosen arbitrarily from all T''' such that $(AS, \mathcal{K}' \cup \{d\}) \sqsubseteq_Q T'''$, we can conclude that t is stable-defended w.r.t. $(AS, \mathcal{K}' \cup \{d\})$ and $\mathcal{Q}$.

Hence by Lemma 3.3, d is defended-relevant for t w.r.t. T.

- *From right to left.* Assume that d is defended-relevant for t w.r.t. T. By Definition 3.14, there is a minimal stable-defended future theory $T' = (AS, \mathcal{K}' \cup \{d\})$ w.r.t. T and $\mathcal{Q}$. Since t is stable-defended w.r.t. $(AS, \mathcal{K}' \cup \{d\})$ and $\mathcal{Q}$, t is defended w.r.t. $(AS, \mathcal{K}' \cup \{d\})$. Then there must be an argument for t and no argument for $\bar{t}$.

  Further, by minimality of $(AS, \mathcal{K}' \cup \{d\})$, t cannot be stable-defended w.r.t. $(AS, \mathcal{K}')$ and $\mathcal{Q}$. Hence there is an argument for t having the observation-based argument for d as a subargument; in particular, this is the argument with top rule $(d, v_1, \ldots, v_n \Rightarrow t)$. Since such an argument exists, for each $x \in X$, either $x \in \mathcal{K}'$ or $\overline{x} \in \mathcal{K}'$. In addition, for each $y \in Y$ we have $y \notin \mathcal{K}'$ and $\overline{y} \notin \mathcal{K}'$. This is because if there would be any $y \in Y$ such that $y \in \mathcal{K}'$ or $\overline{y} \in \mathcal{K}'$, then there would be an argument for t with top rule $y \Rightarrow t$ or $\overline{y} \Rightarrow t$, and hence the argument with top rule $(d, v_1, \ldots, v_n \Rightarrow t)$ would not have been required and $(AS, \mathcal{K}' \cup \{d\})$ would not have been minimal. Now let τ_X be the assignment to variables in X corresponding to $\mathcal{K}'$, i.e., for each $x \in X$, $\tau_X[x] =$ True iff $x \in \mathcal{K}'$ and $\tau_X[x] =$ False iff $\overline{x} \in \mathcal{K}'$.

  Next, we argue that $\phi[\tau_X, \tau_Y]$ is False for each assignment τ_Y to variables in Y. Towards a contradiction, assume that there is an assignment τ_Y such that $\phi[\tau_X, \tau_Y]$ is True. Let $\mathcal{K}' \cup \{d\} \cup \mathcal{K}''$ be the corresponding knowledge base, i.e., $\mathcal{K}'' = \{y \in Y \mid \tau_Y[y] = \text{True}\} \cup \{\overline{y} \in Y \mid \tau_Y[y] = \text{False}\}$. Since $\phi[\tau_X, \tau_Y]$ is True, there is an argument for $\bar{t}$ in $Arg_{(AS, \mathcal{K}' \cup \{d\} \cup \mathcal{K}'')}$, which implies that t is not defended w.r.t. $(AS, \mathcal{K}' \cup \{d\} \cup \mathcal{K}'')$. However, then t cannot be stable-defended w.r.t. $(AS, \mathcal{K}' \cup \{d\})$ and $\mathcal{Q}$; a contradiction. Hence $\phi[\tau_X, \tau_Y]$ is False for each assignment τ_Y to variables in Y, i.e., Φ is True. □

In summary, the justification, stability, and relevance problems are clearly separated in terms of standard complexity-theoretic assumptions, ranging from polynomial-time decidability (justification) to CoNP-completeness (stability) and further to Σ_2^P-completeness (relevance). The results hold regardless of preferences over defeasible rules. In order to obtain this result for deciding the justification status, we developed a reformulation in terms of defeasible rules, a representation that is bounded polynomially in terms of a given AT. The membership results all rely on this reformulation.

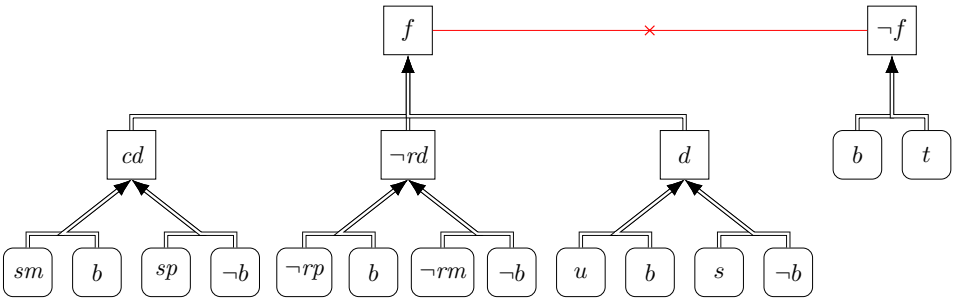


Figure 3.11: Example of an argumentation theory T from the law enforcement domain. (Rounded) rectangles represent literals from $\mathcal{L}$ and rounded rectangles are queryable literals. Rules are represented by double-lined arrows and contradictories as single-lined arrows. Note that the literals b and $\neg b$ are visualised multiple times and and the contrariness relation between b and $\neg b$ is omitted for clarity.

3.6.3. A naive algorithm for stability

Having identified the complexity of the stability and relevance problems, we next illustrate why solving problems in high complexity classes is challenging, by providing a naive algorithm for stability and testing this algorithm on a toy example.

The first solution that may come to one's mind in order to detect if a literal is stable in some argumentation theory w.r.t. the set of queryables could be an algorithm that (1) generates all future argumentation theories for the given argumentation theory; (2) computes the current justification status of the literal and stores this as a justification label; and (3) assigns a stability label based on the justification labels for the future argumentation theories. Such an algorithm, let us call it STABILITY-NAIVE,[3] is sound and complete, but also exponential: assuming that the contradiction function of the argumentation system is instantiated as classical negation, each argumentation theory $T = (AS, \mathcal{K})$ has $3^{\frac{1}{2}|\mathcal{Q}|-|\mathcal{K}|}$ future argumentation theories. This is highly problematic for the runtime, as we demonstrate next with an experiment on a toy example.

Let $T = (AS, \mathcal{K})$ with $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ be an argumentation theory in the domain of online trade fraud, visualised in Figure 3.11. $\mathcal{L}$ consists of the following literals and their negations:

- b: citizen tried to buy a product (as opposed to selling a product);
- sm: citizen sent money;
- sp: citizen sent product;
- rp: citizen received product;
- rm: citizen received money;
- u: suspect url;
- s: screenshot of payment;
- t: trusted web shop;
- cd: citizen delivered;

[3]The source code can be found at `https://github.com/DaphneO/StabilityLabelAlgorithm`.

- *rd*: citizen received delivery;
- *d*: deception;
- *f*: fraud.

Let $\mathcal{K} = \emptyset$. This argumentation system has 24 literals (including negation), 16 of which are queryable, and 8 rules.

For this experiment, we created a data set of all $3^8 = 6561$ future argumentation theories of $(AS, \emptyset)$. For each of these argumentation theories, we executed STABILITY-NAIVE and measured the computation time.[4] The results are shown as boxplots grouped by the number of unknown queryables (i.e. $|\mathcal{Q}_{pos}| - |\mathcal{K}|$ where $\mathcal{Q}_{\text{pos}}$ is the set of all non-negated queryables) in Figure 3.12.

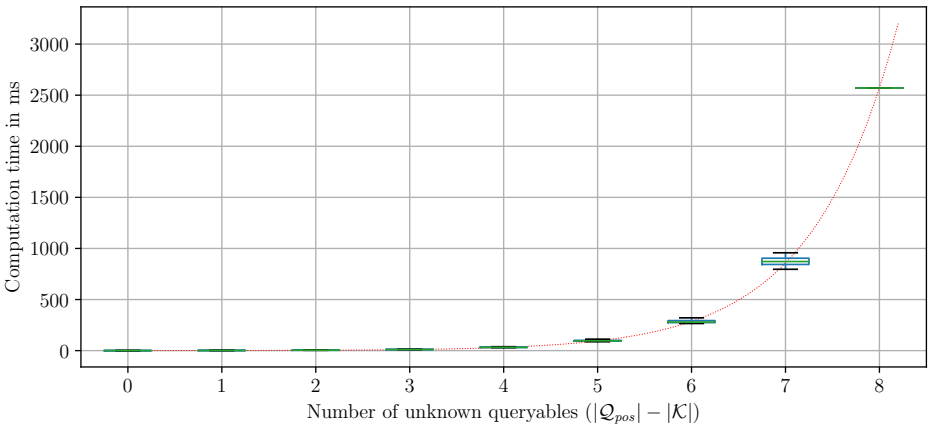


Figure 3.12: Computation time of STABILITY-NAIVE, applied to the 3^8 future argumentation theories of $(AS, \emptyset)$ for the literal f, where AS and $\mathcal{Q}$ are the argumentation system and set of queryables from our running example in Figure 3.1. The results are shown as boxplots grouped by the number of unknown queryables. The red dotted line is the formula $g(x) = 0.392 \cdot x^3$ and indicates that the computation time grows exponentially with the number of unknown queryables.

From the figure it becomes clear that even for a simple argumentation system such as our toy example, the exponential algorithm STABILITY-NAIVE cannot be used in real-time, given that the computation would take multiple seconds after each update of the knowledge base. For the argumentation theory in which the knowledge base is empty, so eight queryables are unknown, the computation time is over 2500ms. To explicate the exponential computation time of STABILITY-NAIVE, we plotted the line $0.392 \cdot 3^{|\mathcal{Q}_{pos}|-|\mathcal{K}|}$, which fits nicely with the measured computation times, in red. Assuming that this formula can be used to extrapolate the computation time to inputs with a larger number of unknown queryables, the expected computation time would be 1 hour and 33 minutes in an application where $|\mathcal{Q}_{pos}| - |\mathcal{K}| = 15$; if

[4]This experiment was run on a Intel(R) Core(TM) i7-7820HQ CPU 2.90 GHz 16 GB RAM machine.

$|\mathcal{Q}_{pos}| - |\mathcal{K}| = 20$, it would take over two weeks. As will become clear in Chapter 5, this number of queryables is common for realistic applications; for example, the argumentation system that we use for actual trade fraud intake has a $\mathcal{Q}_{pos}$ of 15 positive queryables.

3.7. Polynomial heuristic algorithm for stability

In the previous section, we have shown that the naive approach for solving the CoNP-complete problem of deciding stability is impractical. Under the assumption that P $\neq$ NP, problems in this complexity class are considered intractable: each sound and complete algorithm for computing stability has some exponential component. This means that an exact algorithm for stability would need exponential time. Practical applications, such as our decision support systems at the police, require fast computation for arbitrary argumentation theories. An alternative to sound and complete algorithms is to use a heuristic algorithm that is sound, complete or neither. Popular approaches in this area are to apply data-driven techniques or devise direct heuristic algorithms. Data-driven techniques such as neural networks have the benefit of predictable computational cost but provide no soundness guarantee (see e.g. Craandijk and Bex (2020)). For the police this is problematic; often it is better to accidentally collect too much information than to make a wrong decision. We therefore prefer direct heuristic algorithms over data-driven techniques.

The contribution proposed in this section is a polynomial algorithm for estimating stability. The algorithm is designed for argumentation theories without rule preferences, as rule preferences are not required for the applications at the police (cf. Chapter 5). An advantage of not considering preferences between rules is that the notions of rebuttal attack and defeat coincide. This follows directly from Definition 3.6. As we will prove later in this section, the heuristic algorithm is sound, but not complete. This way, we can guarantee fast computation for arbitrary argumentation theories (in the considered fragment, so without rule preferences) and ensure that more information cannot change the justification status of any literal that is labelled stable.

Our algorithm for approximating stability iteratively constructs a labelling that assigns a label to each literal and rule. Before explaining how the algorithm achieves such a labelling, we will first illustrate what such a labelling should look like by showing the correct labelling for an example argumentation theory. Having observed some properties of this labelling, we subsequently explain our proposed procedure to approximate this labelling.

3.7.1. Desired stability labelling

In order to decide if a literal is stable, our algorithm uses a labelling L that assigns a quadruple of four booleans $\langle u, d, o, b\rangle$ (i.e. a label) to each literal and rule. Each of these booleans corresponds to a justification status. Intuitively, the truth value of a boolean belonging to a literal represents the possibility that this literal can still become stable-unsatisfiable (u), stable-defended (d), stable-out (o) or stable-

blocked (b) in a future argumentation theory. Rules are assigned a label as well, because this helps in efficiently computing the labels of their conclusion literals. The label assigned to a given rule aggregates information from labels assigned to its antecedents. If only one of the booleans in a label of some literal or rule is True, then that literal or rule is labelled stable, as formally defined next.

Definition 3.20 (Labelled stable)**.** Given a label $L[x]$ for some literal or rule $x \in \mathcal{L} \cup \mathcal{R}$:

- x is *labelled stable-unsatisfiable* by L iff $L[x] = \langle 1, 0, 0, 0 \rangle$;
- x is *labelled stable-defended* by L iff $L[x] = \langle 0, 1, 0, 0 \rangle$;
- x is *labelled stable-out* by L iff $L[x] = \langle 0, 0, 1, 0 \rangle$;
- x is *labelled stable-blocked* by L iff $L[x] = \langle 0, 0, 0, 1 \rangle$.

A literal or rule $x \in \mathcal{L} \cup \mathcal{R}$ is *labelled stable* by L iff any of the above applies.

In Example 3.17, we will clarify which labelling our algorithm tries to obtain by showing the correct labels for an example argumentation theory. Note that this example does not yet describe the labelling procedure, but rather motivates the literal labels that it tries to achieve.

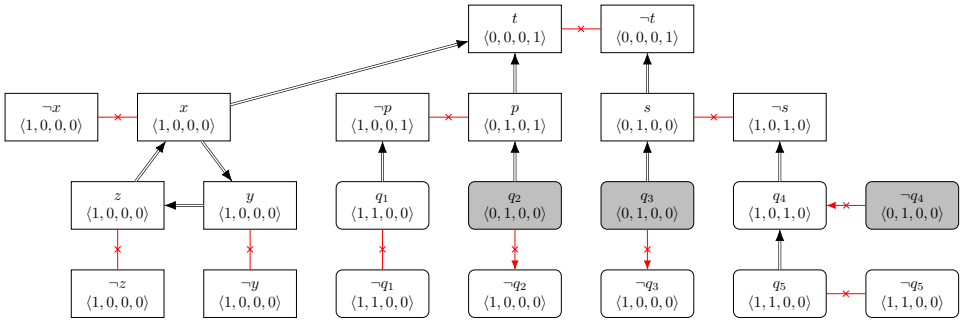


Figure 3.13: Argumentation theory illustrating the ground-truth $\langle u, d, o, b \rangle$-labelling of literals.

Example 3.17 (Stability labelling)**.** In Figure 3.13, we give an example of an argumentation theory $T = (AS, \mathcal{K})$ where $AS = (\mathcal{L}, \bar{\ }, \mathcal{R}, \leq)$ and $\langle u, d, o, b \rangle$ is the stability labelling that *should* be found by a sound and complete stability labelling algorithm. The set of queryables $\mathcal{Q}$ is $\{q_1, \ldots, q_5\} \cup \{\neg q_1, \ldots, \neg q_5\}$. From these queryables, q_2, q_3 and $\neg q_4$ are observed in T (i.e., in $\mathcal{K}$), while q_1, $\neg q_1$, q_5 and $\neg q_5$ can still be observed in a future argumentation theory T' such that $T \sqsubseteq_Q T'$.

Some literals are already stable:

- each literal l in $\{\neg q_2, \neg q_3, x, \neg x, y, \neg y, z, \neg z\}$ is stable-unsatisfiable (having label $\langle 1, 0, 0, 0 \rangle$) because there is no rule for l and either $l \notin \mathcal{Q}$ or there is some contradictory $l' \in \bar{l}$ such that $l' \in \mathcal{K}$;

- q_2, q_3, $\neg q_4$ and s are stable-defended (having label $\langle 0,1,0,0\rangle$) because there is an argument for these literals that cannot be attacked and therefore cannot be defeated; and
- t and $\neg t$ are stable-blocked (having label $\langle 0,0,0,1\rangle$) because the arguments for these literals attack and defeat each other, but neither is in the grounded extension.

Other literals are not yet stable, but can become stable in a future argumentation theory. Consider for example the argumentation theory $T' = (AS, \mathcal{K} \cup \{q_1\})$ such that $T \sqsubseteq_Q T'$. In T', the literal q_1 would be stable-defended; $\neg q_1$ would be stable-unsatisfiable and both $\neg p$ and p become stable-blocked in T' w.r.t. $\mathcal{Q}$. An alternative future argumentation theory of T is $T'' = (AS, \mathcal{K} \cup \{\neg q_1\})$, where $\neg q_1$ and p become stable-defended, while q_1 and $\neg p$ are stable-unsatisfiable in T'' w.r.t. $\mathcal{Q}$. Similarly, adding either q_5 or $\neg q_5$ would make each literal in $\{q_5, \neg q_5, q_4, \neg s\}$ stable.

Finally, recall from Section 3.3.1 that the distinction between the *out* and *blocked* justification status is convenient for our algorithm. This can be seen by considering the difference in statuses for p and s. Both literals have a contradictory that cannot become stable-defended in any future argumentation theory ($\neg p$ and $\neg s$, respectively). However, $\neg s$ cannot be stable-blocked in any future argumentation theory (given that each argument for $\neg s$ is attacked and therefore defeated by the observation-based argument for $\neg q_4$), which means that s is stable-defended. On the other hand, $\neg p$ can still become stable-blocked (by adding q_1 to the knowledge base), so p is not stable-defended but can still become stable-blocked.

Based on the example above, we can make three general remarks about the desired labelling:

Remark 1 The labels of literals depend on the labels of antecedents of rules for those literals, as well as of antecedents of rules for contradictories of those literals. For example, the status of t depends on the status of p and the status of s. In addition, the labels of queryable literals also depend on their presence (or one of their contradictories' presence) in the knowledge base: the status of q_4 depends on the status of q_5, but also on the presence of $\neg q_4$ in the knowledge base.

Remark 2 It is not always required that each of these antecedents is stable for the conclusion literal to be stable. For example: t and $\neg t$ are stable although p is not stable. s is stable although q_4 is not stable.

Remark 3 The rule set of an argumentation theory can contain cycles of support relations, such as the cycle $x \Rightarrow y$, $y \Rightarrow z$, $z \Rightarrow x$ in the example.

We take these three remarks into account in our proposed labelling procedure. By Remark 1, the labels of rules depend on the labels of their antecedent literals, while labels of literals depend on the labels of rules for that literal or for one of its contradictories. Literals and rules are labelled bottom-up, starting from queryable literals and literals for which there is no rule and relabelling literals and rules based on the resulting new labels, until no new label can be added. By Remark 2, the information

contained in the labelling should be more precise than a single label indicating if the literal is stable or not. For example, if we only had one label for literals that are not stable, it would not be possible to label s in Example 3.17 as stable, since it depends on the literal q_4 that would just be labelled as not stable. Therefore, we propose to assign quadruple labels $\langle u, d, o, b\rangle$ to each literal and rule. By Remark 3, it is necessary to correctly handle cycles of support relations. We deal with this in a preprocessing step.

In the following two subsections, we will describe the algorithm in two steps: a preprocessing step and the main labelling procedure.

3.7.2. Preprocessing

Our algorithm starts with a preprocessing step. This enables the algorithm to properly deal with argumentation theories containing *support cycles*, i.e. cycles of inference relations based on which no argument can be constructed. For example, consider the inference relation between the literals x, y and z in Figure 3.13. None of these literals are queryable, so there is no future argumentation theory in which there is an observation-based argument for any of them. Since there are no other rules for any of these literals, other than the three rules $x \Rightarrow y$, $y \Rightarrow z$ and $z \Rightarrow x$ that form a cycle, these three rules form a support cycle. Support cycles can be problematic for defeasible reasoning algorithms, when not handled properly: an initial version of our algorithm for stability, proposed in Testerink et al. (2019a), does not label literals and rules that are dependent on support cycles: in a bottom-up labelling procedure, there is no place to start labelling.

The preprocessing step is specified in Algorithm 3. The idea of this algorithm is that initially, all literals that cannot be in the knowledge base in a future argumentation theory and all rules are labelled $\langle 1, 0, 0, 0\rangle$ (i.e. stable-unsatisfiable). Then, the algorithm incrementally labels those rules for which all antecedents are not labelled $\langle 1, 0, 0, 0\rangle$ and their consequents as $\langle 1, 1, 1, 1\rangle$ (i.e. may still become unsatisfiable, defended, out or blocked) based on the intuition that there may be an argument based on these rules in a future argumentation theory.

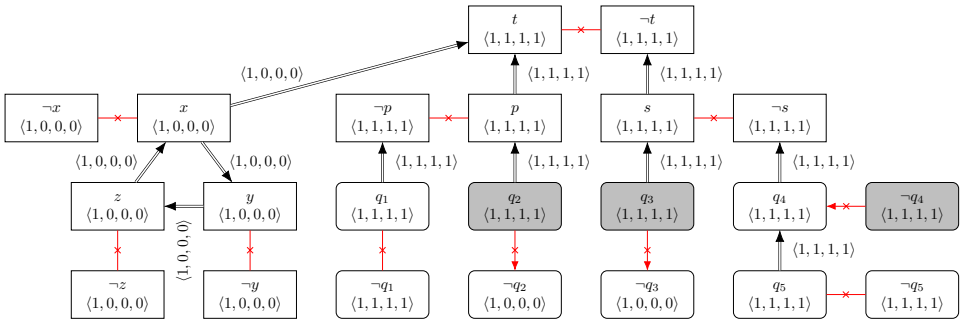


Figure 3.14: Argumentation theory illustrating the $\langle u, d, o, b\rangle$-labelling of literals and rules after the preprocessing step.

Algorithm 3 Preprocessing step

```
1: procedure PREPROCESS(ℒ, ℛ, ¯, 𝒬, 𝒦)
2:     for Literal l in ℒ do
3:         if l ∈ 𝒬 and for each l' ∈ l̄: l' ∉ 𝒦 then L[l] = ⟨1,1,1,1⟩
4:         else L[l] = ⟨1,0,0,0⟩
5:     for Rule r in ℛ do
6:         L[r] = ⟨1,0,0,0⟩
7:     Change = True
8:     while Change do
9:         Change = False
10:        for Rule r in ℛ do
11:            if L[r] = ⟨1,0,0,0⟩ and for each l ∈ ants(r): L[l] ≠ ⟨1,0,0,0⟩ then
12:                L[r] = ⟨1,1,1,1⟩
13:                L[cons(r)] = ⟨1,1,1,1⟩
14:                Change = True
15:    return L
```

Example 3.18 (Preprocessing step)**.** We return to the argumentation theory $T = (AS, \mathcal{K})$ where $AS = (\mathcal{L}, ^{-}, \mathcal{R}, \leq)$, with queryables $\mathcal{Q}$, from Example 3.17 and Figure 3.13.

Let $L_p = \text{PREPROCESS}(\mathcal{L}, \mathcal{R}, ^{-}, \mathcal{Q}, \mathcal{K})$. The labelling L_p is illustrated in Figure 3.14. The literals in $\{\neg q_2, \neg q_3, x, \neg x, y, \neg y, z, \neg z\}$ are labelled $\langle 1,0,0,0\rangle$ by L_p. For each other literal l in $\mathcal{L}$: $L_p[l] = \langle 1,1,1,1\rangle$. Further note that the rules in and from the support cycle, i.e. $x \Rightarrow y$, $y \Rightarrow z$, $z \Rightarrow x$ and $x \Rightarrow t$, are labelled $\langle 1,0,0,0\rangle$. The other rules are labelled $\langle 1,1,1,1\rangle$ by L_p.

3.7.3. Quadruple labelling procedure

The result of the preprocessing procedure is an initial labelling L_p for each of the literals and rules in our argumentation system. After preprocessing, we apply a bottom-up labelling procedure that updates the quadruple of four booleans $\langle u, d, o, b\rangle$ for each literal and rule, resulting in the final labelling L. Algorithm 6 specifies how literals and rules are visited in the labelling procedure, where literals and rules are labelled according to the labelling rules specified in Algorithm 4 and 5. We will discuss both the labelling rules and the labelling procedure, starting with the labelling rules.

In Section 3.7.1 we observed that there are cases in which a literal should be labelled stable, although it is dependent on literals that are not yet labelled stable (Remark 2). Because of this, a labelling that assigns a single label (e.g. stable-unsatisfiable / stable-defended / stable-out / stable-blocked / unstable) to literals and rules does not suffice. The labelling rules in these algorithms require some additional notation for indexing single booleans of labels:

Notation 3.1 (Indexing of label parts)**.** Given a label $L[x] = \langle u, d, o, b\rangle$ for some literal or rule $x \in \mathcal{L} \cup \mathcal{R}$, we refer to the u- (resp. d-, o-, b-) boolean of $L[x]$ with $L[x].u$ (resp. $L[x].d, L[x].o, L[x].b$).

In Relabel-Literal (Algorithm 4) and Relabel-Rule (Algorithm 5) we specify how the labels of literals and rules are updated. Note that each of these labelling rules is able to turn a boolean (u, d, o or b) to False, but none of them is able to turn a boolean to True.

Algorithm 4 Relabel-Literal procedure

procedure Relabel-Literal($\mathcal{L}, \mathcal{R}, \bar{}, \mathcal{Q}, \mathcal{K}, L, l$)
  ▷ ***Labelling rules turning L[l].u to False***
  if $l \in \mathcal{K}$ **then** $L[l].u$ = False ▷ L-U-a
  if there is a rule r for l with $\neg L[r].u$ **then** $L[l].u$ = False ▷ L-U-b

  ▷ ***Labelling rules turning L[l].d to False***
  if some $l' \in \bar{l}$ is in $\mathcal{K}$ **then** $L[l].d$ = False ▷ L-D-a
  if $l \notin \mathcal{Q}$ **then**
    if for each rule r for l: $\neg L[r].d$ **then** $L[l].d$ = False ▷ L-D-b
    if there is some $l' \in \bar{l}$ for which there is a rule r' with $\neg L[r'].u$ and $\neg L[r'].o$ **then** $L[l].d$ = False ▷ L-D-c

  ▷ ***Labelling rules turning L[l].o to False***
  if $l \in \mathcal{K}$ **then** $L[l].o$ = False ▷ L-O-a
  if $l \in \mathcal{Q}$ and for each $l' \in \bar{l}$, some $l'' \in \overline{l'}$ is in $\mathcal{K}$ **then**
    if for each rule r for l: $\neg L[r].o$ **then** $L[l].o$ = False ▷ L-O-b
    if there is a rule r for l with $\neg L[r].u$ and $\neg L[r].o$ **then** $L[l].o$ = False ▷ L-O-c
  if $l \notin \mathcal{Q}$ **then**
    if for each rule r for l: $\neg L[r].o$ **then** $L[l].o$ = False ▷ L-O-d
    if there is a rule r for l with $\neg L[r].u$ and $\neg L[r].o$ **then** $L[l].o$ = False ▷ L-O-e
  if for each rule r for l: $\neg L[r].d$ and $\neg L[r].o$ and $\neg L[r].b$ **then** $L[l].o$ = False ▷ L-O-f

  ▷ ***Labelling rules turning L[l].b to False***
  if $l \in \mathcal{Q}$ **then** $L[l].b$ = False ▷ L-B-a
  if for each rule r for l: $\neg L[r].d$ and $\neg L[r].b$ **then** $L[l].b$ = False ▷ L-B-b
  if for each $l' \in \bar{l}$: for each rule r' for l': $\neg L[r'].d$ and $\neg L[r'].b$ **then**
    if for each rule r for l: $\neg L[r].b$ **then** $L[l].b$ = False ▷ L-B-c
    if there is a rule r for l with $\neg L[r].u$ and $\neg L[r].o$ and $\neg L[r].b$ **then** $L[l].b$ = False ▷ L-B-d
  return L

Example 3.19 (Quadruple labelling rules)**.** We return to the argumentation theory from Figure 3.13 and Examples 3.17 and 3.18. Some labelling rules for literals in Relabel-Literal (Algorithm 4) are not dependent on rules for (a contradictory of) that literal. For example, case L-B-a already labels a literal $l \in \mathcal{L}$ as $\neg L[l].b$ if $l \in \mathcal{Q}$. This labelling rule applies for the literals in $\{q_1, \ldots, q_5\} \cup \{\neg q_1, \ldots, \neg q_5\}$, because they are queryable. Case L-U-a and L-O-a apply if a literal is in the knowledge base

Algorithm 5 RELABEL-RULE procedure

1: **procedure** RELABEL-RULE($\mathcal{L}, \mathcal{R}, L, r$)
2: **if** for each antecedent l of r: $\neg L[l].u$ **then** $L[r].u =$ False ▷ R-U-a
3: **if** there is an antecedent l of r with $\neg L[l].d$ **then** $L[r].d =$ False ▷ R-D-a
4: **if** for each antecedent l of r: $\neg L[l].o$ **then** $L[r].o =$ False ▷ R-O-a
5: **if** for each antecedent l of r: $\neg L[l].b$ **then** $L[r].b =$ False ▷ R-B-a
6: **if** there is an antecedent l of r with $\neg L[l].d$ and $\neg L[l].b$ **then** $L[r].b =$ False ▷ R-B-b
7: **return** L

$\mathcal{K}$. Thanks to these labelling rules, the literals $l \in \{q_2, q_3, \neg q_4\}$ in the example will be labelled $\neg L[l].u$ and $\neg L[l].o$. Finally, case L-D-a does not depend on rules either: for a given literal $l \in \mathcal{L}$, it applies if there is some $l' \in \bar{l}$ such that $l' \in \mathcal{K}$. This case applies for the literals $\neg q_2$, $\neg q_3$ and q_4. Note that the absence of rules for a literal is informative for the justification status as well: for example, the literal q_1 is labelled $\neg L[q_1].o$ by case L-O-f because there is no rule for q_1 in $\mathcal{R}$.

Other labels are based on the rules for (a contradictory of) a literal and propagate properties of (attacks and thus defeats on) subarguments. For example, the rule $q_3 \Rightarrow s$ is labelled $\langle 0, 1, 0, 0\rangle$ by RELABEL-RULE's cases R-U-a, R-O-a and R-B-a (and the fact that its only antecedent q_3 is labelled $\langle 0, 1, 0, 0\rangle$). Similarly, the rule $q_4 \Rightarrow \neg s$ is labelled $\langle 1, 0, 1, 0\rangle$ by the cases R-D-a and R-B-a (or R-B-b). Note that this rule cannot be labelled stable. Still, although the literal s is dependent on a rule that is not labelled stable, we can label s as stable: $L[s] = \langle 0, 1, 0, 0\rangle$ by RELABEL-LITERAL's cases L-U-b, L-O-d and L-B-c (or L-B-d). Other literals, such as p, cannot be labelled stable, but still we can exclude some stability statuses: p can never be stable-unsatisfiable (case L-U-b) or stable-out (case L-O-d) w.r.t. $\mathcal{Q}$ in any future argumentation theory.

Finally, we discuss the algorithm that visits the literals and rules, repeatedly applying Algorithm 4 and 5. As identified in Remark 1 in Section 3.7.1, the labels of literals depend on the antecedents of rules for those literals and for contradictories of those literals; labels of queryable literals also depend on the knowledge base. Therefore, STABILITY-LABEL (Algorithm 6) starts by labelling those literals that are queryable or for which there is no rule, see lines 4–6. Then, literals and rules are labelled incrementally: after considering a literal for the first time (line 6) or changing a literal (line 13 and 16), the label of each rule that has this literal as an antecedent is added to the set TODO-SET and therefore at some point considered for relabelling. The algorithm ends when the TODO-SET is empty (line 7): at this moment, the labelling has reached a fixed point and cannot change any more. We give some intuition by labelling the example from the beginning of this section.

Example 3.20 (Labelling step)**.** We return to the argumentation theory from Figure 3.13 and Examples 3.17–3.19 and show how literals and rules are considered for (re)labelling by Algorithm 6.

Algorithm 6 Labelling procedure STABILITY-LABEL

```
1: procedure STABILITY-LABEL(ℒ, ℛ, ¯, 𝒬, 𝒦)
2:     L = PREPROCESS(ℒ, ℛ, ¯, 𝒬, 𝒦)
3:     TODO-SET = empty set
4:     for Literal l in ℒ such that l ∈ 𝒬 or there is no rule for l in ℛ do
5:         L = RELABEL-LITERAL(ℒ, ℛ, ¯, 𝒬, 𝒦, L, l)
6:         Add all rules having l as antecedent to TODO-SET
7:     while TODO-SET is not empty do
8:         Pop a rule r from TODO-SET
9:         L = RELABEL-RULE(ℒ, ℛ, L, r)
10:        if r's label changed then
11:            L = RELABEL-LITERAL(ℒ, ℛ, ¯, 𝒬, 𝒦, L, cons(r))
12:            if cons(r)'s label changed then
13:                Add all rules having cons(r) as antecedent to TODO-SET
14:            for l' ∈ cons(r)‾ do
15:                L = RELABEL-LITERAL(ℒ, ℛ, ¯, 𝒬, 𝒦, L, l')
16:                if l''s label changed then
17:                    Add all rules having l' as antecedent to TODO-SET
18:    return L
```

First, note that the literals x, y and z are never reconsidered for relabelling after the preprocessing step, since they are not queryable and there is no rule for those literals that can change its label. Therefore, the labels in L assigned to these literals equal the labels in L_p, that is: $\langle 1, 0, 0, 0\rangle$ (as shown in Example 3.18).

All queryables in Q, as well as the literals $\neg x$, $\neg y$ and $\neg z$, for which there is no rule, are considered for relabelling in Algorithm 6 line 5; see Figure 3.15. For the queryable literals, RELABEL-LITERAL case L-B-a applies. As a result, the b-boolean is turned to False: there is no future argumentation theory T' such that queryable literals are stable-blocked in T' w.r.t. $\mathcal{Q}$. Furthermore, q_2, q_3 and $\neg q_4$ are in the knowledge base. Therefore, the u- and o-booleans for these literals are turned to False by case L-U-a and L-O-a. For $\neg q_2$, $\neg q_3$ and q_4, case L-D-a applies, since they have a contradictory in the knowledge base. As a result, the d-boolean is turned to False. Finally, the o-booleans of q_1, $\neg q_1$, $\neg q_2$, $\neg q_3$, q_5 and $\neg q_5$ are turned to False by case L-O-f.

After relabelling of aforementioned literals, the rules $q_1 \Rightarrow \neg p$, $q_2 \Rightarrow p$, $q_3 \Rightarrow s$, $q_5 \Rightarrow q_4$ and $q_4 \Rightarrow \neg s$ are added to TODO-SET. Suppose that the rule $q_1 \Rightarrow \neg p$ is selected first from TODO-SET (this order is arbitrary). This rule is labelled $\langle 1, 1, 0, 0\rangle$ by case R-O-a and R-B-a, as illustrated in Figure 3.16. Since this rule's label changed, the literals $\neg p$ and p are considered for relabelling by RELABEL-LITERAL as well. Then the label of $\neg p$ changes into $\langle 1, 1, 0, 1\rangle$ (by case L-O-d). The label of p does not change, so the rule $p \Rightarrow t$ is not yet added to TODO-SET.

In the next iteration of the while loop, the rule $q_2 \Rightarrow p$ is selected from TODO-SET and relabelled as $\langle 0, 1, 0, 0\rangle$ by case R-U-a, R-O-a and R-B-a of RELABEL-RULE; see Figure 3.17. After this, both p and $\neg p$ are reconsidered for relabelling. Since the label of p changes, the rule $p \Rightarrow t$ is added to TODO-SET.

In additional iterations of the while loop the rules $q_3 \Rightarrow s$, $q_4 \Rightarrow \neg s$ and $q_5 \Rightarrow q_4$ are selected from TODO-SET, so that these rules, their consequents and the contradictories of those contradictories can be relabelled. The result after this iteration is shown in Figure 3.18.

Subsequently, the rules $p \Rightarrow t$ and $s \Rightarrow \neg t$ are considered for relabelling by RELABEL-RULE. For both rules, the u-booleans and o-booleans are turned to False by case R-U-a and R-O-a, respectively. Furthermore, $s \Rightarrow \neg t$ is labelled $\neg L[s \Rightarrow \neg t].b$ by case R-B-a. Finally, t and $\neg t$ are relabelled: $L[t] = L[\neg t] = \langle 0, 0, 0, 1\rangle$ by case L-U-b, L-D-c, L-O-d. Note that the resulting labelling of literals, illustrated in Figure 3.19, exactly matches the desired stability labelling shown in Figure 3.13.

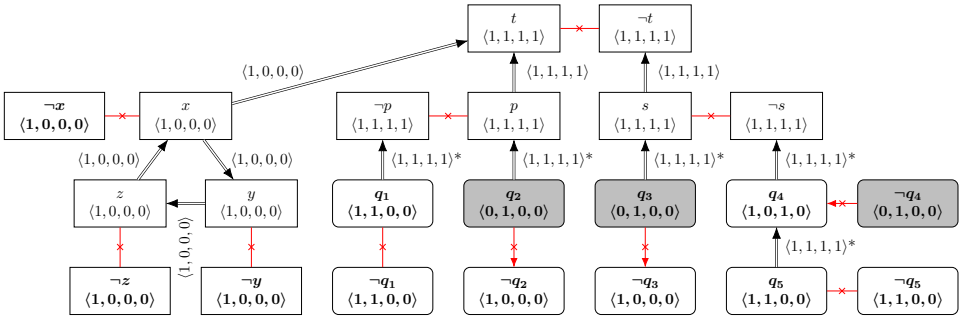


Figure 3.15: Argumentation theory illustrating the $\langle u, d, o, b\rangle$-labelling of STABILITY-LABEL between lines 6 and 7. All literals that have been considered for relabelling by RELABEL-LITERAL are illustrated in boldface. The rules in TODO-SET are indicated with an asterisk.

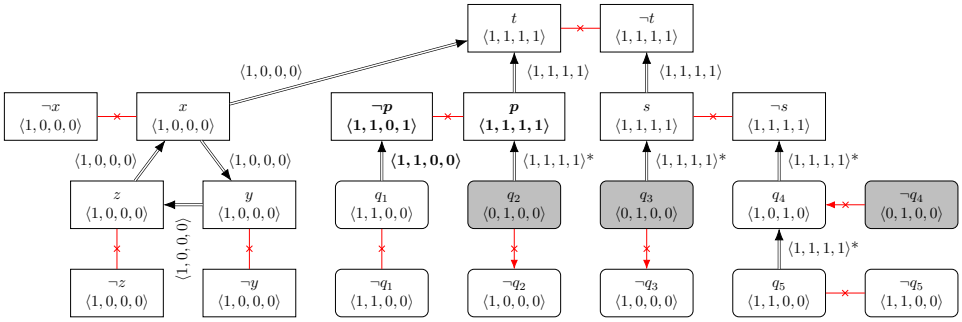


Figure 3.16: Argumentation theory illustrating the $\langle u, d, o, b\rangle$-labelling of STABILITY-LABEL after the first iteration of the while-loop, where the rule $q_1 \Rightarrow \neg p$ has been selected from TODO-SET. All literals that have been considered for relabelling by RELABEL-LITERAL are illustrated in boldface. The rules in TODO-SET are indicated with an asterisk.

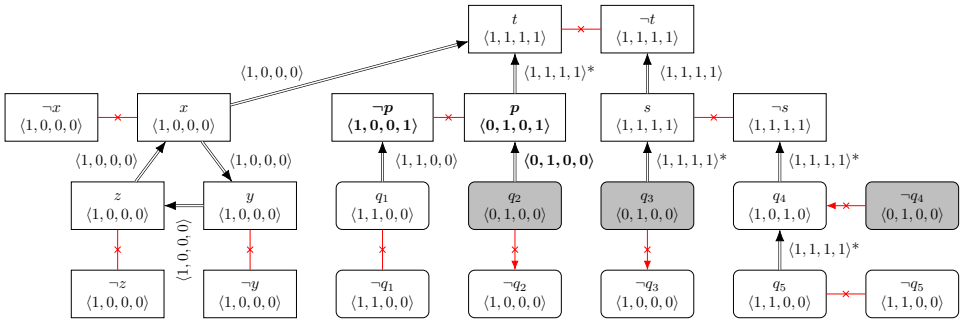


Figure 3.17: Argumentation theory illustrating the $\langle u, d, o, b\rangle$-labelling of Stability-Label after the second iteration of the while-loop, where the rule $q_2 \Rightarrow p$ has been selected from todo-set. All literals that have been considered for relabelling by Relabel-Literal are illustrated in boldface. The rules in todo-set are indicated with an asterisk.

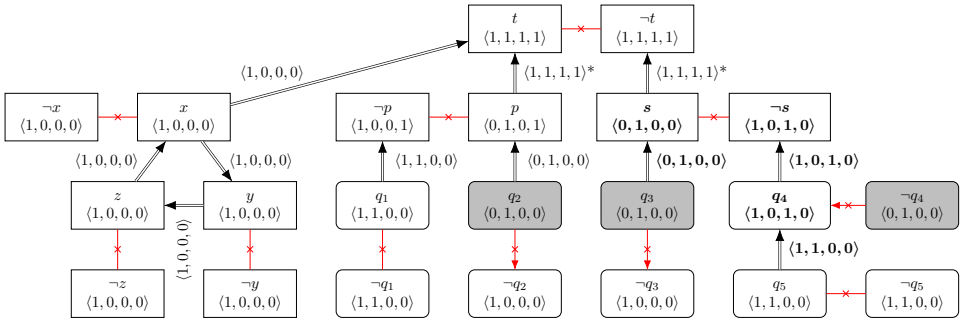


Figure 3.18: Argumentation theory illustrating the $\langle u, d, o, b\rangle$-labelling of Stability-Label after the fifth iteration of the while-loop, where the rules $q_3 \Rightarrow s$, $q_5 \Rightarrow q_4$ and $q_4 \Rightarrow \neg s$ have been selected from todo-set. All literals that have been considered for relabelling by Relabel-Literal are illustrated in boldface. The rules in todo-set are indicated with an asterisk.

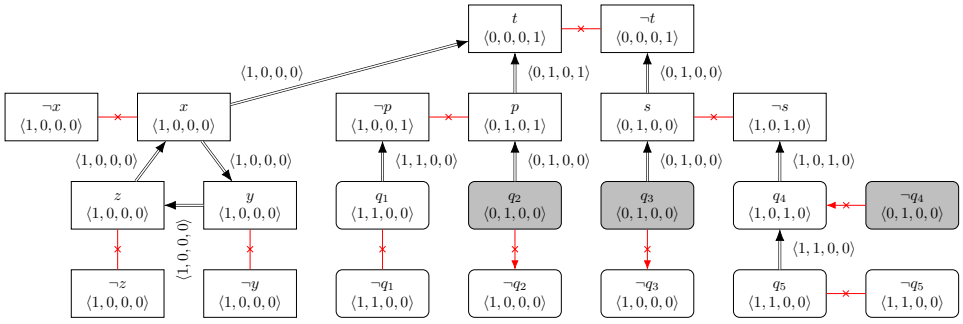


Figure 3.19: Argumentation theory illustrating the $\langle u, d, o, b\rangle$-labelling of Stability-Label after the final iteration of the while-loop, where the rules $p \Rightarrow t$ and $s \Rightarrow \neg t$ have been selected from todo-set. Note that the resulting labels for literals correspond to the desired labels as illustrated in Figure 3.13.

In our running example, we have seen that our algorithm STABILITY-LABEL is able to find the accurate stability labelling for a specific argumentation theory. Keep in mind that this is not the case for every possible argumentation theory: STABILITY-LABEL is a heuristic algorithm, in which we sacrificed perfect accuracy for fast computation. In the next subsections, we will prove properties of the proposed algorithm: we will show that it is sound, identify conditions under which it is complete and prove that it runs in polynomial time.

3.7.4. Soundness

In this section, we show that STABILITY-LABEL is sound: given an argumentation theory T and a set of queryables $\mathcal{Q}$, if the algorithm labels l as stable in T w.r.t. $\mathcal{Q}$, then l is stable in T w.r.t. $\mathcal{Q}$. Soundness is a valuable property in general, but specifically in our applications, because it ensures that the decision support system only gives a piece of advice if no additional information can change the conclusion. As a first step, we show that the preprocessing algorithm is sound in the sense that each literal that is labelled $\langle 1,0,0,0\rangle$ is stable-unsatisfiable in T w.r.t. $\mathcal{Q}$.

Lemma 3.4 (Soundness preprocessing step)**.** *Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and let $\mathcal{Q}$ be a set of queryables. Furthermore let L_p be the labelling obtained by PREPROCESS (Algorithm 3) on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. For each $l \in \mathcal{L}$: if $L_p[l] = \langle 1,0,0,0\rangle$, then l is stable-unsatisfiable in T w.r.t. $\mathcal{Q}$.*

Proof sketch. This can be shown by contraposition: if l is not stable-unsatisfiable in T w.r.t. $\mathcal{Q}$, then there is some T' such that $T \sqsubseteq_{\mathcal{Q}} T'$ and there is an argument A for l in $Arg_{T'}$. If A is observation-based, then $L_p[l] = \langle 1,1,1,1\rangle$ is assigned in Algorithm 3 line 3; if A is rule-based, then there is some rule r for l such that all $a \in \texttt{ants}(r)$ are labelled $L_p[a] = \langle 1,1,1,1\rangle$, hence the label $L_p[r] = \langle 1,1,1,1\rangle$ is assigned by line 13. In both cases, $L_p[l] \neq \langle 1,0,0,0\rangle$. □

The full proof can be consulted in Appendix B.1.4. Given that the preprocessing step is sound, we now need to show that the remainder of the STABILITY-LABEL algorithm is sound as well. Below we give a proof sketch; for the full proof, we refer to Appendix B.1.5.

Proposition 3.8 (Soundness stability labelling)**.** *Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and $\leq = \emptyset$, let $\mathcal{Q}$ be the set of queryable literals and let L be the labelling obtained by STABILITY-LABEL (Algorithm 6) on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$.*

- *If $L[l] = \langle 1,0,0,0\rangle$ then l is stable-unsatisfiable in T w.r.t. $\mathcal{Q}$;*
- *if $L[l] = \langle 0,1,0,0\rangle$, then l is stable-defended in T w.r.t. $\mathcal{Q}$;*
- *if $L[l] = \langle 0,0,1,0\rangle$, then l is stable-out in T w.r.t. $\mathcal{Q}$; and*
- *if $L[l] = \langle 0,0,0,1\rangle$, then l is stable-blocked in T w.r.t. $\mathcal{Q}$.*

Proof sketch. The following items will be used in the soundness proof sketch and can be shown by induction (cf. Lemmas B.5, B.1, B.4 and B.10 in the appendix):

1. For each $r \in \mathcal{R}$ labelled $\neg L[r].d$ and $\neg L[r].b$: for each T' such that $T \sqsubseteq_Q T'$, each argument with top rule r is attacked and therefore defeated by some argument in $G(T')$;
2. For each $l \in \mathcal{L}$ labelled $\neg L[l].u$: there is an argument for l in $Arg_{T'}$ for each T' such that $T \sqsubseteq_Q T'$;
3. For each $r \in \mathcal{R}$ labelled $\neg L[r].u$ and $\neg L[r].o$: if $\texttt{cons}(r) \notin \mathcal{Q}$ then for each T' such that $T \sqsubseteq_Q T'$ there is an argument with top rule r in $Arg_{T'}$ that is not attacked and therefore not defeated by any argument in $G(T')$;
4. For each $l \in \mathcal{L}$ labelled $\neg L[l].d$: there is no argument for l in $G(T')$ for any T' such that $T \sqsubseteq_Q T'$.

There are four cases in which a literal $l \in \mathcal{L}$ is labelled stable by L (see Lemmas B.7, B.8, B.9 and B.11 in the appendix).

- If $L[l] = \langle 1,0,0,0\rangle$ then it was already labelled as such by PREPROCESS ($L_p[l] = \langle 1,0,0,0\rangle$), because each rule r for l must have been labelled $L[r] = \langle 1,0,0,0\rangle$ and either $l \notin \mathcal{Q}$ or there is an $l' \in \bar{l}$ such that $l' \in \mathcal{K}$. By Lemma 3.4 l is stable-unsatisfiable in T w.r.t. $\mathcal{Q}$.
- Suppose $L[l] = \langle 0,1,0,0\rangle$; if l's label was assigned in the $(k+1)$'th iteration then there is a rule r for l that is labelled $L[r] = \langle 0,1,0,0\rangle$ and for each r' for l' where $l' \in \bar{l}$, r' is labelled $\neg L[r'].d$ and $\neg L[r'].b$. Each $a \in \texttt{ants}(r)$ is labelled $\langle 0,1,0,0\rangle$ in or before the k'th iteration; hence by induction each $a \in \texttt{ants}(r)$ is stable-defended in T w.r.t. $\mathcal{Q}$. Furthermore, by Item 1 above: for each T' such that $T \sqsubseteq_Q T'$, each rule-based argument for each $l' \in \bar{l}$ is attacked and therefore defeated by an argument in $G(T')$. Consequently, there is an argument for l, with top rule r, in $G(T')$ of each T' such that $T \sqsubseteq_Q T'$. So l is stable-defended in T w.r.t. $\mathcal{Q}$.
- Suppose $L[l] = \langle 0,0,1,0\rangle$. We distinguish two cases. If $l \in \mathcal{Q}$ then some $l' \in \bar{l}$ is in $\mathcal{K}$. Alternatively, $l \notin \mathcal{Q}$; then all rules r for l are labelled $\neg L[r].d$ and $\neg L[r].b$. By Item 1 above all arguments based on rules for l must be attacked and therefore defeated by an argument in $G(T')$ for each T' such that $T \sqsubseteq_Q T'$. Furthermore, by Item 2 above there is an argument for l in each T' such that $T \sqsubseteq_Q T'$. To conclude, l is stable-out in T w.r.t. $\mathcal{Q}$.
- If $L[l] = \langle 0,0,0,1\rangle$ then $l \notin \mathcal{Q}$ and there is some rule r for l that is labelled $\neg L[r].u$ and $\neg L[r].o$. By Item 3 above, there is an argument with top rule r in each T' such that $T \sqsubseteq_Q T'$ that is not attacked and therefore not defeated by any argument in $G(T')$. Furthermore, by Item 4 above there is no argument for l in $G(T')$ for any T' such that $T \sqsubseteq_Q T'$. As a result, l is stable-blocked in T w.r.t. $\mathcal{Q}$. □

3.7.5. Conditional completeness

Next, we consider the completeness of our algorithm. As we illustrate in Example 3.21, STABILITY-LABEL is not complete for all argumentation theories.

Example 3.21 (Fraud example continued)**.** We return to the example on fraud from Section 3.6.3. Consider the argumentation theory $T = (AS, \mathcal{K})$ where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ where $\leq = \emptyset$ and $\mathcal{L}$, $\mathcal{R}$ and $\mathcal{Q}$ correspond to the language, rules and queryables in Figure 3.11, but $\mathcal{K} = \{\neg sm, rm\}$. STABILITY-LABEL does not label f stable: it expects a future argument for f with top rule $cd, \neg rd, d \Rightarrow f$, where the argument for cd has top rule $sp, \neg b \Rightarrow cd$ and the argument for $\neg rd$ has top rule $\neg rp, b \Rightarrow rd$. However, this argument would require both b and $\neg b$ to be in the knowledge base, which violates the consistency criterion. In fact, for each T' such that $T \sqsubseteq_Q T'$ there is no argument for f in $Arg_{T'}$, so f should be labelled $\langle 1, 0, 0, 0\rangle$.

Example 3.21 shows that there are argumentation theories where the STABILITY-LABEL algorithm wrongfully takes the possibility into account that there exists an argument for a literal in a future argumentation theory. Specifically, this issue is caused by an *inconsistent potential argument*, which we define next.

Definition 3.21 (Potential argument)**.** Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$. A *potential argument* A^p on the basis of an argumentation theory T given a set of queryables $\mathcal{Q}$ is a structure obtainable by applying one or more of the following steps finitely many times:

- c is an *observation-based potential argument* if $c \in \mathcal{Q}$ and for each $c' \in \bar{c}$: $c' \notin \mathcal{K}$.
  $\texttt{prem}(A^p) = \{c\}$; $\texttt{conc}(A^p) = c$; $\texttt{sub}(A^p) = \{c\}$.
- $A_1, \ldots, A_m \Rightarrow c$ is a *rule-based potential argument* if there is a rule $c_1, \ldots, c_m \Rightarrow c$ in $\mathcal{R}$ and for each $i \in [1 \,..\, m]$: A_i is a potential argument on the basis of T given $\mathcal{Q}$ and $\texttt{conc}(A_i) = c_i$.
  $\texttt{prem}(A^p) = \texttt{prem}(A_1) \cup \ldots \cup \texttt{prem}(A_m)$; $\texttt{conc}(A^p) = c$; $\texttt{sub}(A^p) = \texttt{sub}(A_1) \cup \ldots \cup \texttt{sub}(A_m) \cup \{A^p\}$; $\texttt{top-rule}(A^p) = r$.

We denote the set of potential arguments on the basis of T given $\mathcal{Q}$ as $P_{\mathcal{Q}}(T)$ and refer to a potential argument with conclusion c as "a potential argument for c". Given some $A^p \in P_{\mathcal{Q}}(T)$, A^p is *inconsistent* iff there exist $p_1, p_2 \in \texttt{prem}(A^p)$ such that $p_1 \in \overline{p_2}$. Given some set of potential arguments $S \subseteq P_{\mathcal{Q}}(T)$, S is *inconsistent* iff there is some $a \in \mathcal{L}$ such that there is some $a' \in \bar{a}$ and $\{a, a'\} \in \{\texttt{prem}(A^p) \mid A^p \in S\}$.

Just like arguments, potential arguments can be in conflict. Next, we define p-attacks, which are rebuttal attacks between potential arguments.

Definition 3.22 (P-attack)**.** Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$. For two potential arguments $A^p, B^p \in P_{\mathcal{Q}}(T)$ we say that A^p *p-attacks* B^p iff A^p's conclusion is c and:

- *attack on conclusion*: there is some $c' \in \bar{c}$ such that c' is the conclusion of B^p and $c' \notin \mathcal{K}$.
- *attack on subargument*: there is some $c' \in \bar{c}$ such that c' is the conclusion of a subargument B' of B^p such that $B' \neq B^p$ and $c' \notin \mathcal{K}$.

We write A^p *p-attacks* B^p *on* B' if A^p attacks B^p, $B' \in \texttt{sub}(B^p)$ and $\texttt{conc}(A^p) \in \overline{\texttt{conc}(B')}$.

Note that, for a given argumentation theory T, each argument on the basis of T or some future argumentation theory is a potential argument in $P_{\mathcal{Q}}(T)$ since $\mathcal{K} \subseteq \mathcal{Q}$ and for each $(AS, \mathcal{K}')$ in the future argumentation theories of T: $\mathcal{K}' \subseteq \mathcal{Q}$. On the other hand, there may be a potential argument $A^p \in P_{\mathcal{Q}}(T)$ such that there is no T' such that $T \sqsubseteq_{\mathcal{Q}} T'$ and $A^p \in Arg_{T'}$, but then A^p must be inconsistent; we encountered this situation in Example 3.21, where the only potential argument for f in $P_{\mathcal{Q}}(T)$ was inconsistent and therefore not an argument in any future argumentation theory.

Being a heuristic algorithm, Stability-label heuristically reasons with potential arguments rather than with arguments in future argumentation theories. We explicate this in the next lemma, which lists the conditions under which one or more booleans of the stability label are labelled negative. These six items can be shown by induction. For the proofs, we refer to Appendix B.1.6.

Lemma 3.5 (Conditions for labelling)**.** *Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and $\leq = \emptyset$ and let L be the labelling after executing the Stability-label algorithm on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. Let L_p be the labelling after executing Preprocess on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. Let $l \in \mathcal{L}$ be a literal. Then:*

1. *$L_p[l] = \langle 1, 0, 0, 0 \rangle$ iff there is no potential argument A^p for l in $P_{\mathcal{Q}}(T)$.*
2. *If there is an argument for l in Arg_T, then $\neg L[l].u$.*
3. *If each potential argument for l in $P_{\mathcal{Q}}(T)$ is p-attacked by an observation-based argument in Arg_T, then $\neg L[l].d$ and $\neg L[l].b$.*
4. *There is an argument A for l in Arg_T and there is no observation-based potential argument in $P_{\mathcal{Q}}(T)$ that p-attacks A iff $\neg L[l].u$ and $\neg L[l].o$.*
5. *If each potential argument A^p for l in $P_{\mathcal{Q}}(T)$ is p-attacked by an argument B in Arg_T and there is no observation-based potential argument in $P_{\mathcal{Q}}(T)$ that p-attacks B, then $\neg L[l].d$.*
6. *If there is an argument A for l in Arg_T and each potential argument B^p in $P_{\mathcal{Q}}(T)$ that p-attacks A is p-attacked by an observation-based argument in Arg_T, then $L[l] = \langle 0, 1, 0, 0 \rangle$.*

Because of this heuristic of reasoning with potential arguments, Stability-label efficiently recognises many, though not all situations in which a literal is stable. From Lemma 3.5 we can now derive in which situations literals are stable in the argumentation theory, but not labelled as such. Before formally specifying this in Proposition 3.9, we give some intuitions on the three complications of using potential arguments instead of arguments in future argumentation theories:

1. Potential arguments are not required to be consistent, while inconsistent potential arguments cannot be derived in any future argumentation theory – recall Example 3.21.
2. Consistent potential arguments do exist as arguments in some future argumentation theory, but in some situations Stability-label needs to reason with combinations of potential arguments, which may not be derivable from the same future argumentation theory. We will see this in Example 3.22.

3. The existence of a potential argument in a future argumentation theory may cause other potential arguments to become arguments in the theory as well. This is illustrated in Example 3.23.

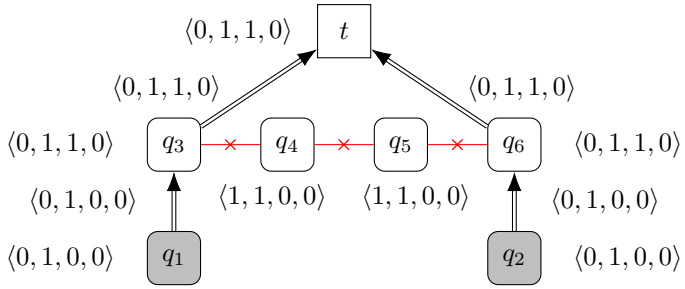


Figure 3.20: t is stable-defended in T, but it is not labelled as such because each argument for t in Arg_T is p-attacked by an observation-based potential argument in $P_{\mathcal{Q}}(T)$. However, there is no T' such that $T \sqsubseteq_Q T'$ and all arguments for t are attacked by an observation-based argument in $Arg_{T'}$. (Proposition 3.9 Case 2a) Note that this inconsistency can be between more than two arguments; for example consider the argumentation theory obtained by adding to $\mathcal{L}$ and $\mathcal{Q}$: q_7, q_8, q_9, q_{10}, adding q_7 to $\mathcal{K}$, contradictories: $\overline{q_6} = \{q_5, q_8\}$, $\overline{q_8} = \{q_6, q_9\}$, $\overline{q_9} = \{q_8, q_{10}\}$, $\overline{q_{10}} = \{q_8\}$ and add the rules $q_7 \Rightarrow q_{10}$ and $q_{10} \Rightarrow t$ to $\mathcal{R}$.

Example 3.22 (Incomplete because of observation-unattackable literal)**.** In the argumentation theory $T = (AS, \mathcal{K})$ of Figure 3.20, there are two arguments for t in Arg_T, i.e. $A : [q_1 \Rightarrow q_3] \Rightarrow t$ and $B : [q_2 \Rightarrow q_6] \Rightarrow t$. These arguments are not attacked by any argument in Arg_T, so t is defended in T. There are potential arguments for q_4, which p-attacks A, and q_5, which p-attacks B. As a consequence, t is not labelled stable-defended by Stability-label, as the algorithm assumes that these potential arguments can become actual arguments in a future argumentation theory. There is, however, no T' such that $T \sqsubseteq_Q T'$ in which both q_4 and q_5 are in $Arg_{T'}$. Therefore, in each T' such that $T \sqsubseteq_Q T'$, some argument for t is not attacked by any argument in $Arg_{T'}$, so t is stable-defended in T w.r.t. $\mathcal{Q}$.

In the example above, t is stable-defended in T w.r.t. $\mathcal{Q}$ but not labelled as such because the attackers of arguments for t are not derivable from the same future argumentation theory. Formally, t is observation-unattackable in T w.r.t. $\mathcal{Q}$, as defined next.

Definition 3.23 (Observation-unattackable)**.** Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and $\leq = \emptyset$ and let $l \in \mathcal{L}$ be a literal. l is *unattackable* in T w.r.t. $\mathcal{Q}$ iff there is no consistent set of potential arguments $T^p \subseteq P_{\mathcal{Q}}(T)$ such that each argument for l in Arg_T is p-attacked by some potential argument in T^p. l is *observation-unattackable* in T w.r.t. $\mathcal{Q}$ iff there is no consistent set of observation-based potential arguments $T^p \subseteq P_{\mathcal{Q}}(T)$ such that each argument for l in Arg_T is p-attacked by some potential argument in T^p.

A final cause of Stability-label's incompleteness is related to the fact that updating a knowledge base so as to add a potential argument to the set of arguments may introduce additional arguments.

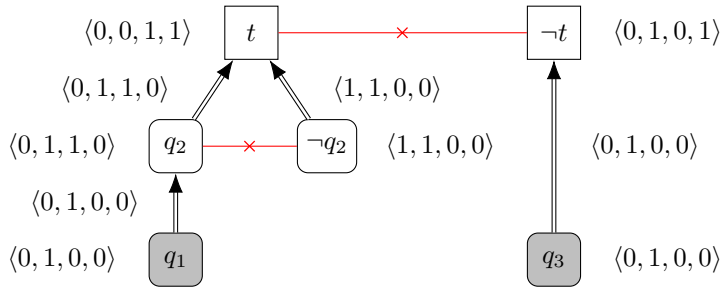


Figure 3.21: For each T' such that $T \sqsubseteq_Q T'$: t and $\neg t$ are blocked in T', but not labelled as such. t is labelled $L[l].o$ because the argument $[q_1 \Rightarrow q_2] \Rightarrow t$ is p-attacked by the observation-based potential argument $\neg q_2$ of which the introduction in T forces a new argument for t (Proposition 3.9 Case 2b). $\neg t$ is labelled $L[\neg t].d$ because there is an argument $q_3 \Rightarrow \neg t$ for $\neg t$ that is attacked by the argument $[q_1 \Rightarrow q_2] \Rightarrow t$, which is in turn p-attacked by the observation-based potential argument $\neg q_2$, but the introduction of $\neg q_2$ in T forces $\neg q_2 \Rightarrow t$, which attacks $q_3 \Rightarrow \neg t$ (Proposition 3.9 Case 5c).

Example 3.23 (Incomplete because of forcing argument)**.** Figure 3.21 visualises an argumentation theory $T = (AS, \mathcal{K})$ where $\mathcal{Q} = \{q_1, \neg q_1, q_2, \neg q_2, q_3, \neg q_3\}$ and $\mathcal{K} = \{q_1, q_3\}$. Both in the current argumentation theory and in the two future argumentation theories $(AS, \mathcal{K} \cup \{q_2\})$ and $(AS, \mathcal{K} \cup \{\neg q_2\})$, t is blocked because there are rule-based arguments for both t and $\neg t$. However, Stability-label does not label t as stable because it reckons with the possibility that $[q_1 \Rightarrow q_2] \Rightarrow t$, the only argument for t in Arg_T, is attacked by $\neg q_2$ in a future argumentation theory. The algorithm does not take into account the side effect of adding $\neg q_2$ to the knowledge base, that is: the inclusion of the argument $\neg q_2 \Rightarrow t$, which attacks and is attacked by the argument $q_3 \Rightarrow \neg t$.

Next, we formally define this phenomenon as potential arguments forcing arguments.

Definition 3.24 (Forcing arguments)**.** Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and $\leq = \emptyset$. Given two potential arguments A^p, B^p in $P_{\mathcal{Q}}(T)$, the introduction of A^p in T *forces* B^p iff $B^p \in Arg_{(AS, \mathcal{K} \cup \mathtt{prem}(A^p))}$. We say that the introduction of A^p in T *forces a new argument* for l iff there is some B^p for l in $P_{\mathcal{Q}}(T)$ such that the introduction of A^p in T forces B^p and $B^p \notin Arg_T$.

At this point, we have addressed the three causes of Stability-label's incompleteness: inconsistent potential arguments, observation-unattackable literals and potential arguments forcing arguments. We use these notions to formally specify the situations in which a literal is stable, but not labelled as such.[5] The full proof of Proposition 3.9, as well as additional examples illustrating the cases of incompleteness, can be found in Appendix B.1.6.

[5]Note that these conditions for incompleteness cannot be checked in polynomial time. In fact, under the assumption that P $\neq$ NP it is generally impossible to check any exact incompleteness conditions for any heuristic stability algorithm in polynomial time: if that would be possible, then this could be used to create an exact polynomial algorithm for the stability problem.

Proposition 3.9 (Conditional completeness stability labelling)**.** *Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and $\leq = \emptyset$ and let L be the labelling after executing the* Stability-label *algorithm on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. Given a literal $l \in \mathcal{L}$, if l is stable in T but l is not labelled stable by L, then some of the following five cases applies:*

1. *l is stable-unsatisfiable in T and each potential argument for l in $P_{\mathcal{Q}}(T)$ is inconsistent.*
2. *l is stable-defended or stable-blocked in T and:*
   (a) *l is observation-unattackable in T w.r.t. $\mathcal{Q}$;*[6] *or*
   (b) *some argument A for l in Arg_T is p-attacked by an observation-based potential argument B^p such that the introduction of B^p in T forces a new argument for l.*
3. *l is stable-defended in T and there is an argument A for l in Arg_T that is p-attacked by a potential argument in $P_{\mathcal{Q}}(T)$ that is not p-attacked by an argument in Arg_T and:*
   (a) *each potential argument in $P_{\mathcal{Q}}(T)$ p-attacking A that is not p-attacked by an argument in Arg_T is inconsistent; or*
   (b) *there is a consistent potential argument B^p in $P_{\mathcal{Q}}(T)$ p-attacking A that is not p-attacked by an argument in Arg_T, but the introduction of B^p in T forces a new argument for l; or*
   (c) *l is unattackable in T w.r.t. $\mathcal{Q}$.*
4. *l is stable-out in T and each potential argument for l in $P_{\mathcal{Q}}(T)$ that is not p-attacked by an observation-based argument in Arg_T has inconsistent premises; or*
5. *l is stable-blocked in T and there is a potential argument A^p for l in $P_{\mathcal{Q}}(T)$ such that each argument in Arg_T p-attacking A^p is p-attacked by an observation-based potential argument C^p in $P_{\mathcal{Q}}(T)$ and:*
   (a) *A^p is inconsistent; or*
   (b) *the introduction of A^p in T forces an argument attacking A^p; or*
   (c) *the introduction of C^p in T forces an argument that p-attacks A^p.*

Proof sketch. There are four cases in which a literal $l \in \mathcal{L}$ is stable, but not labelled as such by L.

- If l is *stable-unsatisfiable* in T but $L[l] \neq \langle 1, 0, 0, 0 \rangle$ then by Lemma 3.5 Item 1, each potential argument for l in $P_{\mathcal{Q}}(T)$ is inconsistent *(Case 1)*.
- If l is *stable-defended* in T but $L[l] \neq \langle 0, 1, 0, 0 \rangle$ then:

[6] This condition is specifically required for argumentation systems that do not have classical negation as a contrariness function.

  - If $L[l].u$ or $L[l].o$ (although l is not unsatisfiable or out in any T' such that $T \sqsubseteq_{\mathcal{Q}} T'$) then either l is observation-unattackable in T *(Case 2a)* or there is an argument $A \in Arg_T$ for l that is p-attacked by some potential argument of which the introduction in T forces a new argument for l *(Case 2b)*.
  - If $\neg L[l].u$ and $\neg L[l].o$ then there must be some rule-based argument A for l in Arg_T that is not p-attacked by any observation-based argument in $P_{\mathcal{Q}}(T)$ (Lemma 3.5 Item 4). Still, l is not labelled stable-defended by L, so there must be at least one potential argument p-attacking A. Then either each potential argument p-attacking A is inconsistent *(Case 3a)*; there is some consistent B^p p-attacking A, but the introduction of this potential argument in T forces some new argument for l *(Case 3b)*; or it is not possible to attack all arguments for l in Arg_T at the same time *(Case 3c)*.

- If l is *stable-out* in T but $L[l] \neq \langle 0, 0, 1, 0 \rangle$ then there is an argument for l in Arg_T, so by Lemma 3.5 Item 2 $\neg L[l].u$ which implies $L[l].d$ or $L[l].b$. Then by Lemma 3.5 Item 3, each potential argument for l in $P_{\mathcal{Q}}(T)$ that is not p-attacked by an observation-based argument in Arg_T has inconsistent premises *(Case 4)*.

- If l is *stable-blocked* in T but $L[l] \neq \langle 0, 0, 0, 1 \rangle$ then:

  - If $L[l].u$ or $L[l].o$ (although l is not unsatisfiable or out in any T' such that $T \sqsubseteq_{\mathcal{Q}} T'$) then either l is observation-unattackable in T *(Case 2a)* or there is an argument $A \in Arg_T$ for l that is p-attacked by some potential argument of which the introduction in T forces a new argument for l *(Case 2b)*.
  - If $\neg L[l].u$ and $\neg L[l].o$ then $L[l].d$, so by Lemma 3.5 Item 5 there is a potential argument A^p for l such that each argument in Arg_T p-attacking A^p is p-attacked by some observation-based potential argument C^p in $P_{\mathcal{Q}}(T)$. Then either A^p is inconsistent *(Case 5a)*, or A^p is consistent, so there is a future argumentation theory T' from which A^p can be derived. However, given that l is blocked in T', either the introduction of A^p or C^p in T forces an argument that attacks A^p *(Case 5b/5c)*. □

3.7.6. Time complexity

In the next two sections, we discuss the computation time of our proposed algorithm. Here we discuss the worst-case time complexity of Stability-label by a formal complexity analysis; in Section 3.9 we will complement this analysis with a series of experiments.

The Stability-label algorithm consists of a preprocessing part (Algorithm 3) and a final labelling part (Algorithm 6 line 2–18). In the next two lemmas, we analyse the individual parts and give proof sketches. Full proofs are available in Appendix B.1.3.

Lemma 3.6 (Time complexity PREPROCESS)**.** *The time complexity of* PREPROCESS *is* $\mathcal{O}(|\mathcal{L}|^2 + |\mathcal{L}| \cdot |\mathcal{R}|^2)$.

Proof sketch. The time complexity of PREPROCESS is bounded polynomially by the size of the language and rule set of the argumentation system. The complexity is mainly caused by line 3 (which checks the at most $|\mathcal{L}|$ contradictories of each of the $|\mathcal{L}|$ literals) and line 11 (which checks the label of each of the at most $|\mathcal{L}|$ antecedents of each rule $r \in \mathcal{R}$ in each of the maximal $|\mathcal{R}|$ iterations of the while loop). □

Proposition 3.10 (Time complexity STABILITY-LABEL)**.** *The time complexity of* STABILITY-LABEL *is* $\mathcal{O}(|\mathcal{L}|^3 \cdot |\mathcal{R}| + |\mathcal{L}|^2 \cdot |\mathcal{R}|^2)$.

Proof sketch. The runtime of STABILITY-LABEL is particularly dominated by line 15, which relabels all contradictories of the conclusion of a rule that is relabelled. A single execution of this line requires labelling a literal, which in the worst case requires checking the presence of that literal and all of its (at most $|\mathcal{L}|$) contradictories in $\mathcal{Q}$ and $\mathcal{K}$, as well as the labels of all (max $|\mathcal{R}|$) rules for that literal or any of its contradictories. Line 15 is executed at most $|\mathcal{L}|$ times for each iteration of the while loop. The total number of iterations of the while loop equals the number of times a rule is added to TODO-SET. A rule is only added to TODO-SET if it was not yet visited (line 6) or if the label of one of its antecedents changed after a relabelling (line 13 or line 17). Since the label of a literal can change at most four times (i.e. at most four booleans can be turned to False), each rule $r \in \mathcal{R}$ is relabelled at most $5 \cdot |\texttt{ants}(r)|$ times. This means that line 15 is executed at most $5 \cdot |\mathcal{L}|^2 \cdot |\mathcal{R}|$ times. Then the total time required for all iterations of line 15 is at most $5 \cdot c \cdot (|\mathcal{L}|^3 \cdot |\mathcal{R}| + |\mathcal{L}|^2 \cdot |\mathcal{R}|^2)$, where c is a positive constant. □

3.8. Exact algorithms based on ASP

As an exact alternative to the heuristic algorithm proposed in the previous section, we now present algorithms for deciding stability and enumerating relevant updates based on the declarative paradigm of answer set programming (ASP) (Gelfond and Lifschitz, 1988; Niemelä, 1999). We do so since ASP has recently successfully been used to solve argumentation problems in ABA (Lehtonen et al., 2021) and ASPIC$^+$ (Lehtonen et al., 2020; Lehtonen et al., 2022).

3.8.1. Encoding justification status

First, we provide encodings for deciding the justification status of literals. These encodings will be used as parts of our approaches to deciding stability and relevance. We represent a given AT $T = (AS, \mathcal{K})$ and set of queryables $\mathcal{Q}$ in ASP as the following set of facts, denoted by $\texttt{AT}(T, \mathcal{Q})$.

In words, the fact $\textbf{axiom}(a)$ indicates that a is an axiom and $\textbf{queryable}(a)$ indicates that a is a queryable. The heads (consequents) and bodies (antecedents) of rules are expressed via predicates $\textbf{head}/2$ and $\textbf{body}/2$. A fact $\textbf{head}(r, b)$ indicates that b is the head/consequent of the rule r, and $\textbf{head}(r, b)$ indicates b is contained

in the body/antecedents of rule r. A fact $\mathbf{ctr}(a, b)$ indicates that a and b are contradictory to each other, and $\mathbf{preferred}(a, b)$ indicates that a is (not necessarily strictly) preferred to b.

$$\begin{aligned}\mathtt{AT}(T, \mathcal{Q}) = \; & \{\mathbf{axiom}(a). \mid a \in \mathcal{K}\} \cup \\ & \{\mathbf{queryable}(a). \mid a \in \mathcal{Q}\} \cup \\ & \{\mathbf{head}(r, b). \mid r \in \mathcal{R}, b = \mathtt{cons}(r)\} \cup \\ & \{\mathbf{body}(r, b). \mid r \in \mathcal{R}, b \in \mathtt{ants}(r)\} \cup \\ & \{\mathbf{ctr}(a, b). \mid a \in \mathcal{L}, b \in \overline{a}\} \cup \\ & \{\mathbf{preferred}(a, b). \mid a, b \in \mathcal{R}, a \geq b\}.\end{aligned}$$

The module π_{common} in Listing 5 computes what is derivable from a given knowledge base. Line 1 collects rules in the AT. Lines 2–4 determine which literals are derivable. The construction "$\mathbf{derivable}(L) : \mathbf{body}(R, L)$" is a conditional literal, representing a list containing $\mathbf{derivable}(L)$ for all L for which $\mathbf{body}(R, L)$ holds (Gebser et al., 2015). Line 5 assigns unsatisfiable literals based on derivability.

Listing 5 Module π_{common}

```
1 rule(R) :- head(R,_).
2 derivable(L) :- axiom(L).
3 derivable(L) :- head(R,L), applicable_rule(R).
4 applicable_rule(R) :- rule(R), derivable(L) : body(R,L).
5 unsat(L) :- not derivable(L), literal(L).
```

For deciding the justification status of literals, we present two separate ASP encodings for deciding the justification status of literals. First, we introduce π_{just}, which can compute the justification status for inputs without rule preferences. The other encoding, $\pi_{<\text{-}just}$, is able to take rule preferences into account. In comparison, π_{just} is conceptually simpler, it is practically more efficient (as we will see in Section 3.9) and it can be used in a comparison to the heuristic algorithm from Section 3.7. On the other hand, $\pi_{<\text{-}just}$ is more generally applicable.

Justification encoding for inputs without preferences

In this section, we introduce π_{just}. This module relies on Lemmas 3.7 and 3.8, which we prove next. By Lemma 3.7, in the context of ATs without preferences, arguments can only be in the grounded extension if they are defended by observation-based arguments. Proofs are given in Appendix B.2.

Lemma 3.7 (Specification "in" grounded extension)**.** *Let $T = (AS, \mathcal{K})$ be an argumentation theory with argumentation system $AS = (\mathcal{L}, \overline{\ }, \mathcal{R}, \leq)$ where $\leq = \emptyset$. An argument $A \in Arg_T$ is in the grounded extension $G(T)$ iff each argument defeating A is defeated by an observation-based argument.*

In addition, for argumentation theories without preferences, arguments can only be out if they are defeated by some observation-based argument.

Lemma 3.8 (Specification "out" arguments)**.** *Let $T = (AS, \mathcal{K})$ be an argumentation theory with argumentation system $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ where $\leq = \emptyset$. An argument $A \in Arg_T$ is defeated by an argument in the grounded extension $G(T)$ (A is "out"), iff A is defeated by an observation-based argument.*

These lemmas are used in the following proposition, in which we collect rules U not defeated by axioms, and defended rules D (i.e., rules that are not defeated by arguments based on U).

Proposition 3.11. *Let $T = (AS, \mathcal{K})$ be an argumentation theory with argumentation system $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ where $\leq = \emptyset$. Let $U = \{r \in \mathcal{R} \mid \overline{\texttt{cons}(r)} \cap \mathcal{K} = \emptyset\}$ and let $D = \{r \in \mathcal{R} \mid$ there is no argument for $\overline{\texttt{cons}(r)}$ based on $U\}$. Then for each l in $\mathcal{L}$ the following holds.*

- *l is defended w.r.t. T if and only if there is an argument A for l such that $\texttt{defrules}(A) \subseteq D$, and*
- *l is out w.r.t. T if and only if l is not unsatisfiable and there is no argument A for l with $\texttt{defrules}(A) \subseteq U$.*

Proof. First note that U contains exactly the rules that do not have an axiom as the contrary of their head, which implies that any argument B is not defeated by an observation-based argument if and only if B is based on U.

- *From left to right.* Assume that l is defended. Then by Definition 3.9 there is an argument A for l in $G(T)$. Suppose, towards a contradiction, that some $r \in \texttt{defrules}(A)$ is not in D. Then by definition of D there is some argument B for $\overline{\texttt{cons}(r)}$ based on U. By construction of U, B is not defeated by any observation-based argument, while B defeats A by Definition 3.6. This contradicts Lemma 3.7, so $\texttt{defrules}(A) \subseteq D$.

  From right to left. If there is an argument A for l such that $\texttt{defrules}(A) \subseteq D$ then, by construction of D and U, for each $r \in \texttt{defrules}(A')$, there is no argument for $\overline{\texttt{cons}(r)}$ that is not defeated by an observation-based argument. Thus l is defended by Lemma 3.7.

- *From left to right.* If l is out then l is not unsatisfiable and each argument for l is defeated by an argument in $G(T)$. By Lemma 3.8, each argument for l is defeated by an observation-based argument. Then there is no argument A for l with $\texttt{defrules}(A) \subseteq U$: if A would exist, then it would not be defeated by an observation-based argument.

  From right to left. Assume that l is not unsatisfiable and there is no argument A for l such that $\texttt{defrules}(A) \subseteq U$. As l is not unsatisfiable, there is an argument A' for l (with $\texttt{defrules}(A') \not\subseteq U$). This argument is defeated by an observation-based argument on a rule $r \in \texttt{defrules}(A') \setminus U$. Thus, A'

is defeated by $G(T)$ due to Lemma 3.8, hence the justification status of l is out. □

Proposition 3.11 is used in the ASP program $\pi_{just} = \pi_{common} \cup \Delta_{just}$ for assigning the justification status of each literal as the program.

Listing 6 Module Δ_{just}

```
1  defeated(R) :- head(R,X), axiom(Y), ctr(X,Y).
2  undefeated(L) :- axiom(L).
3  undefeated(L) :- head(R,L), undefeated_rule(R).
4  undefeated_rule(R) :- rule(R), not defeated(R), undefeated(L) :
       body(R,L).
5  out(L) :- derivable(L), not undefeated(L).
6  defeated_by_undefeated(R) :- head(R,X), undefeated(Y), ctr(X,Y).
7  defended(L) :- axiom(L).
8  defended(L) :- head(R,L), defended_rule(R).
9  defended_rule(R) :- not defeated_by_undefeated(R), rule(R), defended(L)
       : body(R,L).
10 blocked(L) :- literal(L), not unsat(L), not out(L), not defended(L).
```

The module Δ_{just} (Listing 6) assigns the justification statuses (other than unsatisfiable, which can already be obtained from π_{common}). Lines 1–4 of Δ_{just} collect the rules that are undefeated by axioms and the literals that can be concluded via them. Line 5 ensures that a literal is out if it is derivable but not concluded via undefeated rules. Lines 6–9 of Δ_{just} collect defended literals by considering derivations from rules that are undefeated by rules undefeated by axioms. Finally, the justification status of literals whose status is not unsatisfiable, defended or out is inferred to be blocked on Line 10.

The correctness of the ASP encoding is formalised by stating the correspondence between the justification status of literals in an AT and answer sets of our proposed encoding, based on the previous discussion and Proposition 3.11.

Proposition 3.12. *Given an AT $T = (AS, \mathcal{K})$ where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and $\leq = \emptyset$ and set of queryables $\mathcal{Q}$, for each literal $l \in \mathcal{L}$ and $j \in \{$unsatisfiable, defended, out, blocked$\}$, it holds that l is j in T if and only if there is an answer set to the program $\mathtt{AT}(T, \mathcal{Q}) \cup \pi_{just}$ that contains $\mathbf{j}(l)$.*

General justification encoding capturing preferences

Whereas the encoding π_{just} can be used to find the justification status in argumentation theories without rule preferences, it is not suitable for the more general case where such preferences are present. In this section, we therefore present an encoding $\pi_{<\text{-}just} = \pi_{common} \cup \Delta_{<\text{-}just}$ that is able to reason with rule preferences. It is based on the least fixed point of the defense operator from Definition 3.19 (in Section 3.5.1), from which the justification statuses can be inferred by Proposition 3.7.

The module $\Delta_{<\text{-}just}$ (Listing 7) encodes a sequence of applications of the defense operator with explicit indices (up to $|\mathcal{R}|/2$, per Proposition 3.6). Lines 1–2 detect

Listing 7 Module $\Delta_{<\text{-}just}$

```
1  strictly_less_preferred(X,Y) :- not preferred(X,Y), preferred(Y,X).
2  no_less_preferred(X,Y) :- not strictly_less_preferred(X,Y), rule(X),
       rule(Y).
3  n_rules(N) :- #count{X : rule(X)} = N.
4  max_iterations(N) :- n_rules(M), N=(M+1)/2.
5  iteration(0..N) :- max_iterations(N).
6  defended(X,I) :- axiom(X), iteration(I).
7  defended(X,I) :- head(R,X), defended_rule(R,I).
8  defended_rule(R,I) :- iteration(I), usable_rule(R,I), rule(R),
       defended(X,I) : body(R,X).
9  usable_rule(R,I) :- iteration(J), not defeated_by_undefeated(R,J),
       J+1=I, rule(R).
10 defeated(R,I) :- head(R,X), axiom(Y), ctr(X,Y), iteration(I).
11 defeated(R,I) :- head(R,X), defended_rule(DR,I), head(DR,Y), ctr(X,Y),
       no_less_preferred(DR,R).
12 derived_from_undefeated(X,I) :- axiom(X), iteration(I).
13 derived_from_undefeated(X,I) :- head(R,X), undefeated(R,I).
14 undefeated(R,I) :- iteration(I), rule(R), not defeated(R,I),
       derived_from_undefeated(X,I) : body(R,X).
15 defeated_by_undefeated(R,I) :- head(R,X), axiom(Y), ctr(X,Y),
       iteration(I).
16 defeated_by_undefeated(R,I) :- head(R,X), undefeated(IR,I), head(IR,Y),
       ctr(X,Y), no_less_preferred(IR,R).
17 defended_rule(R) :- defended_rule(R,N), max_iterations(N).
18 defended(X) :- defended(X,N), max_iterations(N).
19 out(L) :- derivable(L), not derived_from_undefeated(L,N),
       max_iterations(N).
20 blocked(L) :- literal(L), not unsat(L), not out(L), not defended(L).
```

when a rule is not strictly less preferred than another. Lines 3–5 set the iteration upper bound. For clarity of presentation, in the following we denote the set of defended rules on each iteration i by D^i and rules not defeated by D^i by U^i (corresponding to **defended_rule** and **undefeated** in $\Delta_{<\text{-}just}$). On Lines 6–7, a literal is deemed defended on iteration i if it can be derived by only using rules from D^i. On Lines 8–9, D^i is identified as the applicable rules that are not defeated by U^{i-1}, corresponding to the defense operator. The rules that D^i defeats are identified on Lines 10–11, following Definition 3.17: $r \in \mathcal{R}$ is defeated if either an axiom contradicts r or there is an argument based on D^i whose top rule is not less preferred than r and that concludes $\overline{\texttt{cons}(r)}$. Based on rules defeated by D^i, the undefeated rules U^i and the literals derivable from U^i are identified on Lines 12–14. Defeats from U^i are identified on Lines 15–16. Finally, Lines 17–20 label the literals based on the final iteration.

We state the correspondence between the justification status of literals in an AT that includes preferences between rules, and answer sets of our proposed encoding, based on the previous discussion and Proposition 3.7.

Proposition 3.13. *Given an AT $T = (AS, \mathcal{K})$ where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and set of queryables $\mathcal{Q}$, for each literal $l \in \mathcal{L}$ and $j \in \{$unsatisfiable, defended, out, blocked$\}$, it holds that l is j in T if and only if there is an answer set to the program $\texttt{AT}(T, \mathcal{Q}) \cup \pi_{<\text{-}just}$ that contains $\mathbf{j}(l)$.*

3.8.2. Encoding stability

The stability status of a literal is obtained by checking if there is a future AT where the literal is not j for a justification status j. We implement this by conjoining our encodings for justification with a non-deterministic guess of future ATs. Concretely, we guess a set of queryables (that is a superset of the axioms) and enforce their consistency via the following two rules.

$$\begin{aligned}\pi_{stab} = \{&\{\mathbf{axiom}(L)\} \leftarrow \mathbf{queryable}(L).,\\ &\leftarrow \mathbf{axiom}(L), \mathbf{axiom}(C), \mathbf{ctr}(L, C).\}.\end{aligned}$$

The encoding for checking stability without preferences is $\pi_{stab} = \Delta_{stab} \cup \pi_{just}$, and for the case with preferences it is $\pi_{<\text{-}stab} = \Delta_{stab} \cup \pi_{<\text{-}just}$. One can obtain the stability statuses of all literals via the cautious reasoning mode readily available in modern ASP solvers, directly computing the intersection of all answer sets to a given program: the literals that are $\mathbf{j}$ in the cautious solution are stable-j. If the stability status of a single literal is of interest, it can be decided with one ASP solver call by checking if there is a future theory where the literal is not $\mathbf{j}$. Concretely, we can add the constraint $\leftarrow \mathbf{j}(l).$ for literal $l \in \mathcal{L}$ and justification status $j \in \{$*unsatisfiable*, *defended*, *out*, *blocked*$\}$. The resulting program does not have an answer set iff l is stable-j.

We state the correspondence between the j-stability of literals in an AT and answer sets of our proposed encodings, based on the previous discussion.

Proposition 3.14. *Given an AT $T = (AS, \mathcal{K})$ where $AS = (\mathcal{L}, ^-, \mathcal{R}, \leq)$ and set of queryables $\mathcal{Q}$, a literal $l \in \mathcal{L}$ and a justification status $j \in \{$unsatisfiable, defended, out, blocked$\}$, it holds that l is stable-j in T if and only if there is no answer set to the program $\mathtt{AT}(T, \mathcal{Q}) \cup \pi_{<\text{-}stab} \cup \{\leftarrow \mathbf{j}(l).\}$. Further, if $\leq = \emptyset$ in T, then l is stable-j if and only if there is no answer set to the program $\mathtt{AT}(T, \mathcal{Q}) \cup \pi_{stab} \cup \{\leftarrow \mathbf{j}(l).\}$.*

Example 3.24. To illustrate our encodings, consider Example 3.16 and another AT, $T' = (AS, \{trusted\})$ and the question whether $\neg deception$ is stable-defended w.r.t. T'. Now our encoding $\pi_{<\text{-}stab}$ finds that $\neg deception$ is defended w.r.t. T' (in a single iteration, as $\overline{\neg deception} = deception$ is not derivable). However, as $\pi_{<\text{-}stab}$ (conjoined with the additional constraint "$\leftarrow$ **defended**$(\neg deception)$") non-deterministically guesses all consistent subsets of the queryables to be axioms, it has an answer set corresponding to the AT T illustrated in Example 3.16, in which $\neg deception$ is not defended. This is a counterexample to $\neg deception$ being stable-defended w.r.t. T'.

3.8.3. ASP-based relevance algorithm

For j-relevance, we propose ASP-based counterexample-guided abstraction refinement (CEGAR) (Clarke et al., 2004; Clarke et al., 2003) algorithms, using our stability encodings as subprocedures. As we have seen before in Section 2.5, in CEGAR, an abstraction (an overapproximation of the solution space) is iteratively refined by drawing candidates from this space and verifying if the candidate is an actual solution. Candidate solutions are computed with a solver. If there is a candidate solution, it is checked with another solver if there is a counterexample to the candidate being a solution. If there is no counterexample, the candidate is an actual solution. Otherwise, the abstraction is refined based on an analysis of the counterexample and the search is continued.

The algorithms proposed in this section are designed to find all queryables in $\mathcal{Q}$ that are j-relevant for l w.r.t. T, for a given argumentation theory T, set of queryables $\mathcal{Q}$, (topic) literal l and justification status j. For the abstraction, consistent sets of queryables $\mathcal{K}'$ are considered to be candidates. Checking if the candidate $\mathcal{K}'$ is j-relevant amounts to checking if l is stable-j w.r.t. $(AS, \mathcal{K}')$ but not w.r.t. $(AS, \mathcal{K}' \setminus \{q\})$. If and only if this is the case, q is j-relevant for l by Lemma 3.3. Otherwise, we can further narrow the search space, based on the following proposition. Informally, Proposition 3.15 states that (1) subsets of knowledge bases belonging to ATs in which a literal is not stable cannot form ATs in which the literal is stable, while (2) all supersets of knowledge bases belonging to ATs in which the literal is stable form ATs in which the literal is stable.

Proposition 3.15. *Let $T = (AS, \mathcal{K})$ be an AT, $\mathcal{Q}$ a set of queryables and j a justification status in $\{$unsatisfiable, defended, out, blocked$\}$. Given $l \in \mathcal{L}$ and $q \in \mathcal{Q}$ where $q \notin \mathcal{K}$ and $\overline{q} \cap \mathcal{K} = \emptyset$,*

1. *if $T' = (AS, \mathcal{K}') \sqsupseteq_Q T$ such that l is not stable-j w.r.t. T', then for each $\mathcal{K}'' \subseteq \mathcal{K}'$, l is not stable-j w.r.t. $(AS, \mathcal{K}'')$;*
2. *if $T' = (AS, \mathcal{K}') \sqsupseteq_Q T$ such that l is stable-j w.r.t. T' and $q \notin \mathcal{K}'$, then for*

each consistent $\mathcal{K}'' \supseteq \mathcal{K}'$, l is stable-j w.r.t. $(AS, \mathcal{K}'' \setminus \{q\})$.

Proof. We prove both items separately.

1. For the first item, suppose that $T' = (AS, \mathcal{K}')$ is a future AT with $T \sqsubseteq_Q T'$ such that l is not stable-j w.r.t. T'. Then there is some T^* with $(AS, \mathcal{K}') \sqsubseteq_Q T^*$ such that l is not j w.r.t. T^*. For each $\mathcal{K}'' \subseteq \mathcal{K}'$ it must be that $(AS, \mathcal{K}'') \sqsubseteq_Q T^*$ as well. Therefore l is not stable-j w.r.t. $(AS, \mathcal{K}'')$ and $\mathcal{Q}$.
2. For the second item, suppose that $T' = (AS, \mathcal{K}')$ is a future AT with $T \sqsubseteq_Q T'$ such that l is stable-j w.r.t. T' and $q \notin \mathcal{K}'$. Then for each $\mathcal{K}''$ such that $\mathcal{K}' \subseteq \mathcal{K}''$, l is stable-j w.r.t. $(AS, \mathcal{K}'')$, and since $q \notin \mathcal{K}'$, $\mathcal{K}'' \setminus \{q\} \supseteq \mathcal{K}'$. Thus, l is stable-j w.r.t. $(AS, \mathcal{K}'' \setminus \{q\})$. □

Algorithm 7 ASP-based CEGAR algorithm for finding all j-relevant queryables for $l \in \mathcal{L}$ w.r.t. an argumentation theory $T = (AS, \mathcal{K})$ where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and set of queryables $\mathcal{Q}$ and justification status j (unsatisfiable, defended, out or blocked).

```
1:  procedure RELEVANCE(T, Q, l, j)
2:      π_c := AT(T, Q) ∪ π_candidate
3:      π_v := AT(T, Q) ∪ π_(<-)stab ∪ {← j(l).}
4:      R = ∅
5:      while I := π_c is satisfiable do
6:          K' := {q' ∈ Q | axiom(q') ∈ I}
7:          if C := π_v ∪ {axiom(q') | q' ∈ K'} is satisfiable
8:              then π_c := π_c ∪ no_subsets(C)
9:          else
10:             for q ∈ K' \ R do
11:                 if π_v ∪ {axiom(q') | q' ∈ K' \ {q}} is satisfiable
12:                     then R := R ∪ {q}
13:             π_c := π_c ∪ no_supersets(K')
14:     return R
```

Algorithm 7 explicates how all j-relevant queryables are found for a given literal. The algorithm uses two programs, π_c and π_v. The program π_c is initialised as a guess of a consistent set of queryables within the argumentation theory (Line 2). Concretely, $\pi_{candidate}$ is defined as follows.

$$\pi_{candidate} = \{\{\mathbf{axiom}(L)\} \leftarrow \mathbf{queryable}(L), \\ \leftarrow \mathbf{axiom}(L), \mathbf{axiom}(C), \mathbf{ctr}(L, C)\}.$$

In Line 3, the program π_v is constructed by combining $\mathtt{AT}(T, \mathcal{Q})$ with either π_{stab} (if no preferences need to be taken into account) or $\pi_{<\text{-}stab}$ (if preferences must be taken into account), as well as a constraint enforcing that literal l is not stable-j.

R in Line 4 is a set that will be filled with relevant queryables. Line 5 tries to find a consistent set of queryables $\mathcal{K}' \subseteq \mathcal{Q}$, corresponding to answer sets of π_c. As long

as such set exists, $\mathcal{K}'$ is extracted in Line 6. Line 7 then checks if l is stable-j with respect to $(AS, \mathcal{K}')$. If l is *not* stable-j w.r.t. $\mathcal{K}'$ (which is the case if and only if C, constructed in Line 7, is satisfiable), then all subsets of $\mathcal{K}'$, including $\mathcal{K}'$ itself, can be discarded due to the first item of Proposition 3.15. This refinement is performed in Line 8 and uses the following ASP constraint, where $q_1, \ldots, q_n$ are the queryables *not* contained in the counterexample C.

$$no\_subsets(C) = \{\leftarrow not\ \mathbf{axiom}(q_1), \ldots, not\ \mathbf{axiom}(q_n).\}$$

Otherwise, that is, if l is stable-j w.r.t. $\mathcal{K}'$ and Line 9 is reached, it is checked for each queryable in the candidate whether l is stable-j w.r.t. the candidate *without* this queryable (Lines 10–11). If *not*, this queryable is j-relevant for l due to Lemma 3.3 and added to R in Line 12. After checking this for all queryables in the candidate, supersets of the candidate can be ruled out from further consideration (Line 13). This follows from the second item of Proposition 3.15: l is stable-j w.r.t. $\mathcal{K}'' \setminus \{q\}$ for any consistent $\mathcal{K}'' \supseteq \mathcal{K}'$ and any $q \notin \mathcal{K}'$, so no superset of the candidate $\mathcal{K}''$ can show the relevance of any further queryable. This refinement is accomplished on Line 13 by the ASP constraint $no\_supersets(\mathcal{K}')$ where $q_1, \ldots, q_m$ are the queryables contained in the candidate $\mathcal{K}'$.

$$no\_supersets(\mathcal{K}') = \{\leftarrow \mathbf{axiom}(q_1), \ldots, \mathbf{axiom}(q_m).\}$$

Finally, at the point where no candidates ($\mathcal{K}'$) are found by π_c, the search terminates and R is returned.

Example 3.25. Consider the AT $T = (AS, \{similar\_url\})$ from Example 3.1. To find out which queryables are defended-relevant for *deception*, the abstraction enforces that candidates are consistent and contain the axiom *similar_url*. The following execution of Algorithm 7 might take place:

1. Candidate $\mathcal{K}'_1 = \{similar\_url, too\_cheap, typosquatting\}$. Now *deception* is not stable-defended w.r.t. $\mathcal{K}'_1$ (it may still become blocked) and thus subsets of $\mathcal{K}'_1$ are ruled out as candidates in the future, and the search is continued.
2. Candidate $\mathcal{K}'_2 = \{similar\_url, too\_cheap, typosquatting, \neg trusted\}$. Line 7 finds that *deception* is stable-defended w.r.t. $\mathcal{K}'_2$, so Lines 10–13 are executed. For each queryable $q' \in \mathcal{K}'_2 \setminus R = \mathcal{K}'_2$ it is checked whether *deception* is stable-defended w.r.t. $\mathcal{K}'_2 \setminus \{q'\}$. Now *deception* is also stable-defended w.r.t. $\mathcal{K}'_2 \setminus \{too\_cheap\}$ and $\mathcal{K}'_2 \setminus \{typosquatting\}$. On the other hand, *deception* is not stable-defended w.r.t. $\mathcal{K}'_2 \setminus \{\neg trusted\}$ and thus $\neg trusted$ is added to R. Finally, supersets of $\mathcal{K}'_2$ (in this case only $\mathcal{K}'_2$ itself) are excluded from further consideration in the refinement in Line 13.
3. Candidate $\mathcal{K}'_3 = \{similar\_url, too\_cheap, \neg trusted\}$. Now *deception* is stable-defended w.r.t. $\mathcal{K}'_3$, and is not stable-defended w.r.t. $\mathcal{K}'_3 \setminus \{too\_cheap\}$. Thus *too_cheap* is added to R and supersets of $\mathcal{K}'_3$ are not considered as candidates in subsequent iterations.

4. Candidate $\mathcal{K}'_4 = \{similar\_url, typosquatting, \neg trusted\}$. Now *deception* is stable-defended w.r.t. $\mathcal{K}'_4$ and is not stable-defended w.r.t. $\mathcal{K}'_4 \setminus \{typosquatting\}$. Thus *typosquatting* is added to R and supersets of $\mathcal{K}'_4$ are not considered as candidates in subsequent iterations.
5. Candidate $\mathcal{K}'_5 = \{similar\_url, \neg typosquatting, \neg too\_cheap, trusted\}$. *deception* is not stable-defended w.r.t. $\mathcal{K}'_5$, so subsets of $\mathcal{K}'_5$ are not considered as candidates in subsequent iterations.
6. Some further iterations are required to exhaust the search space, but no further queryables are found to be defended-relevant for *deception*. The algorithm returns $R = \{\neg trusted, typosquatting, too\_cheap\}$.

Proposition 3.16. *Given an AT $T = (AS, \mathcal{K})$, a set of queryables $\mathcal{Q}$, a literal $l \in \mathcal{L}$, a justification status $j \in \{unsatisfiable, defended, out, blocked\}$, it holds that* RELEVANCE$(T, \mathcal{Q}, l, j)$ *equals the set of queryables that are j-relevant for l in T.*

Proof. For an AT $T = (AS, \mathcal{K})$, a set of queryables $\mathcal{Q}$, a literal $l \in \mathcal{L}$, a justification status $j \in \{\text{unsatisfiable, defended, out, blocked}\}$, let $R =$ RELEVANCE$(T, \mathcal{Q}, l, j)$.

First, assume that an arbitrary $q \in \mathcal{Q}$ is j-relevant for l w.r.t. T and $\mathcal{Q}$. Then by Lemma 3.3 there is a consistent set of queryables $\mathcal{K}' \subseteq \mathcal{Q}$ such that $q \in \mathcal{K}'$, l is stable-j w.r.t. $(AS, \mathcal{K}')$ and l is not stable-j w.r.t. $(AS, \mathcal{K}' \setminus \{q\})$. We show that q is added to R in all cases.

- Suppose that the algorithm considers $\mathcal{K}'$ as a candidate. Then after the check in Line 7 it finds that C is *not* satisfiable as l is stable-j w.r.t. $(AS, \mathcal{K}')$ and $\mathcal{Q}$. In Line 11 (which is reached unless q was added to R in some earlier iteration) it is checked if l is stable-j w.r.t. $(AS, \mathcal{K}' \setminus \{q\})$. As this is not the case, Line 12 is reached and q is added to R.
- Alternatively, $\mathcal{K}'$ is not considered as a candidate as it is excluded in Line 8 or 13. We show that $\mathcal{K}'$ cannot be excluded in Line 8. Towards a contradiction, suppose that $\mathcal{K}'$ was excluded in Line 8. Then there is some $\mathcal{K}'' \supseteq \mathcal{K}'$ such that l is not stable-j w.r.t. $(AS, \mathcal{K}'')$ and $\mathcal{Q}$, which implies that l is not stable-j w.r.t. $(AS, \mathcal{K}')$ and $\mathcal{Q}$ and thus contradicts our assumption on $\mathcal{K}'$. So $\mathcal{K}'$ is excluded in Line 13. Then there is some $\mathcal{K}''' \subseteq \mathcal{K}'$ such that l is stable-j w.r.t. $(AS, \mathcal{K}''')$ and $\mathcal{Q}$ (as Line 13 can only be reached if the condition in Line 7 returns False for $\mathcal{K}'''$). Given that l is *not* stable-j w.r.t. $(AS, \mathcal{K}')$ and $\mathcal{Q}$ and considering that $\mathcal{K}''' \subseteq \mathcal{K}'$, l cannot be stable-j w.r.t. $(AS, \mathcal{K}''')$ either. So Line 11 must have evaluated to True and q has been added to R in Line 12.

Finally, the algorithm must terminate as π_c is iteratively expanded with constraints that exclude one or more knowledge bases corresponding to future ATs (and the number of future ATs is finite). At this point, R is returned and includes q.

It remains to be shown that all queryables in R are j-relevant for l w.r.t. T and $\mathcal{Q}$. Let q be an arbitrary queryable in R. It must have been added in Line 12, so the condition in Line 7 must have evaluated False and the condition in Line 11 was True. Then there is a consistent set of queryables $\mathcal{K}' \subseteq \mathcal{Q}$ such that $q \in \mathcal{K}'$, l is

stable-j w.r.t. $(AS, \mathcal{K}')$ and l is not stable-j w.r.t. $(AS, \mathcal{K}' \setminus \{q\})$. By Lemma 3.3 q is j-relevant for l w.r.t. T and $\mathcal{Q}$. □

3.9. Empirical evaluation

In this section, we empirically evaluate the heuristic algorithm for stability proposed in Section 3.7 and the ASP-based algorithms for stability and relevance in Section 3.8. The implementation of all algorithms is available in open source at `https://github.com/DaphneOdekerken/stability_relevance_aspic`. The ASP-based algorithms use Clingo (Gebser et al., 2016) as the ASP solver and its incremental (multi-shot) features (Gebser et al., 2019) for implementing the CEGAR algorithms for relevance. All experiments were run using Clingo 5.4.1 on a Intel(R) Core(TM) i7-7820HQ CPU 2.90 GHz 16 GB RAM machine under a per-instance time limit of 60 seconds.

3.9.1. Benchmarks

As benchmarks, we consider both real-world and synthetic data.

For real-world benchmarks, we generated instances for the stability and relevance problems based on the argumentation system $AS = (\mathcal{L}, ^-, \mathcal{R}, \leq)$ and set of queryables $\mathcal{Q}$ used in a system for the intake of online trade fraud at the Dutch police (see also Chapter 5). In this setting, $|\mathcal{L}| = 60$, $|\mathcal{Q}| = 30$ and $|\mathcal{R}| = 43$. All literals in $\mathcal{L} \setminus \mathcal{Q}$ have a single contradictory: their negation. Considering the queryables in $\mathcal{Q}$, 19 queryables have a single contradictory; three literals have two contradictories; seven literals have three contradictories and one literal has four contradictories. All rules are equally preferred. Most rules have one (13) or two (14) antecedents; four rules have three antecedents; eight rules have four and the remaining four rules have five antecedents. The rules are defined in such a way that they form a tree-like structure, without (support) cycles. Each literal is assigned a layer, which informally is the largest number of rule applications to reach a queryable. Out of the 60 literals, 40 have layer 0 (this includes all 30 queryables); 6 have layer 1; 5 have layer 2; 6 have layer 3 and 3 have layer 4. To generate stability and relevance instances, we obtained knowledge bases by randomly sampling 100 consistent subsets of 7 queryables. We consider one literal as a "topic", for which the relevant updates are needed.

To further study the scalability of our implementations, we also consider *synthetic data*. For this, we generated argumentation theories and queryable sets that are parametrised by the size of the language $|\mathcal{L}|$. We generated a set with larger instances to test our stability algorithms, and a set with smaller ones to test the relevance algorithms. We generated ATs $T = (AS, \mathcal{K})$ with $AS = (\mathcal{L}, ^-, \mathcal{R}, \leq)$ and queryables $\mathcal{Q}$, with the following parameters.

- For the language size ($|\mathcal{L}|$), we generated instances for testing stability with $|\mathcal{L}| \in \{50, 100, 150, 200, 250, 500, 1000\}$ and for relevance with $|\mathcal{L}| \in \{50,$ 60, 70, 80, 90, 100$\}$.
- The number of rules was chosen to be $|\mathcal{R}| = \frac{3}{2} \cdot |\mathcal{L}|$.
- The body size of rules was chosen to be between 1 and 5, with one third of the

rules having one rule antecedent, another third having two antecedents and the remaining third was split equally to have three, four, or five antecedents.

- The literal layer distribution was selected by having $\frac{2}{3} \cdot |\mathcal{L}|$ literals with layer 0, each one-tenth of the literals for layers 1, 2, and 3, and the remaining ones with layer 4.
- The ratio between queryables and literals ($|\mathcal{Q}|/|\mathcal{L}|$) is 0.5.
- The ratio between axioms and queryables ($|\mathcal{K}|/|\mathcal{Q}|$) is 0.25.

Similar to the fraud data set, all queryables are on layer 0, that is, there are no rules for queryable literals. We randomly selected one topic from the literals in layer 4. For the stability benchmarks, we generated 100 different argumentation theories and queryable sets for each value of $|\mathcal{L}|$, for a total of 700 instances for the stability benchmarks and 600 instances for the relevance benchmarks.

3.9.2. Results

We continue with an overview of the results of the empirical evaluation.

Stability

Table 3.1 provides an overview of the results for the task of computing the stability of *each* literal in a given AT. The table provides the number of solved instances within the time limit of 60 seconds (as the primary metric), with mean run times over solved instances (as a secondary metric) for each number of literals $|\mathcal{L}|$ ranging from 50 to 1000.

We observe that on the real-world instances, both the ASP-based approaches and the heuristic algorithm terminate on average in a fraction of a second. On the synthetic instances, the heuristic algorithm takes 0.301 seconds on average for instances with $|\mathcal{L}| = 1000$, while the ASP-based algorithm that does not take preferences into account takes 0.106 seconds on average on the same instances. Note also that while the ASP-based algorithms are exact (always assigning correct stability statuses), the heuristic algorithm mislabelled in total 69 out of 117.521 stable literals in the synthetic instances. Overall, the exact ASP-based algorithm that does not take preferences into account outperforms the heuristic approach in both run times and in terms of accuracy. The mean runtime of the ASP-based algorithm that does take preferences into account for instances with $|\mathcal{L}| = 1000$ is 54 seconds, compared to less than one second for the no-preference algorithm. On the other hand, even for the preference-aware algorithm, the mean run times remain within 5 seconds up to $|\mathcal{L}| = 250$ on the synthetic instances, and the real-world instances are solved particularly fast, with a mean run time of a fraction of a second.

Relevance

Turning to relevance, Table 3.2 shows results for our approach for the task of enumerating all defended-relevant queryables for the given topic. For the real-world instances, the ASP algorithm that does not take preferences into account requires 0.144 seconds on average for this task. The ASP algorithm that takes preferences

		Stability instances solved (mean run time (s))					
Benchmark	$\lvert\mathcal{L}\rvert$	Heuristic		ASP		Pref-aware ASP	
Real	60	100	(0.007)	100	(0.009)	100	(0.032)
Synthetic	50	100	(0.009)	100	(0.009)	100	(0.074)
	100	100	(0.015)	100	(0.013)	100	(0.365)
	150	100	(0.023)	100	(0.017)	100	(0.855)
	200	100	(0.031)	100	(0.021)	100	(1.641)
	250	100	(0.041)	100	(0.025)	100	(2.523)
	500	100	(0.108)	100	(0.049)	100	(11.527)
	1000	100	(0.321)	100	(0.104)	100	(54.005)

Table 3.1: Number of solved instances and mean runtimes over solved instances for detecting stability of all literals. The number of instances per value of $|\mathcal{L}|$ is 100.

		Relevance instances solved (mean run time (s))			
Benchmark	$\lvert\mathcal{L}\rvert$	ASP		Preference-aware ASP	
Real	60	100	(0.150)	100	(0.297)
Synthetic	50	100	(0.144)	100	(0.894)
	60	100	(1.200)	94	(10.262)
	70	100	(4.370)	67	(12.486)
	80	91	(9.341)	56	(13.448)
	90	71	(11.162)	38	(29.667)
	100	58	(30.100)	11	(0.353)

Table 3.2: Number of solved instances and mean runtimes over solved instances for enumerating defended-relevance for a single topic. The number of instances per value of $|\mathcal{L}|$ is 100.

into account takes 0.894 seconds on average. Compared to the results for stability, the run times for relevance on the synthetic instances reflect the fact that deciding relevance is harder in terms of computational complexity. The algorithm that does not take preferences into account solves all instances with up to 70 literals within the run time limit. The preference-aware algorithm solves all instances with up to 50 literals and a majority of the instances with up to 80 literals.

3.9.3. Discussion

The results presented in the previous section show that, although the heuristic algorithm is able to estimate stability in a reasonable amount of time even for large instances, the ASP-based algorithm is faster, while being exact. This shows that ASP-based algorithms (which did not yet exist for ASPIC$^+$ when we started working on stability) are the better choice for determining stability in settings without rule preferences. In fact, further analysis of the runtime revealed that the ASP-based algorithm was faster than the parsing step of the heuristic algorithm alone. Whereas the ASP-based algorithm is based on Clingo, which is highly optimised and mainly

written in C++ and C, the heuristic algorithm is written in the typically slower language Python. This may explain why, at least for the instances that we tested, the exponential algorithm based on ASP is faster than the polynomial heuristic algorithm. Given these results, there is not much potential for speeding up the heuristic algorithm, at least not for the setting without preferences.

On the other hand, for the ASP-algorithm that does take preferences into account, we see an increase in running time. At this point there seems to be potential for improvement. Interesting directions for future research are an adaptation of the heuristic algorithm so that it can handle preferences and an adaptation of the ASP-algorithm so that it does not need to compute so many iterations in the grounding step (most lines of Listing 7 are repeated for each of the $|\mathcal{R}|/2$ iterations). Similarly, there is potential for further research in both heuristic algorithms and ASP-solvers, aiming to obtain faster results for relevance. In particular, it would be interesting to combine them, for instance by developing a heuristic algorithm that restricts the candidates to be tested by the ASP-solver.

3.10. Related work

The notions of stability and relevance are related to research in dynamic argumentation. Dynamic argumentation (Doutre and Mailly, 2018) is a research direction within computational argumentation that assumes that the argumentation framework is not static, but can be altered by, for example, the addition or removal of arguments and/or attacks, possibly resulting in a changed acceptability status of some argument(s). Research on dynamic argumentation includes work on the impact of a change operation (Alfano et al., 2020; Alfano et al., 2021a), enforcement (Borg and Bex, 2021b; Rapberger and Ulbricht, 2023), resolution (Modgil and Prakken, 2012) and the relation with belief revision (Falappa et al., 2009; Snaith and Reed, 2016). Since the task of estimating stability is to decide if any further information (resulting in additional arguments) can change the acceptability status of some claim, it can be placed in the field of dynamic argumentation as well.

Most research on dynamic argumentation only considers the effect of changes in the *abstract* framework, such as adding an argument. In doing so, these approaches abstract away from dependencies between arguments. As we have shown in Section 3.4, this is problematic, especially when these approaches reckon with argumentation frameworks for which no structured instantiation exists.

At present, none of the existing work in dynamic argumentation specifically studies stability on the structured level. In general, dynamics for structured approaches have only recently received more attention. We briefly discuss some related research. In Borg and Bex (2021b) and Rapberger and Ulbricht (2023), enforcement is studied in structured argumentation settings. Enforcement is related to, yet different from stability. Some literal can be enforced if it is possible to alter the knowledge base in such a way that it is justified, whereas it is stable(-defended) if each possible addition to the knowledge base results in an argumentation theory in which the literal is justified. Another related study (Snaith and Reed, 2016) shows how the

acceptability of a specific set of arguments can be altered by a minimal number of changes on the premises and/or rules in an argumentation framework, relating dynamic argumentation to belief revision. However, they do not focus on a specific task, such as stability; they do not consider computational complexity, nor do they provide an efficient (heuristic) algorithm. Two related papers that do take computational complexity into account are Alfano et al. (2020) and Alfano et al. (2021a). The authors propose efficient algorithms to minimise re-computations after a change in an Attack-Support Argumentation Framework or in a DeLP program. However, whereas they study acceptability status after a specific change, we study the status after any possible change in the structured argumentation framework. In Prakken (2023) expansions in ASPIC$^+$ are studied, based on a refinement of the notion from Baumann and Brewka (2010). Finally, the notion of forgetting was recently studied in Berthold et al. (2023b) for assumption-based argumentation.

3.11. Conclusion

In this chapter, we studied stability and relevance from complexity-theoretic and algorithmic perspectives in an instantiation of ASPIC$^+$. We focused on grounded semantics. Whereas stability was introduced for (a limited fragment of) ASPIC$^+$ in earlier work (Testerink et al., 2019a), the introduction of relevance in this context is the first contribution of this chapter. In Section 3.4 we compared the notions of stability and relevance for ASPIC$^+$ to the analogous notions for IAFs (defined in Chapter 2). Although these notions are related, stability and relevance in ASPIC$^+$ cannot be derived directly from stability and relevance in IAFs. Consequently, the algorithms for stability and relevance for IAFs, as proposed in Chapter 2, cannot be reused directly for computing stability and relevance for ASPIC$^+$.

Towards developing efficient algorithms, a second contribution of this chapter is the complexity analysis of the problems of stability and relevance in ASPIC$^+$. We established CoNP-completeness of stability and Σ_2^P-completeness of relevance. As a side-result of the CoNP membership proof for stability, we provided a polynomial-time algorithm for the justification problem in ASPIC$^+$.

The third contribution of this chapter consists of various algorithms for stability and relevance. First, we proposed a polynomial-time algorithm Stability-label for estimating stability. We have shown that the refined algorithm is sound and runs in polynomial time. In addition, we have identified conditions under which Stability-label is complete. Given that the algorithm is not complete for all argumentation theories, there are examples of argumentation theories for which the algorithm does not detect that a literal is stable. As an alternative approach, we developed exact algorithms based on answer set programming for deciding stability and relevance, as well as for the task of finding all relevant queryables. We showed through an extensive empirical evaluation on both real-world and synthetic data that both algorithmic approaches exhibit promising scalability.

These three contributions provide answers to research questions 1, 2 and 3 in the context of ASPIC$^+$.

4

Stability & relevance in precedent-based reasoning

This chapter is based on the following papers:

1. The notions of stability and relevance were introduced in Odekerken et al. (2023a) for incomplete fact situations and *complete* case bases. This conference paper, presented at ICAIL 2023, was written under supervision of Floris Bex and Henry Prakken.
2. Odekerken et al. (2023b) discussed justification and possibility in settings with *incomplete case bases*. This conference paper was presented at JURIX 2023 and written under supervision of Floris Bex and Henry Prakken.
3. In Odekerken et al. (2024), we extend upon the aforementioned conference papers. In this article, which forms the basis of the current chapter, the notions of justification, possibility, stability and relevance are studied in settings with incomplete fact situations and incomplete case bases. This article is accepted for Artificial Intelligence and Law. It was written under supervision of Floris Bex and Henry Prakken.

4.1. Introduction

Modelling reasoning with precedents is an important topic in the computational study of legal argument. This type of reasoning is applied to problems that are not decided by a legal rule but by comparing the characteristics of legal cases. In particular, the characteristics of a new case for which the outcome still has to be decided (called *focus case*) are compared to the characteristics of precedent cases that have already been assigned an outcome. In the seminal work on the HYPO system for US trade secrets law (Rissland and Ashley, 1987), these characteristics were multi-valued and were called *dimensions*. Later, in influential work on the CATO system (Aleven, 1997), the characteristics of a case were considered to be boolean and were called *factors*. Each factor or dimension has some direction: a factor's presence favours one of the two parties in a case and some dimension values are more favourable to one of the parties than other values.

Studies on reasoning with legal cases have multiple applications. The first systems were designed to generate arguments in the legal domain (Rissland and Ashley, 1987), for example for teaching purposes (Aleven, 1997). Later work, initiated by Horty (2011), addressed the question how precedents constrain the possible decisions for a focus case, providing multiple models including a *result model* of *precedential constraint*. Recently, this model has been used for explaining machine-learning-based decisions (e.g. Prakken and Ratsma (2022) and Peters et al. (2022)). In this chapter, we study an alternative application of the result model as we use it in a classifier for human-in-the-loop decision support, as illustrated in Chapter 1. In an ongoing project for the Dutch police, we have implemented a simplified version of this model for the classification of mala fide web shops (see also Chapter 5). This is a classification problem where it is important that decisions are taken consistently. That is, if a decision is taken once, then in similar (or more extreme) situations, the same decision should be taken – which is exactly the kind of reasoning modelled by the result model of precedential constraint.

Whereas in existing work on the result model (e.g. Horty (2011) and Horty (2019)), it is assumed that the characteristics of a focus case are fully investigated, in practical settings the values of some dimensions may not yet have been determined with certainty (Ashley, 1991). This also applies for the use case in web shop classification. Investigation into the exact values of these dimensions requires time and effort. It is therefore useful to identify cases for which the same outcome is enforced regardless of how the uncertainties in dimension values are resolved: in those cases, there is no need for investment in further investigation.

As a first contribution of this chapter, we therefore propose to extend the result model for precedential constraint with a component that provides the possibility to express uncertainties in the dimension values of a focus case, thus enabling precedent-based reasoning with incomplete information. First, we introduce the notion of *incomplete fact situations*, which express not only the certain characteristics of a case, but also the uncertain ones. More specifically, in an incomplete fact situation each dimension is assigned a set of possible dimension values. An incomplete fact

situation can be completed by resolving all uncertainties: a *completion* is defined as a (complete) fact situation that, for each dimension, has exactly one of the possible dimension values. Incomplete fact situations can also be assigned an outcome, forming *incomplete cases*, where the outcome of an incomplete case conveys that this outcome applies to all completions. A set of incomplete cases forms an *incomplete case base*. We require incomplete case bases to be consistent, in the sense that no complete fact situation is a completion of two incomplete cases with conflicting outcomes in that incomplete case base.

Incomplete case bases establish constraints on the outcome of focus cases, based on *a fortiori reasoning*. This type of reasoning works as follows: if a (focus) fact situation is more extreme towards the outcome of a precedent case than the fact situation of this case, then the same outcome should be assigned to the focus fact situation. For any (complete) fact situation, we refer to the outcome enforced by the incomplete case base as its *justification status*. Then, we define *stability* for case-based reasoning, where incomplete fact situations are called *stable* if each completion has the same justification status. For those incomplete fact situations that are not stable, we study the task of identifying *relevance*: which uncertainties in the incomplete focus case should be resolved in order to make the case stable? Finally, in some situations, a user can decide to update the incomplete case base with an outcome that is not decided a fortiori based on the original incomplete case base. To identify whether such an update is allowed, in the sense that it does not make the incomplete case base inconsistent, we introduce the notion of *possible* justification statuses.

Recall from Chapter 1 that the notions of justification, stability, relevance and possibility have applications in human-in-the-loop decision support systems. If an incomplete fact situation is stable, then no further investigation is required. Otherwise, relevant updates reveal which dimensions should be investigated further. Possible justification statuses indicate which updates on the incomplete case base are allowed. For application in decision support systems it is required that the tasks of identifying possibility and stability status, as well as listing the relevant uncertainties are performed efficiently, even if there is a lot of uncertainty on the value of dimensions. This is not trivial. As a second contribution of this chapter, we will prove that these tasks are situated in high complexity classes. However, we will also show that the source of this high complexity disappears under particular conditions on the dimension value ordering. If these conditions hold, the tasks can be performed in polynomial time. As a third and final contribution, we describe efficient and scalable algorithms for identifying justification, stability and relevance.[1] For proofs of all propositions in the chapter, we refer to Appendix C.

Outline We start by recalling the basic definitions of the result model in a setting with dimensions and complete information in Section 4.2. We then show in Section 4.3 how these can be used in a human-in-the-loop decision support system, under the assumption of complete information. As the assumption of complete information does not apply in general, in the remainder of the chapter we drop this

[1] Implementations of these algorithms can be found at `https://git.science.uu.nl/D.Odekerken/lcbr`.

assumption. In Section 4.4, we therefore define the notions of justification, stability and relevance for precedent-based reasoning with incomplete information. We then show in Section 4.5 how algorithms for these tasks can be applied in practice in human-in-the-loop decision support. Practical applications require efficient algorithms, which we discuss in Section 4.6. Related work is discussed in Section 4.7, after which we conclude in Section 4.8.

4.2. Preliminaries

Before formally defining the notions of justification, stability, possibility and relevance, we recall the definitions of (complete) fact situations, cases and the notion of precedential constraint from Horty (2019). In order to define these formally, we introduce the notion of a domain. This is the set of all dimensions related to the cases about which we want to reason.

Definition 4.1 (Domain)**.** Let $\mathcal{D}$ be a set of dimensions, where a dimension is a tuple $d = (V, \leq_{\texttt{PRO}})$ such that:

- V is a set of values that can be assigned to d; and
- $\leq_{\texttt{PRO}}$ is a partial order on V.

We refer to $\mathcal{D}$ as the domain. The notation $V(\mathcal{D}, d)$ refers to the set of values that can be assigned to the dimension d in domain $\mathcal{D}$. In addition, we define $\leq_{\texttt{CONTRA}}$ as a partial order on V such that $v' \leq_{\texttt{CONTRA}} v$ iff $v \leq_{\texttt{PRO}} v'$. Furthermore, $\leq_{\texttt{PRO}} = \geq_{\texttt{CONTRA}}$ and $\geq_{\texttt{PRO}} = \leq_{\texttt{CONTRA}}$. For any $o \in \{\texttt{PRO}, \texttt{CONTRA}\}$ we write $v <_o v'$ iff $v \leq_o v'$ and $v' \not\leq_o v$. Analogously, $v >_o v'$ iff $v \geq_o v'$ and $v' \not\geq_o v$. Finally, $v \approx_o v'$ iff $v \geq_o v'$ and $v' \geq_o v$.

Example 4.1. As a running example, we consider the domain $\mathcal{D}$ of classifying web shops for the decision whether a web shop is trustworthy (PRO) or not (CONTRA). As illustrated in Figure 4.1, $\mathcal{D}$ has two dimensions: $\mathcal{D} = \{d_1, d_2\}$. Dimension d_1 represents the number of complaints filed at the police, taking into account retracted complaints as well. It can be stated that web shops without complaints (v_{14}) at least as trustworthy (PRO) as, for example, web shops with many complaints and some retractions (v_{13}). Not all values of d_1 can be compared to each other: in the $\geq_{\texttt{PRO}}$ ordering illustrated in Figure 4.1, $v_{12} \not\geq_{\texttt{PRO}} v_{13}$ and $v_{13} \not\geq_{\texttt{PRO}} v_{12}$. In words: web shops with few complaints and no retractions (v_{12}) are not as least as trustworthy as web shops with many complaints and some retractions (v_{13}), and vice versa. Dimension d_2 represents the possibility of paying in a safe way. Whereas web shops that do not display this possibility (v_{22}) are in general less trustworthy than web shops that do have this possibility (v_{23}), it is even worse if a web shop (deceptively) displays the safe payment option while this is actually not possible (v_{21}).

In this approach, factors are seen as a special case of dimensions with just two values: $V = \{0, 1\}$, corresponding to the absence or presence of the factor. For factors favoring a PRO outcome, the associated ordering is $\leq_{\texttt{PRO}} = \{(0,0), (0,1), (1,1)\}$, while the ordering for factors favoring a CONTRA outcome is $\leq_{\texttt{PRO}} = \{(0,0), (1,0), (1,1)\}$. This is the same as the approach in Prakken and Ratsma (2022), Section 5.2.

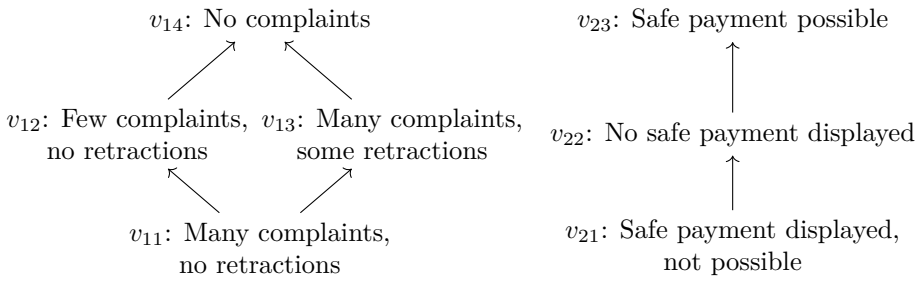


Figure 4.1: Hasse diagrams for the $\leq_{\texttt{PRO}}$ ordering of the two dimensions in (a toy example version of) the web shop domain $\{d_1, d_2\}$. The dimension d_1, displayed on the left, has the four values $\{v_{11}, v_{12}, v_{13}, v_{14}\}$ while dimension d_2, on the right, has three values $\{v_{21}, v_{22}, v_{23}\}$. If there is a path from value a to b then $a \leq_{\texttt{PRO}} b$.

A complete fact situation is then a combination of dimensions with corresponding values, as defined next. Note that each complete fact situation is required to have exactly one value for each dimension.

Definition 4.2 (Complete fact situation)**.** A complete fact situation c within a domain $\mathcal{D}$ is a set D of value assignments to all dimensions in $\mathcal{D}$. A value assignment is a pair (d, v) where $d \in \mathcal{D}$ and $v \in V(\mathcal{D}, d)$. The notation $v(d, c)$ denotes the value of dimension d in complete fact situation c.

Example 4.2 (Complete fact situation)**.** An example of a complete fact situation in the domain $\mathcal{D}$ is $c_1 = \{(d_1, v_{12}), (d_2, v_{23})\}$.

Based on the dimension values of the complete fact situation, an outcome can be assigned. A complete fact situation paired with the outcome is a complete case.

Definition 4.3 (Complete case)**.** Within a domain $\mathcal{D}$, a complete case is a (c, o)-pair, where c is a complete fact situation within the domain $\mathcal{D}$ and o is the assigned outcome of the case, which can be either PRO or CONTRA.

Example 4.3 (Complete case)**.** If c_1 is a complete fact situation, then (c_1, PRO) is a complete case, in which the outcome PRO is assigned. If, alternatively, the outcome CONTRA is assigned, then the case is (c_1, CONTRA).

We then refer to a set of complete cases as a complete case base.

Definition 4.4 (Complete case base)**.** A complete case base $\mathcal{C}$ within a domain $\mathcal{D}$ is a set of complete cases within $\mathcal{D}$.

Example 4.4. A complete case base $\mathcal{C}$ in the web shop domain $\mathcal{D}$ might consist of two complete cases: $\mathcal{C} = \{(c_1, \text{PRO}), (c_2, \text{CONTRA})\}$ where $c_1 = \{(d_1, v_{12}), (d_2, v_{23})\}$ and $c_2 = \{(d_1, v_{13}), (d_2, v_{22})\}$.

Based on the ordering of dimension values, complete fact situations can be compared to each other under some conditions, which are specified in the following definition, adapted from Horty (2019). The idea is that a complete fact situation

with dimensions with values that are "at least as PRO" as a the values in complete fact situation that is already assigned a PRO outcome is constrained to be PRO. Similarly, a complete fact situation that is at least as CONTRA as a complete fact situation with CONTRA outcome is assigned to be CONTRA.

Definition 4.5 (Strength for a side)**.** Let c_1 and c_2 be two complete fact situations within some domain $\mathcal{D}$. Then c_1 is at least as strong as c_2 for the side PRO (notated as $c_1 \geq_{\texttt{PRO}} c_2$) iff for all $d \in \mathcal{D}$: $v(d, c_1) \geq_{\texttt{PRO}} v(d, c_2)$. Analogously, c_1 is at least as strong as c_2 for the side CONTRA ($c_1 \geq_{\texttt{CONTRA}} c_2$) iff for all $d \in \mathcal{D}$: $v(d, c_1) \geq_{\texttt{CONTRA}} v(d, c_2)$.

Example 4.5. Consider again the web shop domain $\mathcal{D}$. We will compare the complete fact situation $c_1 = \{(d_1, v_{12}), (d_2, v_{23})\}$ to the complete fact situation $c' = \{(d_1, v_{14}), (d_2, v_{23})\}$. For each dimension d in $\mathcal{D}$, $v(d, c_1) \geq_{\texttt{PRO}} v(d, c')$. Therefore, $c_1 \geq_{\texttt{PRO}} c'$. As it is not the case that for each $c \in \mathcal{D}$: $v(d, c_1) \geq_{\texttt{CONTRA}} v(d, c')$, it holds that $c_1 \not\geq_{\texttt{CONTRA}} c'$.

Using the cases from the complete case base, in specific situations the outcome of a new complete fact situation can be forced based on the so-called *a fortiori* constraint (Horty, 2011).

Definition 4.6 (A fortiori constraint)**.** Let $\mathcal{C}$ be a complete case base and c a complete fact situation in domain $\mathcal{D}$. Then c is forced towards outcome PRO by $\mathcal{C}$ iff there is some (c', PRO) in $\mathcal{C}$ such that $c \geq_{\texttt{PRO}} c'$; c is forced towards outcome CONTRA by $\mathcal{C}$ iff there is some (c', CONTRA) in $\mathcal{C}$ such that $c \geq_{\texttt{CONTRA}} c'$.

Example 4.6. Continuing our running example, if the complete case base $\mathcal{C}$ in domain $\mathcal{D}$ contains the complete case (c_1, PRO) then the complete fact situation c' is forced towards outcome PRO as $c' \geq_{\texttt{PRO}} c_1$.

Note that a new case only has a forced outcome if there is a precedent in the complete case base for which the a fortiori constraint applies. The notion of justification directly uses the a fortiori constraint to assign an outcome to a complete focus case (i.e. any (new) complete fact situation that is not yet assigned an outcome) based on a *consistent* complete case base. The following definition for consistency is adapted from Horty (2004, Section III) for dimensions.

Definition 4.7 (Consistency of a complete case base)**.** A complete case base $\mathcal{C}$ is *inconsistent* iff there are complete cases (c_1, CONTRA) and (c_2, PRO) in $\mathcal{C}$ such that $c_1 \geq_{\texttt{PRO}} c_2$ and *consistent* otherwise.

Definition 4.8 (Justification status for complete case bases)**.** Let $\mathcal{C}$ be a complete case base and c a complete fact situation. Then the justification status of c given $\mathcal{C}$ is:

- PRO if c is forced towards outcome PRO by $\mathcal{C}$;
- CONTRA if c is forced towards outcome CONTRA by $\mathcal{C}$;
- UNDECIDED otherwise.

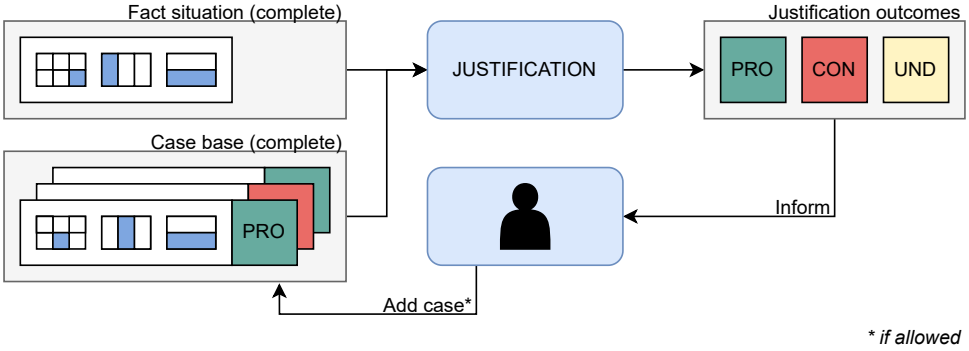


Figure 4.2: High-level overview of a human-in-the-loop decision-making process for complete fact situations and complete case bases, involving algorithms for justification.

Example 4.7. Reconsider the running example with the complete case base $\mathcal{C} = \{(c_1, \text{PRO}), (c_2, \text{CONTRA})\}$, where $c_1 = \{(d_1, v_{12}), (d_2, v_{23})\}$ and $c_2 = \{(d_1, v_{13}), (d_2, v_{22})\}$.

- $c' = \{(d_1, v_{14}), (d_2, v_{23})\}$ has the justification status PRO as $c' \geq_{\texttt{PRO}} c_1$.
- $c'' = \{(d_1, v_{11}), (d_2, v_{21}\}$ is at least as CONTRA as c_2, therefore its justification status is CONTRA.
- $c''' = \{(d_1, v_{13}), (d_2, v_{23}\}$ is not as least as PRO or CONTRA as any case in $\mathcal{C}$, so its justification status is UNDECIDED.

4.3. Reasoning with complete cases

The notion of justification status can be used in a human-in-the-loop decision-making process, as we illustrate in Figure 4.2. Suppose that a complete case base and a complete fact situation are given for a specific domain and that decisions should be taken consistently by *a fortiori* reasoning. The justification status (Definition 4.8) can be computed by an automatic procedure, returning the justification status PRO, CONTRA or UNDECIDED for the fact situation given the case base. In case the justification status is PRO or CONTRA, the decision can be taken automatically, as the only way to retain the consistency of the case base is by adding the fact situation with the justification status as its outcome. This holds in general, as we show in the following proposition.[2] This proposition is in line with earlier observations on case base dynamics for the (factor-based) reason model (Horty, 2011, Section 5).

Proposition 4.1. *Let $\mathcal{C}$ be a consistent complete case base and c a complete fact situation.*

1. *If c is PRO w.r.t. $\mathcal{C}$ then $\mathcal{C} \cup \{(c, PRO)\}$ is consistent.*

[2]Proofs for all propositions can be found in Appendix C.

2. *If c is PRO w.r.t. $\mathcal{C}$ then $\mathcal{C} \cup \{(c, CONTRA)\}$ is inconsistent.*
3. *If c is CONTRA w.r.t. $\mathcal{C}$ then $\mathcal{C} \cup \{(c, PRO)\}$ is inconsistent.*
4. *If c is CONTRA w.r.t. $\mathcal{C}$ then $\mathcal{C} \cup \{(c, CONTRA)\}$ is consistent.*
5. *If c is UNDECIDED w.r.t. $\mathcal{C}$ then $\mathcal{C} \cup \{(c, PRO)\}$ is consistent.*
6. *If c is UNDECIDED w.r.t. $\mathcal{C}$ then $\mathcal{C} \cup \{(c, CONTRA)\}$ is consistent.*

In situations where the justification status is UNDECIDED, the decision cannot be taken automatically. Instead, an analyst (the human in the loop) should decide whether this new complete fact situation should be assigned a PRO or CONTRA outcome. For both outcomes, the result of adding the complete fact situation with outcome as a new case to the complete case base results in a consistent complete case base by Proposition 4.1.

Note that this procedure requires that the complete case base always remains consistent. Under this assumption, any complete fact situation has exactly one justification status. This assumption may be quite strong in general (Peters et al., 2022; Canavotto, 2023), but for applications in human-in-the-loop decision support we consider it to be reasonable as users have control on the complete case base. Still, it may occur that the analyst's decision is incompatible with the justification status obtained from the complete case base: that is, either the analyst decides on outcome o = PRO while the justification status o' is CONTRA, or o' is PRO while o is CONTRA. Such a situation can have multiple causes, including concept drift in the interpretation of dimension values, a mistake in the decision for an earlier case, or the new case being an outlier. In any case, the analyst is not allowed to add the complete fact situation c with outcome o to the complete case base: that would cause the complete case base to become inconsistent. The analyst then has four options:

1. Do not add c with outcome o to the complete case base (as c is considered to be an outlier);
2. Remove all complete cases with outcome o' that force c to be o' from the complete case base and then (optionally) add c with outcome o to the complete case base;
3. Correct the complete fact situations of all cases with outcome o' that force c to be o' from the complete case base and then (optionally) add c with outcome o to the complete case base; or
4. Correct c, so it becomes c' (which is not forced to be o') and then add c' with outcome o to the complete case base.

The choice between these four options is up to the analyst. By always choosing one of these options after an incompatible decision is made, the analyst ensures that the complete case base remains consistent.

4.4. Reasoning with incomplete cases

In the previous section, we assumed that all dimension values are known for both the focus case and the case base, but this does not always match reality. In practice, as stated in Chapter 1, for new fact situations the exact value of each dimension is not immediately known. We therefore would like to take multiple values into account for specific dimensions and reason with such "incomplete cases". In this section, we will first introduce the notion of incomplete cases and then show how we can reason with them using the notions of stability, relevance and possibility.

4.4.1. Incomplete cases

First, we introduce the notion of incomplete fact situations. Analogous to complete fact situations, incomplete fact situations assign values to each of the dimensions in a domain. However, for incomplete fact situations this is not a single value per dimension, but a set of possible values.

Definition 4.9 (Incomplete fact situation)**.** An *incomplete fact situation* c within a domain $\mathcal{D}$ is the assignment of a possible value set for each dimension: for each $d = (V, \leq_{\texttt{PRO}})$ in $\mathcal{D}$ the incomplete fact situation contains a tuple (d, V_p) where $V_p \subseteq V(\mathcal{D}, d)$ is the finite set of possible values. The notation $V_p(d, c)$ denotes the set of possible dimension values of dimension d in incomplete fact situation c. For each $d \in \mathcal{D}$, $V_p(d, c)$ must contain at least one value.

When obtaining more information on an incomplete fact situation, the set of possible values can be reduced. Once all these uncertainties are resolved, the fact situation no longer has to be considered incomplete. Completions of an incomplete fact situation are complete fact situations that are obtained by selecting a single value from the set of possible values for each dimension.

Definition 4.10 (Completions)**.** A *completion* of an incomplete fact situation c is a complete fact situation c' such that for each $(d, v) \in c'$: $v \in V_p(d, c)$.

Example 4.8. In the incomplete fact situation $c_3 = \{(d_1, \{v_{12}, v_{14}\}), (d_2, \{v_{23}\})\}$ in the web shop domain, we do not know the exact value of d_1 (i.e., the number of complaints and retractions). c_3 has two completions: $\{(d_1, v_{12}), (d_2, v_{23})\}$ and $\{(d_1, v_{14}), (d_2, v_{23})\}$.

Next, we extend the definitions of cases and case bases to incomplete variants.

Definition 4.11 (Incomplete case base)**.** Within a domain $\mathcal{D}$, an *incomplete case* is a pair (c', o), where c' is an incomplete fact situation within the domain $\mathcal{D}$ and o is the assigned outcome of the incomplete case, which can be either PRO or CONTRA. An *incomplete case base* $\mathcal{C}$ within a domain $\mathcal{D}$ is a set of incomplete cases within $\mathcal{D}$.

Although an incomplete case base can only contain incomplete cases, it is still possible to add a case for which the precise dimension values are known. This can be done by choosing a singleton for each set of possible dimension values. For example, in a domain with one dimension d_1 and a (complete) fact situation $\{(d_1, v_1)\}$ that

we assign the outcome PRO, the "incomplete" case $(\{(d_1, \{v_1\})\}, \text{PRO})$ is added to the incomplete case base.

Since incomplete case bases consist of incomplete cases rather than complete cases, the notions of a fortiori constraint and consistency need to be redefined.

Definition 4.12 (A fortiori constraint with incomplete case base)**.** A complete fact situation c is *forced* towards outcome o (PRO or CONTRA) by an incomplete case base $\mathcal{C}$ iff there is some incomplete case (c', o) in $\mathcal{C}$ such that c' has at least one completion c'' such that $c \geq_o c''$.

In words, having an incomplete case (c', o) in the incomplete case base with outcome PRO should be interpreted as: each of the completions of c', as well as "more extreme" fact situations, should have the outcome o.

Example 4.9. Let $\mathcal{C}'$ be an incomplete case base in domain $\mathcal{D}$ that is the "incomplete version of" $\mathcal{C}$ from Example 4.7: $\mathcal{C}' = \{(c_1, \text{PRO}), (c_2, \text{CONTRA})\}$ where $c_1 = \{(d_1, \{v_{12}\}), (d_2, \{v_{23}\})\}$ and $c_2 = \{(d_1, \{v_{13}\}), (d_2, \{v_{22}\})\}$. Let $c_5 = \{(d_1, \{v_{11}, v_{12}, v_{13}\}), (d_2, \{v_{23}\})\}$. For $\mathcal{C}' \cup \{(c_5, \text{PRO})\}$, the complete fact situation $c'' = \{(d_1, v_{13}), (d_2, v_{23})\}$ is forced towards outcome PRO because of c_5: c_5 has a completion $c_5' = \{(d_1, v_{13}), (d_2, v_{23})\}$ and $c'' \geq_{\texttt{PRO}} c_5'$. Note that c_5 also has completions that are not "at least as PRO as" c'', for example: $c_5'' = \{(d_1, v_{11}), (d_2, v_{23})\} \not\geq_{\texttt{PRO}} c''$. This does not influence the a fortiori constraint on c''. By adding the incomplete case (c_5, PRO) to the incomplete case base, the analyst expressed that the exact value of d_1 does not matter, as the value assignment (d_2, v_{23}), i.e., the web shop having a safe payment option, was considered to be sufficient for having the PRO outcome in combination with any value in $V_p(d_1, c_5)$.

Note that Definition 4.12 is an incomplete counterpart of Definition 4.5. Analogously, we present an incomplete version of Definition 4.7 of consistency in Definition 4.13. In words, an incomplete case base is consistent if it is not possible to find a completion of a CONTRA incomplete case that is at least as PRO as a completion of a PRO incomplete case.

Definition 4.13 (Consistency of an incomplete case base)**.** An incomplete case base $\mathcal{C}$ is *inconsistent* iff there are incomplete cases (c_1, CONTRA) and (c_2, PRO) in $\mathcal{C}$ such that c_1 has a completion c_1' and c_2 has a completion c_2' for which $c_1' \geq_{\texttt{PRO}} c_2'$. $\mathcal{C}$ is *consistent* iff it is not inconsistent.

Example 4.10. The incomplete case base $\mathcal{C}' = \{(c_1, \text{PRO}), (c_2, \text{CONTRA})\}$ where $c_1 = \{(d_1, \{v_{12}\}), (d_2, \{v_{23}\})\}$ and $c_2 = \{(d_1, \{v_{13}\}), (d_2, \{v_{22}\})\}$ from Example 4.9 is consistent. Now let $c_4 = \{(d_1, \{v_{11}, v_{12}, v_{13}\}), (d_2, \{v_{21}, v_{23}\})\}$.

- Then $\mathcal{C}' \cup \{(c_4, \text{PRO})\}$ is inconsistent, as $(c_2, \text{CONTRA}) \in \mathcal{C}'$ and c_2 has a completion $(\{(d_1, v_{13}), (d_2, v_{22})\})$ that is as least as PRO as, for example, the completion $\{(d_1, v_{11}), (d_2, v_{21})\}$ of c_4.
- Similarly, $\mathcal{C}' \cup \{(c_4, \text{CONTRA})\}$ is inconsistent as (c_1, PRO) has a completion $(\{(d_1, v_{12}), (d_2, v_{23})\})$ that is as least as CONTRA as the completion $\{(d_1, v_{12}), (d_2, v_{23})\}$ of c_4.

Analogous to the definition of justification w.r.t. complete case bases (Definition 4.8), Definition 4.14 specifies justification of a complete fact situation w.r.t. incomplete (and consistent) case bases.

Definition 4.14 (Justification status for incomplete case bases)**.** Let $\mathcal{C}$ be an incomplete consistent case base and c a complete fact situation. Then the justification status of c given $\mathcal{C}$ is:

- PRO if c is forced towards outcome PRO by $\mathcal{C}$;
- CONTRA if c is forced towards outcome CONTRA by $\mathcal{C}$; and
- UNDECIDED otherwise.

4.4.2. Stability: drawing conclusions for incomplete focus cases

In practical settings, the fact situation for which we would like to derive the outcome using a fortiori reasoning may be incomplete as well. We therefore cannot directly use the notion of justification status, as this is only defined for *complete* focus cases (even if the case base is incomplete). Instead, we introduce the notion of stability. The stability status identifies situations in which the justification status is the same for all completions.

Definition 4.15 (Stability status)**.** Let $\mathcal{C}$ be a consistent incomplete case base and c an incomplete fact situation. Then the *stability status* of c given $\mathcal{C}$ is

- Stable-PRO if the justification status of each completion c' of c given $\mathcal{C}$ is PRO;
- Stable-CONTRA if the justification status of each completion c' of c given $\mathcal{C}$ is CONTRA;
- Stable-UNDECIDED if the justification status of each completion c' of c given $\mathcal{C}$ is UNDECIDED; and
- Unstable otherwise.

Example 4.11. Recall from Example 4.8 that the incomplete fact situation $c_3 = \{(d_1, \{v_{12}, v_{14}\}), (d_2, \{v_{23}\})\}$ in the web shop domain has two completions, as we do not know the exact value of d_1. For each completion c' of c_3: $c' \geq_{\texttt{PRO}} c_1$; therefore, c_3 is Stable-PRO w.r.t. the incomplete case base $\mathcal{C}'$.

Incomplete fact situations that are stable are interesting as they indicate that further investigation on the exact dimension values is not required: the derived outcome would be the same for all completions anyway.

Finally, note that adding incomplete fact situations with a Stable-PRO or Stable-CONTRA outcome to a consistent incomplete case base with the same outcome can never make the resulting incomplete case base inconsistent.

Proposition 4.2 (Impossibility of inconsistency)**.** *Let $\mathcal{C}$ be a consistent incomplete case base and c an incomplete fact situation. For each $o \in \{PRO, CONTRA\}$: if c is Stable-o w.r.t. $\mathcal{C}$ then $\mathcal{C} \cup \{(c, o)\}$ is consistent.*

4.4.3. Relevance: collecting information towards stability

Even though it is quite common that a decision can be taken without full investigation of the incomplete focus case, given that it is Stable-PRO or Stable-CONTRA w.r.t. the incomplete case base, there are also many situations where the incomplete fact situation is Unstable. In such situations, there are dimensions for which it is still relevant to investigate if some of their possible values can be excluded. In this section, we define these relevant updates. In order to do so, we first need the additional notions of partial completions and minimal stable partial completions.

A partial completion is some refinement of the possible dimension values of the incomplete focus case, obtained after further investigation by the user. In a partial completion, part of the information may still be uncertain, whereas in a completion the exact value for each of the dimensions is certain.

Definition 4.16 (Partial completion)**.** A *partial completion* of an incomplete fact situation c is an incomplete fact situation c' such that for each $(d, V_p') \in c'$: $V_p' \subseteq V_p(d, c)$. We denote all partial completions of some incomplete fact situation c by $F(c)$. Note that for each c, $c \in F(c)$.

Example 4.12. Consider the incomplete case base $\mathcal{C}' = \{(c_1, \text{PRO}), (c_2, \text{CONTRA})\}$ where $c_1 = \{(d_1, \{v_{12}\}), (d_2, \{v_{23}\})\}$ and $c_2 = \{(d_1, \{v_{13}\}), (d_2, \{v_{22}\})\}$ from Example 4.9. The incomplete fact situation $c_4 = \{(d_1, \{v_{11}, v_{12}, v_{13}\}), (d_2, \{v_{21}, v_{23}\})\}$ in the web shop domain $\mathcal{D}$ is Unstable, as its completions can have various justification statuses.

- The completion $\{(d_1, v_{12}), (d_2, v_{23})\}$ has the justification status PRO (because of (c_1, PRO)),
- The completion $\{(d_1, v_{12}), (d_2, v_{21})\}$ has the justification status UNDECIDED and
- The completion $\{(d_1, v_{11}), (d_2, v_{21})\}$ has the justification status CONTRA (because of (c_2, CONTRA)).

An analyst can decide upon further investigation, for example by trying to make a payment using the safe option (analysing d_2). If the analyst succeeds, c_4 should be replaced by $c_5 = \{(d_1, \{v_{11}, v_{12}, v_{13}\}), (d_2, \{v_{23}\})\}$. Note that c_5 is a partial completion of c_4.

Partial completions and completions are related in the following way: for each completion c' of some incomplete focus case c, there is some partial completion $c^* \in F(c)$ where $c^* = \{(d, \{v\}) \mid (d, v) \in c'\}$. Using this notion of partial completions, we next introduce minimal stable partial completions w.r.t. an incomplete fact situation c and justification status j. Minimal stable-j partial completions c' are required to be Stable-j and, informally, there should not be any other partial completion "in between" c and c' that is Stable-j as well.

Definition 4.17 (Minimal stable partial completions)**.** Let $\mathcal{C}$ be an incomplete case base and let c be an incomplete fact situation. Let j be a justification status

(PRO, CONTRA or UNDECIDED). Then $c' \in F(c)$ is a minimal stable-j partial completion w.r.t. $\mathcal{C}$ and c iff:

- c' is Stable-j given $\mathcal{C}$; and
- there is no partial completion $c'' \in F(c) \setminus \{c'\}$ such that c'' is Stable-j given $\mathcal{C}$ and $c' \in F(c'')$.

Example 4.13. Continuing Example 4.12, suppose we are interested in the minimal stable-PRO partial completions w.r.t. $\mathcal{C}'$ and $c_4 = \{(d_1, \{v_{11}, v_{12}, v_{13}\}), (d_2, \{v_{21}, v_{23}\})\}$. $c_5 = \{(d_1, \{v_{11}, v_{12}, v_{13}\}), (d_2, \{v_{23}\})\}$ is not minimal stable-PRO as it has a completion (for example $\{(d_1, v_{13}), (d_2, v_{23})\}$) that is not PRO w.r.t. $\mathcal{C}'$. The only minimal stable-PRO partial completion w.r.t. $\mathcal{C}'$ and c_4 is $\{(d_1, \{v_{12}\}), (d_2, \{v_{23}\})\}$.

Now let us consider the minimal stable-CONTRA partial completions w.r.t. $\mathcal{C}'$ and c_4. There are three stable-CONTRA partial completions of c_4 w.r.t. $\mathcal{C}'$:

1. $\{(d_1, \{v_{11}, v_{13}\}), (d_2, \{v_{21}\})\}$,
2. $\{(d_1, \{v_{11}\}), (d_2, \{v_{21}\})\}$ and
3. $\{(d_1, \{v_{13}\}), (d_2, \{v_{21}\})\}$.

However, only the first of these is a minimal stable-CONTRA partial completion.

Next, we define relevant updates, which are those updates that lead towards a minimal stable partial completion.

Definition 4.18 (Relevance)**.** Within some domain $\mathcal{D}$, let $\mathcal{C}$ be an incomplete case base and let c be an incomplete fact situation. Let j be a justification status (PRO, CONTRA or UNDECIDED). Then for a given dimension $d \in \mathcal{D}$, the removal of possible value $v \in V_p(d, c)$ is j-relevant w.r.t. $\mathcal{C}$ and c iff there is some minimal stable-j partial completion c' w.r.t. $\mathcal{C}$ and c such that $v \notin V_p(d, c')$.

Example 4.14. As stated in Example 4.13, the only minimal stable-PRO partial completion w.r.t. $\mathcal{C}'$ and $c_4 = \{(d_1, \{v_{11}, v_{12}, v_{13}\}), (d_2, \{v_{21}, v_{23}\})\}$ is $\{(d_1, \{v_{12}\}), (d_2, \{v_{23}\})\}$. Then by Definition 4.18 the removal of values v_{11} and v_{13} are PRO-relevant for dimension d_1, while the removal of value v_{21} is PRO-relevant for dimension d_2 w.r.t. $\mathcal{C}'$ and c_4.

Regarding CONTRA-relevance, recall that the only minimal stable-CONTRA partial completion w.r.t. $\mathcal{C}'$ and c_4 is $\{(d_1, \{v_{11}, v_{13}\}), (d_2, \{v_{21}\})\}$. Then the removal of value v_{12} is CONTRA-relevant for dimension d_1, while the removal of value v_{23} is CONTRA-relevant for dimension d_2 w.r.t. $\mathcal{C}'$ and c_4.

Updates can have multiple relevance statuses at the same time. We illustrate this in the following example.

Example 4.15. Let $\mathcal{D} = \{d_1\}$ be a domain with $d_1 = (\{1, 2, 3\}, \leq, \geq)$ and $\mathcal{C}$ is a consistent incomplete case base $\{(c_1, \text{PRO}), (c_2, \text{CONTRA})\}$ where $c_1 = \{(d_1, \{3\})\}$ and $c_2 = \{(d_1, \{1\})\}$. Let $c = \{(d_1, \{1, 2, 3\})\}$ be an incomplete fact situation. Then the minimal stable-PRO partial completion is $\{(d_1, \{3\})\}$; the minimal stable-CONTRA partial completion is $\{(d_1, \{1\})\}$ and the minimal stable-UNDECIDED

partial completion is $\{(d_1, \{2\})\}$. Consequently, the removal of value 1 for d_1 is both PRO- and UNDECIDED-relevant, whereas the removal of value 3 is both CONTRA- and UNDECIDED-relevant and the removal of value 2 is both PRO- and CONTRA-relevant.

However, for specific dimensions the removal of a value cannot be both PRO- and CONTRA-relevant. This applies in particular for dimensions that can have two values that are comparable to each other by the given ordering (such as factor-based dimensions). We formally prove this in the following proposition.

Proposition 4.3. *For any incomplete case base $\mathcal{C}$, incomplete fact situation c and dimension d in domain $\mathcal{D}$ such that $d = (\{v_1, v_2\}, \leq_{PRO})$ and $\leq_{PRO} = \{(v_1, v_1), (v_1, v_2), (v_2, v_2)\}$, the removal of some possible value v for d cannot be both PRO- and CONTRA-relevant w.r.t. $\mathcal{C}$ and c.*

Finally, we prove that iteratively performing j-relevant updates to any incomplete fact situation c eventually results in a minimal j-stable partial completion, provided that c has a completion with justification status j.

Proposition 4.4. *For any incomplete case base $\mathcal{C}$, incomplete fact situation c and justification status j, let ϕ_j be an arbitrary function from $F(c)$ to $F(c)$ such that, for a given $c' \in F(c)$:*

1. *if there is some d for which there is some v such that removing v is j-relevant w.r.t. $\mathcal{C}$ and c', then $\phi_j(c')$ returns a partial completion c'' of c' such that $V_p(d, c'') = V_p(d, c') \setminus \{v\}$ while for all d' such that $d' \neq d$: $V_p(d', c'') = V_p(d', c')$.*
2. *otherwise, $\phi_j(c') = c'$.*

Then for any partial completion $c' \in F(c)$ that has a completion that is j w.r.t. $\mathcal{C}$, there is some minimal stable-j partial completion c'' that can be obtained within a finite number n of applications of ϕ_j (so $\phi_j^n(\phi_j^{n-1}(\ldots \phi_j^1(c') \ldots)) = c''$).

4.4.4. Possibility: checking statuses for maintaining consistency

As we have seen in Section 4.4.3, incomplete fact situations that are Unstable can be refined by performing relevant updates, which results in a partial completion that may or may not be stable. Now suppose that the analyst decides that an outcome should apply for each completion of the incomplete fact situation and wants to add this information to the case base. Let us assume, for example, that the analyst decides that having a safe payment option is a very strong sign that a web shop is trustworthy and that therefore each of c_5's completions should have the outcome PRO. In other words, the analyst wants to add the case (c_5, PRO) to the incomplete case base. Just like in the human-in-the-loop procedure described in Section 4.3, updates to the incomplete case base are allowed as long as the incomplete case base remains consistent, in accordance with Definition 4.13. This consistency is guaranteed by posing restrictions on the incomplete cases that are allowed to be added to the incomplete case base. In the system described in Section 4.3, where only complete fact situations can be added to the complete case base, it suffices

to check that complete cases with justification status PRO are not added with a CONTRA outcome and vice versa. This does not apply to the extension presented in this section, where *incomplete* fact situations are added to the incomplete case base, since the notion of justification status is only defined for *complete* fact situations that are used as focus case. We therefore introduce the notion of *possible* justification statuses and show how they can be used to guarantee that the consistency of the incomplete case base is maintained after adding an incomplete case.

Informally, the possibility status is the natural counterpart of the stability status: whereas an incomplete fact situation is Stable-PRO if the justification status of *each* of its completions is PRO, it is Possible-PRO if *some* of them are PRO.

Definition 4.19 (Possibility statuses)**.** Given a consistent incomplete case base $\mathcal{C}$ and a justification status j (PRO, CONTRA or UNDECIDED), an incomplete fact situation c is *Possible-j* if the justification status of some completion c' of c given $\mathcal{C}$ is j.

Note that an incomplete fact situation can be both Possible-PRO and Possible-CONTRA. It is also possible that it is neither Possible-PRO nor Possible-CONTRA: this is exactly the case if the incomplete fact situation is Stable-UNDECIDED. Finally, each incomplete fact situation has at least one possibility status – so it is Possible-PRO, -CONTRA or -UNDECIDED (or a combination) – since each incomplete fact situation has at least one completion. Next, we show how this new notion of possibility can be applied for verifying if the incomplete case base remains consistent after adding an incomplete fact situation with a specific outcome.

Proposition 4.5. *[Maintaining consistency] Let $\mathcal{C}$ be an incomplete case base that is consistent and let c be an incomplete fact situation.*

1. *$\mathcal{C} \cup (c, PRO)$ is consistent iff c is **not** Possible-CONTRA given $\mathcal{C}$; and*
2. *$\mathcal{C} \cup (c, CONTRA)$ is consistent iff c is **not** Possible-PRO given $\mathcal{C}$.*

4.5. Human-in-the-loop decision support

Having defined stability, possibility and relevance, we now illustrate how these notions can be applied in decision support systems that can be used by human analysts to make consistent decisions on incomplete focus cases, based on inconsistent case bases. This has been implemented in a human-in-the-loop decision support system for web shop classification at the Dutch police, albeit in a simplified version (see Chapter 5). From practical experience, we know that for many, though not all, dimensions the values can be extracted automatically. In modelling the incomplete fact situation c, the availability and quality of automatic extraction methods can be taken into account as follows.

- For some dimensions d, the value v can be extracted accurately. In these cases, $(d, \{v\})$ is included in c. An example of such a dimension is the registration status of a web shop at trustmark companies. The value of this dimension can be obtained automatically and reliably by an API call.

Possible PRO	Possible CONTRA	Possible UNDEC	Stability	Add PRO	Add CONTRA	Refine
Yes	No	No	S-PRO	Obsolete	Forbidden	Obsolete
No	Yes	No	S-CONTRA	Forbidden	Obsolete	Obsolete
No	No	Yes	S-UNDEC	Allowed	Allowed	Allowed
Yes	Yes	No	Unstable	Forbidden	Forbidden	Required
Yes	No	Yes	Unstable	Allowed	Forbidden	Allowed
No	Yes	Yes	Unstable	Forbidden	Allowed	Allowed
Yes	Yes	Yes	Unstable	Forbidden	Forbidden	Required

Table 4.1: Possibility statuses, their relation to stability and the allowed actions for the user. In this table, S is short for Stable and UNDEC is short for UNDECIDED.

- For other dimensions, the automatic extraction procedure is imperfect. Then all values that are possible according to the extraction procedure are in $V_p(d, c)$. This applies for example for the dimension related to the safe payment option (d_2 in our toy example). Although a classifier can check automatically whether or not the web shop displays a safe payment option (and assign the values $V_p(d_2, c) = \{v_{21}, v_{23}\}$ to a fact situation c), it is up to the human analyst to decide whether it is actually possible to perform this safe payment (and distinguish between v_{21} and v_{23}).
- Alternatively, no automatic procedure is possible or implemented for a dimension. In that case, $V_p(d, c) = V(\mathcal{D}, d)$. An example of this case is a dimension representing whether or not the web shop is reachable by phone.

In the remainder of this section, we show how the analyst uses the system to make a consistent and well-informed decision on c, without spending more resources than necessary. A high-level overview of the decision support system is illustrated in Figure 4.3. As a first step, the analyst obtains the stability status. If this status turns out to be Unstable, then the completions of the incomplete focus case have at least two different justification statuses. All PRO- and CONTRA-relevant updates are then identified using the algorithm for relevance and presented to the analyst. The analyst can then choose to further investigate (some of) the dimensions related to these updates and, after this investigation, update the incomplete fact situation. If the analyst obtains more information on the possible values V_p' to be assigned to some dimension d where $(d, V_p) \in c$, then in the updated incomplete fact situation c', (d, V_p) is replaced by (d, V_p').

After this update, the algorithms for stability (and, if necessary, relevance) can be executed again, until a stable situation is reached (which, by Proposition 4.4, will eventually happen if only relevant dimension values are removed) or the analyst decides that further investigation on the fact situation is no longer opportune given the available resources. At this point, the analyst makes a decision based on c', that is, on the current dimension values of c', the most updated version of the incomplete fact situation – which may or may not contain dimensions with multiple possible values.

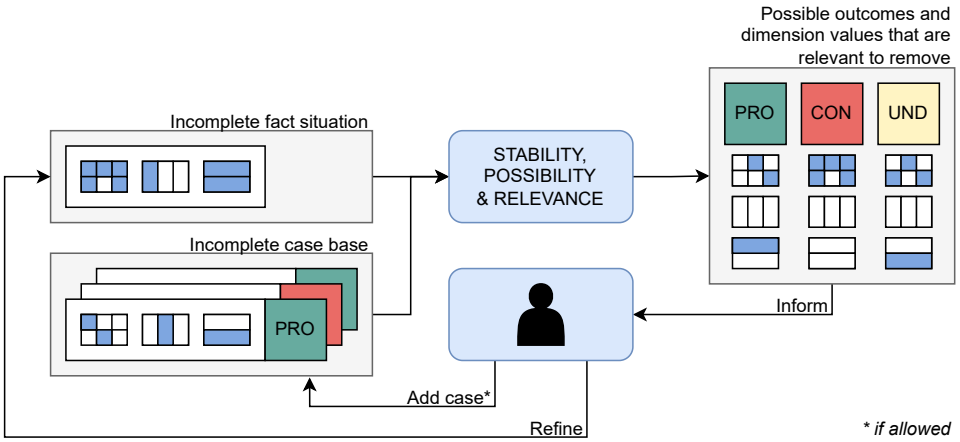


Figure 4.3: High-level overview of the proposed human-in-the-loop decision-making process for incomplete fact situations and incomplete case bases, involving algorithms for possibility, stability and relevance identification.

In Table 4.1 we summarise the relation between each combination of possibility statuses, the corresponding stability status and the actions that the user can or should take (i.e., refine the incomplete case or add it to the incomplete case base with some outcome). For completeness, we also include the Possible-UNDECIDED status, although it does not play a role in maintaining consistency. Actions that introduce an inconsistency according to Proposition 4.5 are forbidden; all other actions are allowed. For incomplete fact situations that are both Possible-PRO and Possible-CONTRA, the only allowed action is refinement. Therefore, we marked this action as required. Finally, if an incomplete fact situation c is Stable-o (o being PRO or CONTRA) then adding (c, o) to the incomplete case base $\mathcal{C}$ is obsolete, in the sense that for any new fact situation c', the justification status of c' given $\mathcal{C} \cup \{(c, o)\}$ is exactly the same as its justification status given $\mathcal{C}$. Refinement does not have any effect either, as every partial completion of c will be Stable-o as well.

4.6. Complexity and algorithms

At this point we have formally introduced the notions of justification, stability, relevance and possibility. In addition, we have shown how these notions are applicable for human-in-the-loop decision support systems. In this section, we will focus on complexity-theoretic aspects, provide algorithms for these tasks and prove under which conditions these algorithms run in polynomial time.

4.6.1. Computing justification

First, we will consider the task of computing the justification status given an incomplete case base and a complete fact situation. From Definition 4.12 we recall that a complete fact situation c is forced towards outcome o (PRO or CONTRA) by an

incomplete case base $\mathcal{C}$ iff there is some incomplete case (c', o) in $\mathcal{C}$ such that c' has at least one completion c'' such that $c \geq_o c''$. So the justification status could be computed by first constructing all possible completions of incomplete cases in the incomplete case base. However, this approach is very inefficient, as we illustrate with the following example.

Example 4.16 (Exponential blowup of naive justification algorithm)**.** Suppose that we are given some incomplete case base $\mathcal{C}$ within the domain $\mathcal{D}$ and some fact situation c within this domain. Assume that for each $d_i \in \mathcal{D}$, $d_i = (\{1, \ldots, 6\}, \leq)$, so each dimension has six possible values and higher values are considered to be more PRO. Now we consider the worst-case number of completions for a varying number of dimensions. First suppose that $|\mathcal{D}| = 1$ and that there is some incomplete case c' in the incomplete case base with $c' = \{(d_1, \{1, \ldots, 6\})\}$. A naive algorithm for justification that first computes all completions would need to check each of the 6 completions of c'. As the number of dimensions increases, the number of completions grows exponentially. Suppose, for example, that $|\mathcal{D}| = 3$ and consider an incomplete case (c'', o) with $c'' = \{(d_i, \{1, \ldots, 6\}) \mid i \in \{1, 2, 3\}\}$ in the incomplete case base. As c'' has $6^3 = 216$ completions, a naive justification algorithm requires 216 iterations. Similarly, if we have 6 dimensions that can take 6 different values, the number of iterations is 46656.

This naive approach is clearly impractical for actual applications, in which there can be a large number of dimensions, each of which may have many possible values. Therefore, we propose an alternative, more efficient, algorithm in this section. This algorithm is based on the following proposition.

Proposition 4.6. *[Reformulation of the a fortiori constraint for incomplete case bases] Let $\mathcal{C}$ be a consistent incomplete case base in domain $\mathcal{D}$ and c a complete fact situation. Then for outcome o (PRO or CONTRA): c is forced towards outcome o by $\mathcal{C}$ iff there is some incomplete (c', o) in $\mathcal{C}$ where c' is such that for each $d \in \mathcal{D}$ there is some $v' \in V_p(d, c')$ such that $v(d, c) \geq_o v'$.*

We use Proposition 4.6 to design a polynomial algorithm for computing the justification status of fact situation c given a consistent incomplete case base $\mathcal{C}$. Algorithm 8 works by iterating over $\mathcal{C}$, searching for an incomplete case $(c', o) \in \mathcal{C}$ such that for each dimension in the domain, there is at least one possible value v' of the incomplete fact situation such that $v(d, c) \geq_o v'$. If such an incomplete case is found, then by Proposition 4.6, the justification status is o; otherwise, it is UNDECIDED.

Algorithm 8 Computing the justification status

```
1: procedure COMPUTE-JUSTIFICATION(D, C, c)
2:     for incomplete case (c', o) in C do
3:         if for each d ∈ D: there is a v' in V_p(d, c') such that v(d, c) ≥_o v' then
4:             return o
5:     return UNDECIDED
```

Example 4.17. For the consistent incomplete case base $\mathcal{C}' \cup \{(c_5, \text{PRO})\}$, the complete fact situation $c'' = \{(d_1, v_{13}), (d_2, v_{23})\}$ is forced towards outcome PRO because of c_5. Note that for each dimension, just one possible value in c_5 needs to be "at least as PRO" as the value in c''. In this example, $V_p(d_1, c_5) = \{v_{11}, v_{12}, v_{13}\}$. To derive the PRO status of c'' it suffices that $v(d_1, c'') = v_{13} \geq_{\texttt{PRO}} v_{13}$, even though $v_{11} \not\geq_{\texttt{PRO}} v_{13}$ and $v_{12} \not\geq_{\texttt{PRO}} v_{13}$.

The algorithm requires a single run on the incomplete case base and for each of the incomplete cases it performs one comparison with the complete focus case for each possible dimension value. This implies that the algorithm is polynomial, therefore the problem of deciding justification can be computed in polynomial time (so is in P).

Proposition 4.7. *Deciding if a complete fact situation is PRO/CONTRA/UNDECIDED w.r.t. some consistent incomplete case base is in P.*

4.6.2. Computing PRO/CONTRA-possibility

In this section, we provide polynomial algorithms to check whether an incomplete fact situation is Possible-PRO or Possible-CONTRA w.r.t. some incomplete case base. Although the notion of possibility (Definition 4.19) is defined in terms of completions, it is not necessary to generate all completions in order to compute PRO/CONTRA-possibility. Instead, we propose an algorithm based on the following proposition.

Proposition 4.8. *Let $\mathcal{C}$ be a consistent incomplete case base in domain $\mathcal{D}$ and c an incomplete fact situation. Then for outcome o (where o is PRO or CONTRA): c is Possible-o w.r.t. $\mathcal{C}$ iff there is some incomplete (c', o) in $\mathcal{C}$ where c' is such that for each $d \in \mathcal{D}$ there is some $v \in V_p(d, c)$ such that there is some $v' \in V_p(d, c')$ such that $v \geq_o v'$.*

The result from Proposition 4.8 is directly used in Algorithm 9 for checking if the incomplete fact situation c is Possible-PRO w.r.t. some incomplete case base $\mathcal{C}$ and domain $\mathcal{D}$. The runtime of this algorithm is polynomial in the size of the input. A similar algorithm (Check-Possible-CONTRA) can be made by replacing all occurrences of PRO by CONTRA. Therefore, deciding whether c is Possible-PRO and/or Possible-CONTRA w.r.t. $\mathcal{C}$ and $\mathcal{D}$ can be done in polynomial time.

Proposition 4.9. *Deciding if an incomplete fact situation is Possible-PRO and/or Possible-CONTRA w.r.t. some incomplete case base is in P.*

4.6.3. Computing PRO- and CONTRA-stability

To determine the stability status of an incomplete fact situation, one could construct all possible completions and run the justification algorithm for each of them. However, this approach would be very inefficient for applications with a large number of dimensions, in particular if each of them has many possible values. Fortunately, it is possible to compute the stability status in a faster way, strongly limiting the number of calls to the justification algorithm.

Algorithm 9 Check if an incomplete fact situation is Possible-PRO

```
procedure CHECK-POSSIBLE-PRO(𝒟, 𝒞, c)
    for incomplete case (c′, o) in 𝒞 do
        if o = PRO then
            if for all d ∈ 𝒟 there are v ∈ V_p(d, c) and v′ ∈ V_p(d, c′) s.t. v ≥_PRO v′
then
                return True
    return False
```

In fact, for deciding whether an incomplete fact situation c is Stable-PRO, Stable-CONTRA or Unstable it suffices to only consider those completions that are most likely to have a different justification status. We will prove this later in this section (in Proposition 4.10). Informally, our proposed strategy for computing the stability statuses works as follows:

- To determine whether c is Stable-PRO, construct the "most CONTRA completions" by choosing the "most CONTRA" values for each dimension within the set of possible values. If the justification algorithm classifies each of the resulting complete fact situations as PRO, then c is Stable-PRO.
- To determine if c is Stable-CONTRA, construct the "most PRO completions". If these are all CONTRA, then c is Stable-CONTRA.
- In order to decide if c is Unstable, construct both all most PRO and all most CONTRA completions and compute their justification statuses. If not all of the resulting justification statuses are the same then c is Unstable.

The following definition (corrected from Odekerken et al. (2023a, Definition 14)[3]) specifies how these most PRO or CONTRA completions are constructed.

Definition 4.20 (Extreme completions)**.** Let c be an incomplete fact situation in domain $\mathcal{D}$.

- For each of the dimensions $d \in \mathcal{D}$, let the most PRO values of this dimension be $\text{MPD}(d, c) = \{v \in V_p(d, c) \mid$ there is no $v' \in V_p(d, c)$ such that $v' >_{\texttt{PRO}} v\}$.
- For each of the dimensions $d \in \mathcal{D}$, let the most CONTRA values of d be $\text{MCD}(d, c) = \{v \in V_p(d, c) \mid$ there is no $v' \in V_p(d, c)$ such that $v' >_{\texttt{CONTRA}} v\}$.
- The most PRO completion set of c is $\text{MP}(c) = \bigtimes_{d \in \mathcal{D}} \{(d, v) \mid v \in \text{MPD}(d, c)\}$ (where $\bigtimes$ is the cartesian product).
- The most CONTRA completion set of c is $\text{MC}(c) = \bigtimes_{d \in \mathcal{D}} \{(d, v) \mid v \in \text{MCD}(d, c)\}$.

Example 4.18. Consider a dimension with six possible values $d = (\{1, \ldots, 6\}, \leq'_{\texttt{PRO}})$ where $\leq'_{\texttt{PRO}}$ is the ordering illustrated in Figure 4.4. Let $c = \{(d, \{1, \ldots, 6\})\}$ be an

[3]Due to a notational error of Definition 14 in Odekerken et al. (2023a), a completion could contain multiple values for the same dimension, which should not be possible.

Algorithm 10 Get most PRO completions

```
1: procedure GET-MOST-PRO-COMPLETIONS(𝒟, c)
2:     initialise MPD
3:     for d ∈ 𝒟 do
4:         MPD[d, c] = ∅
5:         for v ∈ V_p(d, c) do
6:             if for each v' ∈ V_p(d, c): v' ⊁_PRO v then
7:                 add v to MPD[d, c]
8:     most_pro_completions = {∅}
9:     for d ∈ 𝒟 do
10:        new_most_pro_completions = ∅
11:        for completion_in_progress in most_pro_completions do
12:            for v ∈ MPD[d, c] do
13:                new_comp_in_progress = completion_in_progress + (d, v)
14:                add new_comp_in_progress to new_most_pro_completions
15:        most_pro_completions = new_most_pro_completions
16:    return most_pro_completions
```

Algorithm 11 Check if an incomplete fact situation is Stable-CONTRA

```
1: procedure CHECK-STABLE-CONTRA(𝒟, 𝒞, c)
2:     most_pro = GET-MOST-PRO-COMPLETIONS(𝒟, c)
3:     for c' ∈ most_pro do
4:         if COMPUTE-JUSTIFICATION(𝒟, 𝒞, c') ≠ CONTRA then
5:             return False
6:     return True
```

incomplete fact situation in domain $\{d\}$. Then $\text{MPD}(d, c) = \{5, 6\}$ and $\text{MCD}(d, c) = \{1, 2\}$. $\text{MP}(c) = \{\{(d, 5)\}, \{(d, 6)\}\}$ and $\text{MC}(c) = \{\{(d, 1)\}, \{(d, 2)\}\}$.

Algorithm 10 explicates this construction of extreme completions from Definition 4.20. Lines 2–7 compute the most pro values of each dimension (MPD) and Lines 9–16 compute the most PRO completions (MP).

In the following proposition, we prove that checking the justification status of these extreme completions suffices to determine the stability status of any incomplete fact situation.

Proposition 4.10. *Given an incomplete case base $\mathcal{C}$ and an incomplete fact situation c:*

1. *c is Stable-PRO given $\mathcal{C}$ iff each MC(c) is PRO given $\mathcal{C}$;*
2. *c is Stable-CONTRA given $\mathcal{C}$ iff each MP(c) is CONTRA given $\mathcal{C}$; and*
3. *c is Unstable given $\mathcal{C}$ iff the justification status of some MP(c) does not equal the justification status of some MC(c) given $\mathcal{C}$.*

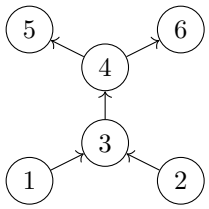


Figure 4.4: Hasse diagram of the ordering $\leq'_{\texttt{PRO}} = \{(1,1),(1,3),(1,4),(1,5),(1,6),(2,2),(2,3),(2,4),(2,5),(2,6),(3,3),(3,4),(3,5),(3,6),(4,4),(4,5),(4,6),(5,5),(6,6)\}$ on the set $\{1,\ldots,6\}$.

Using this result, the stability status of an incomplete fact situation c can be computed in a more efficient way, since it suffices to only call the justification algorithm for the most extreme completions in $\text{MP}(c) \cup \text{MC}(c)$. Algorithm 11 provides pseudocode that checks if an incomplete fact situation c is Stable-CONTRA w.r.t. incomplete case base $\mathcal{C}$ and domain $\mathcal{D}$. The algorithms for checking the Stable-PRO and Unstable status are similar.

Example 4.19. Suppose that we are given some incomplete case base $\mathcal{C}$ within the domain $\mathcal{D}$. Assume that for each $d_i \in \mathcal{D}$, $d_i = (\{1,\ldots,6\}, \leq'_{\texttt{PRO}})$ where $\leq'_{\texttt{PRO}}$ is defined as in Example 4.18 (illustrated in Figure 4.4). Suppose that $|\mathcal{D}| = 6$ and consider an incomplete focus case $c = \{(d_i, \{1,\ldots,6\}) \mid i \in \{1,\ldots,6\}\}$. As c' has $6^6 = 46656$ completions, the naive stability algorithm requires 46656 calls to Compute-Justification (Algorithm 8). However, the number of extreme completions $|\text{MP}(c)| + |\text{MC}(c)|$ is $2^{|\mathcal{D}|} + 2^{|\mathcal{D}|} = 2^6 + 2^6 = 128$, so the number of calls to the justification algorithm is greatly reduced compared to the 46656 calls of the naive approach.

If the number of completions in $\text{MP}(c) \cup \text{MC}(c)$ is small compared to the total number of completions of c, then this stability algorithm is much faster than the naive version. In fact, if for each dimension d, the set $V_p(d,c)$ has a single minimum and a single maximum value, then the stability algorithm runs in polynomial time.

Proposition 4.11. *Given an incomplete case base $\mathcal{C}$ in domain $\mathcal{D}$ and some incomplete fact situation c, if for each $d \in \mathcal{D}$ the set $V_p(d,c)$ has a single minimum and a single maximum value, then Check-Stable-PRO/Check-Stable-CONTRA runs in polynomial time for $\mathcal{D}$, $\mathcal{C}$ and c.*

Proposition 4.11 implies that computing stability is fast, even for very big inputs, if the possible values of each dimension have a single minimum and maximum. This is, for example, always the case if the ordering on each of the dimensions is total (such as $\leq$ or $\geq$). However, the exponential blowup that occurs if the possible dimension values have multiple maxima or minima cannot be circumvented in general, because the problem of deciding if a case is Stable-PRO/Stable-CONTRA is CoNP-complete, as we prove next.

Proposition 4.12. *Deciding whether an incomplete fact situation is Stable-PRO or Stable-CONTRA w.r.t. some consistent incomplete case base is CoNP-complete.*

To complete the complexity analysis of stability statuses, we prove that the problem of deciding whether an incomplete fact situation is Unstable is NP-complete.

Proposition 4.13. *Deciding whether an incomplete fact situation is Unstable w.r.t. some consistent incomplete case base is NP-complete.*

To summarise this section, the tasks of deciding on the stability statuses PRO-stable, CONTRA-stable and Unstable are inherently complex. This could yield problems for applications with many possible dimension values. We have provided the algorithms CHECK-STABLE-PRO and CHECK-STABLE-CONTRA for deciding on these stability statuses in a relatively efficient way and proved that these algorithms are polynomial for specific inputs.

4.6.4. Computing relevance

In this section, we consider computational aspects of the relevance problem. First, we will study its complexity. In order to prove an upper bound on the complexity, we will use the property that relevant operations can be verified by a single completion c' of c for which the value v for d causes a difference in justification status: c' has the justification status j, but replacing d's value by v results in a completion with a different justification status. Before formally proving this property in Proposition 4.14, we illustrate it with an example.

Example 4.20. We return to the web shop domain $\mathcal{D}$. Recall the incomplete case base $\mathcal{C}' = \{(c_1, \text{PRO}), (c_2, \text{CONTRA})\}$ where $c_1 = \{(d_1, \{v_{12}\}), (d_2, \{v_{23}\})\}$ and $c_2 = \{(d_1, \{v_{13}\}), (d_2, \{v_{22}\})\}$. Let $c_4 = \{(d_1, \{v_{11}, v_{12}, v_{13}\}), (d_2, \{v_{21}, v_{23}\})\}$ be an incomplete focus case. In Example 4.14 we showed that, among other things, the removal of value v_{11} is PRO-relevant for dimension d_1. Indeed, c_4 has a completion $\{(d_1, v_{12}), (d_2, v_{23})\}$ that is PRO w.r.t. $\mathcal{C}'$, but replacing the value of d_1 to v_{11} results in the alternative completion $\{(d_1, v_{11}), (d_2, v_{23})\}$, which is not PRO w.r.t. $\mathcal{C}$.

This property holds in general, as we show in Proposition 4.14.

Proposition 4.14. *Let $\mathcal{C}$ be an incomplete case base in domain $\mathcal{D}$ and let c be an incomplete fact situation. Let j be a justification status (PRO, CONTRA or UNDECIDED). Then for a given dimension $d \in \mathcal{D}$, the removal of possible value $v \in V_p(d, c)$ is j-relevant w.r.t. $\mathcal{C}$ and c iff there is some completion c' of c such that:*

- *c' is j w.r.t. $\mathcal{C}$; and*
- *c'' is **not** j w.r.t. $\mathcal{C}$ where c'' is equal to c' except for the value of d, which is v. Formally: $c'' = \{(d', v(d', c')) \mid d' \in \mathcal{D} \setminus \{d\}\} \cup \{(d, v)\}$.*

Proposition 4.14 is the crux of the proof that all three variants of the relevance problems are in NP. In addition, these variants are NP-hard. This implies that all three variants of the relevance problem are NP-complete.

Proposition 4.15. *For each justification status j (PRO, CONTRA or UNDECIDED), deciding whether the removal of a possible value for a given dimension of an incomplete fact situation is j-relevant w.r.t. some incomplete case base is NP-complete.*

Having identified the complexity of the relevance problem, we propose an algorithm for enumerating all PRO-and CONTRA-relevant removals, given an incomplete fact situation c and incomplete case base $\mathcal{C}$ in domain $\mathcal{D}$. Informally, this algorithm works as follows:

1. For each incomplete case $(c', o) \in \mathcal{C}$, construct the completions of c that "just suffice to match" c' and collect these complete cases in an alternative complete case base $\mathcal{C}^*$.
2. Let $\mathcal{C}'$ be a minimised version of $\mathcal{C}^*$, where complete cases forced by another complete case in $\mathcal{C}'$ are removed.
3. For each complete case (c', o) in $\mathcal{C}'$, for each $d \in \mathcal{D}$ and for each $v \in V_p(d, c)$ such that $v \not\geq_o v(d, c')$: construct an alternative complete fact situation c'' by replacing the value of d in c' by v. Store v as an o-relevant removal if the justification status is no longer o.

To give some intuition on this procedure: constructing $\mathcal{C}'$ is convenient as it allows us to efficiently find completions c' and c'' mentioned in Proposition 4.14, where the justification status switches between PRO/CONTRA and an alternative status. Next, we formally define the alternative case base with minimal matching completions.

Definition 4.21 (Alternative case base with minimal matching completions)**.** Given an incomplete fact situation c, an incomplete case (c', o) and dimension d in domain $\mathcal{D}$, the set of minimal matching values is $V_m(d, (c', o))$, which consists of all values v in $V_p(d, c)$ such that there is some $v' \in V_p(d, c')$ such that:

1. $v \geq_o v'$ (so they match) and
2. there is no $v'' \in V_p(d, c)$ such that both $v'' <_o v$ and for some $v''' \in V_p(d, c')$ it holds that $v'' \geq_o v'''$ (so they are minimal).

The set of minimal matching completions consists of all combinations of minimal matching values for each dimension: $C_m(c', o) = \bigtimes_{d \in \mathcal{D}} \{(d, v) \mid v \in V_m(d, (c', o))\}$.

Then the alternative case base $\mathcal{C}^*$ is constructed by combining the minimal matching completions and the outcome for each incomplete case in the original incomplete case base: $\mathcal{C}^* = \{(c^*, o) \mid (c', o) \in \mathcal{C} \text{ and } c^* \in C_m(c', o)\}$.

The procedure for obtaining the alternative case base w.r.t. some domain $\mathcal{D}$, original incomplete case base $\mathcal{C}$ and incomplete focus case c is given in Algorithm 12. The minimal matching values are computed in Lines 7–15; the minimal matching completions are found by Lines 16–23 and Lines 24–25 combine these into the alternative case base.

Example 4.21. We demonstrate how the alternative case base is constructed for our running example in the web shop domain $\mathcal{D}$. Recall that the incomplete case base is $\mathcal{C}' = \{(c_1, \text{PRO}), (c_2, \text{CONTRA})\}$ with $c_1 = \{(d_1, \{v_{12}\}), (d_2, \{v_{23}\})\}$ and $c_2 = \{(d_1, \{v_{13}\}), (d_2, \{v_{22}\})\}$. For the incomplete focus case $c_4 = \{(d_1, \{v_{11}, v_{12}, v_{13}\}),$

Algorithm 12 Get alternative case base within domain $\mathcal{D}$ for the original incomplete case base $\mathcal{C}$ w.r.t. the incomplete focus case c.

```
1: procedure GET-ALTERNATIVE-CASE-BASE(𝒟, 𝒞, c)
2:     initialise Vm and Cm
3:     𝒞* = ∅
4:     for (c', o) ∈ 𝒞 do
5:         for d ∈ 𝒟 do
6:             Vm[d, c', o] = ∅
7:             for v ∈ V_p(d, c) do
8:                 if there is some v' ∈ V_p(d, c') such that v ≥_o v' then
9:                     is_minimum = True
10:                    for v'' ∈ V_p(d, c) do
11:                        if v'' <_o v and for some v''' ∈ V_p(d, c'): v'' ≥_o v''' then
12:                            is_minimum = False
13:                            break
14:                    if is_minimum then
15:                        Add v to Vm[d, c', o]
16:        Cm[c', o] = {∅}
17:        for d ∈ 𝒟 do
18:            new_min_match_comp = ∅
19:            for comp_in_progress in Cm[c', o] do
20:                for v ∈ Vm[d, c', o] do
21:                    new_comp_in_progress = comp_in_progress + (d, v)
22:                    add new_comp_in_progress to new_min_match_comp
23:            Cm[c', o] = new_min_match_comp
24:        for c* ∈ Cm[c', o] do
25:            add (c*, o) to 𝒞*
26:    return 𝒞*
```

$(d_2, \{v_{21}, v_{23}\})\}$, we first compute the minimal matching values for each of the cases in $\mathcal{C}'$.

- $V_m(d_1, (c_1, \text{PRO})) = \{v_{12}\}$ and $V_m(d_2, (c_1, \text{PRO})) = \{v_{23}\}$.
- $V_m(d_1, (c_2, \text{CONTRA})) = \{v_{13}\}$ and $V_m(d_2, (c_2, \text{CONTRA})) = \{v_{21}\}$.

These are then used to compute the set of minimal matching completions.

- $C_m(c_1, \text{PRO}) = \{(d_1, v_{12}), (d_2, v_{23})\}$.
- $C_m(c_2, \text{CONTRA}) = \{(d_1, v_{13}), (d_2, v_{21})\}$.

Finally, these completions and their corresponding outcomes form the alternative case base $\mathcal{C}'' = \{(\{(d_1, v_{12}), (d_2, v_{23})\}, \text{PRO}), (\{(d_1, v_{13}), (d_2, v_{21})\}, \text{CONTRA})\}$.

After computing the alternative case base, an additional step reduces the number of cases, as one complete case forces another complete case with the same outcome.

Such complete cases are filtered out in the minimised case base, defined as follows.

Definition 4.22 (Minimised case base)**.** Given a complete case base $\mathcal{C}$, the minimised version is $\texttt{min}(\mathcal{C})$ where $\texttt{min}(\mathcal{C}) = \{(c', o) \in \mathcal{C} \mid$ there is no c'' such that (c'', o) is in $\mathcal{C}$ and $c'' <_o c'\}$.

After obtaining the minimal case base $\mathcal{C}'$, the procedure to obtain PRO-relevant updates for some incomplete focus case c is as follows. For each $(c', \text{PRO}) \in \mathcal{C}'$, for each $d \in \mathcal{D}$ and for each $v \in V_p(d, c)$ such that $v \not\succeq_{\texttt{PRO}} v(d, c')$, construct c'' by replacing the value of d in c' by v. Let j be the justification status of c'' w.r.t. $\mathcal{C}'$. If j is not PRO, then for the dimension d, the removal of v is PRO-relevant. The procedure for obtaining all CONTRA-relevant updates is analogous.

This procedure is given in pseudocode in Algorithm 13. Line 2 calls Algorithm 12 to find the alternative case base $\mathcal{C}^*$ Then $\mathcal{C}' = \texttt{min}(\mathcal{C}^*)$ is computed in Lines 3–6. Finally, Lines 7–14 compute the set R of relevant updates, where for $o \in \{\text{PRO}, \text{CONTRA}\}$: the removal of value v for dimension d is o-relevant for c w.r.t. $\mathcal{C}$ and $\mathcal{D}$ iff $(d, v, o) \in R$. In order to check whether (d, v, o) should be added to R, the code in Line 12 makes a variation c'' of complete case c' in the minimised case base $\mathcal{C}'$ in which the value of d is replaced by v and checks the justification status of c'' w.r.t. the original incomplete case base $\mathcal{C}$.

Algorithm 13 List all relevant updates for some incomplete focus case c w.r.t. (the original) incomplete case base $\mathcal{C}$ in domain $\mathcal{D}$.

```
1:  procedure List-relevant-updates(D, C, c)
2:      C* = Get-alternative-case-base(D, C, c)
3:      C' = ∅
4:      for (c', o) ∈ C* do
5:          if there is no (c'', o) ∈ C* such that c'' <_o c' then
6:              Add (c', o) to C'
7:      R = ∅
8:      for (c', o) ∈ C' do
9:          for d ∈ D do
10:             for v ∈ V_p(d, c) do
11:                 if v ⋡_o v(d, c') then
12:                     c'' = {(d', v(d', c')) | d' ∈ D \ {d}} ∪ {(d, v)}
13:                     if Compute-Justification(D, C, c'') ≠ o then
14:                         Add (d, v, o) to R.
15:     return R
```

Example 4.22. We continue Example 4.21. Note that, in this example, the alternative case base $\mathcal{C}'' = \{(\{(d_1, v_{12}), (d_2, v_{23})\}, \text{PRO}), (\{(d_1, v_{13}), (d_2, v_{21})\}, \text{CONTRA})\}$ is already minimised: $\mathcal{C}'' = \texttt{min}(\mathcal{C}'')$. First, we search for PRO-relevant updates w.r.t. incomplete focus case $c_4 = \{(d_1, \{v_{11}, v_{12}, v_{13}\}), (d_2, \{v_{21}, v_{23}\})\}$, by constructing variations on $(\{(d_1, v_{12}), (d_2, v_{23})\})$, the complete fact situation of the only PRO complete case in $\mathcal{C}''$.

- For dimension d_1 we check v_{11} and v_{13} as they are not at least as PRO as v_{12}.
  - $(\{(d_1, v_{11}), (d_2, v_{23})\})$ is not PRO w.r.t. $\mathcal{C}'$, so the removal of v_{11} is PRO-relevant.
  - $(\{(d_1, v_{13}), (d_2, v_{23})\})$ is not PRO w.r.t. $\mathcal{C}'$, so the removal of v_{13} is PRO-relevant.
- For dimension d_2 we check v_{21} as it is not as least as PRO as v_{23}. $(\{(d_1, v_{12}), (d_2, v_{21})\})$ is not PRO w.r.t. $\mathcal{C}''$, so the removal of v_{21} is PRO-relevant.

Similarly, we search for CONTRA-relevant updates by constructing variations on $(\{(d_1, v_{13}), (d_2, v_{21})\})$, the complete fact situation of the only CONTRA complete case in $\mathcal{C}''$.

- For dimension d_1 we only check v_{12}. $(\{(d_1, v_{12}), (d_2, v_{21})\})$ is not CONTRA w.r.t. $\mathcal{C}'$, so the removal of v_{12} is CONTRA-relevant.
- For dimension d_2 we check v_{23} as it is not as least as CONTRA as v_{21}. $(\{(d_1, v_{13}), (d_2, v_{23})\})$ is not CONTRA w.r.t. $\mathcal{C}'$. Therefore, the removal of v_{23} is CONTRA-relevant.

The algorithm for relevance is exact, as stated in the following proposition.

Proposition 4.16. *Within some domain $\mathcal{D}$, let $\mathcal{C}$ be an incomplete case base and let c be an incomplete focus case. For any dimension $d \in \mathcal{D}$, $(d, v, o) \in$ LIST-RELEVANT-UPDATES$(\mathcal{D}, \mathcal{C}, c)$ iff v is o-relevant to remove for dimension d w.r.t. c, $\mathcal{D}$ and $\mathcal{C}$.*

Similar to the stability algorithm from Section 4.6.3, the proposed algorithm has a worst-case exponential runtime but runs in polynomial time under specific conditions on the dimension ordering.

Proposition 4.17. *Within some domain $\mathcal{D}$, let $\mathcal{C}$ be an incomplete case base and let c be an incomplete focus case. If for each dimension $d = (V, \leq_{PRO})$ in $\mathcal{D}$, $\leq_{PRO}$ is a total order on V then* LIST-RELEVANT-UPDATES *runs in polynomial time.*

In this section, we have proposed efficient algorithms for the problems of justification, stability and relevance. Although some of these problems are in high complexity classes, we have identified conditions on the dimensions for which the algorithms run in polynomial time.

4.7. Related work

The research area of reasoning with legal cases originated from Rissland and Ashley's work on the HYPO system for US trade secret's law (Rissland and Ashley, 1987). Interestingly, the suggestion of studying hypothetical variations to the focus case by varying dimension values was already put forward for HYPO by Ashley and Rissland (1988), motivated by similar reasons as ours: as explained by Ashley (1991), suggestions for hypothetical variations are useful in lawyers' preparation for trial, because they cannot assume to be certain of all elements of the fact situation, and hypotheticals suggest dimensions that they can elicit from a client or prepare to

rebut. Whereas many ideas in HYPO have greatly influenced the AI and Law field (Bench-Capon, 2017), this aspect seems to have attracted less attention.

Compared to HYPO, our approach differs in, for instance, the evaluation of cases, the selection of the citeable case(s) and the strategy for generating hypothetical variations. Whereas HYPO focused on generating debates as they can take place between lawyers, we focus on classifying the outcome of legal cases, in particular those cases for which the outcome is forced by a precedent. Related work in this area includes Prakken and Sartor (1998), Ashley and Brüninghaus (2009), and Zurek (2012) and, in particular, research on the models for precedential constraint proposed by Horty (2011) and Horty (2019), especially the result model. The main difference between existing work on the result model and this chapter is that we do not constrain the dimensions of precedent and focus cases to have a single value, but rather allow for a set of possible values. Moreover, whereas the models for precedential constraint in Horty (2011) and Horty (2019) are defined on cases that either contain only factors or only dimensions, in our work we also consider cases that combine factors and dimensions in a similar way as Prakken and Ratsma (2022). An alternative way of combining factors and dimensions in a single model of precedential constraint is presented by Prakken (2021). Finally, whereas the result model has been applied for explaining outcomes of machine-learning-based decision-making applications (Prakken and Ratsma, 2022), we propose an alternative application for human-in-the-loop decision support.

In recent work, the idea of modeling precedents with multiple interpretations has been proposed by Rigoni (2024), with the aim of incorporating a theory of vertical precedent in U.S. federal appellate courts. More specifically, Rigoni sketches a variant of the reason model where each case is replaced by a set of possible interpretations of the factors of the case. In contrast to our approach, the case base in Rigoni (2024)'s approach may be exponential in size compared to the original case base. As additional differences, Rigoni (2024) focuses on the reason model of precedential constraint where a case is represented by (interpretations of) factors, while we study the result model where a case is represented by (possible) dimension values.

Another area of related research is factor ascription (Bench-Capon and Atkinson, 2022), which provides argumentation schemes for the presence of factors in a case. This could be used for instantiating possible dimension values.

Finally, the idea of defining and computing a stability status and relevance notion, based on possible future information updates, was developed in other settings as well. Our definitions for stability and relevance are inspired by the definitions presented for argumentation-based reasoning in Chapters 2 and 3. In addition, Governatori et al. (2022) define stability for Defeasible Deontic Logic theories.

4.8. Conclusion

In this chapter, we have defined and studied the notions of justification, stability, possibility and relevance for Horty's result model of precedential constraint. In con-

trast to earlier work on this model, we do not assume that all factors and dimension value assignments of the focus case have been established with certainty. In order to account for this, we have introduced the notion of incomplete fact situations. We have defined an incomplete fact situation as *stable* if its justification status, that is, the enforced outcome by the precedential constraint, does not change, regardless of any change in the uncertain elements. For incomplete fact situations that are not stable, we have studied the task of identifying *relevance*: which updates on dimensions can be performed to make the resulting incomplete fact situation stable? By introducing the notions of stability and relevance, we answered research question 1 in the context of precedent-based reasoning.

We have described how these tasks can be applied in a human-in-the-loop decision support system: if an incomplete fact situation is stable, then further investigation is not required. Otherwise, relevant updates reveal which dimensions should be investigated further. In addition, we have identified the notion of *possibility*, which is required to guarantee the consistency of the incomplete case base. By establishing the complexity classes in which the problems of deciding stability and relevance (as well as justification and possibility) are situated, research question 2 was answered as well.

Finally, we have answered research question 3 by showing that the tasks of identifying the justification, possibility and stability status of a focus case w.r.t. an incomplete case base can be performed efficiently, just like the task of identifying relevant uncertainties.

5

Applications in law enforcement

This chapter is based on the following papers:

1. In Odekerken and Bex (2020) we introduce the web shop classification tool. This paper, presented at JURIX 2020, was written in collaboration with Floris Bex.
2. In Odekerken et al. (2022a) we provided implementation details on the systems discussed in this section. This paper was written under supervision of Floris Bex and AnneMarie Borg. Bas Testerink contributed to the description of the applications. The paper is published in Intelligent Systems with Applications.

The following demo papers describe PyArg, discussed in Section 5.3:

1. In Borg and Odekerken (2022), we introduced PyArg. This demo paper, presented at COMMA 2022, was written in collaboration with AnneMarie Borg. My contributions consisted of setting up the project and implementing the ASPIC$^+$ backend code.
2. Odekerken et al. (2023c) demonstrates a new release of PyArg with generators, an updated visualisation, a learning environment, a page for demonstrating algorithms and one for showcasing applications. This demo paper was written in collaboration with AnneMarie Borg and Matti Berthold. I contributed to the visualisation, learning environment and the application page and wrote the paper. It was presented at the Arg&App workshop in 2023.
3. Odekerken et al. (2023d) gives an overview of the features of PyArg. I

wrote this demo paper in collaboration with AnneMarie Borg and Matti Berthold. It was presented at the AI^3 workshop in 2023.

4. Xia et al. (2024), accepted as a demo paper for COMMA 2024, demonstrates a new layered visualisation for grounded labellings of abstract argumentation frameworks. It was written in collaboration with Yilin Xia, Shawn Bowers and Bertram Ludäscher. My role in the collaboration was the implementation of this alternative visualisation in PyArg.

5.1. Introduction

In the previous chapters, we studied the theoretic notions of stability and relevance in incomplete argumentation frameworks (Chapter 2), ASPIC$^+$ (Chapter 3) and precedent-based reasoning (Chapter 4). In addition to introducing these notions for all three formalisms, we studied the computational complexity and provided efficient algorithms, thus answering research questions 1, 2 and 3. In this chapter, the fourth and final research question is answered as we illustrate how these algorithms can be applied in practical tools and systems. First, we present two applications for the algorithms for justification, stability and relevance at the Dutch police in Section 5.2. Subsequently, we demonstrate PyArg, an open-source package and web interface in which these and other algorithms can be used. This software is aimed at researchers in the computational argumentation community, as well as anyone aiming to set up an argumentation-based system.

5.2. Case studies at the Dutch police

In this section, we illustrate how the algorithms for estimating stability and relevance can be applied in transparent human-in-the-loop decision support systems at the Dutch police. We present two use cases: fraud intake and the classification of web shops. These use cases were already mentioned in the introduction and used as running examples throughout this thesis. For both use cases, we developed systems that are implemented at the Dutch police. In particular, the system for fraud intake has been used over 300.000 times since its launch in September 2019.

5.2.1. Fraud intake

The first use case at the Dutch police concerns a system that assists in the intake of online trade fraud. This involves cases such as fake web shops and malicious second-hand traders on trading platforms such as eBay. Every year, the police receives tens of thousands of complaints on online trade fraud, which are typically sent by citizens via an online form, and this number is increasing. Traditional online forms for intake have proven to be insufficient. Citizens tend to have a hard time identifying which facts are relevant or not from a legal standpoint. It is also not always clear for a citizen whether a case is a criminal or civil case. The system therefore helps the citizen to provide all relevant legal facts and then proceeds to advise the citizen on the best course of action given these facts and the legal context.

Example 5.1 (Trade fraud complaint example)**.** This is a fictitious example of the free text in a complaint.

> *I was looking for a playstation and contacted fred fraudster via eBay. he said that he lives in aberdeen which is far away so we agreed that he would send it to me. But after two weeks waiting I still have not recieved anything.*

Except for the fact that this text is written in English and not in Dutch, this is a typical example of the free text submitted in a complaint on online trade fraud. In

this complaint, some crucial information is missing: the complainant in this example does not say if they paid for the product.

Architecture

The system for fraud intake is implemented according to the architecture illustrated in Figure 5.1. Given the initial complaint text provided by a citizen, the *information extraction* component uses natural language processing techniques to automatically extract initial observations. These observations are then combined with an argumentation system to build arguments for and against the topic literal. The *stability* component uses Stability-Label from Section 3.7 to decide if any additional observations that the user could possibly add in the future can change the justification status of the topic literal. If there are still relevant potential observations, then the *relevance-based policy* module returns the best query given current observations. Otherwise, the dialogue terminates and an explanation of the outcome is generated.

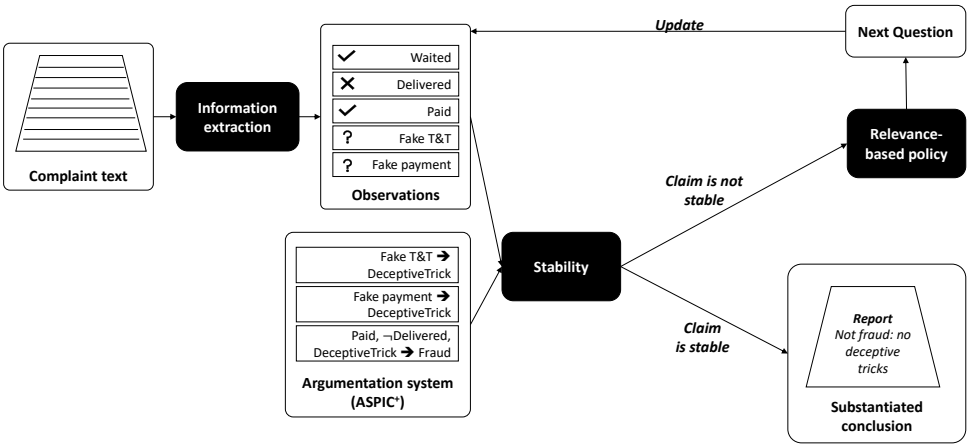


Figure 5.1: Illustration of the process in the fraud intake system.

Information extraction

For the information extraction component, observations are extracted from the initial free-text user input. Observations in this domain are, for example, "the complainant paid" or "the product was not delivered". After experimenting with various methods (Schraagen et al., 2017; Schraagen et al., 2018; Schraagen and Bex, 2019), including machine learning approaches and hand-made classifiers, we decided to use document-level regular expressions (regexes) for information extraction. We furthermore enable the citizen to correct any classification mistakes in a separate screen, which is shown immediately after the text input step. From this point, the observations are used as axiom literals in the initial knowledge base.

Argumentation system

The argumentation system was developed in collaboration with domain experts at the national centre for counteracting online trade fraud and the Dutch Public

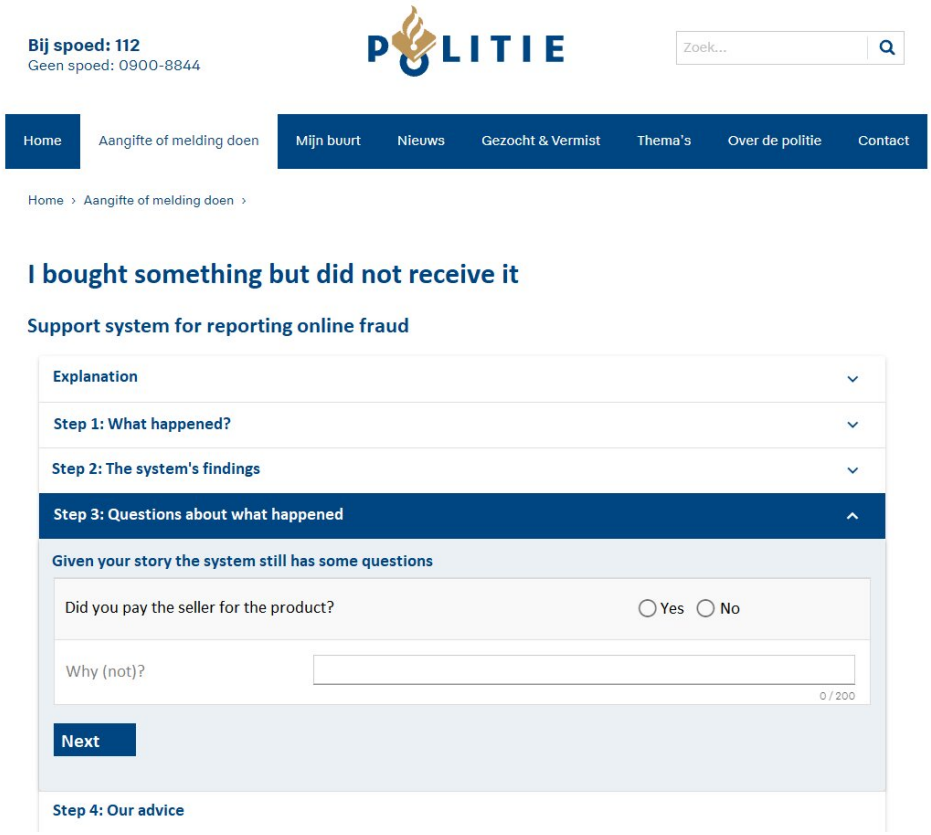


Figure 5.2: The system on the web site of the Dutch police. The English translation is ours; the system on the police web site is only in Dutch.

Prosecution Service. Table 5.1 reports on some specifications. The argumentation system has a layered structure in that there is a high-level rule, based on Article 326 of the Dutch Criminal Code, specifying whether there is evidence of fraud. We use seven different topic literals, which correspond to the follow-up actions. These actions include, for instance, accepting the complaint, consulting a human expert, rejecting the complaint but accepting it as a civil case or completely rejecting it and are dependent on the rule on fraud. Other rules relate general legal terms to more specific literals, where the most specific literals are concrete enough to ask to the citizen using the system. These queryable literals make up half of the language. There are no rules for queryable literals.

For contradiction, we use a function in which not only each literal and its negation are contradictories, but we add an additional contradictory relation between some literals. For example, the observation $\neg$*package_ordered* and the observation *delivery_proof_fake* are contradictory, because alleged victims of online trade fraud that did not order a package will not receive a fake delivery proof. This way, we

Number of literals	30 (and negations)
Number of queryable literals	15 (and negations)
Number of topic literals	7
Number of rules	43
Number of future argumentation theories from $(AS, \emptyset)$	5.443.200
Number of potential arguments	126
Number of arguments in at least one future argumentation theory	124

Table 5.1: Specification of the argumentation system used for fraud inquiry at the police

restrict the number of questions in some dialogues. Note that, if the knowledge base is empty, the number of future argumentation theories is 5.443.200; if we did not have these additional contradictories, we would have $3^{15} = 14.348.907$ future argumentation theories (as there are 15 non-negated queryables and for each of these there are three options: they can be present in the knowledge base, their negation is present or neither).

Whereas the process of identifying rules *for* fraud was relatively straightforward, it was more challenging to construct rules on which counterarguments could be based. For domain experts without a background in logic or knowledge representation, it is not trivial to fully understand the consequences of an attack between arguments. In particular, we experienced difficulties in identifying on which level a counterargument should be placed. To give an example: we recently extended the rule set so as to accept not only cases of fraud, but also fraud attempts. In case of a fraud attempt in the context of online trade fraud, the complainant realised in time that the counterparty would not deliver goods as promised and therefore did not pay. For fraud, the police maintains a directive that the complainant should have waited at least five days after the promised date of delivery. The complainant not having waited long enough is a reason for the police not to accept a complaint on trade fraud. However, the question if it would be a reason not to accept a complaint on fraud attempt turned out to be hard to answer. We tried to simplify this process in two ways.

First, we developed a user interface that visualises the argumentation system and directly shows the effect of adding or removing a literal to the knowledge base on the literal labelling. This gave not only domain experts but also the software developers implementing the intake system, insight in the working of the algorithm. Second, we iteratively developed the rules by providing the domain experts various example observation sets and asking them if this would be a case of (attempted) fraud or if more information would be required.

Given the size of the argumentation system and our experiments in Section 3.9, we expected that running stability takes no more than a few milliseconds. We verified this with the following experiment. First, we generated a data set in which we sampled 10.000 knowledge bases, i.e. consistent subsets of $\mathcal{Q}$, for each size between 0 and 15. This resulted in 160.000 argumentation theories based on the fraud intake argumentation system. For each of these argumentation theories, we measured the

time required for running Stability-label ten times. The average computation time is 1.7 ms and is not dependent on the number of unknown literals. This shows that our algorithm can be used in real time for estimating stability in applied argumentation-based inquiry.

Relevance-based policy

In argumentation theories where not every topic literal is stable, it is necessary to query the citizen using the system for more information. This is handled by the policy module. In the fraud intake system, the policy module consists of two parts: a relevance listing step that identifies which questions are still relevant and a strategy for choosing the next question of this relevant set.

The relevance listing step is argumentation-based and uses a heuristic algorithm that is similar to Stability-label in order to find out which queryable literals are still relevant in that observing them in the future would lead to a stable situation. The details of this algorithm are outside the scope of this thesis, but the general idea is that each literal and rule is associated with a set of relevant queryables for each of the justification statuses.[1] This set is iteratively filled with literals that could cause a boolean in the labelling to turn from True to False. Once we have a set of relevant literals for each of the topics, the inquiry system should select the next question from these relevant literals. In the current system, the system asks for the literal in the relevant set with the highest priority.

Explaining stability

After all relevant questions have been asked, every topic label is stable and the advice (corresponding to stable-defended topic literals) can be presented to the citizen. Especially in those situations where the system advises citizens not to submit a complaint, it is important to provide an explanation. An initial experimental study on the effect of various types of explanation on the trust of citizens shows that a substantive explanation that gives reasons for its advice (e.g. "The advice is not to submit a complaint. This is probably not a case of fraud since you tried to buy a product from a trusted web shop") can help in improving citizens' trust in the fraud intake system (Nieuwenhuizen, 2020). Being argumentation-based, our inquiry system architecture has high potential for explainability. For instance, in order to explain the justification or stability status of a particular literal, multiple explanations can be derived, in various levels of detail. For example, to explain the justification status of a topic literal we can provide the complete argument for the literal, the argument and its attackers or only the observations that were provided by the citizen.

Explainable AI in argumentation is a research area on itself (see e.g. Čyras et al. (2021), Vassiliades et al. (2021), and Borg and Bex (2024)), in which various types

[1] This heuristic algorithm works well for the argumentation system on fraud, but we did not test its accuracy for general argumentation systems. Instead of using a heuristic algorithm, the ASP-based algorithm for relevance proposed in Section 3.8 could be used. When the fraud intake system was implemented in 2018-2019, the ASP-based algorithms did not yet exist.

of explanations have been discussed. Still, it is far from trivial to automatically generate compact explanations that are understandable to a wide audience with a high variance of language proficiency. For our use case, we chose the following approach. First, we determine the minimal stable-defended future theories (Definition 3.13) from the argumentation theory with an empty knowledge base, that is: the smallest sets of queryable literals for which a topic literal is labelled stable-defended. It turns out that for this use case, the number of minimal stable subsets of the literal *reject complaint* is relatively small, which made it feasible to manually formulate a natural-language explanation for each of them. These natural-language explanations are similar to the one provided at the beginning of this paragraph. When the inquiry system has reached the point where the literal *reject complaint* is stable-defended, it matches its knowledge base to the minimal stable-defended future theories and their corresponding explanations and it returns the (first) matching natural language explanation.

Discussion

The development of this fraud intake system has been a multi-year effort. In the beginning of the design process, many techniques were considered (Bex et al., 2016), including direct classification of the advice based on machine learning (Kos et al., 2017). The results of the latter approach were, however, not satisfactory, as the incompleteness of the initial text was not considered. In addition, the advice produced by the machine learning methods could not be explained well. Finally, predicting the advice is just one task in the fraud intake process, which also requires the collection of additional information. We briefly experimented with machine learning to learn follow-up questions, but did not continue this branch of research, due to an additional issue of the machine learning approach: it requires a data set of complaints with high-quality labels. For this use case, no such data set was available because, in practice, police analysts often did not have the time to ask all relevant questions. In fact, this was the motivation for developing an intake system. We therefore decided to focus on argumentation-based approaches, which are transparent and do not require training data.

Meanwhile, the fraud intake system presented in this chapter has been running on the police website since September 2019 and has been used over 300.000 times.[2] The performance of the intake system is monitored in several ways. First, the submitted complaints (enriched with the questions asked by the system and the answers provided by the citizen) are assessed by human analysts, who typically make their decision independent of the system's advice (Soares et al., 2023). In around 85% of these complaints, the advice of the system agrees with the analyst's decision. One of the reasons for disagreement is that, when the analyst makes a decision, some time has passed and there is more information available regarding the complaint than when the complaint was investigated by the system. This happens, for example, if the complaint concerned a web shop that was investigated in response to earlier complaints and classified as mala fide. Apart from this accuracy evaluation, we evaluated the user experience in the first years of deployment of the system. In a

[2]It is available on the police website on `https://aangifte.politie.nl/iaai-preintake/`.

poll that popped up immediately after usage of the system, we asked citizens about their experience. On average, citizens gave a score of 4.3 out of five, which suggests that they are quite satisfied with the system.

5.2.2. Web shop classification

Our second use case concerns a human-in-the loop classifier that distinguishes bona fide from mala fide web shops. Many complaints on online trade fraud concern reports on web shops that do not deliver goods. Nonetheless, not each of these shops has bad intentions: in many cases, the customer fell victim to malfunctioning delivery service, rather than fraud. Therefore, analysts at the national centre for counteracting online trade fraud check suspicious web shops and classify them as bona fide or mala fide. This classification task is a combination of routine work (that can be automated) and more detailed investigation (that should be done by humans). As we suggested at the end of the previous section, the process for counteracting trade fraud benefits from efficient web shop classification. In this section, we therefore introduce a system for human-in-the-loop web shop classification that relies on static and dynamic algorithms for both argumentation- and precedent-based reasoning.

The classification outcome has serious implications: web shops classified as mala fide will be taken offline, while web shops classified as bona fide can be placed on a white list for fraud intake (which is then used by the fraud intake system from Section 5.2.1). In view of this, it is required that a human analyst checks each piece of advice given by the classifier. However, we should be alert to the control problem (Zerilli et al., 2019), i.e. the situation that a human analyst devolves too much responsibility to the classifier and fails to detect cases where the classifier is wrong. To prevent this, analysts should be kept actively in the loop: they should, for example, be notified of possible mistakes by the classifier and be encouraged to check these situations. An additional motivation for a human-in-the-loop approach is that some factors influencing the decision can only be found by a manual investigation, such as making a payment. In cases where these values could be relevant, the analyst should be invited to investigate these factors and return the resulting information to the system.

As an additional requirement, the classifier should be transparent. Currently, the result of a web shop check by human analysts is a well-founded advice that includes the factors that made them decide on their conclusion. This is required for various purposes, such as alerting citizens and informing the registrar in a web site take-down request, so our system should be able to produce similar explanations.

Web shop classification system architecture: from URL to initial advice

Given these requirements, we propose a system that combines rule- and precedent-based reasoning with incomplete information, thus using algorithms from Chapters 3 and 4. Informally, the rule-based part is used to derive which factors are present for a given web shop. Factors are either bona fide (e.g. *"uses https"*) or mala fide (e.g. *"uses fake hallmark logo"*). Relating to Chapter 4, a factor can be considered as a

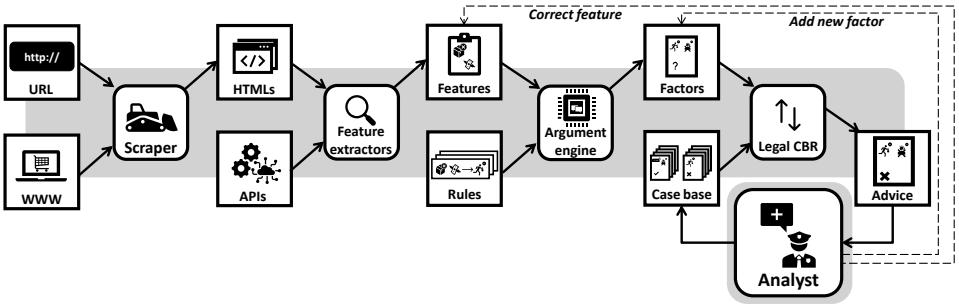


Figure 5.3: Web shop checker architecture.

dimension with two values: "present" and "absent". The precedent-based part of the classification procedure then compares the factors of the new web shop to a case base that contains factors of precedent web shops and the corresponding decision ("bona fide" or "mala fide").

The system is illustrated in Figure 5.3. The initial input is a URL that is considered suspicious. As a first step, a *web scraper* scrapes the web shop's HTML pages. Subsequently, features are extracted from the HTML by *feature extractors*. Some feature extractors require *API* calls to obtain additional information from external organisations. The resulting feature vector is the initial knowledge base used by the argumentation engine, which handles the rule-based reasoning part.

Apart from this knowledge base, the *argumentation engine* has access to an argumentation system and a set of queryables that model the requirements for a factor to be present. For example, Figure 5.4 illustrates the rules related to the factor *b_trusted* (stating that the web shop is on the trusted list of some trustmark company) and *m_fake_logo* (stating that the web shop misuses a trustmark logo). In this example, *b_trusted* is unsatisfiable in the current argumentation theory, as well as in each argumentation theory that can be obtained in the future. The reason for this is that its antecedent *a_registered* is unsatisfiable, given that a contradictory of this queryable literal is observed. The factor *m_fake_logo* is defended in the current argumentation theory, since there is an argument for this literal that is not attacked. However, note that we defined the rule set in such a way that the analyst can overrule feature extractors in stating that a feature is present or absent. The queryable $\neg$*e_logo* can still be added to the knowledge base and would change the justification status of *m_fake_logo* from defended to out. Such a correction could lead to a change in the factors corresponding to the web shop, which may indirectly influence the advice. The stability and relevance algorithms for ASPIC$^+$ from Chapter 3 are used to identify which features can still be obtained after manual investigation by the analyst.

The *precedent-based reasoning (Legal CBR)* module compares the factors of the tested web shop to a case base of earlier cases. Using the algorithm for justification from Chapter 4, it identifies precedential constraints (Horty, 2011) based on a fortiori

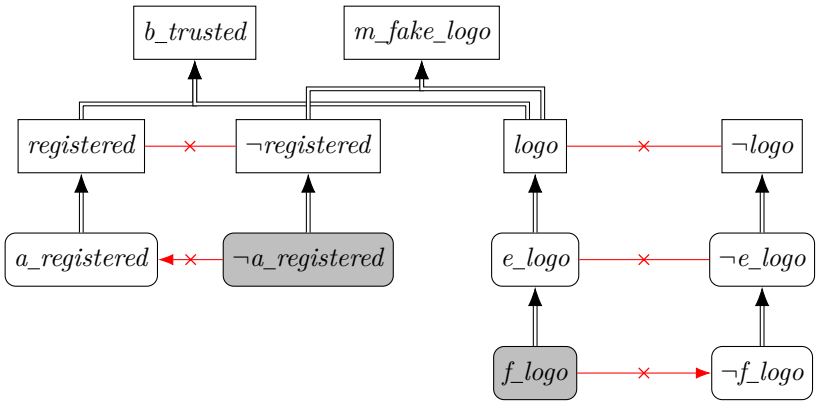


Figure 5.4: Excerpt from the argumentation system and queryables. If a feature extractor found a trustmark logo at the web site (*f_logo*), but an API call returns that this site is not registered at the trustmark company (¬*a_registered*), there is an argument for the factor that the web shop uses a fake trustmark logo (*m_fake_logo*). However, if the analyst searches for the logo on the web shop, the image found by the feature extractor may not be a trustmark logo (¬*e_logo*). In that case, each argument for *m_fake_logo* is attacked by the observation-based argument for ¬*e_logo*. Consequently, *m_fake_logo* is out in this future argumentation theory, hence is not considered as a factor in the (best-case) set of factor.

reasoning. A web shop is constrained to be mala fide if its factors are at least as "bad" as those of a precedent case labelled mala fide. Similarly, our web shop is constrained to be bona fide if its factors are at least as "good" as those of a precedent case labelled bona fide. If no precedential constraint applies, the tested web shop is classified as undecided. This way, an initial advice (bona fide, mala fide or undecided) for the web shop is obtained. However, all factors that are not stable (according to the argumentation part) have two possible values: present and absent. These possible values are used to compute precedent-based notions of stability and relevance (using the algorithms presented in Chapter 4).

The *advice* that is presented to the analyst contains not only the initial advice (based on and explained by the defended factors), but also suggestions for alternative outcomes. If such an alternative outcome exists, the system presents relevant factors to be added or removed, as well as the queryables that are relevant for the addition (defended) or removal (other statuses) of that factor. An example is given in Figure 5.5.

Discussion

This system has been implemented for the classification of fraudulent web shops at the Dutch Police. The implementation of the proposed system required a significant amount of knowledge engineering, since the rules for the argumentation engine are identified manually. We consider this effort to be worthwhile, since the rule set provides a way to reason with incomplete information and to generate human-readable explanations. Finally, note that many rules can be obtained easily since they fit a certain scheme (reminiscent to argument schemes in Walton et al. (2008)) – for

The web shop `www.suspicious-shop.com` seems to be bona fide. This advice is based on following dimension values:

- The Chamber of Commerce number mentioned on the web site exists;
- The VAT number on the web site is valid.

This advice is based on a comparable advice for `www.bona-fide-shop.com`. However, the following information would change the advice:

- Payments are transferred to a foreign bank account.
  This dimension value can be obtained by making a payment.

Figure 5.5: Example of explanation for a new advice, based on the factors of a precedent web shop.

example, some feature is present if it is detected by a feature extractor or if it is observed by an analyst; see the rules for logo in Figure 5.4. In case of conflict, the rule based on the analyst's observation is stronger.

5.3. PyArg

In the previous sections, we showed how algorithms for stability and relevance can be used in decision support systems at the Dutch police. In addition, we made these and other algorithms available for the research community in PyArg. PyArg is an open-source software implementation in Python that provides practical algorithms for theoretical problems in various argumentation formalisms and makes (potential) applications of these algorithms visible in a web interface. PyArg is intended to be a software solution for researchers within the argumentation community, students who may become part of it, as well as stakeholders outside the community. Depending on the users' goals, they can (A) validate and extend the open-source implementations of argumentation algorithms on GitHub; (B) install the Python package in one line and use it as a dependency in other Python projects; and/or (C) explore PyArg's functionalities in the web interface. In this section, we give a brief overview of PyArg's functionalities from the frontend and from the backend.

For an extended overview of software related to computational argumentation, we refer to Cerutti et al. (2018). In this section, we relate to implementations that are most similar to PyArg. Tweety (Thimm, 2018) is a comprehensive collection of Java libraries that includes algorithms for both abstract and structured approaches to argumentation. It is in fact more comprehensive than PyArg, but does not have a visualisation option. The Online Argument Structures Tool (TOAST) (Snaith and Reed, 2012) does provide a visualisation for ASPIC$^+$, but the source code is not openly available. Gorgias Cloud (Spanoudakis et al., 2023) is a recent system that is similar to PyArg, but is based on the Gorgias argumentation system. NEXAS (Dachselt et al., 2022) is an alternative approach to visualising extensions of abstract argumentation frameworks, which, compared to PyArg, is more aimed towards (large) frameworks with many extensions. Finally, many algorithms for argumentation-

related problems have been submitted to the ICCMA competition.[3] However, these are mostly focused on fast implementation of limited problems, mainly in the context of abstract argumentation.

5.3.1. Backend functionalities

In this section, we describe the functionalities that can be used from the backend, by cloning the Python code from GitHub[4] or installing it from PyPI[5] using `pip install python-argumentation`. Specific usage examples can be found on the documentation website.[6]

The Python package PyArg currently supports abstract argumentation (Dung, 1995), ASPIC$^+$ (Prakken, 2010) and assumption-based argumentation (ABA) (Bondarenko et al., 1993; Čyras et al., 2018). It provides algorithms for *evaluating* argumentation settings in different semantics (Baroni et al., 2011). In addition, it has functionality for *explaining* the (non-)acceptance of arguments and formulas in abstract argumentation frameworks (AFs) and ASPIC$^+$ argumentation theories (Borg and Bex, 2021a; Borg and Bex, 2021c). Furthermore, PyArg provides algorithms for *dynamic* argumentation problems. In particular, the package contains an implementation of the heuristic algorithm for the stability problem from Section 3.7.3, as well as an inexact but efficient algorithm for estimating relevance based on the labels from the aforementioned stability algorithm. Finally, PyArg provides algorithms for *realisability* in abstract argumentation (Dunne et al., 2015) and ABA (Berthold et al., 2023c), that is: given a semantics and a set of extensions, is there an (assumption-based) AF that, for the given semantics, has exactly these extensions?

In addition, PyArg provides several *generators*. For generating ASPIC$^+$ argumentation systems, PyArg uses the "layered" generator from Section 3.9.1. For abstract AFs, PyArg provides a basic random generator. Furthermore, PyArg provides various *importers and exporters* to convert argumentation settings to various formats.

5.3.2. Web interface

In order to demonstrate how PyArg's algorithms can be applied in various settings, we have provided a web interface.[7] In this section, we describe each of its five pages.

On the *generator* pages, users can generate an ASPIC$^+$ argumentation system or abstract AF, parameterised by specific settings; see Figure 5.6.

For *visualisation*, the user can choose between abstract argumentation (Dung, 1995), ASPIC$^+$ (Prakken, 2010), ABA (Bondarenko et al., 1993; Čyras et al., 2018) and IAFs (Cayrol et al., 2007). As a first step, the user either chooses a predefined argumentation setting or gives a specification of a new one. For the abstract AFs, users can provide arguments and the attacks between them; in the ASPIC$^+$ setting users

[3] `http://argumentationcompetition.org/`
[4] `https://github.com/DaphneOdekerken/PyArg`
[5] `https://pypi.org/project/python-argumentation/`
[6] `https://daphneodekerken.github.io/PyArg/`
[7] `https://pyarg.npai.science.uu.nl/`

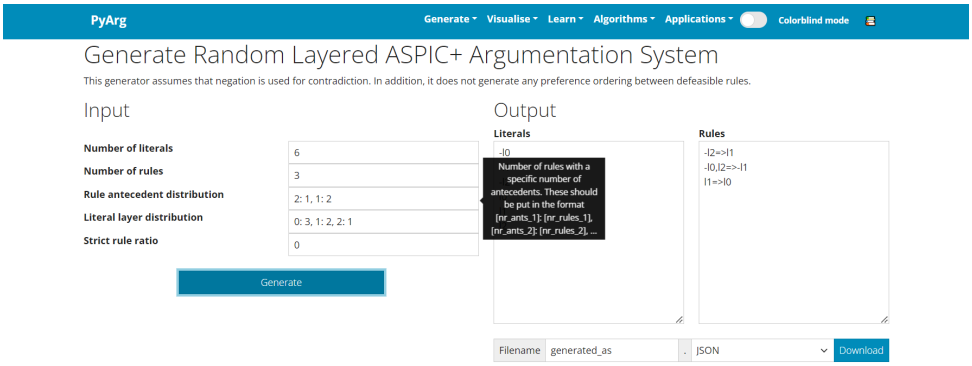


Figure 5.6: The generator page for ASPIC$^+$ argumentation systems.

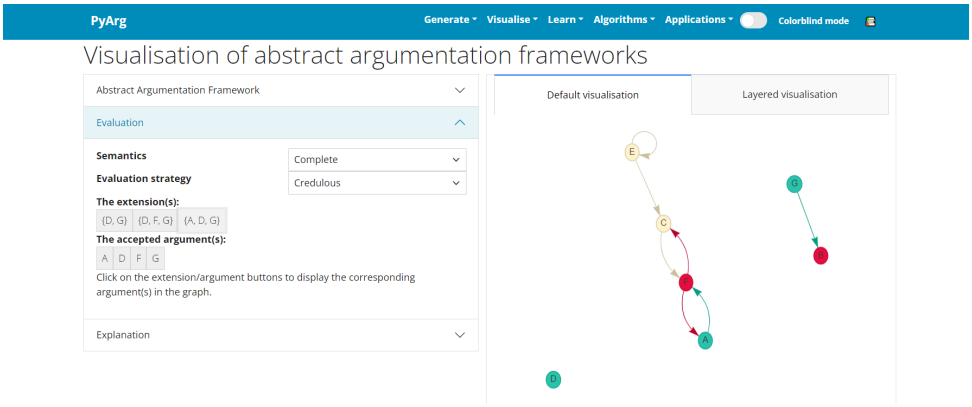


Figure 5.7: The visualisation page for abstract AFs.

can provide axioms, ordinary premises with their preferences, strict rules, defeasible rules with their preferences and a choice in how to derive an ordering from these preferences; and in the ABA setting, users provide atoms, rules, assumptions and contraries. Given this input, the corresponding AF is visualised as a graph. Next, this input is evaluated based on a large variety of extension-based semantics (Baroni et al., 2011). The extensions of a given semantics are presented as buttons; by clicking on a button, the corresponding extension is visualised by coloring the graph – see Figures 5.7 and 5.8. PyArg features both a regular mode (in which arguments in the extension are coloured green, while other arguments are yellow or red) and a colourblind-friendly mode that uses an adapted colour palette. For abstract argumentation and ASPIC$^+$, the user can additionally request explanations for (non-)accepted arguments and formulas (Borg and Bex, 2021a; Borg and Bex, 2021c).

The *learning* environment is intended for anyone interested in learning computational argumentation. In this functionality in the web interface, a learner can choose

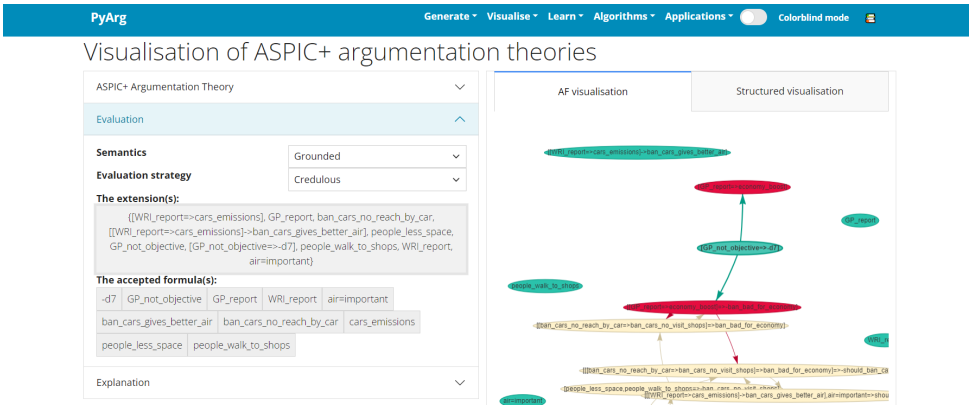


Figure 5.8: The visualisation page for ASPIC$^+$.

between various exercises. As the learner starts an exercise, PyArg generates a random abstract AF using its generators (see Figure 5.9). The learner then gives the extensions, and PyArg uses its semantics algorithms for validating the learner's solutions.

The *algorithms* pages showcase research on realisability, see Figure 5.10. The user can input sets of extensions in the text field. PyArg then computes and displays properties of these extensions (Dunne et al., 2015). Depending on these properties and the semantics, it may be possible to generate a canonical AF with exactly these semantics. If that is the case, the corresponding semantics button becomes active and the user can generate the canonical AF.

The chat interface (Figure 5.11) is an *application* for the algorithms for stability and relevance (Chapter 3). First, the user chooses an ASPIC$^+$ argumentation system, a set of queryables, a topic formula for the chat and an initial knowledge base. PyArg then uses the stability algorithm to find out if it makes sense to ask for more information – if so, it uses the relevance algorithm for identifying relevant questions.

5.4. Conclusion

Efficient algorithms for justification, stability and relevance have promising applications in the law enforcement domain. From a user perspective, these applications can take many forms, including dialogue systems (Section 5.2.1) and dashboards (Section 5.2.2). Although the information extraction and user interface functionalities of these systems are quite different, the argumentation engine is in both cases the same and uses the algorithms proposed in this thesis. Finally, in order to make these algorithms applicable for other use cases within and outside law enforcement, we have made them available in PyArg, an open-source Python package and web interface.

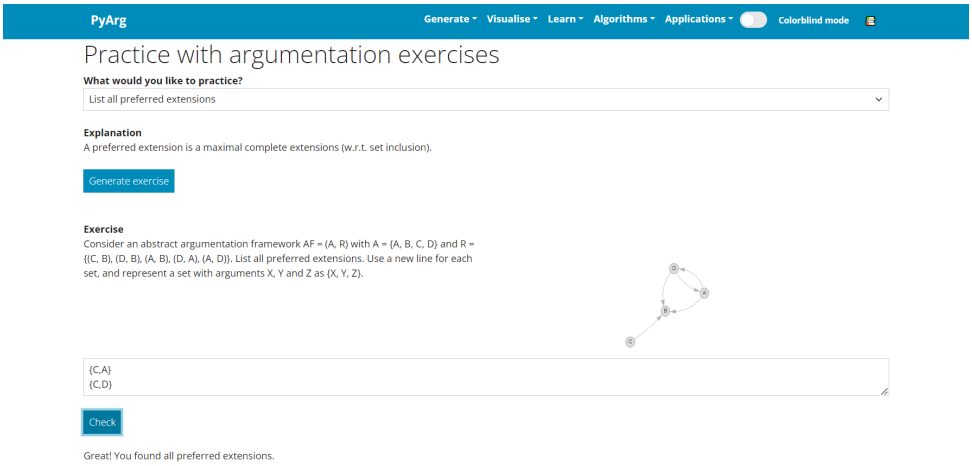


Figure 5.9: One of the learning pages, where a user can practice identifying preferred extensions.

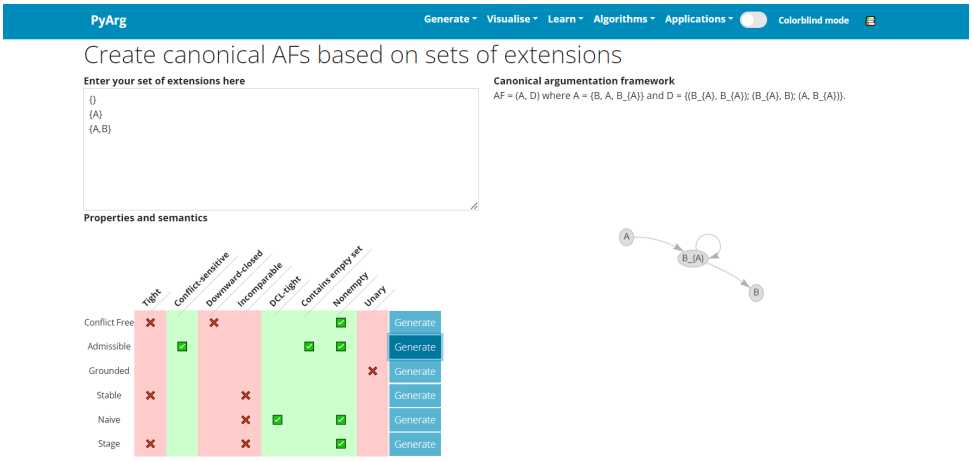


Figure 5.10: One of the algorithms pages showcasing research on realisability.

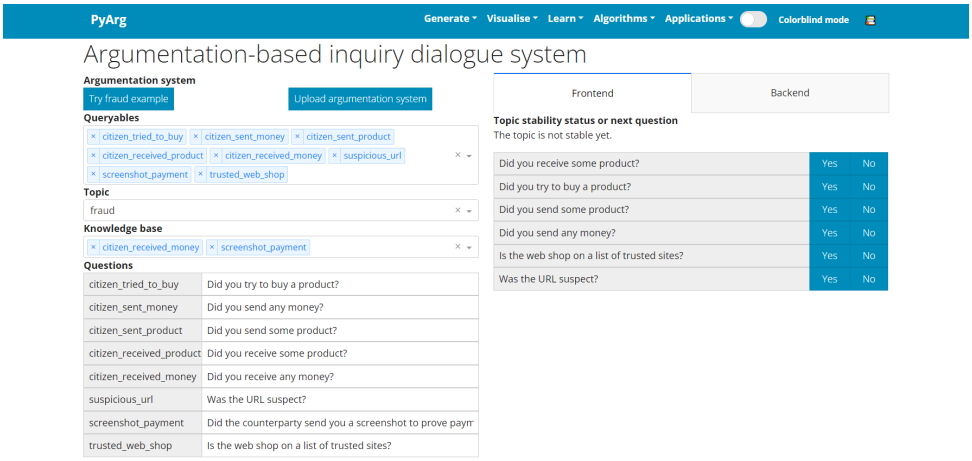


Figure 5.11: The chat interface on the applications page.

6

Conclusion

In this thesis, we have studied the notions of stability and relevance in various formalisms in order to enable transparent and cautious reasoning with incomplete information in law enforcement. After introducing the four research questions in Chapter 1, we studied complexity and algorithms for reasoning in abstract argumentation in Chapter 2; for ASPIC$^+$ in Chapter 3 and for precedent-based reasoning in Chapter 4. Subsequently, Chapter 5 presented two applications in law enforcement, as well as an open-source Python implementation of the argumentation-based algorithms. At this point, the four research questions of this thesis can be answered. We provide answers to the research questions in Section 6.1. In Section 6.2, we then use the results presented in this thesis to construct an argument for the claim that argumentation-based systems are useful for decision support in law enforcement. Finally, we illustrate in Section 6.3 how future work can make this claim stable, taking into account potential challenges arising from future use cases.

6.1. Answers to research questions

The first research question concerned the modelling of incomplete information in the three studied formalisms. This is required in practical settings such as fraud intake and web shop classification, where not all information that is needed to make a decision may be known in advance.

Research question 1. How can argumentation- and precedent-based formalisms be extended to enable reasoning with incomplete information?

A How can incomplete information be represented?

B In which situations does new information not update the justification status?

C Which new information can still change the justification status?

For abstract approaches to argumentation, an approach for handling qualitative uncertainty about the presence of information was already proposed in earlier work on Incomplete Argumentation Frameworks (IAFs). IAFs are an extension to abstract argumentation frameworks that contain not only certain arguments and attacks, but also uncertain ones. IAFs can be completed by selecting a subset of these uncertain arguments and attacks and considering them to be certain, just like the set of arguments and attacks that was already defined to be certain. This work is described in Chapter 2. As a new contribution, we redefined the notion of stability (Mailly and Rossit, 2020) and introduced the notion of relevance for IAFs. Informally, a (certain) argument is stable if it has the same justification status in all completions. For arguments that are not yet stable, the notion of relevance is useful to identify updates that lead towards a stable situation. After performing (a sequence of) relevant updates, the argument will eventually be stable. These relevant updates thus help to identify which information should still be collected in order to give a final advice.

For structured approaches to argumentation, in particular ASPIC$^+$, a similar approach was proposed in Chapter 3. In ASPIC$^+$, arguments are constructed based on a knowledge base and a set of rules. Whereas in earlier work the knowledge base is considered to be complete, in Chapter 3 the ASPIC$^+$ argumentation framework is extended to an incomplete variant, to which a set of queryables is added. These queryables are statements for which it is not known whether they are part of the knowledge base. Future argumentation theories, the natural counterpart of completions, are any argumentation theory that arises from choosing a consistent subset of queryables and considering them to be part of the knowledge base. Thanks to this consistency requirement, future argumentation theories are more restrictive than the completions of IAFs. Stability and relevance are defined in a similar way as for IAFs. A claim is stable if its justification status is the same in all future argumentation theories, while (for unstable claims) those queryables that lead to a stable situation are relevant. Even though these definitions are similar, Section 3.4 showed that there is a discrepancy between stability and relevance in an abstract setting compared to stability and relevance in a structured setting. From this result we conclude that the structured approach is preferred for modelling rule-based reasoning with incomplete information.

Finally, the result model of precedential constraint was not yet equipped to handle incomplete information about dimension values. In order to enable this, in Chapter 4 we proposed to assign a set of possible values to each dimension, rather than a single value. In this proposal, both new (focus) cases and precedent cases can be incomplete, in the sense that dimensions may have multiple possible values. These incomplete cases can be completed by selecting one value from the possible values for each dimension. If an incomplete focus case has the same justification status for all its completions, then we defined it as stable. For incomplete focus cases that are not yet stable, one of the actions that can be performed by a human analyst is further investigation into the dimension values of the case. By removing possible dimension values, the analyst will eventually end up in a stable situation. In order

to decide which dimension values lead towards such a stable situation, we proposed the notion of relevance.

For each of the studied formalisms, the extension for modelling incomplete information basically represents a space of multiple possible situations. As a consequence, the number of completions (or, in the terminology of ASPIC$^+$, future argumentation theories) grows exponentially with the amount of uncertain information (arguments, knowledge or dimension values). This raised concerns related to the computation of stability and relevance, as the investigation of an exponential number of completions may require an algorithm that takes exponential time. In order to investigate whether these concerns are valid, we answered the following research question.

Research question 2. In which complexity classes are the problems of justification, stability and relevance in the context of argumentation- and precedent-based formalisms situated?

This question was answered by complexity analyses for abstract approaches to argumentation in Chapter 2, for ASPIC$^+$ in Chapter 3 and for the result model of precedential constraint in Chapter 4.

For abstract approaches to argumentation, the complexity of stability and relevance was studied for various semantics. Even for grounded semantics, the problem of identifying whether a given argument is stable is CoNP-complete. Regarding relevance, the problem of deciding whether adding or removing a given uncertain argument or attack is relevant for some topic argument is NP-complete for grounded semantics. Apart from some trivial problems, the stability and relevance problems for other semantics are at least as hard.

For structured approaches to argumentation, the complexity of stability and relevance was studied for grounded semantics, as this was the most suitable semantics for the practical applications at the Dutch police, which require cautious reasoning. The first step in this complexity analysis was to study the complexity of the justification problem, which was not studied before for ASPIC$^+$. Even though an argumentation theory may give rise to an exponential number of arguments, we were able to provide a polynomial algorithm for establishing the justification status under grounded semantics in Section 3.5. This algorithm does not rely on the enumeration of arguments, but instead iteratively applies a newly introduced defense operator on sets of defeasible rules until a fixed point is reached. Given that justification can be computed in polynomial time, the upper bounds for stability (in CoNP) and relevance (in Σ_2^P) directly follow. By providing reductions from UNSAT and quantified Boolean formulas, we have shown that stability is CoNP-complete and relevance is Σ_2^P-complete. Although the stability problem is in the same complexity class for abstract and structured approaches to argumentation (CoNP-complete), this does not hold for the relevance problem, which is harder for structured approaches (Σ_2^P-complete compared to NP-complete). This is caused by the requirement in structured approaches that future argumentation theories have consistent knowledge bases.

Finally, for the incomplete version of the model of precedential constraint we performed a complexity analysis in Chapter 4. The justification status of a focus case can be computed in polynomial time, even if the focus case and case base are both incomplete. In contrast to the argumentation-based approaches, there is not necessarily a complexity jump from justification to stability and relevance. We have shown that the complexity classes of stability and relevance are dependent on the ordering of the dimension values. If this ordering is total, stability and relevance can be computed in polynomial time. However, if we do not assume a total ordering on the dimension values, then stability is CoNP-complete while relevance is NP-complete. An additional problem that is required for decision support systems based on precedent-based reasoning is the possibility problem, which verifies if there is some completion of the incomplete focus case that has a specific justification status. This problem can be computed in polynomial time for both the PRO and the CONTRA justification status. Consequently, it can be checked in polynomial time that an update on the incomplete case base does not make the incomplete case base inconsistent.

Given that, in general, the problems of stability and relevance are situated in high complexity classes, the question arises as to which extent the stability status and relevant updates can still be computed within reasonable time.

Research question 3. How can the problems of justification, stability and relevance in the context of argumentation- and precedent-based formalisms be solved efficiently?

This question was answered for the three different representations in Chapters 2, 3 and 4.

For abstract approaches to argumentation under grounded and complete semantics, we proposed algorithms based on answer set programming (ASP) in Chapter 2. We have shown that the algorithms for relevance are sufficiently fast for instances with hundreds of (certain or uncertain) arguments under grounded semantics and tens of (certain or uncertain) arguments under complete semantics. In line with the lower complexity of the stability problem, the algorithms for stability are faster.

For structured approaches to argumentation, we proposed various algorithms for stability in Chapter 3. In particular, both a polynomial-time heuristic algorithm and (two variations of) a worst-case exponential exact ASP-based algorithm were discussed. Even though the former is a heuristic algorithm and therefore not exact in general, it was proven to be exact under specific conditions on the input. Both algorithms are sufficiently fast to be used in practice. Surprisingly, the ASP-based algorithm designed for instances without rule preferences was even faster than the polynomial heuristic algorithm. As a solution to the relevance problem, we proposed exact ASP-based algorithms.

For the result model of precedential constraint, we proposed exact algorithms for the problems of stability, relevance and possibility in Chapter 4. These algorithms

run in polynomial time under conditions of the ordering of the dimension values. In particular, if the dimension values are in a total ordering, all algorithms run in polynomial time. For other orderings, the algorithms can be exponential. This is in line with the complexity results.

The extension of formalisms (answers to Research question 1), complexity results (Research question 2) and algorithms (Research question 3) form the theoretical contributions of this thesis. The answers to these research questions, provided in Chapters 2, 3 and 4, enable the representation of uncertain information in both argument- and precedent-based reasoning and provide sufficiently fast algorithms for the (generally complex) problems of stability and relevance. The only remaining research question asks how these algorithms can be applied in practical decision support systems at the Dutch police.

Research question 4. How can the algorithms for stability and relevance be applied in practical decision support systems at the police?

As an answer to this research question, Chapter 5 reports on our experiences with two use cases. First, we discussed a system for fraud intake, which has been used by hundreds of thousands of users since its launch in September 2019. Second, we described a system for the classification of mala fide web shops. These use cases show that the algorithms for stability and relevance have been used in combination with other techniques for natural language processing and web scraping. In addition, the algorithms have been implemented in two open source packages (PyArg, introduced in Section 5.3 and LCBR, referred to in Chapter 4), which are freely available to the research community.

6.2. An argument for argumentation

Having answered the main research questions, in this section, we use our results to argue that argumentation-based systems are useful for decision support in law enforcement in general. We will do so by constructing an argument by ASPIC$^+$ definitions. This argument is illustrated in Figure 6.1. The literals in the language are abbreviated and have the following meanings:

- 1: Argumentation-based systems fulfill all requirements for AI in law enforcement.
- 11: There are three requirements for AI in law enforcement.
- 111: Three requirements for AI in law enforcement were identified in Chapter 1.
- 12: Argumentation-based systems fulfill the transparency requirement.
- 121: Argumentation-based systems are transparent.
- 1211: Argumentation-based systems can be explained by the defense relation.
- 1212: Argumentation-based systems can be explained by the structure of arguments.

- 122: AI in law enforcement is required to be transparent.
- 1221: Transparency increases trust.
- 1222: Transparent AI is easier to correct.
- 13: Argumentation-based systems fulfill the cautiousness requirement.
- 131: Argumentation-based systems are cautious.
- 1311: Reasoning under grounded semantics is sceptical.
- 1312: Reasoning with stability is even more sceptical.
- 132: AI in law enforcement is required to be cautious (Chapter 1).
- 1321: Decisions in law enforcement have serious implications (Chapter 5).
- 14: Argumentation-based systems fulfill the requirement of handling incomplete information.
- 141: AI in law enforcement is required to handle incomplete information.
- 1411: In practice, some information is missing (Chapter 5).
- 142: Argumentation-based systems are able to handle incomplete information.
- 1421: Argumentation-based systems fulfill the representation requirement.
- 14211: Handling incomplete information requires the representation of incomplete information (Chapter 1).
- 14212: Argumentation-based systems are able to represent incomplete information (Chapter 3).
- 1422: Argumentation-based systems fulfill the termination requirement.
- 14221: Handling incomplete information requires identification of termination conditions (Chapter 1).
- 14222: Stability algorithms in argumentation-based systems identify termination (Chapter 3).
- 1423: Argumentation-based systems fulfill the relevance requirement.
- 14321: Handling incomplete information requires the identification of relevant updates (Chapter 1).
- 14322: Relevance algorithms in argumentation-based systems identify relevant updates (Chapter 3).
- 1424: There are three requirements for handling incomplete information.
- 14241: Chapter 1 distinguishes three requirements for handling incomplete information.
- 2: Domain knowledge can easily be modelled in argumentation theories and queryables.
- 21: For the fraud intake system, domain knowledge was modelled in an argumentation theory and queryables (Chapter 5).

- 22: For the fraud web shop classification system, domain knowledge was modelled in an argumentation theory and queryables (Chapter 5).
- 3: There are algorithms that are sufficiently fast.
- 31: There are algorithms that are fast for instances that are not too large (Chapter 3).
- 32: Real-life instances are not very large.
- 321: Instances identified in Chapter 5 are not very large.

6.3. Future work: remaining stable

The argument presented in Section 6.2 is based on a "complete" argumentation theory, in which our topic claim is stable-defended. However, there is still a lot of information that is unknown. In particular, new use cases may reveal new challenges. In this section, we identify three uncertain arguments attacking our argument for argumentation. For each of these counterarguments, we describe how future work can help us to defend against them or introduce new arguments for argumentation.

6.3.1. Modelling domain knowledge

In the first use cases, described in Chapter 5, we collaborated with experts to translate their domain knowledge into argumentation systems. However, in future applications, it would be convenient if the domain experts are able to expand the argumentation system themselves. It is likely that they will find that challenging. If this is the case, then there is an argument attacking the argument for literal 2 (domain knowledge can easily be modelled in argumentation theories and queryables). To defend against this argument, future work is required. A possible counterargument would be that maintaining domain knowledge is doable for domain experts without formal background if they can use a suitable tool. Such a tool could for example assist in navigating through the language of the argumentation system or visualise the effects of an update in the rule set.

6.3.2. Large argumentation theories

Another challenge that may arise if users are able to extend the argumentation system in future applications is that these argumentation systems become very large. If that happens, there is an argument attacking the argument for literal 32 (real-life instances are not very large). Rather than attacking this argument, we propose to investigate whether an alternative argument for literal 3 (there are algorithms that are sufficiently fast) can be constructed. This argument would not be based on literal 31 (there are algorithms that are fast for instances *that are not too large*) but on another claim, like "there are algorithms that are fast for large instances". In order to support such a claim, faster algorithms should be developed, in particular for the computationally complex problem of identifying relevant updates.

Figure 6.1: Illustration of the argument for argumentation.

6.3.3. More than relevance

A third uncertain argument attacks our argument for argumentation on literal 1424: there are three requirements for handling incomplete information. In particular, future applications may show that listing all relevant updates is insufficient, as there may be too many relevant updates. It may be required to automatically obtain an ordering on the relevant updates. We therefore propose to investigate in future work how such an ordering can be derived. It could, for example, be based on data of interactions with the system or on a probability distribution on queryables.

References

Aleven, Vincent (1997). "Teaching case-based argumentation through a model and examples". Ph.D. thesis. University of Pittsburgh.

Alfano, Gianvincenzo, Andrea Cohen, Sebastian Gottifredi, Sergio Greco, Francesco Parisi, and Guillermo Simari (2020). "Dynamics in abstract argumentation frameworks with recursive attack and support relations". In: *Proceedings of the 24th European Conference on Artificial Intelligence*. Ed. by Giuseppe De Giacomo, Alejandro Catalá, Bistra Dilkina, Michela Milano, Senén Barro, Alberto Bugarín, and Jérôme Lang, pp. 577–584. DOI: 10.3233/FAIA200141.

Alfano, Gianvincenzo, Sergio Greco, Francesco Parisi, Gerardo Simari, and Guillermo Simari (2021a). "Incremental computation for structured argumentation over dynamic DeLP knowledge bases". In: *Artificial Intelligence* 300, p. 103553. DOI: 10.1016/j.artint.2021.103553.

— (2021b). "On the incremental computation of semantics in dynamic argumentation". In: *Journal of Applied Logics* 8.6, pp. 1749–1792. DOI: 10.1109/MIS.2021.3077292.

Alfano, Gianvincenzo, Sergio Greco, Francesco Parisi, and Irina Trubitsyna (2022). "Incomplete argumentation frameworks: Properties and complexity". In: *Proceedings of the 36th AAAI Conference on Artificial Intelligence*. Ed. by Michael Wooldridge, Jennifer Dy, and Sriraam Natarajan, pp. 5451–5460. DOI: 10.1609/aaai.v36i5.20483.

Araszkiewicz, Michał, Ilaria Angela Amantea, Saurabh Chakravarty, Robert van Doesburg, Maria Dymitruk, Marie Garin, Leilani Gilpin, Daphne Odekerken, and Seyedeh Sajedeh Salehi (2020). "ICAIL doctoral consortium, Montreal 2019". In: *Artificial Intelligence and Law* 28, pp. 267–280. DOI: 10.1007/s10506-020-09267-z.

Ashley, Kevin (1991). "Reasoning with cases and hypotheticals in HYPO". In: *International Journal of Man-Machine studies* 34.6, pp. 753–796. DOI: 10.1016/0020-7373(91)90011-U.

Ashley, Kevin and Stefanie Brüninghaus (2009). "Automatically classifying case texts and predicting outcomes". In: *Artificial Intelligence and Law* 17.2, pp. 125–165. DOI: 10.1007/s10506-009-9077-9.

Ashley, Kevin and Edwina Rissland (1988). "A case-based approach to modeling legal expertise". In: *IEEE Intelligent Systems* 3.3, pp. 70–77. DOI: 10.1109/64.21892.

Atkinson, Katie, Pietro Baroni, Massimiliano Giacomin, Anthony Hunter, Henry Prakken, Chris Reed, Guillermo Simari, Matthias Thimm, and Serena Villata (2017). "Towards artificial argumentation". In: *AI magazine* 38.3, pp. 25–36. DOI: 10.1609/aimag.v38i3.2704.

Atkinson, Katie, Trevor Bench-Capon, and Danushka Bollegala (2020). "Explanation in AI and law: Past, present and future". In: *Artificial Intelligence* 289, p. 103387. DOI: 10.1016/j.artint.2020.103387.

Baroni, Pietro, Martin Caminada, and Massimiliano Giacomin (2011). "An introduction to argumentation semantics". In: *The Knowledge Engineering Review* 26.4, pp. 365–410. DOI: 10.1017/S0269888911000166.

Baroni, Pietro and Massimiliano Giacomin (2007). "On principle-based evaluation of extension-based argumentation semantics". In: *Artificial Intelligence* 171.10-15, pp. 675–700. DOI: 10.1016/j.artint.2007.04.004.

Baumann, Ringo and Gerhard Brewka (2010). "Expanding argumentation frameworks: Enforcing and monotonicity Results". In: *Computational Models of Argument. Proceedings of COMMA 2010*. Ed. by Pietro Baroni, Federico Cerutti, Massimiliano Giacomin, and Guillermo Simari. IOS Press, pp. 75–86. DOI: 10.3233/978-1-60750-619-5-75.

Baumeister, Dorothea, Matti Järvisalo, Daniel Neugebauer, Andreas Niskanen, and Jörg Rothe (2021). "Acceptance in incomplete argumentation frameworks". In: *Artificial Intelligence* 295, p. 103470. DOI: 10.1016/j.artint.2021.103470.

Baumeister, Dorothea, Daniel Neugebauer, Jörg Rothe, and Hilmar Schadrack (2018). "Verification in incomplete argumentation frameworks". In: *Artificial Intelligence* 264, pp. 1–26. DOI: 10.1016/j.artint.2018.08.001.

Bench-Capon, Trevor (2017). "HYPO's legacy: Introduction to the virtual special issue". In: *Artificial Intelligence and Law* 25.2, pp. 205–250. DOI: 10.1007/s10506-017-9201-1.

Bench-Capon, Trevor and Katie Atkinson (2022). "Argument schemes for factor ascription". In: *Computational Models of Argument. Proceedings of COMMA 2022*. Ed. by Francesca Toni, Sylwia Polberg, Richard Booth, Martin Caminada, and Hiroyuki Kido. IOS Press, pp. 68 –79. DOI: 10.3233/FAIA220142.

Berthold, Matti, Matthias Knorr, and Daphne Odekerken (2023a). "ForgettingWeb". In: *Technical Communications of the 39th International Conference on Logic Programming*, pp. 321–323. DOI: 10.48550/arXiv.2308.14898.

Berthold, Matti, Anna Rapberger, and Markus Ulbricht (2023b). "Forgetting aspects in assumption-based argumentation". In: *Proceedings of the 20th International Conference on Principles of Knowledge Representation and Reasoning*. Ed. by Pierre Marquis, Tran Cao Son, and Gabriele Kern-Isberner, pp. 86–96. DOI: 10.24963/kr.2023/9.

— (2023c). "On the expressive power of assumption-based argumentation". In: *Proceedings of the 18th European Conference on Logics in Artificial Intelligence*. Ed. by Sarah A Gaggl, Maria V Martinez, and Magdalena Ortiz, pp. 145–160. DOI: 10.1007/978-3-031-43619-2_11.

Besnard, Philippe, Alejandro Garcia, Anthony Hunter, Sanjay Modgil, Henry Prakken, Guillermo Simari, and Francesca Toni (2014). "Introduction to structured argumentation". In: *Argument & Computation* 5.1, pp. 1–4. DOI: 10.1080/19462166.2013.869764.

Besnard, Philippe and Anthony Hunter (2008). *Elements of argumentation*. MIT press Cambridge. ISBN: 9780262026437.

Bex, Floris, Joeri Peters, and Bas Testerink (2016). "AI for online criminal complaints: From natural dialogues to structured scenarios". In: *Proceedings of the ECAI 2016 workshop on Artificial Intelligence for Justice*, pp. 22–29.

Bex, Floris, Henry Prakken, Chris Reed, and Douglas Walton (2003). "Towards a formal account of reasoning about evidence: Argumentation schemes and generalisations". In: *Artificial Intelligence and Law* 11, pp. 125–165. DOI: 10.1023/B:ARTI.0000046007.11806.9a.

Bex, Floris and Bart Verheij (2013). "Legal stories and the process of proof". In: *Artificial Intelligence and Law* 21, pp. 253–278. DOI: 10.1007/s10506-012-9137-4.

Bondarenko, Andrei, Phan Minh Dung, Robert A. Kowalski, and Francesca Toni (1997). "An abstract, argumentation-theoretic approach to default reasoning". In: *Artificial intelligence* 93.1-2, pp. 63–101. DOI: 10.1016/S0004-3702(97)00015-5.

Bondarenko, Andrei, Francesca Toni, and Robert A. Kowalski (1993). "An assumption-based framework for non-monotonic reasoning". In: *Proceedings of the 2nd International Workshop on Logic Programming and Non-Monotonic Reasoning*, pp. 171–189.

Borg, AnneMarie and Floris Bex (2021a). "A basic framework for explanations in argumentation". In: *IEEE Intelligent Systems* 36.2, pp. 25–35. DOI: 10.1109/MIS.2021.3053102.

— (2021b). "Enforcing sets of formulas in structured argumentation". In: *Proceedings of the 18th International Conference on Principles of Knowledge Representation and Reasoning*. Ed. by M. Bienvenu, G. Lakemeyer, and E. Erdem, pp. 130–140. DOI: 10.24963/kr.2021/13.

— (2021c). "Necessary and sufficient explanations for argumentation-based conclusions". In: *Proceedings of the 16th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty*. Ed. by Jiřina Vejnarová and Nic Wilson. Vol. 12897. Lecture Notes in Artificial Intelligence. Springer, pp. 45–58. DOI: 10.1007/978-3-030-86772-0_4.

— (2022). "Contrastive explanations for argumentation-based conclusions". In: *Proceedings of the 21st International Conference on Autonomous Agents and MultiAgent Systems*. Ed. by Piotr Faliszewski, Viviana Mascardi, Catherine Pelachaud, and Matthew E. Taylor. International Foundation for Autonomous Agents and Multiagent Systems, pp. 1551–1553.

— (2024). "Minimality, necessity and sufficiency for argumentation and explanation". In: *International Journal of Approximate Reasoning* 168, p. 109143. DOI: 10.1016/j.ijar.2024.109143.

Borg, AnneMarie and Daphne Odekerken (2022). "PyArg for solving and explaining argumentation in Python: Demonstration". In: *Computational Models of Argument. Proceedings of COMMA 2022*. Ed. by Francesca Toni, Sylwia Polberg, Richard Booth, Martin Caminada, and Hiroyuki Kido. IOS Press, pp. 349–350. DOI: 10.3233/FAIA220167.

Caminada, Martin (2006). "On the issue of reinstatement in argumentation". In: *Proceedings of the 10th European Conference on Logics in Artificial Intelligence*.

Ed. by Michael Fisher, Wiebe Hoeka, Boris Konev, and Alexei Lisitsa. Springer, pp. 111–123. DOI: 10.1007/11853886_11.

Caminada, Martin and Leila Amgoud (2007). "On the evaluation of argumentation formalisms". In: *Artificial Intelligence* 171.5-6, pp. 286–310. DOI: 10.1016/j.artint.2007.02.003.

Canavotto, Ilaria (2023). "Reasoning with inconsistent precedents". In: *Artificial Intelligence and Law*, pp. 1–30. DOI: 10.1007/s10506-023-09382-7.

Cayrol, Claudette, Caroline Devred, and Marie-Christine Lagasquie-Schiex (2007). "Handling ignorance in argumentation: Semantics of partial argumentation frameworks". In: *Proceedings of the 9th European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty*. Ed. by Khaled Mellouli. Springer, pp. 259–270. DOI: 10.1007/978-3-540-75256-1_25.

Cayrol, Claudette, Florence Dupin de Saint-Cyr, and Marie-Christine Lagasquie-Schiex (2010). "Change in abstract argumentation frameworks: Adding an argument". In: *Journal of Artificial Intelligence Research* 38, pp. 49–84. DOI: 10.1613/jair.2965.

Cerutti, Federico, Sarah A Gaggl, Matthias Thimm, and Johannes P Wallner (2018). "Foundations of implementations for formal argumentation". In: *Handbook of Formal Argumentation*. Ed. by Pietro Baroni, Dov Gabbay, Massimiliano Giacomin, and Leendert van der Torre. Vol. 1. College Publications, pp. 689–767.

Cerutti, Federico, Matthias Thimm, and Mauro Vallati (2020). "An experimental analysis on the similarity of argumentation semantics". In: *Argument & Computation* 11.3, pp. 1–36. DOI: 10.3233/AAC-200907.

Clarke, Edmund M., Orna Grumberg, Somesh Jha, Yuan Lu, and Helmut Veith (2003). "Counterexample-guided abstraction refinement for symbolic model checking". In: *Journal of the ACM* 50.5, pp. 752–794. DOI: 10.1145/876638.876643.

Clarke, Edmund M., Anubhav Gupta, and Ofer Strichman (2004). "SAT-based counterexample-guided abstraction refinement". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 23.7, pp. 1113–1123. DOI: 10.1109/TCAD.2004.829807.

Coste-Marquis, Sylvie, Caroline Devred, and Pierre Marquis (2005). "Symmetric argumentation frameworks". In: *Proceedings of the 8th European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty*. Ed. by Lluís Godo. Springer, pp. 317–328. DOI: 10.1007/11518655_28.

Craandijk, Dennis and Floris Bex (2020). "Deep learning for abstract argumentation semantics". In: *Proceedings of the 29th International Joint Conference on Artificial Intelligence*. Ed. by Christian Bessiere, pp. 1667–1673. DOI: 10.24963/ijcai.2020/231.

Čyras, Kristijonas, Xiuyi Fan, Claudia Schulz, and Francesca Toni (2018). "Assumption-based argumentation: Disputes, explanations, preferences". In: *Handbook of Formal Argumentation*. Ed. by Pietro Baroni, Dov Gabbay, Massimiliano Giacomin, and Leendert van der Torre. Vol. 1. College Publications. Chap. 7, pp. 365–408.

Čyras, Kristijonas, Antonio Rago, Emanuele Albini, Pietro Baroni, and Francesca Toni (2021). "Argumentative XAI: A survey". In: *Proceedings of the 30th In-*

ternational Joint Conference on Artificial Intelligence. Ed. by Zhi-Hua Zhou, pp. 4392–4399. DOI: 10.24963/ijcai.2021/600.

Dachselt, Raimund, Sarah Gaggl, Markus Krötzsch, Julián Méndez, Dominik Rusovac, and Mei Yang (2022). "NEXAS: A visual tool for navigating and exploring argumentation solution spaces". In: *Computational Models of Argument. Proceedings of COMMA 2022*. Ed. by Francesca Toni, Sylwia Polberg, Richard Booth, Martin Caminada, and Hiroyuki Kido. Vol. 353. IOS Press, pp. 116–127. DOI: 10.3233/FAIA220146.

Dietz, Emmanuelle, Antonis Kakas, and Loizos Michael (2022). "Argumentation: a calculus for human-centric AI". In: *Frontiers in Artificial Intelligence* 5, p. 955579. DOI: doi.org/10.3389/frai.2022.955579.

Dimopoulos, Yannis and Alberto Torres (1996). "Graph theoretical structures in logic programs and default theories". In: *Theoretical Computer Science* 170.1-2, pp. 209–244. DOI: 10.1016/S0304-3975(96)80707-9.

Doutre, Sylvie and Jean-Guy Mailly (2018). "Constraints and changes: A survey of abstract argumentation dynamics". In: *Argument & Computation* 9, pp. 223–248. DOI: 10.3233/AAC-180425.

Dung, Phan Minh (1995). "On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games". In: *Artificial Intelligence* 77, pp. 321–357. DOI: 10.1016/0004-3702(94)00041-X.

Dunne, Paul E and Trevor JM Bench-Capon (2002). "Coherence in finite argument systems". In: *Artificial Intelligence* 141.1-2, pp. 187–203. DOI: 10.1016/S0004-3702(02)00261-8.

Dunne, Paul E., Wolfgang Dvořák, Thomas Linsbichler, and Stefan Woltran (2015). "Characteristics of multiple viewpoints in abstract argumentation". In: *Artificial Intelligence* 228, pp. 153–178. DOI: 0.1016/j.artint.2015.07.006.

Dunne, Paul E and Michael Wooldridge (2009). "Complexity of abstract argumentation". In: *Argumentation in Artificial Intelligence*. Ed. by Guillermo Simari and Iyad Rahwan. Springer, pp. 85–104. DOI: 10.1007/978-0-387-98197-0_5.

Dvořák, Wolfgang (2012). "On the complexity of computing the justification status of an argument". In: *International Workshop on Theory and Applications of Formal Argumentation*. Springer, pp. 32–49. DOI: 10.1007/978-3-642-29184-5_3.

Dvorák, Wolfgang and Paul E Dunne (2018). "Computational problems in formal argumentation and their complexity". In: ed. by Pietro Baroni, Dov Gabbay, Massimiliano Giacomin, and Leendert van der Torre. Vol. 1. College Publications. Chap. 13, pp. 631–687.

Dvořák, Wolfgang, Anna Rapberger, Johannes P Wallner, and Stefan Woltran (2020). "ASPARTIX-V19 - an answer-set programming based system for abstract argumentation". In: *Proceedings of the 11th International Symposium on Foundations of Information and Knowledge Systems*. Springer, pp. 79–89. DOI: 10.1007/978-3-030-39951-1_5.

Egly, Uwe, Sarah Alice Gaggl, and Stefan Woltran (2010). "Answer-set programming encodings for argumentation frameworks". In: *Argument & Computation* 1.2, pp. 147–177. DOI: 10.1080/19462166.2010.486479.

Falappa, Marcelo, Gabriele Kern-Isberner, and Guillermo Simari (2009). "Belief revision and argumentation theory". In: *Argumentation in Artificial Intelligence*. Ed. by Guillermo Simari and Iyad Rahwan. Springer, pp. 341–360. DOI: 10.1007/978-0-387-98197-0_17.

Fazzinga, Bettina, Sergio Flesca, Filippo Furfaro, et al. (2020). "Revisiting the notion of extension over incomplete abstract argumentation frameworks". In: *Proceedings of the 29th International Joint Conference on Artificial Intelligence*. Ed. by Christian Bessiere, pp. 1712–1718. DOI: 10.24963/ijcai.2020/237.

García, Alejandro and Guillermo Simari (2004). "Defeasible logic programming: An argumentative approach". In: *Theory and practice of logic programming* 4.1-2, pp. 95–138. DOI: 10.1017/S1471068403001674.

Gebser, Martin, Amelia Harrison, Roland Kaminski, Vladimir Lifschitz, and Torsten Schaub (2015). "Abstract gringo". In: *Theory and Practice of Logic Programming* 15.4-5, pp. 449–463. DOI: 10.1017/S1471068415000150.

Gebser, Martin, Roland Kaminski, Benjamin Kaufmann, Max Ostrowski, Torsten Schaub, and Philipp Wanko (2016). "Theory solving made easy with Clingo 5". In: *Technical Communications of the 32nd International Conference on Logic Programming*, 2:1–2:15. DOI: 10.4230/OASIcs.ICLP.2016.2.

Gebser, Martin, Roland Kaminski, Benjamin Kaufmann, and Torsten Schaub (2019). "Multi-shot ASP solving with Clingo". In: *Theory and Practice of Logic Programming* 19.1, pp. 27–82. DOI: 10.1017/S1471068418000054.

Gelfond, Michael and Vladimir Lifschitz (1988). "The stable model semantics for logic programming". In: *Proceedings of International Logic Programming Conference and Symposium*. MIT Press, pp. 1070–1080.

Governatori, Guido, Francesco Olivieri, Antonino Rotolo, and Matteo Cristani (2022). "Stable normative explanations". In: *Legal Knowledge and Information Systems. Proceedings of JURIX 2022*. Ed. by Enrico Francesconi, Georg Borges, and Christoph Sorge. IOS Press, pp. 43–52. DOI: 10.3233/FAIA220447.

Hecham, Abdelraouf, Pierre Bisquert, and Madalina Croitoru (2018). "On a flexible representation for defeasible reasoning variants". In: *Proceedings of the 17th International Conference on Autonomous Agents and Multiagent Systems*. Ed. by Elisabeth André, Sven Koenig, Mehdi Dastani, and Gita Sukthankar, pp. 1123–1131.

Horty, John (2004). "The result model of precedent". In: *Legal Theory* 10.1, pp. 19–31. DOI: 10.1017/s1352325204000151.

— (2011). "Rules and reasons in the theory of precedent". In: *Legal Theory* 17, pp. 1–33. DOI: 10.1017/S1352325211000036.

— (2019). "Reasoning with dimensions and magnitudes". In: *Artificial Intelligence and Law* 27, pp. 309–345. DOI: 10.1007/s10506-019-09245-0.

Kakas, Antonis and Pavlos Moraitis (2003). "Argumentation based decision making for autonomous agents". In: *Proceedings of the 2nd International Joint Conference on Autonomous Agents and Multiagent Systems*. Ed. by Jeffrey S Rosenschein, pp. 883–890. DOI: 10.1145/860575.860717.

Kos, William, Marijn Schraagen, Matthieu Brinkhuis, and Floris Bex (2017). "Classification in a skewed online trade fraud complaint corpus". In: *Preproceedings of the 29th Benelux Conference on Artificial Intelligence*, pp. 172–183.

Lehtonen, Tuomo, Daphne Odekerken, Johannes P. Wallner, and Matti Järvisalo (2024). "Complexity results and algorithms for preferential argumentative reasoning in ASPIC+". In: *Proceedings of the 21th International Conference on Principles of Knowledge Representation and Reasoning*. Ed. by Pierre Marquis, Magdalena Ortiz, and Maurice Pagnucco, pp. 520–530. DOI: 10.24963/kr.2024/49.

Lehtonen, Tuomo, Johannes P Wallner, and Matti Järvisalo (2020). "An answer set programming approach to argumentative reasoning in the ASPIC+ framework". In: *Proceedings of the 17th International Conference on Principles of Knowledge Representation and Reasoning*. Ed. by Diego Calvanese, Esra Erdem, and Michael Thielscher, pp. 636–646. DOI: 10.24963/kr.2020/63.

Lehtonen, Tuomo, Johannes P. Wallner, and Matti Järvisalo (2021). "Declarative algorithms and complexity results for assumption-based argumentation". In: *Journal of Artificial Intelligence Research* 71, pp. 265–318. DOI: 10.1613/jair.1.12479.

Lehtonen, Tuomo, Johannes P Wallner, and Matti Järvisalo (2022). "Computing stable conclusions under the weakest-link principle in the ASPIC+ argumentation formalism". In: *Proceedings of the 19th International Conference on Principles of Knowledge Representation and Reasoning*. Ed. by Gabriele Kern-Isberner, Gerhard Lakemeyer, and Thomas Meyer, pp. 215–225. DOI: 10.24963/kr.2022/22.

Mailly, Jean-Guy (2023). "Extension-based semantics for incomplete argumentation frameworks: Properties, complexity and algorithms". In: *Journal of Logic and Computation* 33.2, pp. 406–435. DOI: 10.1093/logcom/exac099.

Mailly, Jean-Guy and Julien Rossit (2020). "Stability in abstract argumentation". In: *Workshop Notes of the 18th International Workshop on Non-Monotonic Reasoning*, pp. 93–99.

Miller, Tim (2019). "Explanation in artificial intelligence: Insights from the social sciences". In: *Artificial intelligence* 267, pp. 1–38. DOI: 10.1016/j.artint.2018.07.007.

Modgil, Sanjay and Henry Prakken (2012). "Resolutions in structured argumentation". In: *Computational Models of Argument. Proceedings of COMMA 2012*. Ed. by Bart Verheij, Stefan Szeider, and Stefan Woltran. IOS Press, pp. 310–321. DOI: 10.3233/978-1-61499-111-3-310.

— (2013). "A general account of argumentation with preferences". In: *Artificial Intelligence* 195, pp. 361–397. DOI: 10.1016/j.artint.2012.10.008.

Niemelä, Ilkka (1999). "Logic programs with stable model semantics as a constraint programming paradigm". In: *Annals of Mathematics and Artificial Intelligence* 25.3-4, pp. 241–273. DOI: 10.1023/A:1018930122475.

Nieuwenhuizen, Esther (2020). "Artificiële intelligentie, is dat wel te vertrouwen? Een experimentele studie naar het effect van uitleg over beslissingen van het intelligente aangiftesysteem van de politie op het vertrouwen van burgers in deze beslissingen. (*Can we trust Artificial Intelligence? An experimental study into the effect of explaining decisions of the police's intelligent intake system on citizen*

trust.)" MA thesis. Utrecht University. URL: http://dspace.library.uu.nl/handle/1874/400688.

Niskanen, Andreas and Matti Järvisalo (2020). "Algorithms for dynamic argumentation frameworks: An incremental SAT-based approach". In: *Proceedings of the 24th European Conference on Artificial Intelligence*. Ed. by Giuseppe De Giacomo, Alejandro Catalá, Bistra Dilkina, Michela Milano, Senén Barro, Alberto Bugarín, and Jérôme Lang. IOS Press, pp. 849–856. DOI: 10.3233/FAIA200175.

Odekerken, Daphne (2022). "Justification, stability and relevance for transparent and efficient human-in-the-loop decision support". In: *Online Handbook of Argumentation for AI* 3, pp. 28–32. DOI: 10.48550/ARXIV.2212.07996.

— (2023). "Argumentative reasoning with incomplete information in law enforcement". In: *Online Handbook of Argumentation for AI* 4. DOI: 10.48550/arXiv.2401.09444.

— (2024). "Finding relevant updates in incomplete argumentation frameworks". In: *Computational Models of Argument. Proceedings of COMMA 2024*. Ed. by Chris Reed, Matthias Thimm, and Tjitze Rienstra. IOS Press, pp. 181–192. DOI: 10.3233/FAIA240320.

Odekerken, Daphne and Floris Bex (2020). "Towards transparent human-in-the-loop classification of fraudulent web shops". In: *Legal Knowledge and Information Systems. Proceedings of JURIX 2020*. Ed. by Serena Villata, Jakub Harašta, and Petr Křemen. IOS Press, pp. 239–242. DOI: 10.3233/FAIA200873.

Odekerken, Daphne, Floris Bex, AnneMarie Borg, and Bas Testerink (2022a). "Approximating stability for applied argument-based inquiry". In: *Intelligent Systems with Applications* 16, p. 200110. DOI: 10.1016/j.iswa.2022.200110.

Odekerken, Daphne, Floris Bex, and Henry Prakken (2023a). "Justification, stability and relevance for case-based reasoning with incomplete focus cases". In: *Proceedings of the 19th International Conference on Artificial Intelligence and Law*. Ed. by Matthias Grabmair, Francisco Andrade, and Paulo Novais, pp. 177–186. DOI: 10.1145/3594536.3595136.

— (2023b). "Precedent-based reasoning with incomplete cases". In: *Legal Knowledge and Information Systems. Proceedings of JURIX 2023*. Ed. by Giovanni Sileno, Jerry Spanakis, and Gijs van Dijck. IOS Press, pp. 33–42. DOI: 10.3233/FAIA230943.

— (2024). "Precedent-based reasoning with incomplete information for human-in-the-loop decision support". Accepted for Artificial Intelligence and Law.

Odekerken, Daphne, AnneMarie Borg, and Matti Berthold (2023c). "Accessible algorithms for applied argumentation". In: *First International Workshop on Argumentation and Applications*, pp. 92–98. URL: https://www.ceur-ws.org/Vol-3472/short3.pdf.

— (2023d). "Demonstrating PyArg 2.0". In: *Seventh Workshop on Advances in Argumentation in Artificial Intelligence*. URL: https://ceur-ws.org/Vol-3546/paper14.pdf.

Odekerken, Daphne, AnneMarie Borg, and Floris Bex (2020). "Estimating stability for efficient argument-based inquiry". In: *Computational Models of Argument. Proceedings of COMMA 2020*. Ed. by Henry Prakken, Stefano Bistarelli,

Francesco Santini, and Carlo Taticchi. IOS Press, pp. 307–318. DOI: 10.3233/FAIA200514.

— (2022b). "Stability and relevance in incomplete argumentation frameworks". In: *Computational Models of Argument. Proceedings of COMMA 2022*. Ed. by Francesca Toni, Sylwia Polberg, Richard Booth, Martin Caminada, and Hiroyuki Kido. IOS Press, pp. 272–283. DOI: 10.3233/FAIA220159.

— (2023e). "Justification, stability and relevance in incomplete argumentation frameworks". In: *Argument & Computation* 15.3, pp. 1–58. DOI: 10.3233/AAC-230002.

Odekerken, Daphne, Tuomo Lehtonen, AnneMarie Borg, Johannes P. Wallner, and Matti Järvisalo (2023f). "Argumentative reasoning in ASPIC+ under incomplete information". In: *Proceedings of the 20th International Conference on Principles of Knowledge Representation and Reasoning*. Ed. by Pierre Marquis, Tran Cao Son, and Gabriele Kern-Isberner, pp. 531–541. DOI: 10.24963/kr.2023/52.

Odekerken, Daphne, Tuomo Lehtonen, Johannes P. Wallner, and Matti Järvisalo (n.d.). "Argumentative reasoning in ASPIC+ under incomplete information". Submitted to Journal of Artificial Intelligence Research.

Papadimitriou, Christos H. (1994). *Computational Complexity*. Addison-Wesley. ISBN: 0470864125.

Peters, Joeri, Floris Bex, and Henry Prakken (2022). "Justifications derived from inconsistent case bases using authoritativeness". In: *Proceedings of the 1st International Workshop on Argumentation for eXplainable AI*. Vol. 3209. CEUR WS, pp. 1–13.

Pollock, John L (1987). "Defeasible reasoning". In: *Cognitive Science* 11.4, pp. 481–518. DOI: 10.1207/s15516709cog1104_4.

Prakken, Henry (2010). "An abstract framework for argumentation with structured arguments". In: *Argument & Computation* 1.2, pp. 93–124. DOI: 10.1080/19462160903564592.

— (2020). "An argumentation-based analysis of the Simonshaven case". In: *Topics in Cognitive Science* 12.4, pp. 1068–1091. DOI: 10.1111/tops.12418.

— (2021). "A formal analysis of some factor-and precedent-based accounts of precedential constraint". In: *Artificial Intelligence and Law* 29, pp. 559–585. DOI: 10.1007/s10506-021-09284-6.

— (2022). "Formalising an aspect of argument strength: degrees of attackability". In: *Computational Models of Argument: Proceedings of COMMA 2022*. Ed. by Francesca Toni, Sylwia Polberg, Richard Booth, Martin Caminada, and Hiroyuki Kido. IOS Press, pp. 296–307. DOI: 10.3233/FAIA220161.

— (2023). "Relating abstract and structured accounts of argumentation dynamics: The case of expansions". In: *Proceedings of the 20th International Conference on Principles of Knowledge Representation and Reasoning*. Ed. by Pierre Marquis, Tran Cao Son, and Gabriele Kern-Isberner, pp. 562–571. DOI: 10.24963/kr.2023/55.

Prakken, Henry and Rosa Ratsma (2022). "A top-level model of case-based argumentation for explanation: formalisation and experiments". In: *Argument & Computation* 13.2, pp. 159–194. DOI: 10.3233/AAC-210009.

Prakken, Henry and Giovanni Sartor (1998). "Modelling reasoning with precedents in a formal dialogue game". In: *Artificial Intelligence and Law* 6, pp. 231–287. DOI: 10.1023/A:1008278309945.

Prakken, Henry and Gerard Vreeswijk (2001). "Logics for defeasible argumentation". In: *Handbook of Philosophical Logic*. Ed. by Dov Gabbay and Franz Guenthner. Vol. 4. Springer, pp. 219–318. DOI: 10.1007/978-94-017-0456-4_3.

Prakken, Henry and Michiel De Winter (2018). "Abstraction in argumentation: Necessary but dangerous". In: *Computational Models of Argument. Proceedings of COMMA 2018*. Ed. by Sanjay Modgil, Katarzyna Budzynska, and John Lawrence. IOS Press, pp. 85–96. DOI: 10.3233/978-1-61499-906-5-85.

Rapberger, Anna and Markus Ulbricht (2023). "On dynamics in structured argumentation formalisms". In: *Journal of Artificial Intelligence Research* 77, pp. 563–643. DOI: 10.1613/jair.1.14481.

Rienstra, Tjitze, Matthias Thimm, Kristian Kersting, and Xiaoting Shao (2020). "Independence and D-separation in abstract argumentation". In: *Proceedings of the 17th International Conference on Principles of Knowledge Representation and Reasoning*. Ed. by Diego Calvanese, Esra Erdem, and Michael Thielscher, pp. 713–722. DOI: 10.24963/kr.2020/73.

Rigoni, Adam (2024). "Toward representing interpretation in factor-based models of precedent". In: *Artificial Intelligence and Law*, pp. 1–28. DOI: 10.1007/s10506-023-09384-5.

Rissland, Edwina and Kevin Ashley (1987). "A case-based system for trade secrets law". In: *Proceedings of the 1st International Conference on Artificial Intelligence and Law*. ACM, pp. 60–66. DOI: 10.1145/41735.41743.

Schraagen, Marijn and Floris Bex (2019). "Extraction of semantic relations in noisy user-generated law enforcement data". In: *Proceedings of the 13th International Conference on Semantic Computing*. IEEE, pp. 79–86. DOI: 10.1109/ICOSC.2019.8665497.

Schraagen, Marijn, Floris Bex, Daphne Odekerken, and Bas Testerink (2018). "Argumentation-driven information extraction for online crime reports". In: *Proceedings of the International Workshop on Legal Data Analysis and Mining*, pp. 20–25. DOI: https://ceur-ws.org/Vol-2482/paper32.pdf.

Schraagen, Marijn, Matthieu Brinkhuis, and Floris Bex (2017). "Evaluation of named entity recognition in Dutch online criminal complaints". In: *Computational Linguistics in The Netherlands Journal* 7, pp. 3–16. URL: https://clinjournal.org/clinj/article/view/65.

Skiba, Kenneth, Daniel Neugebauer, and Jörg Rothe (2020). "Complexity of non-empty existence problems in incomplete argumentation frameworks". In: *IEEE Intelligent Systems* 36.2, pp. 13–24. DOI: 10.1109/MIS.2020.3046782.

Snaith, Mark and Chris Reed (2012). "TOAST: Online ASPIC+ implementation". In: *Computational Models of Argument. Proceedings of COMMA 2012*. Ed. by Bart Verheij, Stefan Szeider, and Stefan Woltran. IOS Press, pp. 509–510. DOI: 10.3233/978-1-61499-111-3-509.

— (2016). "Argument revision". In: *Journal of Logic and Computation* 27.7, pp. 2089–2134. DOI: 10.1093/logcom/exw028.

Soares, Carlos, Stephan Grimmelikhuijsen, and Albert Meijer (2023). "Screen-level bureaucrats in the age of algorithms. An ethnographic study of algorithmically supported public service workers in the Netherlands Police". In: *Information Polity* Preprint, pp. 1–16.

Spanoudakis, Nikolaos I, Georgios Gligoris, Adamos Koumi, and Antonis C Kakas (2023). "Explainable argumentation as a service". In: *Journal of Web Semantics*, p. 100772. DOI: 10.1016/j.websem.2023.100772.

Stockmeyer, Larry J (1976). "The polynomial-time hierarchy". In: *Theoretical Computer Science* 3.1, pp. 1–22. DOI: 10.1016/0304-3975(76)90061-X.

Strass, Hannes, Adam Wyner, and Martin Diller (2019). "*EMIL*: Extracting meaning from inconsistent language: Towards argumentation using a controlled natural language interface". In: *International Journal of Approximate Reasoning* 112, pp. 55–84. DOI: 10.1016/j.ijar.2019.04.010.

Testerink, Bas, Daphne Odekerken, and Floris Bex (2019a). "A method for efficient argument-based inquiry". In: *Proceedings of the 13th International Conference on Flexible Query Answering Systems*. Ed. by Alfredo Cuzzocrea, Sergio Greco, Henrik Legind Larsen, Domenico Saccà, Troels Andreasen, and Henning Christiansen. Springer, pp. 114–125. DOI: 10.1007/978-3-030-27629-4_13.

— (2019b). "AI-assisted message processing for the Netherlands National Police". In: *Proceedings of the ICAIL 2019 Workshop on AI and the Administrative State*. Ed. by Karl Branting, pp. 10–13. URL: https://ceur-ws.org/Vol-2471/paper2.pdf.

Thimm, Matthias (2018). "The formal argumentation libraries of Tweety". In: *Proceedings of the 4th International Workshop on Theory and Practice of Formal Argumentation*. Springer, pp. 137–142. DOI: 10.1007/978-3-319-75553-3_9.

Ulbricht, Markus and Ringo Baumann (2019). "If nothing is accepted–Repairing argumentation frameworks". In: *Journal of Artificial Intelligence Research* 66, pp. 1099–1145. DOI: 10.1613/jair.1.11791.

Vassiliades, Alexandros, Nick Bassiliades, and Theodore Patkos (2021). "Argumentation and explainable artificial intelligence: A survey". In: *The Knowledge Engineering Review* 36, e5. DOI: 10.1017/S0269888921000011.

Wabeke, Thymen, Giovane Moura, Nanneke Franken, and Cristian Hesselman (2020). "Counterfighting counterfeit: Detecting and taking down fraudulent webshops at a ccTLD". In: *Proceedings of the 21st International Conference on Passive and Active Network Measurement*. Ed. by Anna Sperotto, Alberto Dainotti, and Burkhard Stiller. Springer, pp. 158–174. DOI: 10.1007/978-3-030-44081-7_10.

Walton, Douglas, Christopher Reed, and Fabrizio Macagno (2008). *Argumentation schemes*. Cambridge University Press. DOI: 10.1017/CBO9780511802034.

Wrathall, Celia (1976). "Complete sets and the polynomial-time hierarchy". In: *Theoretical Computer Science* 3.1, pp. 23–33. DOI: 10.1016/0304-3975(76)90062-1.

Wu, Yining and Martin Caminada (2010). "A labelling-based justification status of arguments". In: *Studies in Logic* 3.4, pp. 12–29.

Xia, Yilin, Daphne Odekerken, Shawn Bowers, and Bertram Ludäscher (2024). "Layered visualization of argumentation frameworks". In: *Computational Models of Argument. Proceedings of COMMA 2024*. Ed. by Chris Reed, Matthias Thimm, and Tjitze Rienstra. IOS Press, pp. 373–374. DOI: 10.3233/FAIA240346.

Zerilli, John, Alistair Knott, James Maclaurin, and Colin Gavaghan (2019). "Algorithmic decision-making and the control problem". In: *Minds and Machines* 29.4, pp. 555–578. DOI: 10.1007/s11023-019-09513-7.

Zurek, Tomasz (2012). "Modelling of a'fortiori reasoning". In: *Expert Systems with Applications* 39.12, pp. 10772–10779. DOI: 10.1016/j.eswa.2012.02.188.

A

Appendix to Chapter 2

A.1. Proofs justification status

Lemma 2.2 (Complementary relation IN- and UNDEC-JUSTIFICATION)**.** *For any given $\sigma \in \{GR, CP, PR\}$, for each argumentation theory $AF = \langle \mathcal{A}, \mathcal{C} \rangle$ and argument $A \in \mathcal{A}$, each of the following holds:*

1. *A is σ-credulous-IN in AF iff A' is **not** σ-sceptical-UNDEC in $\langle \mathcal{A} \cup \{A'\}, \mathcal{C} \cup \{(A, A'), (A', A')\}\rangle$;*
2. *A is σ-sceptical-IN in AF iff A' is **not** σ-credulous-UNDEC in $\langle \mathcal{A} \cup \{A'\}, \mathcal{C} \cup \{(A, A'), (A', A')\}\rangle$;*
3. *A is σ-credulous-UNDEC in AF iff A' is **not** σ-sceptical-IN in $\langle \mathcal{A} \cup \{A', B, C\}, \mathcal{C} \cup \{(A, B), (A, C), (B, C), (C, A')\}\rangle$; and*
4. *A is σ-sceptical-UNDEC in AF iff A' is **not** σ-credulous-IN in $\langle \mathcal{A} \cup \{A', B, C\}, \mathcal{C} \cup \{(A, B), (A, C), (B, C), (C, A')\}\rangle$.*

Proof. Consider an arbitrary semantics $\sigma \in \{\text{GR}, \text{CP}, \text{PR}\}$, argumentation theory $AF = \langle \mathcal{A}, \mathcal{C} \rangle$ and argument $A \in \mathcal{A}$. We prove the four items separately.

1. Construct $AF' = \langle \mathcal{A} \cup \{A'\}, \mathcal{C} \cup \{(A, A'), (A', A')\}\rangle$; for an illustration, see the first and second columns of Figure 2.10.

   $\Rightarrow$ Suppose that A is σ-credulous-IN in AF: then there is some σ-extension S of AF containing A. Note that S also must be a σ-extension of AF': all arguments in $\mathcal{A}$ attacking attackers of A are still in $\mathcal{A} \cup \{A'\}$ and $S \cup \{A'\}$ is not a σ-extension as it is not conflict-free. Then there exists some σ-extension (i.e. S) of AF' in which A' is attacked by S, so A' is not σ-sceptical-UNDEC in AF'.

   $\Leftarrow$ Suppose that A' is not σ-sceptical-UNDEC in AF'; then there exists some σ-extension S of AF' such that either $A' \in S$ or some argument attacking

A' is in S. Given that A' is self-attacking, $A' \notin S$, so A' is attacked by some argument in S, which can only be A. Furthermore note that S is also a σ-extension of AF, since the arguments that are defended by S in AF are exactly the same as the arguments that are defended by S in AF'. To conclude, there exists some σ-extension (i.e. S) of AF in which $A \in S$, so A is σ-credulous-IN in AF.

2. Construct $AF' = \langle \mathcal{A} \cup \{A'\}, \mathcal{C} \cup \{(A, A'), (A', A')\}\rangle$.

   $\Rightarrow$ Suppose that A is σ-sceptical-IN in AF: A is in each σ-extension of AF. Then A is also in each σ-extension of AF', so for each σ-extension S of AF', A' is attacked by S. Accordingly, A' is not σ-credulous-UNDEC in AF'.

   $\Leftarrow$ If A' is not σ-credulous-UNDEC in AF' then each σ-extension of AF' contains an argument attacking A', which can only be A. Given that each σ-extension of AF is also a σ-extension of AF', A must be in each σ-extension of AF, and therefore is σ-sceptical-IN w.r.t. AF.

3. Construct $AF' = \langle \mathcal{A} \cup \{A', B, C\}, \mathcal{C} \cup \{(A, B), (A, C), (B, C), (C, A')\}\rangle$; for an illustration, see the first and third columns of Figure 2.10.

   $\Rightarrow$ Suppose that A is σ-credulous-UNDEC in AF: there is some σ-extension S of AF such that neither A, nor any attacker of A is in S. Given that all arguments in S are defended by S and all attackers of A in AF' are attackers of A in AF, there must be some σ-extension S' of AF' such that $S \subseteq S'$. Having that neither A, nor any of its attackers is in S', B is not in S' or attacked by any argument in S'. Then, C is not attacked by any argument in S', so A' cannot be in S'. This implies that $S' = S$ is a σ-extension of AF' not containing A', so A' is not σ-sceptical-IN in AF'.

   $\Leftarrow$ If A' is not σ-sceptical-IN in AF' then some σ-extension S of AF' does not contain A'. This implies that the arguments A and B are not in S either: otherwise, they would defend A' against C. So S contains exactly those arguments in $\mathcal{A}$ that are defended by S, which implies that S is a σ extension of AF as well. Finally note that S cannot attack any argument in $\mathcal{A}$ attacking A: otherwise, such an argument would defend B against A and then B would be in S. Given that A is not in S, nor attacked by any argument in S, while S is a σ extension of AF, we derive that A is σ-credulous-UNDEC in AF.

4. Construct $AF' = \langle \mathcal{A} \cup \{A', B, C\}, \mathcal{C} \cup \{(A, B), (A, C), (B, C), (C, A')\}\rangle$.

   $\Rightarrow$ If A is σ-sceptical-UNDEC in AF then no σ-extension of AF contains A or any attacker of A. This implies that no σ-extension of AF' would contain A or any of its attackers. Since each σ-extension of AF' is complete, it could not contain B or C, and therefore not A'. This implies that A' is not σ-credulous-IN in AF'.

   $\Leftarrow$ If A' is not σ-credulous-IN in AF' then no σ-extension of AF' contains A'. Because of the completeness criterion of σ semantics, no σ-extension of

AF' would contain A, B or any argument in $\mathcal{A}$ attacking A. Therefore for each σ-extension S of AF', all arguments in S are in $\mathcal{A} \setminus \{A\}$ and do not attack A. Then no σ-extension of AF can contain A or any attacker of A either. This implies that A is σ-sceptical-UNDEC in AF. □

Lemma 2.3 (Complexities UNDEC-JUSTIFICATION)**.** *For any given semantics σ in $\{GR, CP, PR\}$:*

1. *If the complexity of σ-credulous-IN-JUSTIFICATION is $\boldsymbol{C}$, then the complexity of σ-sceptical-UNDEC-JUSTIFICATION is co-$\boldsymbol{C}$; and*
2. *If the complexity of σ-sceptical-IN-JUSTIFICATION is $\boldsymbol{C}$, then the complexity of σ-credulous-UNDEC-JUSTIFICATION is co-$\boldsymbol{C}$.*

Proof. We prove these two items separately:

1. The first item can be proved by two reductions:
   - Each instance $I_1 = (\langle \mathcal{A}, \mathcal{C} \rangle, A)$ of σ-credulous-IN-JUSTIFICATION can, in polynomial time, be converted to an instance $I_2 = (\langle \mathcal{A} \cup \{A'\}, \mathcal{C} \cup \{(A, A'), (A', A')\} \rangle, A')$ of σ-sceptical-UNDEC-JUSTIFICATION where, by Lemma 2.2 item 1, I_1 is a positive instance iff I_2 is a negative instance. This implies that σ-credulous-IN-JUSTIFICATION reduces to σ-sceptical-UNDEC-JUSTIFICATION.
   - Each instance $I_1 = (\langle \mathcal{A}, \mathcal{C} \rangle, A)$ of σ-sceptical-UNDEC-JUSTIFICATION can in polynomial time be converted to an instance $I_2 = (\langle \mathcal{A} \cup \{A', B, C\}, \mathcal{C} \cup \{(A, B), (A, C), (B, C), (C, A')\} \rangle, A')$ of σ-credulous-IN-JUSTIFICATION. By Lemma 2.2 item 4, I_1 is a positive instance iff I_2 is a negative instance. So σ-sceptical-UNDEC-JUSTIFICATION reduces to σ-credulous-IN-JUSTIFICATION.
2. The second item can be proved by two other reductions:
   - Each instance $I_1 = (\langle \mathcal{A}, \mathcal{C} \rangle, A)$ of σ-sceptical-IN-JUSTIFICATION can, in polynomial time, be converted to an instance $I_2 = (\langle \mathcal{A} \cup \{A'\}, \mathcal{C} \cup \{(A, A'), (A', A')\} \rangle, A')$ of σ-credulous-UNDEC-JUSTIFICATION where, by Lemma 2.2 item 2, I_1 is a positive instance iff I_2 is a negative instance. This implies that σ-sceptical-IN-JUSTIFICATION reduces to σ-credulous-UNDEC-JUSTIFICATION.
   - Each instance $I_1 = (\langle \mathcal{A}, \mathcal{C} \rangle, A)$ of σ-credulous-UNDEC-JUSTIFICATION can, in polynomial time, be converted to an instance $I_2 = (\langle \mathcal{A} \cup \{A', B, C\}, \mathcal{C} \cup \{(A, B), (A, C), (B, C), (C, A')\} \rangle, A')$ of σ-sceptical-IN-JUSTIFICATION. By Lemma 2.2 item 3, I_1 is a positive instance iff I_2 is a negative instance. Thus σ-credulous-UNDEC-JUSTIFICATION does reduce to σ-sceptical-IN-JUSTIFICATION. □

A.2. Proofs stability

Lemma 2.5. *For any given $\sigma \in \{GR, CP, PR, ST\}$ and $c \in \{sceptical, credulous\}$, the complexity of σ-c-OUT-STABILITY equals the complexity of σ-c-IN-STABILITY.*

Proof. We prove this by a reduction from σ-c-OUT-STABILITY to σ-c-IN-STABILITY and by a reduction in the other direction.

- First, consider an arbitrary instance $(\mathcal{I}, A)$ of the σ-c-OUT-STABILITY problem where $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ and $A \in \mathcal{A}$. Construct $\mathcal{I}^* = \langle \mathcal{A} \cup \{B\}, \mathcal{A}^?, \mathcal{C} \cup \{(A, B)\}, \mathcal{C}^? \rangle$, where $B \notin \mathcal{A} \cup \mathcal{A}^?$.
  - If the instance $(\mathcal{I}, A)$ is positive, then A is stable-σ-c-OUT in $\mathcal{I}$. Let $AF^{*\prime} = \langle \mathcal{A}^{*\prime}, \mathcal{C}^{*\prime} \rangle$ be an arbitrary completion of $\mathcal{I}^*$. Note that $\langle \mathcal{A}^{*\prime} \setminus \{B\}, \mathcal{C}^{*\prime} \setminus \{(A, B)\} \rangle$ must be a completion of $\mathcal{I}$ where A is σ-c-OUT. Then A must also be σ-c-OUT in $AF^{*\prime}$, which implies that B is σ-c-IN in $AF^{*\prime}$. Generalising over all completions of $\mathcal{I}^*$, we derive that B is stable-σ-c-IN w.r.t. $\mathcal{I}^*$. So $(\mathcal{I}^*, B)$ is a positive instance of σ-c-IN-STABILITY.
  - Alternatively, the instance is negative: A is not stable-σ-c-OUT in $\mathcal{I}$. Then there is some completion $AF' = \langle \mathcal{A}', \mathcal{C}' \rangle$ of $\mathcal{I}$ such that A is not σ-c-OUT in AF'. Construct $AF^{*\prime} = \langle \mathcal{A}' \cup \{B\}, \mathcal{C}' \cup \{(A, B)\} \rangle$ and note that $AF^{*\prime}$ is a completion of $\mathcal{I}^*$. Since A was not σ-c-OUT in AF', it cannot be σ-c-OUT in $AF^{*\prime}$ either. As a consequence, B is not σ-c-IN in $AF^{*\prime}$, hence $(\mathcal{I}^*, B)$ is a negative instance of σ-c-IN-STABILITY.

  To conclude, σ-c-OUT-STABILITY can be reduced to σ-c-IN-STABILITY.
- Now, consider an arbitrary instance $(\mathcal{I}, A)$ of the σ-c-IN-STABILITY problem where $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ and $A \in \mathcal{A}$. Construct $\mathcal{I}^* = \langle \mathcal{A} \cup \{B\}, \mathcal{A}^?, \mathcal{C} \cup \{(A, B)\}, \mathcal{C}^? \rangle$ with $B \notin \mathcal{A} \cup \mathcal{A}^?$.
  - If the instance $(\mathcal{I}, A)$ is positive, then A is stable-σ-c-IN in $\mathcal{I}$. Let $AF^{*\prime} = \langle \mathcal{A}^{*\prime}, \mathcal{C}^{*\prime} \rangle$ be an arbitrary completion of $\mathcal{I}^*$. Note that $\langle \mathcal{A}^{*\prime} \setminus \{B\}, \mathcal{C}^{*\prime} \setminus \{(A, B)\} \rangle$ must be a completion of $\mathcal{I}$ where A is σ-c-IN. Then A must also be σ-c-IN in $AF^{*\prime}$, which implies that B is σ-c-OUT in $AF^{*\prime}$. Generalising over all completions of $\mathcal{I}^*$, we derive that B is stable-σ-c-OUT w.r.t. $\mathcal{I}^*$. So $(\mathcal{I}^*, B)$ is a positive instance of σ-c-OUT-STABILITY.
  - Alternatively, the instance is negative: A is not stable-σ-c-IN in $\mathcal{I}$. Then there is some completion $AF' = \langle \mathcal{A}', \mathcal{C}' \rangle$ of $\mathcal{I}$ such that A is not σ-c-IN in AF'. Construct $AF^{*\prime} = \langle \mathcal{A}' \cup \{B\}, \mathcal{C}' \cup \{(A, B)\} \rangle$ and note that $AF^{*\prime}$ is a completion of $\mathcal{I}^*$. Since A was not σ-c-IN in AF', it cannot be σ-c-IN in $AF^{*\prime}$ either. As a consequence, B is not σ-c-OUT in $AF^{*\prime}$, hence $(\mathcal{I}^*, B)$ is a negative instance of σ-c-OUT-STABILITY.

  To conclude, we have also shown that σ-c-IN-STABILITY can be reduced to σ-c-OUT-STABILITY.

From these two reductions, it follows that σ-c-IN-STABILITY and σ-c-OUT-STABILITY have the same complexity. □

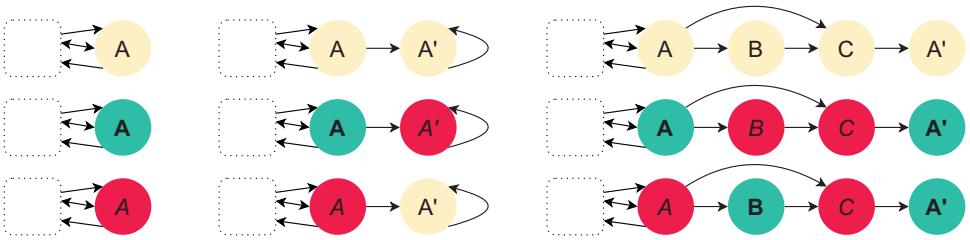


Figure A.1: Illustration of the incomplete argumentation frameworks that are used in transformations between stability problem instances that are used for proving Lemma 2.6. This figure is a repetition of Figure 2.10 that was used for proving Lemma 2.2, but now the dotted, rounded rectangles represent the part of the *incomplete* argumentation framework except (attacks related to) A.

Lemma 2.6 (Complexities UNDEC-STABILITY)**.** *For any given* $\sigma \in \{GR, CP, PR\}$*:*

1. *If possible credulous acceptance w.r.t.* σ *semantics is in the complexity class* $\boldsymbol{C}$*, then* σ*-sceptical-UNDEC-STABILITY is in the complexity class co-*$\boldsymbol{C}$*; and*
2. *If possible sceptical acceptance w.r.t.* σ *semantics is in the complexity class* $\boldsymbol{C}$*, then* σ*-credulous-UNDEC-STABILITY is in the complexity class co-*$\boldsymbol{C}$*.*

Proof. We prove the two items of this lemma separately.

1. The proof for the first item consists of two reductions: we show that possible credulous acceptance w.r.t. σ semantics reduces to the complementary problem of σ-sceptical-UNDEC-STABILITY and the other way around.

   - Let $I_1 = (\mathcal{I}, A)$ be an instance of possible credulous acceptance w.r.t. σ semantics where $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ and $A \in \mathcal{A}$. Construct $I_2 = (\mathcal{I}^*, A')$, an instance of σ-sceptical-UNDEC-STABILITY, where $\mathcal{I}^* = \langle \mathcal{A} \cup \{A'\}, \mathcal{A}^?, \mathcal{C} \cup \{(A, A'), (A', A')\}, \mathcal{C}^? \rangle$ and $A' \notin \mathcal{A} \cup \mathcal{A}^?$. For an illustration, see the first and second columns in Figure A.1. I_1 is a positive instance iff I_2 is a negative instance, as we show next:

     $\Rightarrow$ Suppose that I_1 is positive: $\mathcal{I}$ has some completion $\langle \mathcal{A}', \mathcal{C}' \rangle$ for which there is a σ extension S containing A. Then S is also a σ extension of $AF' = \langle \mathcal{A}' \cup \{A'\}, \mathcal{C}' \cup \{(A, A'), (A', A')\} \rangle$, so A is *not* σ-sceptical-UNDEC in AF'. Note that AF' is a completion of $\mathcal{I}^*$, so A' is *not* stable-σ-sceptical-UNDEC w.r.t. $\mathcal{I}^*$. Therefore, I_2 is a negative instance.

     $\Leftarrow$ Suppose that I_2 is negative: there is some completion $\langle \mathcal{A} \cup \mathcal{A}^{*\prime} \cup \{A'\}, \mathcal{C} \cup \mathcal{C}^{*\prime} \cup \{(A, A'), (A', A')\} \rangle$ of $\mathcal{I}^*$ with some σ extension S such that S either contains A' or some argument attacking A'. S cannot contain A' as S must be conflict-free under σ semantics, so there must be some argument attacking A' in S, which can only be A. Since S is also a σ extension of $\langle \mathcal{A} \cup \mathcal{A}^*, \mathcal{C} \cup \mathcal{C}^* \rangle$, which is a completion

of $\mathcal{I}'$, I_1 is a positive instance of possible credulous acceptance w.r.t. σ semantics.

- Let $I_1 = (\mathcal{I}, A)$ be an instance of σ-sceptical-UNDEC-STABILITY where $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ and $A \in \mathcal{A}$. Now let $I_2 = (\mathcal{I}^*, A')$ be an instance of possible credulous acceptance w.r.t. σ semantics, where $\mathcal{I}^* = \langle \mathcal{A} \cup \{A', B, C\}, \mathcal{A}^?, \mathcal{C} \cup \{(A, B), (A, C), (B, C), (C, A')\}, \mathcal{C}^? \rangle$ and none of A', B and C is in $\mathcal{A} \cup \mathcal{A}^?$. For an illustration, see the first and third columns in Figure A.1. We claim that I_1 is a positive instance iff I_2 is a negative instance:

  $\Rightarrow$ Suppose that I_1 is positive: there is no completion of $\mathcal{I}$ such that A nor any argument attacking A is in any σ extension of that completion. Now suppose towards a contradiction that A' is possibly credulously accepted w.r.t. $\mathcal{I}^*$. In other words, there is some completion $\langle \mathcal{A}^{*\prime}, \mathcal{C}^{*\prime} \rangle$ of $\mathcal{I}^*$ which has some extension S containing A'. Given that A' is only attacked by C in $\langle \mathcal{A}^{*\prime}, \mathcal{C}^{*\prime} \rangle$ and C is attacked by A and B (and σ is complete), it must be that either A or B is in S. In case B is in S, it must be defended against A, so some argument attacking A must be in S. But that would imply that there is some completion of $\mathcal{I}$ having some σ extension S' such that either $A \in S'$ or some argument attacking A is in S', which would contradict our assumption. So A' is *not* possibly credulously accepted w.r.t. $\mathcal{I}^*$. Therefore, I_2 is a negative instance.

  $\Leftarrow$ Suppose that I_2 is negative: no completion of $\mathcal{I}^*$ has a σ extension that contains A'. Towards a contradiction, suppose that I_1 is negative as well. Then there is some completion $\langle \mathcal{A}', \mathcal{C}' \rangle$ of $\mathcal{I}$ that has some σ extension S containing A or an argument attacking A.

  - First suppose that $A \in S$. Now consider the argumentation framework $AF^{*\prime} = \langle \mathcal{A}' \cup \{B, C, A'\}, \mathcal{C}' \cup \{(A, B), (A, C), (B, C), (C, A')\} \rangle$ and the set $S^{*\prime} = S \cup \{A'\}$. Given that S is a σ extension of $\langle \mathcal{A}', \mathcal{C}' \rangle$, it must be that $S^{*\prime}$ is a σ extension of $AF^{*\prime}$.
  - Alternatively, suppose that some argument attacking A is in S. Now consider the argumentation framework $AF^{*\prime} = \langle \mathcal{A}' \cup \{B, C, A'\}, \mathcal{C}' \cup \{(A, B), (A, C), (B, C), (C, A')\} \rangle$ and the set $S^{*\prime} = S \cup \{B, A'\}$. Given that S is a σ extension of $\langle \mathcal{A}', \mathcal{C}' \rangle$, it must be that $S^{*\prime}$ is a σ extension of $AF^{*\prime}$.

  In both cases, there is a completion of $\mathcal{I}^*$ with a σ extension that contains A'. But this contradicts our assumption that I_2 is negative, so I_1 must have been positive.

2. The proof for the second item is similar and consists of two reductions as well.

   - Let $I_1 = (\mathcal{I}, A)$ be an instance of possible sceptical acceptance w.r.t. σ semantics where $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ and $A \in \mathcal{A}$. Let $I_2 = (\mathcal{I}^*, A')$ be an instance of σ-credulous-UNDEC-STABILITY, where $\mathcal{I}^* = \langle \mathcal{A} \cup \{A'\}, \mathcal{A}^?, \mathcal{C} \cup$

$\{(A, A'), (A', A')\}, \mathcal{C}^?\rangle$ and $A' \notin \mathcal{A} \cup \mathcal{A}^?$. For an illustration, see the first and second columns in Figure A.1. I_1 is positive iff I_2 is negative:

$\Rightarrow$ If I_1 is positive then there exists some completion $\langle \mathcal{A}', \mathcal{C}' \rangle$ of $\mathcal{I}$ such that each of its σ extensions contains A. Each σ extension of $AF' = \langle \mathcal{A}' \cup \{A'\}, \mathcal{C}' \cup \{(A, A'), (A', A')\}\rangle$ is also a σ extension of $\langle \mathcal{A}', \mathcal{C}' \rangle$, so A' is *not* σ-credulous-UNDEC in AF'. Since AF' is a completion of $\mathcal{I}^*$, A' is *not* stable-σ-credulous-UNDEC w.r.t. $\mathcal{I}^*$, hence I_2 is a negative instance.

$\Leftarrow$ If I_2 is negative then there exists some completion $AF^* = \langle \mathcal{A} \cup \mathcal{A}^* \cup \{A'\}, \mathcal{C} \cup \mathcal{C}^* \cup \{(A, A'), (A', A')\}\rangle$ of $\mathcal{I}^*$ such that each σ extension contains some argument attacking A', which can only be A. Since each σ extension of $AF' = \langle \mathcal{A} \cup \mathcal{A}^*, \mathcal{C} \cup \mathcal{C}^* \rangle$ is also a σ extension of AF^*, A must be σ-sceptical-IN in AF'. Since AF' is a completion of $\mathcal{I}$, I_1 is a positive instance of possible sceptical acceptance w.r.t. σ semantics.

- Let $I_1 = (\mathcal{I}, A)$ be an instance of σ-sceptical-UNDEC-STABILITY where $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ and $A \in \mathcal{A}$. Now let $I_2 = (\mathcal{I}^*, A')$ be an instance of possible credulous acceptance w.r.t. σ semantics, where $\mathcal{I}^* = \langle \mathcal{A} \cup \{A', B, C\}, \mathcal{A}^?, \mathcal{C} \cup \{(A, B), (A, C), (B, C), (C, A')\}, \mathcal{C}^? \rangle$ and none of A', B and C is in $\mathcal{A} \cup \mathcal{A}^?$. For an illustration, see the first and third columns in Figure A.1. I_1 is a positive instance iff I_2 is a negative instance:

  $\Rightarrow$ If I_1 is positive then for each completion of $\mathcal{I}$ and for each σ extension S of that completion, A is not in S and not attacked by any argument in S. If I_2 would be positive as well, then there would be some completion $AF^{*\prime}$ of $\mathcal{I}^*$ such that A' is in some σ extension S. Then, by completeness of σ, either A or B (and therefore an argument attacking A) must have been in S. This implies that A was not σ-sceptical-UNDEC (but -IN/-OUT) in $AF^{*\prime} = \langle \mathcal{A} \cup \mathcal{A}^{*\prime} \cup \{A', B, C\}, \mathcal{C} \cup \mathcal{C}^{*\prime} \cup \{(A, B), (A, C), (B, C), (C, A')\}\rangle$. However, then $S \setminus \{A', B, C\}$ must have been a σ extension of AF' where $AF' = \langle \mathcal{A} \cup \mathcal{A}^{*\prime}, \mathcal{C} \cup \mathcal{C}^{*\prime} \rangle$, which is a completion of $\mathcal{I}$. This contradicts our earlier assumption that I_1 is positive, so I_2 must have been negative.

  $\Leftarrow$ If I_2 is a negative instance then for each completion of $\mathcal{I}^*$ and for each σ extension S in that completion, A' was not in S. Assume towards a contradiction that I_1 was a negative instance; then there is some completion $AF' = \langle \mathcal{A}', \mathcal{C}' \rangle$ of $\mathcal{I}$ having some σ extension S such that A is either in S or attacked by any argument in S. Now construct $AF^* = \langle \mathcal{A}' \cup \{A', B, C\}, \mathcal{C}' \cup \{(A, B), (A, C), (B, C), (C, A')\}\rangle$. Given that S is a σ extension of AF', either $S \cup \{A'\}$ (if $A \in S$) or $S \cup \{B, A'\}$ (if $A \notin S$) must be a σ extension of AF^*. In any case, A' is contained in a σ extension of AF^*, while AF^* is a completion of $\mathcal{I}^*$, which contradicts the fact that I_2 is negative. We must therefore conclude that I_1 was negative. $\square$

A.3. Proofs relevance

Lemma 2.7. *Given an IAF $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$, a certain argument $A \in \mathcal{A}$ and a justification status j:*

1. *For each $U \in \mathcal{A}^?$, addition of U is j-relevant for A w.r.t. $\mathcal{I}$ iff there exists some $\mathcal{I}' = \langle \mathcal{A}', \{U\}, \mathcal{C}', \emptyset \rangle \in \mathtt{part}(\mathcal{I})$ such that A is **not** j in the certain projection of $\mathcal{I}'$, while A is j in the certain projection of $\langle \mathcal{A}' \cup \{U\}, \emptyset, \mathcal{C}', \emptyset \rangle$.*
2. *For each $U \in \mathcal{C}^?$, addition of U is j-relevant for A w.r.t. $\mathcal{I}$ iff there exists some $\mathcal{I}' = \langle \mathcal{A}', \emptyset, \mathcal{C}', \{U\} \rangle \in \mathtt{part}(\mathcal{I})$ such that A is **not** j in the certain projection of $\mathcal{I}'$, while A is j in the certain projection of $\langle \mathcal{A}', \emptyset, \mathcal{C}' \cup \{U\}, \emptyset \rangle$.*
3. *For each $U \in \mathcal{A}^?$, removal of U is j-relevant for A w.r.t. $\mathcal{I}$ iff there exists some $\mathcal{I}' = \langle \mathcal{A}', \{U\}, \mathcal{C}', \emptyset \rangle \in \mathtt{part}(\mathcal{I})$ such that A is j in the certain projection of $\mathcal{I}'$, while A is **not** j in the certain projection of $\langle \mathcal{A}' \cup \{U\}, \emptyset, \mathcal{C}', \emptyset \rangle$.*
4. *For each $U \in \mathcal{C}^?$, removal of U is j-relevant for A w.r.t. $\mathcal{I}$ iff there exists some $\mathcal{I}' = \langle \mathcal{A}', \{U\}, \mathcal{C}', \emptyset \rangle \in \mathtt{part}(\mathcal{I})$ such that A is j in the certain projection of $\mathcal{I}'$, while A is **not** j in the certain projection of $\langle \mathcal{A}', \emptyset, \mathcal{C}' \cup \{U\}, \emptyset \rangle$.*

Proof. Let $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ be an incomplete argumentation framework, $A \in \mathcal{A}$ a certain argument and j a justification status.

1. We prove both directions of the first item.

   $\Rightarrow$ See Figure 2.11 for an illustration of the steps in this proof.

   (a) Suppose that addition of $U \in A^?$ is j-relevant for A w.r.t. $\mathcal{I}$.

   (b) Then by Definition 2.11 there is a minimal stable-j partial completion $\mathcal{I}^* = \langle \mathcal{A}^*, \mathcal{A}^{?*}, \mathcal{C}^*, \mathcal{C}^{?*} \rangle$ for A w.r.t. $\mathcal{I}$ such that $U \in \mathcal{A}^*$.

   (c) Now construct the IAF $\mathcal{I}'$ from $\mathcal{I}^*$ by moving U from the certain to the uncertain part: $\mathcal{I}' = \langle \mathcal{A}^* \setminus \{U\}, \mathcal{A}^{?*} \cup \{U\}, \mathcal{C}^*, \mathcal{C}^{?*} \rangle$.

   (d) Given that $\mathcal{I}^*$ was minimal and $\mathcal{I}^* \in \mathtt{part}(\mathcal{I}')$, A cannot be stable-j w.r.t. $\mathcal{I}'$. So there must be some $\mathcal{I}'' = \langle \mathcal{A}'', \mathcal{A}^{?\prime\prime}, \mathcal{C}'', \mathcal{C}^{?\prime\prime} \rangle$ in $\mathtt{part}(\mathcal{I}')$ such that A's justification status in the certain projection of $\mathcal{I}''$ is not j – note that this means that U is not in $\mathcal{A}''$ (since A was stable-j in $\mathcal{I}^*$).

   (e) Then the justification status of A in the certain projection of $\mathcal{I}''' = \langle \mathcal{A}'', \{U\}, \mathcal{C}'', \emptyset \rangle$ is not j (because this is the same as the certain projection of $\mathcal{I}''$, i.e. $\langle \mathcal{A}'', \mathcal{C}''|_{\mathcal{A}''} \rangle$).

   (f) Next, construct $\mathcal{I}^{*\prime} = \langle \mathcal{A}'' \cup \{U\}, \emptyset, \mathcal{C}'', \emptyset \rangle$ from $\mathcal{I}'''$ by moving U from the uncertain part to the certain part. Since $\mathcal{I}^{*\prime}$ is in $\mathtt{part}(\mathcal{I}^*)$ and A is stable-j in $\mathcal{I}^*$, A must be j in the certain projection of $\mathcal{I}^{*\prime}$.

   $\Leftarrow$ Suppose that there exists some $\mathcal{I}' = \langle \mathcal{A}', \{U\}, \mathcal{C}', \emptyset \rangle \in \mathtt{part}(\mathcal{I})$ such that A is not j in $\mathtt{cert}(\mathcal{I}')$ and A is j in $\mathtt{cert}(\langle \mathcal{A}' \cup \{U\}, \emptyset, \mathcal{C}', \emptyset \rangle)$. Given that $\langle \mathcal{A}' \cup \{U\}, \emptyset, \mathcal{C}', \emptyset \rangle$ has only one completion (i.e., its certain projection), A must be stable-j w.r.t. $\langle \mathcal{A}' \cup \{U\}, \emptyset, \mathcal{C}', \emptyset \rangle$. Consequently, there must

be some minimal stable-j partial completion $\mathcal{I}'' = \langle \mathcal{A}'', \mathcal{A}^{?\prime\prime}, \mathcal{C}'', \mathcal{C}^{?\prime\prime} \rangle$ for A w.r.t. $\mathcal{I}$ such that $\langle \mathcal{A}' \cup \{U\}, \emptyset, \mathcal{C}', \emptyset \rangle \in \texttt{part}(\mathcal{I}'')$. Note that $U \in \mathcal{A}''$: otherwise $\langle \mathcal{A}', \emptyset, \mathcal{C}', \emptyset \rangle$ would also be in $\texttt{part}(\mathcal{I}'')$, which contradicts the assumption that A is not j in $\texttt{cert}(\langle \mathcal{A}', \{U\}, \mathcal{C}', \emptyset \rangle)$. To conclude, addition of U is j-relevant for A w.r.t. $\mathcal{I}$.

2. This proof is analogous to the proof of item 1, except that U is moved between certain and uncertain attacks rather than between certain and uncertain arguments.

3. We prove both directions of the third item.

   $\Rightarrow$ The steps in this proof are similar to the steps in item 1.

   (a) Suppose that removal of $U \in A^?$ is j-relevant for A w.r.t. $\mathcal{I}$.

   (b) Then by Definition 2.11 there is a minimal stable-j partial completion $\mathcal{I}^* = \langle \mathcal{A}^*, \mathcal{A}^{?*}, \mathcal{C}^*, \mathcal{C}^{?*} \rangle$ for A w.r.t. $\mathcal{I}$ such that $U \notin \mathcal{A}^* \cup \mathcal{A}^{?*}$.

   (c) Now construct the IAF $\mathcal{I}'$ by adding U to the uncertain arguments and adding the attacks from $\mathcal{C}$ related to U to the certain attacks:

   $\mathcal{I}' = \langle \mathcal{A}^*, \mathcal{A}^{?*} \cup \{U\}, \mathcal{C}^* \cup \{(X,Y) \in \mathcal{C} \mid U \in \{X,Y\} \text{ and } \{X,Y\} \subseteq \mathcal{A}^{?*} \cup \{U\}\}, \mathcal{C}^{?*} \rangle$.

   Note that $\mathcal{I}^* \in \texttt{part}(\mathcal{I}')$.

   (d) Given that $\mathcal{I}^*$ was minimal, A cannot be stable-j w.r.t. $\mathcal{I}'$, hence there must be some $\mathcal{I}'' = \langle \mathcal{A}'', \mathcal{A}^{?\prime\prime}, \mathcal{C}'', \mathcal{C}^{?\prime\prime} \rangle$ in $\texttt{part}(\mathcal{I}')$ such that A is not j in $\texttt{cert}(\mathcal{I}'')$.

   (e) Then the justification status of A in the certain projection of $\mathcal{I}''' = \langle \mathcal{A}'', \emptyset, \mathcal{C}'', \emptyset \rangle$ (which is the same as $\texttt{cert}(\mathcal{I}'')$,i.e. $\langle \mathcal{A}'', \mathcal{C}''|_{\mathcal{A}''} \rangle$) is not j either. Note that $\mathcal{A}''$ must contain U: otherwise, $\mathcal{I}'''$ would have been in $\texttt{part}(\mathcal{I}^*)$.

   (f) Next, construct $\mathcal{I}^{*\prime\prime} = \langle \mathcal{A}'' \setminus \{U\}, \emptyset, \mathcal{C}''|_{\mathcal{A}'' \setminus \{U\}}, \emptyset \rangle$ from $\mathcal{I}'''$ by removing U from the uncertain arguments. Since $\mathcal{I}^{*\prime\prime}$ is in $\texttt{part}(\mathcal{I}^*)$ and A is stable-j in $\mathcal{I}^*$, A must be j in $\texttt{cert}(\mathcal{I}^{*\prime})$. This implies that A's justification status is j as well in the certain projection of $\mathcal{I}^{*\prime} = \langle \mathcal{A}'' \setminus \{U\}, \{U\}, \mathcal{C}'', \emptyset \rangle$, which is the same as $\texttt{cert}(\mathcal{I}^{*\prime\prime})$.

   $\Leftarrow$ Suppose that there exists some $\mathcal{I}' = \langle \mathcal{A}', \{U\}, \mathcal{C}', \emptyset \rangle \in \texttt{part}(\mathcal{I})$ such that A is j in $\texttt{cert}(\mathcal{I}')$ and A is not j in $\texttt{cert}(\langle \mathcal{A}' \cup \{U\}, \emptyset, \mathcal{C}', \emptyset \rangle)$. Then A is stable-j w.r.t. $\langle \mathcal{A}', \emptyset, \mathcal{C}', \emptyset \rangle$. Consequently, there must be some minimal stable-j partial completion $\mathcal{I}'' = \langle \mathcal{A}'', \mathcal{A}^{?\prime\prime}, \mathcal{C}'', \mathcal{C}^{?\prime\prime} \rangle$ for A w.r.t. $\mathcal{I}$ such that $\mathcal{I} \in \texttt{part}(\mathcal{I}'')$. Note that $U \notin \mathcal{A}'' \cup \mathcal{A}^{?\prime\prime}$: otherwise $\langle \mathcal{A}' \cup \{U\}, \emptyset, \mathcal{C}', \emptyset \rangle$ would be in $\texttt{part}(\mathcal{I}'')$, which contradicts the assumption that A is not j in $\texttt{cert}(\langle \mathcal{A}' \cup \{U\}, \emptyset, \mathcal{C}', \emptyset \rangle)$. To conclude, removal of U is j-relevant for A w.r.t. $\mathcal{I}$.

4. This proof is analogous to the proof of item 3, except that U is moved between

certain and uncertain attacks rather than between certain and uncertain arguments. □

Lemma 2.8. *[Reduction INSTABILITY to RELEVANCE] Given an incomplete argumentation framework $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$, a certain argument $A \in \mathcal{A}$, semantics $\sigma \in \{GR, CP, PR\}$ and $c \in \{sceptical, credulous\}$:*

1. *Construct $\mathcal{I}'$ and $\mathcal{I}''$ as follows (see Figure 2.12), where A', A'', U and U' are not in $\mathcal{A} \cup \mathcal{A}^?$:*
   - $\mathcal{A}' = \mathcal{A} \cup \{A', A''\}$;
   - $\mathcal{C}' = \mathcal{C} \cup \{(A, A'), (A', A''), (U, A')\}$;
   - $\mathcal{I}' = \langle \mathcal{A}', \mathcal{A}^? \cup \{U\}, \mathcal{C}', \mathcal{C}^? \rangle$; *and*
   - $\mathcal{I}'' = \langle \mathcal{A}' \cup \{U\}, \mathcal{A}^? \cup \{U'\}, \mathcal{C}' \cup \{(U', U)\}, \mathcal{C}^? \rangle$.

   The following three items are equivalent:
   (a) *A is **not** stable-σ-c-IN w.r.t. $\mathcal{I}$; and*
   (b) *addition of U is σ-c-IN-relevant for A'' w.r.t. $\mathcal{I}'$; and*
   (c) *removal of U' is σ-c-IN-relevant for A'' w.r.t. $\mathcal{I}''$.*
2. *Let $\mathcal{I}' = \langle \mathcal{A}, \mathcal{A}^? \cup \{U\}, \mathcal{C} \cup \{(U, A)\}, \mathcal{C}^? \rangle$ and $\mathcal{I}'' = \langle \mathcal{A} \cup \{U\}, \mathcal{A}^? \cup \{U'\}, \mathcal{C} \cup \{(U, A), (U', U)\}, \mathcal{C}^? \rangle$ where U and U' are not in $\mathcal{A} \cup \mathcal{A}^?$. The following three items are equivalent:*
   (a) *A is **not** stable-σ-c-OUT w.r.t. $\mathcal{I}$; and*
   (b) *addition of U is σ-c-OUT-relevant for A w.r.t. $\mathcal{I}'$; and*
   (c) *removal of U' is σ-c-OUT-relevant for A w.r.t. $\mathcal{I}''$.*
3. *Construct $\mathcal{I}'$ and $\mathcal{I}''$ as follows, where A', A'', U and U' are not in $\mathcal{A} \cup \mathcal{A}^?$:*
   - $\mathcal{A}' = \mathcal{A} \cup \{A', A''\}$;
   - $\mathcal{C}' = \mathcal{C} \cup \{(A, A'), (A', A''), (U, A'), (U, A''), (U, U)\}$;
   - $\mathcal{I}' = \langle \mathcal{A}', \mathcal{A}^? \cup \{U\}, \mathcal{C}', \mathcal{C}^? \rangle$; *and*
   - $\mathcal{I}'' = \langle \mathcal{A}' \cup \{U\}, \mathcal{A}^? \cup \{U'\}, \mathcal{C}' \cup \{(U', U)\}, \mathcal{C}^? \rangle$.

   The following three items are equivalent:
   (a) *A is **not** stable-σ-c-UNDEC w.r.t. $\mathcal{I}$; and*
   (b) *addition of U is σ-c-UNDEC-relevant for A'' w.r.t. $\mathcal{I}'$; and*
   (c) *removal of U' is σ-c-UNDEC-relevant for A'' w.r.t. $\mathcal{I}''$.*

Proof. In the following, we prove all three items separately.

1. We start by proving the first item, related to IN justification statuses. Let $\mathcal{I}' = \langle \mathcal{A} \cup \{A', A''\}, \mathcal{A}^? \cup \{U\}, \mathcal{C} \cup \{(A, A'), (A', A'')\}, \mathcal{C}^? \rangle$ and let $\mathcal{I}'' = \langle \mathcal{A} \cup \{A', A'', U\}, \mathcal{A}^? \cup \{U'\}, \mathcal{C} \cup \{(A, A'), (A', A''), (U, A'), (U', U)\}, \mathcal{C}^? \rangle$ (see Figure 2.12).

   (a) ⇒ (b) and (c) Suppose that A is not stable-σ-c-IN w.r.t. $\mathcal{I}$ (a). Then by Definition 2.7 of stability there is some completion $AF^* = \langle \mathcal{A}^*, \mathcal{C}^* \rangle$ in which A is not σ-c-IN. Next, we construct three argumentation frameworks based on AF, containing the argument A'', and discuss its status.

   - First, construct $AF_1 = \langle \mathcal{A}^* \cup \{A', A''\}, \mathcal{C}^* \cup \{(A, A'), (A', A'')\} \rangle$. Given that A'' is attacked by A', which is only attacked by A in AF_1, A'' cannot be σ-c-IN in AF_1.
   - Let $AF_2 = \langle \mathcal{A}^* \cup \{A', A'', U\}, \mathcal{C}^* \cup \{(A, A'), (A', A''), (U, A')\} \rangle$. A'' is σ-c-IN in AF_2, since the unattacked argument U attacks the only attacker of A'' (i.e. A').
   - Next, let $AF_3 = \langle \mathcal{A}^* \cup \{A', A'', U, U'\}, \mathcal{C}^* \cup \{(A, A'), (A', A''), (U, A'), (U', U)\} \rangle$. Given that A is not σ-c-IN in AF^*, A cannot be σ-c-IN in AF_3 either. Since the argument A'' in AF_3 is attacked by A', which is only attacked by A, A'' cannot be σ-c-IN in AF_3.

   Now item *(b)* (addition of U is σ-c-IN-relevant for A'' w.r.t. $\mathcal{I}'$) follows from Lemma 2.7, the fact that the incomplete argumentation framework $\langle \mathcal{A}^* \cup \{A', A''\}, \{U\}, \mathcal{C}^* \cup \{(A, A'), (A', A''), (U, A')\}, \emptyset \rangle$ is in $\texttt{part}(\mathcal{I}')$ and the status of A'' in AF_1 and AF_2.

   Similarly, item *(c)* (removal of U' is σ-c-IN-relevant for A'' w.r.t. $\mathcal{I}''$) follows from Lemma 2.7, the fact that the incomplete argumentation framework $\langle \mathcal{A}^* \cup \{A', A'', U\}, \{U'\}, \mathcal{C}^* \cup \{(A, A'), (A', A''), (U, A'), (U', U)\}, \emptyset \rangle$ is in $\texttt{part}(\mathcal{I}'')$ and the status of A'' in AF_2 and AF_3.

   (b) ⇒ (a) Suppose that addition of U is σ-c-IN-relevant for A'' w.r.t. $\mathcal{I}'$. Then by Lemma 2.7, there is some $\mathcal{I}^{*\prime}$ in $\texttt{part}(\mathcal{I}')$ such that A'' is *not* σ-c-IN in the certain projection of $\mathcal{I}^{*\prime}$ – let us call this certain projection $AF^{*\prime} = \langle \mathcal{A}^{*\prime}, \mathcal{C}^{*\prime} \rangle$. Then A cannot be σ-c-IN in $AF^{*\prime}$ either (since A attacks A', which is the only attacker of A''). Now construct $AF' = \langle \mathcal{A}', \mathcal{C}' \rangle$ where $\mathcal{A}' = \mathcal{A}^{*\prime} \setminus \{A', A'', U\}$ and $\mathcal{C}' = \mathcal{C}^{*\prime} \setminus \{(A, A'), (A', A''), (U, A')\}$. Since A is not σ-c-IN in $AF^{*\prime}$, it cannot be σ-c-IN in AF'. Given that AF' is a completion of $\mathcal{I}$, A cannot be stable-σ-c-IN w.r.t. $\mathcal{I}$.

   (c) ⇒ (a) Suppose that removal of U' is σ-c-IN-relevant for A'' w.r.t. $\mathcal{I}''$. Then by Lemma 2.7, there is some $\mathcal{I}^{*\prime\prime}$ in $\texttt{part}(\mathcal{I}'')$ such that A'' is *not* σ-c-IN in the certain projection of $\mathcal{I}^{*\prime\prime}$ – without loss of generality, let this certain projection be $AF^{*\prime\prime} = \langle \mathcal{A}^{*\prime\prime}, \mathcal{C}^{*\prime\prime} \rangle$. Now construct $AF' = \langle \mathcal{A}', \mathcal{C}' \rangle$ where $\mathcal{A}' = \mathcal{A}^{*\prime\prime} \setminus \{A', A'', U, U'\}$ and $\mathcal{C}' = \mathcal{C}^{*\prime\prime} \setminus \{(A, A'), (A', A''), (U, A'), (U', U)\}$; since A'' is not σ-c-IN in $AF^{*\prime}$, it cannot be that A is σ-c-IN in AF'. As AF' is a completion of $\mathcal{I}$, A cannot be stable-σ-c-IN w.r.t. $\mathcal{I}$.

2. We proceed by proving the second item, related to OUT justification statuses. Let $\mathcal{I}' = \langle \mathcal{A}, \mathcal{A}^? \cup \{U\}, \mathcal{C} \cup \{(U, A)\}, \mathcal{C}^? \rangle$ and let $\mathcal{I}'' = \langle \mathcal{A} \cup \{U\}, \mathcal{A}^? \cup \{U'\}, \mathcal{C} \cup \{(U, A), (U', U)\}, \mathcal{C}^? \rangle$ (see Figure A.2).

   (a) ⇒ (b) and (c) Suppose that A is not stable-σ-c-OUT w.r.t. $\mathcal{I}$ (a). Then there is some completion $AF_1 = \langle \mathcal{A}^*, \mathcal{C}^* \rangle$ of $\mathcal{I}$ such that A is not σ-c-OUT. Next, we construct two additional argumentation frameworks based on AF_1, containing the argument A.

   - First, construct $AF_2 = \langle \mathcal{A}^* \cup \{U\}, \mathcal{C}^* \cup \{(U, A)\} \rangle$. A is σ-c-OUT in AF_2, since the unattacked argument U attacks A.
   - Let $AF_3 = \langle \mathcal{A}^* \cup \{U, U'\}, \mathcal{C}^* \cup \{(U, A), (U', U)\} \rangle$. Given that A is not σ-c-OUT in AF_1, A cannot be σ-c-OUT in AF_3 either.

   Now item *(b)* (addition of U is σ-c-OUT-relevant for A w.r.t. $\mathcal{I}'$) follows from Lemma 2.7, the fact that the IAF $\langle \mathcal{A}^*, \{U\}, \mathcal{C}^* \cup \{(U, A)\}, \emptyset \rangle$ is in $\texttt{part}(\mathcal{I}')$ and the status of A in AF_1 and AF_2.

   Similarly, item *(c)* (removal of U' is σ-c-OUT-relevant for A w.r.t. $\mathcal{I}''$) follows from Lemma 2.7, the fact that the IAF $\langle \mathcal{A}^* \cup \{U\}, \{U'\}, \mathcal{C}^* \cup \{(U, A), (U', U)\}, \emptyset \rangle$ is in $\texttt{part}(\mathcal{I}'')$ and the status of A in AF_2 and AF_3.

   (b) ⇒ (a) Suppose that addition of U is σ-c-OUT-relevant for A w.r.t. $\mathcal{I}'$. Then by Lemma 2.7, there is some $\mathcal{I}^{*\prime}$ in $\texttt{part}(\mathcal{I}')$ such that A is *not* σ-c-OUT in its certain projection $AF^{*\prime} = \langle \mathcal{A}', \mathcal{C}' \rangle$. Note that U cannot be in $\mathcal{A}'$: otherwise, A would be σ-c-OUT. Given that $AF^{*\prime}$ is a completion of $\mathcal{I}$, A cannot be stable-σ-c-OUT w.r.t. $\mathcal{I}$.

   (c) ⇒ (a) Suppose that removal of U' is σ-c-OUT-relevant for A w.r.t. $\mathcal{I}''$. Then by Lemma 2.7, there is some $\mathcal{I}^{*\prime\prime}$ in $\texttt{part}(\mathcal{I}'')$ such that A is *not* σ-c-OUT in $AF^{*\prime\prime} = \langle \mathcal{A}^{*\prime\prime}, \mathcal{C}^{*\prime\prime} \rangle$, the certain projection of $\mathcal{I}^{*\prime\prime}$. Now construct $AF' = \langle \mathcal{A}', \mathcal{C}' \rangle$ where $\mathcal{A}' = \mathcal{A}^{*\prime\prime} \setminus \{U, U'\}$ and $\mathcal{C}' = \mathcal{C}^{*\prime\prime} \setminus \{(U, A), (U', U)\}$; since A is not σ-c-OUT in $AF^{*\prime}$, it cannot be that A is σ-c-OUT in AF'. Given that AF' is a completion of $\mathcal{I}$, A cannot be stable-σ-c-OUT w.r.t. $\mathcal{I}$.

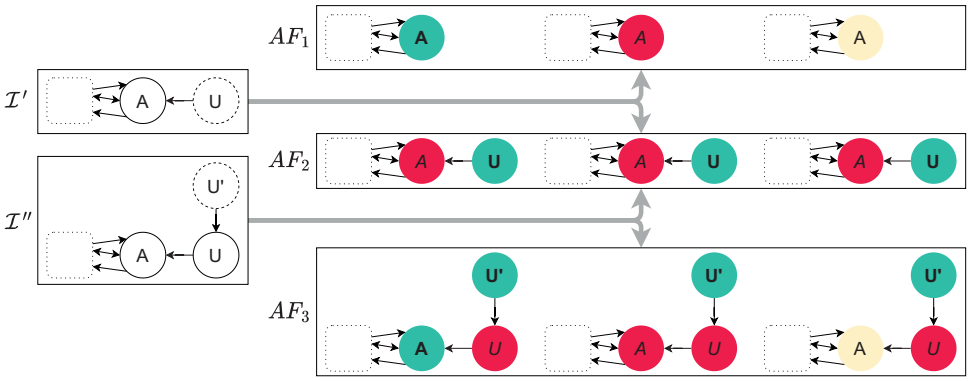


Figure A.2: Illustration of the IAF used to show Lemma 2.8 item 2. The IAFs given on the left are $\mathcal{I}'$ (upper) and $\mathcal{I}''$ (lower). The rounded rectangle with dotted borders represents the original IAF $\mathcal{I}$ (without A and in- and outgoing attacks). The grey arrows point to certain projections AF_1, AF_2 and AF_3 of partial completions. For each of these AFs, the possible justification statuses are colour-coded: green arguments with boldface font are IN, yellow arguments are UNDEC and red arguments with italic font are OUT. Note that, for a given justification status of A, there is only one possible justification status for each of the additional arguments in $\{U, U'\}$.

3. Finally, we prove the third item, related to UNDEC justification statuses. Let $\mathcal{I}' = \langle \mathcal{A} \cup \{A', A''\}, \mathcal{A}^? \cup \{U\}, \mathcal{C} \cup \{(A, A'), (A', A''), (U, A'), (U, A''), (U, U)\}, \mathcal{C}^? \rangle$ and let $\mathcal{I}'' = \langle \mathcal{A} \cup \{A', A'', U\}, \mathcal{A}^? \cup \{U'\}, \mathcal{C} \cup \{(A, A'), (A', A''), (U, A'), (U, A''), (U, U), (U', U)\}, \mathcal{C}^? \rangle$ (see Figure A.3).

   (a) $\Rightarrow$ (b) and (c) If A is not stable-σ-c-UNDEC w.r.t. $\mathcal{I}$ (a) then there is some completion $AF^* = \langle \mathcal{A}^*, \mathcal{C}^* \rangle$ of $\mathcal{I}$ in which A is not σ-c-UNDEC. Next, we construct three argumentation frameworks based on AF^*, containing the argument A'', and discuss its status.

   - First, construct $AF_1 = \langle \mathcal{A}^* \cup \{A', A''\}, \mathcal{C}^* \cup \{(A, A'), (A', A'')\} \rangle$. Given that A'' is attacked by A', which is only attacked by A in AF^*, A'' cannot be σ-c-UNDEC in AF_1.
   - Construct $AF_2 = \langle \mathcal{A}^* \cup \{A', A'', U\}, \mathcal{C}^* \cup \{(A, A'), (A', A''), (U, A'), (U, A''), (U, U)\} \rangle$. A'' is σ-c-UNDEC in AF_2, since the self-attacking argument U attacks each attacker of A'' (i.e. A' and U itself).
   - Next, let $AF_3 = \langle \mathcal{A}^* \cup \{A', A'', U, U'\}, \mathcal{C}^* \cup \{(A, A'), (A', A''), (U, A'), (U, A''), (U, U), (U', U)\} \rangle$. Given that A is not σ-c-UNDEC in AF^*, A cannot be σ-c-UNDEC in AF_3 either. Since the argument A'' in AF_3 is, apart from the argument U that is definitely σ-c-OUT, attacked by A', which is only attacked by A, A'' cannot be σ-c-UNDEC in AF_3.

   Now item *(b)* (addition of U is σ-c-UNDEC-relevant for A'' w.r.t. $\mathcal{I}'$) follows from Lemma 2.7, the fact that the incomplete argumentation framework $\langle \mathcal{A}^* \cup \{A', A''\}, \{U\}, \mathcal{C}^* \cup \{(A, A'), (A', A''), (U, A'), (U, A''), (U, U)\}, \emptyset \rangle$ is in $\texttt{part}(\mathcal{I}')$ and the status of A'' in AF_1 and AF_2.

Similarly, item *(c)* (removal of U' is σ-c-UNDEC-relevant for A'' w.r.t. $\mathcal{I}''$) follows from Lemma 2.7, the fact that the incomplete argumentation framework $\langle \mathcal{A}^* \cup \{A', A'', U\}, \{U'\}, \mathcal{C}^* \cup \{(A, A'), (A', A''), (U, A'), (U, A''), (U, U), (U', U)\}, \emptyset \rangle$ is in $\texttt{part}(\mathcal{I}'')$ and the status of A'' in AF_2 and AF_3.

(b) ⇒ (a) Suppose that addition of U is σ-c-UNDEC-relevant for A'' w.r.t. $\mathcal{I}'$. Then by Lemma 2.7, there is some $\mathcal{I}^{*\prime}$ in $\texttt{part}(\mathcal{I}')$ such that A'' is *not* σ-c-UNDEC in the certain projection of $\mathcal{I}^{*\prime}$ – let us call this certain projection $AF^{*\prime} = \langle \mathcal{A}^{*\prime}, \mathcal{C}^{*\prime} \rangle$. Then A cannot be σ-c-UNDEC in $AF^{*\prime}$ either. Now construct $AF' = \langle \mathcal{A}', \mathcal{C}' \rangle$ where $\mathcal{A}' = \mathcal{A}^{*\prime} \setminus \{A', A'', U\}$ and $\mathcal{C}' = \mathcal{C}^{*\prime} \setminus \{(A, A'), (A', A''), (U, A'), (U, A''), (U, U)\}$. Since A is not σ-c-UNDEC in $AF^{*\prime}$, it cannot be σ-c-UNDEC in AF'. Given that AF' is a completion of $\mathcal{I}$, A cannot be stable-σ-c-UNDEC w.r.t. $\mathcal{I}$.

(c) ⇒ (a) If removal of U' is σ-c-UNDEC-relevant for A'' w.r.t. $\mathcal{I}''$ then by Lemma 2.7, there is some $\mathcal{I}^{*\prime\prime}$ in $\texttt{part}(\mathcal{I}'')$ such that A'' is *not* σ-c-UNDEC in the certain projection of $\mathcal{I}^{*\prime\prime}$ – let this certain projection be $AF^{*\prime\prime} = \langle \mathcal{A}^{*\prime\prime}, \mathcal{C}^{*\prime\prime} \rangle$. Now construct $AF' = \langle \mathcal{A}', \mathcal{C}' \rangle$ where $\mathcal{A}' = \mathcal{A}^{*\prime\prime} \setminus \{A', A'', U, U'\}$ and $\mathcal{C}' = \mathcal{C}^{*\prime\prime} \setminus \{(A, A'), (A', A''), (U, A'), (U, A''), (U, U), (U', U)\}$; since A'' is not σ-c-UNDEC in $AF^{*\prime}$, it cannot be that A is σ-c-UNDEC in AF'. Given that AF' is a completion of $\mathcal{I}$, A cannot be stable-σ-c-UNDEC w.r.t. $\mathcal{I}$. □

Proposition 2.12. *The following problems are NP-complete:*

1. *CP-credulous-UNDEC-RELEVANCE;*
2. *CP-sceptical-IN-RELEVANCE;*
3. *CP-sceptical-OUT-RELEVANCE;*
4. *GR-credulous-IN-RELEVANCE;*
5. *GR-credulous-OUT-RELEVANCE;*
6. *GR-credulous-UNDEC-RELEVANCE;*
7. *GR-sceptical-IN-RELEVANCE;*
8. *GR-sceptical-OUT-RELEVANCE; and*
9. *GR-sceptical-UNDEC-RELEVANCE.*

Proof. For each of these problems, membership in NP follows from membership in P of the corresponding JUSTIFICATION problems (Dung, 1995), listed in Table 2.1, and Proposition 2.11 (since $\text{NP} = \text{NP}^{\text{P}}$).

For NP-hardness, we can give a reduction from the complementary of the corresponding STABILITY problem, using Lemma 2.8.

1. By Proposition 2.6, CP-credulous-UNDEC-STABILITY is CoNP-complete, which means that the complementary problem CP-credulous-UNDEC-INSTABILITY is

NP-complete. By Lemma 2.8 item 3, each instance $I = \langle \mathcal{I}, A \rangle$ of CP-credulous-UNDEC-INSTABILITY can be transformed into an instance $I' = \langle \mathcal{I}', A'', U \rangle$ (for addition) or $I' = \langle \mathcal{I}'', A'', U' \rangle$ (for removal) such that I is a positive instance of CP-credulous-UNDEC-INSTABILITY iff I' is a positive instance of CP-credulous-UNDEC-RELEVANCE.

2. The problem CP-sceptical-IN-STABILITY is CoNP-complete (Baumeister et al., 2021), hence CP-sceptical-IN-INSTABILITY is NP-complete. By Lemma 2.8 item 1, each instance $I = \langle \mathcal{I}, A \rangle$ of CP-sceptical-IN-INSTABILITY can be transformed into an instance $I' = \langle \mathcal{I}', A'', U \rangle$ (for addition) or $I' = \langle \mathcal{I}'', A'', U' \rangle$ (for removal) such that I is a positive instance of CP-sceptical-IN-INSTABILITY iff I' is a positive instance of CP-sceptical-IN-RELEVANCE.

3. The problem CP-sceptical-OUT-STABILITY is CoNP-complete by Baumeister et al. (2021) in combination with Lemma 2.5, hence CP-sceptical-OUT-INSTABILITY is NP-complete. By Lemma 2.8 item 2, each instance $I = \langle \mathcal{I}, A \rangle$ of CP-sceptical-OUT-INSTABILITY can be transformed into an instance $I' = \langle \mathcal{I}', A, U \rangle$ (for addition) or $I' = \langle \mathcal{I}'', A, U' \rangle$ (for removal) such that I is a positive instance of CP-sceptical-OUT-INSTABILITY iff I' is a positive instance of CP-sceptical-OUT-RELEVANCE.

4. The problem GR-credulous-IN-STABILITY is CoNP-complete (Baumeister et al., 2021), hence GR-credulous-IN-INSTABILITY is NP-complete. By Lemma 2.8 item 1, each instance $I = \langle \mathcal{I}, A \rangle$ of GR-credulous-IN-INSTABILITY can be transformed into an instance $I' = \langle \mathcal{I}', A'', U \rangle$ (for addition) or $I' = \langle \mathcal{I}'', A'', U' \rangle$ (for removal) such that I is a positive instance of GR-credulous-IN-INSTABILITY iff I' is a positive instance of GR-credulous-IN-RELEVANCE.

5. The problem GR-credulous-OUT-STABILITY is CoNP-complete by Baumeister et al. (2021) in combination with Lemma 2.5, so GR-credulous-OUT-INSTABILITY is NP-complete. By Lemma 2.8 item 2, each instance $I = \langle \mathcal{I}, A \rangle$ of CP-sceptical-OUT-INSTABILITY can be transformed into an instance $I' = \langle \mathcal{I}', A, U \rangle$ (for addition) or $I' = \langle \mathcal{I}'', A, U' \rangle$ (for removal) such that I is a positive instance of GR-credulous-OUT-INSTABILITY iff I' is a positive instance of GR-credulous-OUT-RELEVANCE.

6. By Proposition 2.6, GR-credulous-UNDEC-STABILITY is CoNP-complete, which means that the complementary problem GR-credulous-UNDEC-INSTABILITY is NP-complete. By Lemma 2.8 item 3, each instance $I = \langle \mathcal{I}, A \rangle$ of GR-credulous-UNDEC-INSTABILITY can be transformed into an instance $I' = \langle \mathcal{I}', A'', U \rangle$ (for addition) or $I' = \langle \mathcal{I}'', A'', U' \rangle$ (for removal) such that I is a positive instance of GR-credulous-UNDEC-INSTABILITY iff I' is a positive instance of GR-credulous-UNDEC-RELEVANCE.

7. The problem GR-sceptical-IN-STABILITY is CoNP-complete (Baumeister et al., 2021), thus GR-sceptical-IN-INSTABILITY is NP-complete. By Lemma 2.8 item 1, each instance $I = \langle \mathcal{I}, A \rangle$ of GR-sceptical-IN-INSTABILITY can be transformed into an instance $I' = \langle \mathcal{I}', A'', U \rangle$ (for addition) or $I' = \langle \mathcal{I}'', A'', U' \rangle$ (for removal) such that I is a positive instance of GR-sceptical-IN-INSTABILITY iff I'

is a positive instance of GR-sceptical-IN-RELEVANCE.

8. The problem GR-sceptical-OUT-STABILITY is CoNP-complete by Baumeister et al. (2021) in combination with Lemma 2.5, hence GR-sceptical-OUT-INSTABILITY is NP-complete. By Lemma 2.8 item 2, each instance $I = \langle \mathcal{I}, A \rangle$ of GR-sceptical-OUT-INSTABILITY can be transformed into an instance $I' = \langle \mathcal{I}', A, U \rangle$ (for addition) or $I' = \langle \mathcal{I}'', A, U' \rangle$ (for removal) such that I is a positive instance of GR-sceptical-OUT-INSTABILITY iff I' is a positive instance of GR-sceptical-OUT-RELEVANCE.
9. By Proposition 2.6, GR-sceptical-UNDEC-STABILITY is CoNP-complete, which means that the complementary problem GR-sceptical-UNDEC-INSTABILITY is NP-complete. By Lemma 2.8 item 3, each instance $I = \langle \mathcal{I}, A \rangle$ of GR-sceptical-UNDEC-INSTABILITY can be transformed into an instance $I' = \langle \mathcal{I}', A'', U \rangle$ (for addition) or $I' = \langle \mathcal{I}'', A'', U' \rangle$ (for removal) such that I is a positive instance of GR-sceptical-UNDEC-INSTABILITY iff I' is a positive instance of GR-sceptical-UNDEC-RELEVANCE. □

Lemma 2.9. *Given an incomplete argumentation framework $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$, a certain argument $A \in \mathcal{A}$, semantics $\sigma \in \{GR, CP, PR\}$:*

1. *Construct $\mathcal{I}'$ and $\mathcal{I}''$ as follows, where A', U and U' are fresh arguments not in $\mathcal{A} \cup \mathcal{A}^?$:*
   - $\mathcal{A}' = \mathcal{A} \cup \{A'\}$;
   - $\mathcal{C}' = \mathcal{C} \cup \{(A, A'), (A', A'), (U, A')\}$;
   - $\mathcal{I}' = \langle \mathcal{A}', \mathcal{A}^? \cup \{U\}, \mathcal{C}', \mathcal{C}^? \rangle$; *and*
   - $\mathcal{I}'' = \langle \mathcal{A}' \cup \{U\}, \mathcal{A}^? \cup \{U'\}, \mathcal{C}' \cup \{(U', U)\}, \mathcal{C}^? \rangle$.

   The following items are equivalent:
   (a) *A is **not** stable-σ-credulous-IN w.r.t. $\mathcal{I}$; and*
   (b) *addition of U' is σ-sceptical-UNDEC-relevant for A' w.r.t. $\mathcal{I}''$; and*
   (c) *removal of U is σ-sceptical-UNDEC-relevant for A' w.r.t. $\mathcal{I}'$.*
2. *Construct $\mathcal{I}'$ and $\mathcal{I}''$ as follows, where A_1, A_2, A_3, A_4, U and U' are fresh arguments not in $\mathcal{A} \cup \mathcal{A}^?$:*
   - $\mathcal{A}' = \mathcal{A} \cup \{A_1, A_2, A_3, A_4\}$;
   - $\mathcal{C}' = \mathcal{C} \cup \{(A, A_1), (A, A_2), (A_1, A_2), (A_2, A_3), (A_3, A_4), (U, U), (U, A_3)\}$;
   - $\mathcal{I}' = \langle \mathcal{A}', \mathcal{A}^? \cup \{U\}, \mathcal{C}', \mathcal{C}^? \rangle$; *and*
   - $\mathcal{I}'' = \langle \mathcal{A}' \cup \{U\}, \mathcal{A}^? \cup \{U'\}, \mathcal{C}' \cup \{(U', U)\}, \mathcal{C}^? \rangle$.

   The following items are equivalent:
   (a) *A is **not** stable-σ-credulous-UNDEC w.r.t. $\mathcal{I}$; and*
   (b) *addition of U' is σ-sceptical-IN-relevant for A_3 w.r.t. $\mathcal{I}''$; and*

(c) removal of U is σ-sceptical-IN-relevant for A_3 w.r.t. $\mathcal{I}'$; and

(d) addition of U' is σ-sceptical-OUT-relevant for A_4 w.r.t. $\mathcal{I}''$; and

(e) removal of U is σ-sceptical-OUT-relevant for A_4 w.r.t. $\mathcal{I}'$.

Proof. We prove these two items separately.

1. Let $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ be an IAF, $A \in \mathcal{A}$ be a certain argument and σ some semantics in $\{\text{GR}, \text{CP}, \text{PR}\}$. Construct $\mathcal{I}'$ and $\mathcal{I}''$ as specified above (see also Figure A.4).

   (a) $\Rightarrow$ ***(b) and (c)*** Suppose that A is *not* stable-σ-credulous-IN w.r.t. $\mathcal{I}$. Then by Definition 2.7 of stability there is some completion $AF^* = \langle \mathcal{A}^*, \mathcal{C}^* \rangle$ of $\mathcal{I}$ such that A is not in any σ-extension. Next, construct and consider the following three AFs:

   - Let $AF_1 = \langle \mathcal{A}^* \cup \{A'\}, \mathcal{C}^* \cup \{(A, A'), (A', A')\} \rangle$. Given that A is not in any σ-extension of $\langle \mathcal{A}^*, \mathcal{C}^* \rangle$ and A' is self-attacking, A' cannot be in, nor attacked by any argument in any σ-extension of AF_1. Consequently, A' is σ-sceptical-UNDEC in AF_1.
   - Now consider $AF_2 = \langle \mathcal{A}^* \cup \{A', U\}, \mathcal{C}^* \cup \{(A, A'), (A', A'), (U, A')\} \rangle$. Since A' is attacked by the unattacked argument U, A' is attacked by an argument in some σ-extension of AF_2 and therefore *not* σ-sceptical-UNDEC in AF_2.
   - Construct $AF_3 = \langle \mathcal{A}^* \cup \{A', U, U'\}, \mathcal{C}^* \cup \{(A, A'), (A', A'), (U, A'), (U', U)\} \rangle$. Note that U is attacked by the unattacked argument U', which means that U' is in each σ-extension of AF_3. Consequently, none of the arguments attacking A' (A, U and A' itself) is in any σ-extension of AF_3. In addition, A' cannot be in any σ-extension of AF_3 as it is self-attacking. This implies that A' is σ-sceptical-UNDEC in AF_3.

   We continue proving item *(b)*. Note that the partial completion $\langle \mathcal{A}^* \cup \{A', U\}, \{U'\}, \mathcal{C}^* \cup \{(A, A'), (A', A'), (U, A'), (U', U)\}, \emptyset \rangle$ is in $\texttt{part}(\mathcal{I}'')$. Then by Lemma 2.7 and the status of A' in AF_2 and AF_3, addition of U' is σ-sceptical-UNDEC-relevant for A' w.r.t. $\mathcal{I}'$.

   Item *(c)* follows from the fact that $\langle \mathcal{A}^* \cup \{A'\}, \{U\}, \mathcal{C}^* \cup \{(A, A'), (A', A'), (U, A')\}, \emptyset \rangle$ is in $\texttt{part}(\mathcal{I}')$, Lemma 2.7 and the status of A' in AF_1 and AF_2.

   (b) $\Rightarrow$ ***(a)*** Suppose that addition of U' is σ-sceptical-UNDEC-relevant for A' w.r.t. $\mathcal{I}''$. Then by Lemma 2.7, there is some $\mathcal{I}^*$ in $\texttt{part}(\mathcal{I}'')$ such that A' is σ-sceptical-UNDEC w.r.t. its certain projection $AF^* = \langle \mathcal{A}^*, \mathcal{C}^* \rangle$. This implies that A is not in any σ-extension of AF^* (otherwise, A' would be out in that extension). Now construct $AF' = \langle \mathcal{A}', \mathcal{C}' \rangle$ where $\mathcal{A}' = \mathcal{A}^* \setminus \{A', U, U'\}$ and $\mathcal{C}' = \mathcal{C}^* \setminus \{(A, A'), (A', A'), (U, A'), (U', U)\}$. Since A' is not in any σ-extension of AF^*, it cannot be in any σ-extension

of AF' either (because all arguments defending A in $\mathcal{A}^*$ are also in $\mathcal{A}'$). Therefore, given that AF' is a completion of $\mathcal{I}$, A is not stable-σ-credulous-IN w.r.t. $\mathcal{I}$.

(c)* $\Rightarrow$ *(a) Finally, suppose that removal of U is σ-sceptical-UNDEC-relevant for A' w.r.t. $\mathcal{I}'$. Then by Lemma 2.7, there is some $\mathcal{I}^*$ in $\texttt{part}(\mathcal{I}')$ such that A' is σ-sceptical-UNDEC w.r.t. its certain projection $AF^* = \langle \mathcal{A}^*, \mathcal{C}^* \rangle$. Consequently, A cannot be in any σ-extension of AF^*. Construct $AF' = \langle \mathcal{A}', \mathcal{C}' \rangle$ where $\mathcal{A}' = \mathcal{A}^* \setminus \{A', U\}$ and $\mathcal{C}' = \mathcal{C}^* \setminus \{(A, A'), (A', A'), (U, A')\}$ and note that A cannot be in any σ-extension of AF either. Given that AF' is a completion of $\mathcal{I}$, A cannot be stable-σ-credulous-IN w.r.t. $\mathcal{I}$.

2. Let $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ be an IAF, $A \in \mathcal{A}$ be a certain argument and σ some semantics in $\{\text{GR}, \text{CP}, \text{PR}\}$. Construct $\mathcal{I}'$ and $\mathcal{I}''$ as specified above (see also Figure A.5).

   (a)* $\Rightarrow$ *(b), (c), (d) and (e) If A is *not* stable-σ-credulous-UNDEC w.r.t. $\mathcal{I}$, then there is some completion $AF^* = \langle \mathcal{A}^*, \mathcal{C}^* \rangle$ of $\mathcal{I}$ such that for each σ-extension, A is either in, or attacked by some argument in this extension. Next, construct and consider the following three AFs:

   - Let $AF_1 = \langle \mathcal{A}_1, \mathcal{C}_1 \rangle$ where $\mathcal{A}_1 = \mathcal{A}^* \cup \{A_1, A_2, A_3, A_4\}$ and $\mathcal{C}_1 = \mathcal{C}^* \cup \{(A, A_1), (A, A_2), (A_1, A_2), (A_2, A_3), (A_3, A_4)\}$. Consider an arbitrary σ-extension S of AF_1 and let $S' = S \setminus \{A_1, A_2, A_3, A_4\}$. Note that S' must be a σ-extension of AF^*. Recall that A is either in, or attacked by an argument in S'. This implies that A is either in S or attacked by some argument in S. In both cases, A_2 is attacked by an argument in S (either A or A_1), which by completeness of σ semantics implies that $A_3 \in S$, while A_4 is attacked by an argument (A_3) in S. Thus A_3 is σ-sceptical-IN in AF_1 and A_4 is σ-sceptical-OUT in AF_1.
   - Now consider $AF_2 = \langle \mathcal{A}_2, \mathcal{C}_2 \rangle$ where $\mathcal{A}_2 = \mathcal{A}^* \cup \{A_1, A_2, A_3, A_4, U\}$ and $\mathcal{C}_2 = \mathcal{C}^* \cup \{(A, A_1), (A, A_2), (A_1, A_2), (A_2, A_3), (A_3, A_4), (U, U), (U, A_3)\}$. Since AF_2 has at least one extension under σ semantics, there is some σ extension S. Given that A_3 is attacked by the self-attacking argument U, which is not attacked by any other argument, A_3 cannot be in S and A_4 is not attacked by any argument in S. This implies that A_3 is *not* σ-sceptical-IN in AF_2 and A_4 is *not* σ-sceptical-OUT in AF_2.
   - Finally construct $AF_3 = \langle \mathcal{A}_3, \mathcal{C}_3 \rangle$ where $\mathcal{A}_3 = \mathcal{A}^* \cup \{A_1, A_2, A_3, A_4, U, U'\}$ and $\mathcal{C}_3 = \mathcal{C}^* \cup \{(A, A_1), (A, A_2), (A_1, A_2), (A_2, A_3), (A_3, A_4), (U, U), (U, A_3), (U', U)\}$. Consider an arbitrary σ-extension S of AF_3 and let $S' = S \setminus \{A_1, A_2, A_3, A_4, U, U'\}$. Note that S' must be a σ-extension of AF^*. Given that A is in S' or attacked by an argument in S', it must be that either A is in S and A_1 is attacked by an argument in S or A is attacked by an argument in S and A_1 is in S. In both cases, A_2 is attacked by an argument in S. Given that U'

is unattacked as well, U is also attacked by an argument in S. This implies that A_3 is in S and A_4 is attacked by an argument in S. In other words, A_3 is σ-sceptical-IN in AF_3 and A_4 is σ-sceptical-OUT in AF_3.

We continue proving items *(b)* and *(d)*. Note that the partial completion $\langle \mathcal{A}_2, \{U'\}, \mathcal{C}_3, \emptyset \rangle$ is in $\texttt{part}(\mathcal{I}'')$. Then by Lemma 2.7 and the status of A_3 in AF_2 and AF_3, addition of U' is σ-sceptical-IN-relevant for A_3 w.r.t. $\mathcal{I}''$ and σ-sceptical-OUT-relevant for A_4 w.r.t. $\mathcal{I}''$.

Items *(c)* and *(e)* follow from the fact that $\langle \mathcal{A}_1, \{U\}, \mathcal{C}_2, \emptyset \rangle$ is in $\texttt{part}(\mathcal{I}')$, Lemma 2.7 and the justification statuses of A_3 and A_4 in AF_1 and AF_2.

(d) $\Rightarrow$ ***(b)*** If addition of U' is σ-sceptical-out-relevant for A_4 w.r.t. $\mathcal{I}''$ then by Lemma 2.7 there exists some $\mathcal{I}^* = \langle \mathcal{A}', \{U\}, \mathcal{C}', \emptyset \rangle \in \texttt{part}(\mathcal{I}'')$ such that A_4 is *not* σ-sceptical-out-relevant in the certain projection of $\mathcal{I}^*$, while A_4 is σ-sceptical-out-relevant in the certain projection of $\langle \mathcal{A}' \cup \{U\}, \emptyset, \mathcal{C}', \emptyset \rangle$. Given that A_4 is (only) attacked by A_3, A_3 is *not* σ-sceptical-in-relevant in the certain projection of $\mathcal{I}^*$, while A_3 is σ-sceptical-in-relevant in the certain projection of $\langle \mathcal{A}' \cup \{U\}, \emptyset, \mathcal{C}', \emptyset \rangle$. By Lemma 2.7, addition of U' is σ-sceptical-in-relevant for A_3 w.r.t. $\mathcal{I}''$.

(e) $\Rightarrow$ ***(c)*** Similarly, if removal of U is σ-sceptical-out-relevant for A_4 w.r.t. $\mathcal{I}'$ then by Lemma 2.7 removal of U is σ-sceptical-in-relevant for A_3 w.r.t. $\mathcal{I}'$.

(b) $\Rightarrow$ ***(a)*** Suppose that addition of U' is σ-sceptical-IN-relevant for A_3 w.r.t. $\mathcal{I}''$. Then by Lemma 2.7, there is some $\mathcal{I}^*$ in $\texttt{part}(\mathcal{I}'')$ such that A_3 is σ-sceptical-IN w.r.t. its certain projection $AF^* = \langle \mathcal{A}^*, \mathcal{C}^* \rangle$. Construct $AF' = \langle \mathcal{A}', \mathcal{C}' \rangle$ where $\mathcal{A}' = \mathcal{A}^* \setminus \{A_1, A_2, A_3, A_4, U, U'\}$ and $\mathcal{C}' = \mathcal{C}^* \setminus \{(A, A_1), (A, A_2), (A_1, A_2), (A_2, A_3), (A_3, A_4), (U, U), (U, A_3), (U', U)\}$.

Suppose, towards a contradiction, that A is stable-σ-credulous-UNDEC w.r.t. $\mathcal{I}$. Then there would be some σ extension S' of AF' such that $A \notin S'$ and A is not attacked by any argument in S' (since AF' is a completion of $\mathcal{I}$). Reconsidering AF^*, the set $S = S' \cup \{U'\}$ would be a σ extension of AF^*. Note that A_3 is not in S, which contradicts our assumption that A_3 is σ-sceptical-IN w.r.t. AF^*. Hence A is not stable-σ-credulous-UNDEC w.r.t. $\mathcal{I}$.

(c) $\Rightarrow$ ***(a)*** Finally, suppose that removal of U is σ-sceptical-UNDEC-relevant for A_3 w.r.t. $\mathcal{I}'$. Then by Lemma 2.7, there is some $\mathcal{I}^*$ in $\texttt{part}(\mathcal{I}')$ such that A_3 is σ-sceptical-UNDEC w.r.t. its certain projection $AF^* = \langle \mathcal{A}^*, \mathcal{C}^* \rangle$. Construct $AF' = \langle \mathcal{A}', \mathcal{C}' \rangle$ where $\mathcal{A}' = \mathcal{A}^* \setminus \{A_1, A_2, A_3, A_4, U\}$ and $\mathcal{C}' = \mathcal{C}^* \setminus \{(A, A_1), (A, A_2), (A_1, A_2), (A_2, A_3), (A_3, A_4), (U, U), (U, A_3)\}$. If A would be stable-σ-credulous-UNDEC w.r.t. $\mathcal{I}$, then there would be some σ extension S' of AF' such that $A \notin S'$ and A is not attacked by any argument in S' (since AF' is a completion of $\mathcal{I}$), thus $S = S' \cup \{U'\}$ would be a σ extension of AF^* not containing A_3. Since this contradicts

our earlier assumption, A cannot be in any σ-extension of AF^*. □

Lemma 2.10. *Let (ϕ, X, Y) be an instance of Σ_2-SAT and let $\phi = \bigwedge_i c_i$ and $c_i = \bigvee_j \alpha_j$ for each clause c_i in ϕ, where α_j are the literals over $X \cup Y$ that occur in clause c_i. Now let $\mathcal{I}_1 = T_1(\phi, X, Y)$ and let $\mathcal{I}_2 = T_2(\phi, X, Y)$, using the transformations T_1 and T_2 specified in Definition 2.12. The following items are equivalent:*

1. *(ϕ, X, Y) is a positive instance of Σ_2-SAT;*
2. *Removal of χ is ST-sceptical-IN-relevant for $\overline{\phi}$ w.r.t. $\mathcal{I}_1$;*
3. *Addition of χ is ST-credulous-IN-relevant for ϕ w.r.t. $\mathcal{I}_1$;*
4. *Addition of $\overline{\chi}$ is ST-sceptical-IN-relevant for $\overline{\phi}$ w.r.t. $\mathcal{I}_2$;*
5. *Removal of $\overline{\chi}$ is ST-credulous-IN-relevant for ϕ w.r.t. $\mathcal{I}_2$;*
6. *Removal of χ is ST-sceptical-OUT-relevant for ϕ w.r.t. $\mathcal{I}_1$;*
7. *Addition of χ is ST-credulous-OUT-relevant for $\overline{\phi}$ w.r.t. $\mathcal{I}_1$;*
8. *Addition of $\overline{\chi}$ is ST-sceptical-OUT-relevant for ϕ w.r.t. $\mathcal{I}_2$; and*
9. *Removal of $\overline{\chi}$ is ST-credulous-OUT-relevant for $\overline{\phi}$ w.r.t. $\mathcal{I}_2$.*

Proof. We introduce an auxiliary statement, for which we prove that it equals all of the items above:

(0) There is some $\mathcal{I}^* \in \texttt{part}(\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \emptyset \rangle)$ such that $\overline{\phi}$ is ST-sceptical-IN in its certain projection AF^* (where $\mathcal{A}$, $\mathcal{A}^?$ and $\mathcal{C}$ are chosen as in Definition 2.12).

Using this additional item, we prove these items separately.

(0) $\Rightarrow$ ***(1)*** Suppose that there is some $\mathcal{I}^* \in \texttt{part}(\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \emptyset \rangle)$ such that $\overline{\phi}$ is ST-sceptical-IN in the certain projection $AF^* = \langle \mathcal{A}^*, \mathcal{C}^* \rangle$ of $\mathcal{I}^*$. Let τ_X be an assignment to variables in X such that it assigns True to all $x_i \in X$ such that $g_i \in \mathcal{A}^*$ and False otherwise. Let τ_Y be an arbitrary assignment to all variables in Y. Given that $\overline{\phi}$ is ST-sceptical-IN in AF^*, for each ST extension S of AF^*, at least one of the arguments $\overline{c_i}$ must have been in S, so there is at least one clause in the formula Φ for which none of the variables was assigned True by τ_X and τ_Y. Since we chose τ_Y arbitrarily, we have that (ϕ, X, Y) is a positive instance of Σ_2-SAT.

(1) $\Rightarrow$ ***(0)*** Let (ϕ, X, Y) be a positive instance of Σ_2-SAT. Then there is some assignment τ_X to all variables of X such that for each assignment τ_Y to the variables of Y, $\Phi[\tau_X, \tau_Y]$ is False. Let $G = \{g_i \mid x_i \in X \text{ and } x_i \text{ is assigned True by } \tau_X\}$. Construct $\mathcal{I}^* = \langle \mathcal{A} \cup G, \emptyset, \mathcal{C}|_{\mathcal{A} \cup G}, \emptyset \rangle$ and let AF^* be its certain projection. Note that $\mathcal{I}^* \in \texttt{part}(\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \emptyset \rangle)$. Given that all arguments in G are unattacked, each ST extension of AF^* contains all arguments in G. Furthermore, for each argument $x \in X$, each ST extension of AF^* contains either x (if x is assigned True by τ_X) or $\overline{y}$ (if x is assigned False by τ_Y). Additionally, for each argument $y \in Y$, each ST extension of AF^* contains either y or $\overline{y}$.

Given that for each assignment τ_Y to the variables of Y, $\phi[\tau_X, \tau_Y]$ is False, it must be that for each ST extension S of this AF, at least one of the clause arguments $\overline{c_i}$ is in S, so ϕ' is attacked by an argument in S; therefore $\overline{\phi} \in S$. Thus, $\overline{\phi}$ is ST-sceptical-IN in AF^*.

(0) $\Rightarrow$ ***(2)*** Suppose that for some $\mathcal{I}^* = \langle \mathcal{A}^*, \mathcal{A}^{?*}, \mathcal{C}|_{\mathcal{A}^* \cup \mathcal{A}^{?*}}, \emptyset \rangle$ in $\texttt{part}(\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \emptyset \rangle)$ $\overline{\phi}$ is ST-sceptical-IN in its certain projection AF^*. This implies that for $\mathcal{I}' = \langle \mathcal{A}^*, \{\chi\}, \mathcal{C}|_{\mathcal{A}^* \cup \{\chi\}}, \emptyset \rangle$ it also holds that $\overline{\phi}$ is ST-sceptical-IN in its certain projection $AF' = AF^* = \langle \mathcal{A}^*, \mathcal{C}|_{\mathcal{A}^*} \rangle$. Note that $\mathcal{I}' \in \texttt{part}(\mathcal{I}_1)$. Now consider $\mathcal{I}'' = \langle \mathcal{A}^* \cup \{\chi\}, \emptyset, \mathcal{C}|_{\mathcal{A}^* \cup \{\chi\}}, \emptyset \rangle$ and its certain projection AF''. In AF'', the argument $\overline{\phi}$ is attacked by the unattacked argument χ, so $\overline{\phi}$ is not ST-sceptical-IN in AF''. Then by Lemma 2.7 item 3, removal of χ is ST-sceptical-IN-relevant for $\overline{\phi}$ w.r.t. $\mathcal{I}_1$.

(2) $\Rightarrow$ ***(0)*** Suppose that removal of χ is ST-sceptical-IN-relevant for $\overline{\phi}$ w.r.t. $\mathcal{I}_1$. Then by Lemma 2.7 item 3 there is some $\mathcal{I}' = \langle \mathcal{A}^*, \emptyset, \mathcal{C}|_{\mathcal{A}^*}, \emptyset \rangle$ in $\mathcal{I}_1$ (where $\chi \notin \mathcal{A}^*$, so $\mathcal{I}' \in \texttt{part}(\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \emptyset \rangle)$) such that $\overline{\phi}$ is ST-sceptical-IN in $\texttt{cert}(\mathcal{I}')$.

(0) $\Rightarrow$ ***(3)*** Suppose that for some $\mathcal{I}^* = \langle \mathcal{A}^*, \mathcal{A}^{?*}, \mathcal{C}|_{\mathcal{A}^* \cup \mathcal{A}^{?*}}, \emptyset \rangle$ in $\texttt{part}(\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \emptyset \rangle)$ $\overline{\phi}$ is ST-sceptical-IN in its certain projection AF^*. Then ϕ is ST-sceptical-OUT and therefore not ST-credulous-IN in AF^*. This implies that for $\mathcal{I}' = \langle \mathcal{A}^*, \{\chi\}, \mathcal{C}|_{\mathcal{A}^* \cup \{\chi\}}, \emptyset \rangle$ it also holds that ϕ is not ST-credulous-IN in its certain projection $AF' = AF^* = \langle \mathcal{A}^*, \mathcal{C}|_{\mathcal{A}^*} \rangle$. Note that $\mathcal{I}' \in \texttt{part}(\mathcal{I}_1)$. Now consider $\mathcal{I}'' = \langle \mathcal{A}^* \cup \{\chi\}, \emptyset, \mathcal{C}|_{\mathcal{A}^* \cup \{\chi\}}, \emptyset \rangle$ and its certain projection AF''. In AF'', the argument $\overline{\phi}$ is attacked by the unattacked argument χ, so $\overline{\phi}$ is not ST-sceptical-IN in AF'', so ϕ is ST-credulous-IN in AF''. Then by Lemma 2.7 item 1, addition of χ is ST-credulous-IN-relevant for ϕ w.r.t. $\mathcal{I}_1$.

(3) $\Rightarrow$ ***(0)*** Suppose that addition of χ is ST-credulous-IN-relevant for ϕ w.r.t. $\mathcal{I}_1$. Then by Lemma 2.7 item 1 there is some $\mathcal{I}' = \langle \mathcal{A}^*, \{\chi\}, \mathcal{C}|_{\mathcal{A}^* \cup \{\chi\}}, \emptyset \rangle$ such that ϕ is not ST-credulous-IN in $\texttt{cert}(\mathcal{I}')$, so ϕ is ST-sceptical-OUT in $\texttt{cert}(\mathcal{I}')$, which implies that $\overline{\phi}$ is ST-sceptical-IN in $\texttt{cert}(\mathcal{I}')$.

(0) $\Rightarrow$ ***(4)*** Suppose that for some $\mathcal{I}^* = \langle \mathcal{A}^*, \mathcal{A}^{?*}, \mathcal{C}|_{\mathcal{A}^* \cup \mathcal{A}^{?*}}, \emptyset \rangle$ in $\texttt{part}(\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \emptyset \rangle)$ $\overline{\phi}$ is ST-sceptical-IN in its certain projection AF^*. This implies that for $\mathcal{I}' = \langle \mathcal{A}^* \cup \{\chi, \overline{\chi}\}, \emptyset, (\mathcal{C} \cup \{(\overline{\chi}, \chi)\})|_{\mathcal{A}^* \cup \{\overline{\chi}, \chi\}}, \emptyset \rangle$ it also holds that $\overline{\phi}$ is ST-sceptical-IN in its certain projection AF'. Note that $\mathcal{I}' \in \texttt{part}(\mathcal{I}_2)$. Now consider $\mathcal{I}'' = \langle \mathcal{A}^* \cup \{\chi\}, \{\overline{\chi}\}, (\mathcal{C} \cup \{(\overline{\chi}, \chi)\})|_{\mathcal{A}^* \cup \{\overline{\chi}, \chi\}}, \emptyset \rangle$ and its certain projection AF''. In AF'', the argument $\overline{\phi}$ is attacked by the unattacked argument χ, so $\overline{\phi}$ is not ST-sceptical-IN in AF''. Then by Lemma 2.7 item 1, addition of $\overline{\chi}$ is ST-sceptical-IN-relevant for $\overline{\phi}$ w.r.t. $\mathcal{I}_2$.

(4) $\Rightarrow$ ***(0)*** Suppose that addition of $\overline{\chi}$ is ST-sceptical-IN-relevant for $\overline{\phi}$ w.r.t. $\mathcal{I}_2$. By Lemma 2.7 item 1 there is an $\mathcal{I}' = \langle \mathcal{A}^* \cup \{\chi, \overline{\chi}\}, \emptyset, (\mathcal{C} \cup \{(\overline{\chi}, \chi)\})|_{\mathcal{A}^* \cup \{\overline{\chi}, \chi\}}, \emptyset \rangle$ in $\texttt{part}(\mathcal{I}_2)$ such that $\overline{\phi}$ is ST-sceptical-IN in $\texttt{cert}(\mathcal{I}')$. Then $\overline{\phi}$ would also be ST-sceptical-IN in $\texttt{cert}(\mathcal{I}'')$ where $\mathcal{I}'' = \langle \mathcal{A}^*, \emptyset, \mathcal{C}|_{\mathcal{A}^*}, \emptyset \rangle$ is in $\texttt{part}(\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \emptyset \rangle)$.

(0) $\Rightarrow$ ***(5)*** Suppose that for some $\mathcal{I}^* = \langle \mathcal{A}^*, \mathcal{A}^{?*}, \mathcal{C}|_{\mathcal{A}^* \cup \mathcal{A}^{?*}}, \emptyset \rangle$ in $\texttt{part}(\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \emptyset \rangle)$ $\overline{\phi}$ is ST-sceptical-IN in its certain projection AF^*; then ϕ is ST-credulous-IN in

AF^*. This implies that for $\mathcal{I}' = \langle \mathcal{A}^* \cup \{\chi, \overline{\chi}\}, \emptyset, (\mathcal{C} \cup \{(\overline{\chi}, \chi)\})|_{\mathcal{A}^* \cup \{\overline{\chi}, \chi\}}, \emptyset\rangle$ it also holds that ϕ is not ST-credulous-IN in its certain projection AF'. Note that $\mathcal{I}' \in \mathtt{part}(\mathcal{I}_2)$. Now consider $\mathcal{I}'' = \langle \mathcal{A}^* \cup \{\chi\}, \{\overline{\chi}\}, (\mathcal{C} \cup \{(\overline{\chi}, \chi)\})|_{\mathcal{A}^* \cup \{\overline{\chi}, \chi\}}, \emptyset\rangle$ and its certain projection AF''. In AF'', the argument $\overline{\phi}$ is attacked by the unattacked argument χ, so ϕ is ST-credulous-IN in AF''. Then by Lemma 2.7 item 3, removal of $\overline{\chi}$ is ST-credulous-IN-relevant for ϕ w.r.t. $\mathcal{I}_2$.

(5) $\Rightarrow$ ***(0)*** Suppose that removal of $\overline{\chi}$ is ST-credulous-IN-relevant for ϕ w.r.t. $\mathcal{I}_2$. By Lemma 2.7 item 3 there is an $\mathcal{I}' = \langle \mathcal{A}^* \cup \{\chi, \overline{\chi}\}, \emptyset, (\mathcal{C} \cup \{(\overline{\chi}, \chi)\})|_{\mathcal{A}^* \cup \{\overline{\chi}, \chi\}}, \emptyset\rangle$ in $\mathtt{part}(\mathcal{I}_2)$ such that ϕ is not ST-credulous-IN in $\mathtt{cert}(\mathcal{I}')$, so ϕ is ST-sceptical-OUT and $\overline{\phi}$ is ST-sceptical-IN in $\mathtt{cert}(\mathcal{I}')$. Then $\overline{\phi}$ would also be ST-sceptical-IN in $\mathtt{cert}(\mathcal{I}'')$ where $\mathcal{I}'' = \langle \mathcal{A}^*, \emptyset, \mathcal{C}|_{\mathcal{A}^*}, \emptyset\rangle$, which is in $\mathtt{part}(\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \emptyset\rangle)$.

(0) $\Rightarrow$ ***(6)*** Suppose that for some $\mathcal{I}^* = \langle \mathcal{A}^*, \mathcal{A}^{?*}, \mathcal{C}|_{\mathcal{A}^* \cup \mathcal{A}^{?*}}, \emptyset\rangle$ in $\mathtt{part}(\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \emptyset\rangle)$ $\overline{\phi}$ is ST-sceptical-IN in its certain projection AF^*; then ϕ is ST-sceptical-OUT in AF^*. This implies that for $\mathcal{I}' = \langle \mathcal{A}^*, \{\chi\}, \mathcal{C}|_{\mathcal{A}^* \cup \{\chi\}}, \emptyset\rangle$ it also holds that ϕ is ST-sceptical-OUT in its certain projection $AF' = AF^*$. Note that $\mathcal{I}' \in \mathtt{part}(\mathcal{I}_1)$. Now consider $\mathcal{I}'' = \langle \mathcal{A}^* \cup \{\chi\}, \emptyset, \mathcal{C}|_{\mathcal{A}^* \cup \{\chi\}}, \emptyset\rangle$ and its certain projection AF''. In AF'', the argument $\overline{\phi}$ is attacked by the unattacked argument χ, so ϕ is not ST-sceptical-OUT in AF''. Then by Lemma 2.7 item 3, removal of χ is ST-sceptical-OUT-relevant for ϕ w.r.t. $\mathcal{I}_1$.

(6) $\Rightarrow$ ***(0)*** Suppose that removal of χ is ST-sceptical-OUT-relevant for ϕ w.r.t. $\mathcal{I}_1$. Then by Lemma 2.7 item 3 there is some $\mathcal{I}' = \langle \mathcal{A}^*, \emptyset, \mathcal{C}|_{\mathcal{A}^*}, \emptyset\rangle$ in $\mathtt{part}(\mathcal{I}_1)$ such that ϕ is ST-sceptical-OUT in $\mathtt{cert}(\mathcal{I}')$, so $\overline{\phi}$ is ST-sceptical-IN in $\mathtt{cert}(\mathcal{I}')$. Since χ is not in $\mathcal{A}^*$, $\mathcal{I}'$ is also in $\mathtt{part}(\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \emptyset\rangle)$.

(0) $\Rightarrow$ ***(7)*** Suppose that for some $\mathcal{I}^* = \langle \mathcal{A}^*, \mathcal{A}^{?*}, \mathcal{C}|_{\mathcal{A}^* \cup \mathcal{A}^{?*}}, \emptyset\rangle$ in $\mathtt{part}(\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \emptyset\rangle)$ $\overline{\phi}$ is ST-sceptical-IN in its certain projection AF^*; then $\overline{\phi}$ is not ST-credulous-OUT in AF^*. This implies that for $\mathcal{I}' = \langle \mathcal{A}^*, \{\chi\}, \mathcal{C}|_{\mathcal{A}^* \cup \{\chi\}}, \emptyset\rangle$ it also holds that $\overline{\phi}$ is not ST-credulous-OUT in its certain projection $AF' = AF^*$. Note that $\mathcal{I}' \in \mathtt{part}(\mathcal{I}_1)$. Now consider $\mathcal{I}'' = \langle \mathcal{A}^* \cup \{\chi\}, \emptyset, \mathcal{C}|_{\mathcal{A}^* \cup \{\chi\}}, \emptyset\rangle$ and its certain projection AF''. In AF'', the argument $\overline{\phi}$ is attacked by the unattacked argument χ, so $\overline{\phi}$ is ST-sceptical-OUT in AF''. Then by Lemma 2.7 item 1, addition of χ is ST-credulous-OUT-relevant for $\overline{\phi}$ w.r.t. $\mathcal{I}_1$.

(7) $\Rightarrow$ ***(0)*** Suppose that addition of χ is ST-credulous-OUT-relevant for $\overline{\phi}$ w.r.t. $\mathcal{I}_1$. Then by Lemma 2.7 item 1 there is some $\mathcal{I}' = \langle \mathcal{A}^*, \{\chi\}, \mathcal{C}|_{\mathcal{A}^* \cup \{\chi\}}, \emptyset\rangle$ in $\mathtt{part}(\mathcal{I}_1)$ such that $\overline{\phi}$ is not ST-credulous-OUT in $\mathtt{cert}(\mathcal{I}')$, so $\overline{\phi}$ is ST-sceptical-IN in $\mathtt{cert}(\mathcal{I}')$. Since χ is not in $\mathcal{A}^*$, $\mathcal{I}'$ is also in $\mathtt{part}(\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \emptyset\rangle)$.

(0) $\Rightarrow$ ***(8)*** Suppose that for some $\mathcal{I}^* = \langle \mathcal{A}^*, \mathcal{A}^{?*}, \mathcal{C}|_{\mathcal{A}^* \cup \mathcal{A}^{?*}}, \emptyset\rangle$ in $\mathtt{part}(\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \emptyset\rangle)$ $\overline{\phi}$ is ST-sceptical-IN in its certain projection AF^*; then ϕ is ST-sceptical-OUT in AF^*. This implies that for $\mathcal{I}' = \langle \mathcal{A}^* \cup \{\chi, \overline{\chi}\}, \emptyset, (\mathcal{C} \cup \{(\overline{\chi}, \chi)\})|_{\mathcal{A}^* \cup \{\overline{\chi}, \chi\}}, \emptyset\rangle$ it also holds that ϕ is ST-sceptical-OUT in its certain projection AF'. Note that $\mathcal{I}' \in \mathtt{part}(\mathcal{I}_2)$. Now consider $\mathcal{I}'' = \langle \mathcal{A}^* \cup \{\chi\}, \{\overline{\chi}\}, (\mathcal{C} \cup \{(\overline{\chi}, \chi)\})|_{\mathcal{A}^* \cup \{\overline{\chi}, \chi\}}, \emptyset\rangle$ and its certain projection AF''. In AF'', the argument $\overline{\phi}$ is attacked by the unattacked argument χ, so ϕ is not ST-sceptical-OUT in AF''. Then by Lemma 2.7 item 1,

addition of $\overline{\chi}$ is ST-sceptical-OUT-relevant for ϕ w.r.t. $\mathcal{I}_2$.

(8) $\Rightarrow$ ***(0)*** Suppose that addition of $\overline{\chi}$ is ST-sceptical-OUT-relevant for ϕ w.r.t. $\mathcal{I}_2$. By Lemma 2.7 item 1 there is an $\mathcal{I}' = \langle \mathcal{A}^* \cup \{\chi, \overline{\chi}\}, \emptyset, (\mathcal{C} \cup \{(\overline{\chi}, \chi)\})|_{\mathcal{A}^* \cup \{\overline{\chi}, \chi\}}, \emptyset \rangle$ in $\texttt{part}(\mathcal{I}_2)$ such that ϕ is ST-sceptical-OUT in $\texttt{cert}(\mathcal{I}')$, so $\overline{\phi}$ is ST-sceptical-IN in $\texttt{cert}(\mathcal{I}')$. Then $\overline{\phi}$ would also be ST-sceptical-IN in $\texttt{cert}(\mathcal{I}'')$ where $\mathcal{I}'' = \langle \mathcal{A}^*, \emptyset, \mathcal{C}|_{\mathcal{A}^*}, \emptyset \rangle$, which is in $\texttt{part}(\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \emptyset \rangle)$.

(0) $\Rightarrow$ ***(9)*** Suppose that for some $\mathcal{I}^* = \langle \mathcal{A}^*, \mathcal{A}^{?*}, \mathcal{C}|_{\mathcal{A}^* \cup \mathcal{A}^{?*}}, \emptyset \rangle$ in $\texttt{part}(\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \emptyset \rangle)$ $\overline{\phi}$ is ST-sceptical-IN in its certain projection AF^*; then $\overline{\phi}$ is not ST-credulous-OUT in AF^*. Then for $\mathcal{I}' = \langle \mathcal{A}^* \cup \{\chi, \overline{\chi}\}, \emptyset, (\mathcal{C} \cup \{(\overline{\chi}, \chi)\})|_{\mathcal{A}^* \cup \{\overline{\chi}, \chi\}}, \emptyset \rangle$ it also holds that $\overline{\phi}$ is not ST-credulous-OUT in its certain projection AF'. Note that $\mathcal{I}' \in \texttt{part}(\mathcal{I}_2)$. Now consider $\mathcal{I}'' = \langle \mathcal{A}^* \cup \{\chi\}, \{\overline{\chi}\}, (\mathcal{C} \cup \{(\overline{\chi}, \chi)\})|_{\mathcal{A}^* \cup \{\overline{\chi}, \chi\}}, \emptyset \rangle$ and its certain projection AF''. In AF'', the argument $\overline{\phi}$ is attacked by the unattacked argument χ, so $\overline{\phi}$ is ST-credulous-OUT in AF''. Then by Lemma 2.7 item 3, removal of $\overline{\chi}$ is ST-credulous-OUT-relevant for $\overline{\phi}$ w.r.t. $\mathcal{I}_2$.

(9) $\Rightarrow$ ***(0)*** Suppose that removal of $\overline{\chi}$ is ST-credulous-OUT-relevant for $\overline{\phi}$ w.r.t. $\mathcal{I}_2$. By Lemma 2.7 item 3 there is an $\mathcal{I}' = \langle \mathcal{A}^* \cup \{\chi, \overline{\chi}\}, \emptyset, (\mathcal{C} \cup \{(\overline{\chi}, \chi)\})|_{\mathcal{A}^* \cup \{\overline{\chi}, \chi\}}, \emptyset \rangle$ in $\texttt{part}(\mathcal{I}_2)$ such that $\overline{\phi}$ is not ST-credulous-OUT in $\texttt{cert}(\mathcal{I}')$, so $\overline{\phi}$ is ST-sceptical-IN in $\texttt{cert}(\mathcal{I}')$. Then $\overline{\phi}$ would also be ST-sceptical-IN in $\texttt{cert}(\mathcal{I}'')$ where $\mathcal{I}'' = \langle \mathcal{A}^*, \emptyset, \mathcal{C}|_{\mathcal{A}^*}, \emptyset \rangle$, which is in $\texttt{part}(\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \emptyset \rangle)$. □

Proposition 2.19. *ST-sceptical-UNDEC-RELEVANCE is Σ_2^p-complete.*

Proof. First, we will show that ST-sceptical-UNDEC-RELEVANCE is in Σ_2^p. By Proposition 2.5, ST-sceptical-UNDEC-JUSTIFICATION is CoNP-complete. By Proposition 2.11, this implies that ST-sceptical-UNDEC-RELEVANCE is in Σ_2^p.

For the hardness proof, we use an existing result on the problem of necessary nonempty existence under ST semantics from Skiba et al. (2020), which is defined as follows: given an IAF $\mathcal{I}$, does each completion AF' of $\mathcal{I}$ have a nonempty ST extension? It is shown in Skiba et al. (2020, Theorem 21) that this problem is Π_2^p-hard.

Let $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ be an arbitrary instance of the necessary nonempty existence problem under ST semantics. If $\mathcal{A} = \emptyset$ then $\mathcal{I}$ is a negative instance, because there is a completion $AF' = \langle \mathcal{A}', \mathcal{C}' \rangle$ of $\mathcal{I}$ where $\mathcal{A}' = \emptyset$, which means that AF' has no nonempty ST extension. Alternatively, assume that $\mathcal{A}$ contains at least one argument and let A be an arbitrary argument in $\mathcal{A}$. Then we transform $\mathcal{I}$ into an instance $(\mathcal{I}', A, U)$ of the *argument removal variant* of the ST-sceptical-UNDEC-RELEVANCE problem, where:

- U is a fresh uncertain argument, not occurring in $\mathcal{A} \cup \mathcal{A}^?$; and
- $\mathcal{I}' = \langle \mathcal{A}, \mathcal{A}^? \cup \{U\}, \mathcal{C} \cup \{(U, B) \mid B \in \mathcal{A} \cup \mathcal{A}^?\}, \mathcal{C}^? \rangle$.

Next, we show that $\mathcal{I}$ is a *positive* instance of necessary nonempty existence under ST semantics iff $(\mathcal{I}', A, U)$ is a *negative* instance of ST-sceptical-UNDEC-RELEVANCE:

⇒ First suppose that $\mathcal{I}$ is a *positive* instance of necessary nonempty existence under ST semantics. Now let $\mathcal{I}'' = \langle \mathcal{A}'', \mathcal{A}^{?\prime\prime}, \mathcal{C}'', \mathcal{C}^{?\prime\prime} \rangle$ be an arbitrary partial completion of $\mathcal{I}'$ (which is the IAF in the transformed problem). We will prove that there is no completion of $\mathcal{I}''$ where A is ST-sceptical-UNDEC by distinguishing two options:

- If $U \in \mathcal{A}''$ then each argument other than U (including A) in each completion of $\mathcal{I}''$ is attacked by the unattacked argument U, so $\{U\}$ is a ST extension. Since A is attacked by an argument in the ST extension, it cannot be ST-sceptical-UNDEC. This holds for every completion of $\mathcal{I}''$.
- Otherwise, $U \notin \mathcal{A}''$. Then every completion of $\mathcal{I}''$ is also a completion of $\mathcal{I}$. Given that $\mathcal{I}$ is a positive instance of necessary nonempty existence under ST semantics, each completion of $\mathcal{I}$ has a (nonempty) ST extension, which must include either A or some argument attacking A. Then there is no completion of $\mathcal{I}$ in which A is ST-sceptical-UNDEC, which implies that there is no completion of $\mathcal{I}''$ either in which A is ST-sceptical-UNDEC.

Given that there is no completion of $\mathcal{I}''$ in which A is ST-sceptical-UNDEC, there can be no partial completion $\mathcal{I}'''$ of $\mathcal{I}''$ such that A is stable-ST-sceptical-UNDEC w.r.t. $\texttt{cert}(\mathcal{I}''')$. Then there is no minimal stable-ST-sceptical-UNDEC partial completion for A w.r.t. $\mathcal{I}$, which implies that the removal of U is not ST-sceptical-UNDEC relevant for A w.r.t. $\mathcal{I}''$. In other words: $(\mathcal{I}', A, U)$ is a *negative* instance of ST-sceptical-UNDEC-RELEVANCE.

⇐ Now suppose that $\mathcal{I}$ is a *negative* instance of necessary nonempty existence under ST semantics. Then there is some completion $AF' = \langle \mathcal{A}', \mathcal{C}' \rangle$ of $\mathcal{I}$ that has no nonempty ST extension. Recall that $\mathcal{A}'$ contains at least one argument (A), so AF' cannot have an empty ST extension (as any ST extension must contain either A or an attacker of A). This implies that AF' has no ST extension at all – which means that A must be ST-sceptical-UNDEC in AF'. Now construct $\mathcal{I}^* = \langle \mathcal{A}', \{U\}, \mathcal{C}' \cup \{(U, B) \mid B \in \mathcal{A} \cup \mathcal{A}^?\}, \emptyset \rangle$. Note that $\mathcal{I}^*$ is a partial completion of $\mathcal{I}'$ and that $\texttt{cert}(\mathcal{I}^*) = AF'$. So $\mathcal{I}'$ has a partial completion $\mathcal{I}^*$ such that A is ST-sceptical-UNDEC in $\texttt{cert}(\mathcal{I}^*)$.

Now consider the IAF $\mathcal{I}^{*\prime} = \langle \mathcal{A}' \cup \{U\}, \emptyset, \mathcal{C}' \cup \{(U, B) \mid B \in \mathcal{A} \cup \mathcal{A}^?\}, \emptyset \rangle$, which is also a partial completion of $\mathcal{I}'$. A is not ST-sceptical-UNDEC in $\texttt{cert}(\mathcal{I}^*)$, because $\texttt{cert}(\mathcal{I}^*)$ has a ST extension $\{U\}$ and U attacks A.

Given that $\mathcal{I}'$ has a partial completion $\mathcal{I}^*$ such that A is ST-sceptical-UNDEC in $\texttt{cert}(\mathcal{I}^*)$ while A is not ST-sceptical-UNDEC in $\texttt{cert}(\mathcal{I}^{*\prime})$ and thanks to specific properties of $\mathcal{I}^*$ and $\mathcal{I}^{*\prime}$ (having only U and no uncertain element), by Lemma 2.7 it must be that the removal of U is ST-sceptical-UNDEC-relevant w.r.t. $\mathcal{I}'$. So $(\mathcal{I}', A, U)$ is a *positive* instance of ST-sceptical-UNDEC-RELEVANCE.

Similarly, we can transform $\mathcal{I}$ into an instance $(\mathcal{I}'', A, U')$ of the *argument addition* variant of the ST-sceptical-UNDEC-RELEVANCE problem, where:

- U is a fresh argument, not occurring in $\mathcal{A} \cup \mathcal{A}^?$;

- U' is a fresh uncertain argument, not occurring in $\mathcal{A} \cup \mathcal{A}^?$; and
- $\mathcal{I}'' = \langle \mathcal{A} \cup \{U\}, \mathcal{A}^? \cup \{U'\}, \mathcal{C} \cup \{(U, B) \mid B \in \mathcal{A} \cup \mathcal{A}^?\} \cup \{(U', U)\}, \mathcal{C}^? \rangle$.

Then $\mathcal{I}$ is a *positive* instance of necessary nonempty existence under ST semantics iff $(\mathcal{I}'', A, U')$ is a *negative* instance of ST-sceptical-UNDEC-RELEVANCE.

For the *attack addition* variant of the ST-sceptical-UNDEC-RELEVANCE problem, the transformation from $\mathcal{I}$ into $(\mathcal{I}''', A, (U', U))$ is very similar:

- U and U' are fresh arguments, not occurring in $\mathcal{A} \cup \mathcal{A}^?$; and
- $\mathcal{I}''' = \langle \mathcal{A} \cup \{U, U'\}, \emptyset, \mathcal{C} \cup \{(U, B) \mid B \in \mathcal{A} \cup \mathcal{A}^?\}, \mathcal{C}^? \cup \{(U', U)\} \rangle$.

Then $\mathcal{I}$ is a *positive* instance of necessary nonempty existence under ST semantics iff $(\mathcal{I}''', A, (U', U))$ is a *negative* instance of ST-sceptical-UNDEC-RELEVANCE.

Finally, the transformation for the *attack removal* variant of the ST-sceptical-UNDEC-RELEVANCE problem from $\mathcal{I}$ into $(\mathcal{I}'''', A, (U'', U'))$ is:

- U, U' and U'' are fresh arguments, not occurring in $\mathcal{A} \cup \mathcal{A}^?$; and
- $\mathcal{I}'''' = \langle \mathcal{A} \cup \{U, U', U''\}, \emptyset, \mathcal{C} \cup \{(U, B) \mid B \in \mathcal{A} \cup \mathcal{A}^?\} \cup \{(U', U)\}, \mathcal{C}^? \cup \{(U'', U')\} \rangle$.

Then $\mathcal{I}$ is a *positive* instance of necessary nonempty existence under ST semantics iff $(\mathcal{I}'''', A, (U'', U'))$ is a *negative* instance of ST-sceptical-UNDEC-RELEVANCE.

We have shown for each of the four variants of the relevance problem that $\mathcal{I}$ is a *positive* instance of necessary nonempty existence under ST semantics iff the transformed instance is a *negative* instance of ST-sceptical-UNDEC-RELEVANCE. Given that the necessary nonempty existence problem under ST semantics is Π_2^p-hard, the complementary problem of ST-sceptical-UNDEC-RELEVANCE must be Σ_2^p-hard. Together with the membership result from the beginning of this proof, this implies that ST-sceptical-UNDEC-RELEVANCE is Σ_2^p-complete. □

Proposition 2.20. *ST-sceptical-existent-IN-RELEVANCE and ST-sceptical-existent-OUT-RELEVANCE are Σ_2^p-complete.*

Proof. Membership in Σ_2^p directly follows from the complexity of ST-sceptical-existent-IN- and -OUT-JUSTIFICATION and Proposition 2.11, in the following way: ST-sceptical-existent-IN-JUSTIFICATION is DP-complete by Dunne and Wooldridge (2009, page 92). By Lemma 2.1, ST-sceptical-existent-OUT-JUSTIFICATION is DP-complete as well. Then by Proposition 2.11 the problems of ST-sceptical-existent-IN-RELEVANCE and ST-sceptical-existent-OUT-RELEVANCE are in $NP^{DP} = \Sigma_2^p$; note that $NP^{DP} \subseteq \Sigma_2^p$ as any DP query can be answered by two (adaptive) SAT queries.

In order to prove Σ_2^p-hardness, we reduce from possible sceptical-existent acceptance under ST semantics (Definition 2.5), which was proven to be Σ_2^p-hard in Baumeister et al. (2021, Theorem 25). Let $(\mathcal{I}, A)$ be an arbitrary instance of possible sceptical-existent acceptance under ST semantics where $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$. We transform this into an instance $(\mathcal{I}', A, U)$ of ST-sceptical-existent-IN-RELEVANCE where U is a fresh uncertain argument that is not in $\mathcal{A} \cup \mathcal{A}^?$ and $\mathcal{I}' = \langle \mathcal{A}, \mathcal{A}^? \cup \{U\}, \mathcal{C} \cup \{(U, U)\}, \mathcal{C}^? \rangle$.

Next, we will prove that $(\mathcal{I}, A)$ is a positive instance of possible sceptical-existent acceptance under ST semantics iff $(\mathcal{I}', A, U)$ is a positive instance of ST-sceptical-existent-IN-RELEVANCE.

$\Rightarrow$ If $(\mathcal{I}, A)$ is a positive instance of possible sceptical-existent acceptance under ST semantics then there is some completion $AF' = \langle \mathcal{A}', \mathcal{C}' \rangle$ of $\mathcal{I}$ that has a ST extension and such that A is in each ST extension of AF'.

Construct the IAF $\mathcal{I}'' = \langle \mathcal{A}', \{U\}, \mathcal{C}' \cup \{(U,U)\}, \emptyset \rangle$; note that $\texttt{cert}(\mathcal{I}'') = AF'$ and $\mathcal{I}'' \in \texttt{part}(\mathcal{I}')$. So A is ST-sceptical-existent-IN w.r.t. $\texttt{cert}(\mathcal{I}'')$.

Also construct $\mathcal{I}''' = \langle \mathcal{A}' \cup \{U\}, \emptyset, \mathcal{C}' \cup \{(U,U)\}, \emptyset \rangle$ – this is a partial completion of $\mathcal{I}'$ as well. The certain projection of $\mathcal{I}'''$ contains the self-attacking argument U, which is not attacked by any other argument. Consequently, $\texttt{cert}(\mathcal{I}''')$ cannot have a ST extension, so A cannot be ST-sceptical-existent-IN w.r.t. $\texttt{cert}(\mathcal{I}''')$.

Then by Lemma 2.7, the removal of U is ST-sceptical-existent-IN-relevant w.r.t. $\mathcal{I}$. In other words, $(\mathcal{I}', A, U)$ is a positive instance of ST-sceptical-existent-IN-RELEVANCE.

$\Leftarrow$ If $(\mathcal{I}, A)$ is a negative instance of possible sceptical-existent acceptance under ST semantics then there is no partial completion $\mathcal{I}''$ of $\mathcal{I}$ such that A is ST-sceptical-existent-IN in $\texttt{cert}(\mathcal{I}'')$. Next, we will show that there is no partial completion $\mathcal{I}'''$ of $\mathcal{I}'$ either for which A is ST-sceptical-existent-IN in $\texttt{cert}(\mathcal{I}''')$. Let $\mathcal{I}''' = \langle \mathcal{A}''', \mathcal{A}^{?\prime\prime\prime}, \mathcal{C}''', \mathcal{C}^{?\prime\prime\prime} \rangle$ be an arbitrary partial completion of $\mathcal{I}'$.

- If $U \in \mathcal{A}'''$ then $\texttt{cert}(\mathcal{I}''')$ contains the self-attacking argument U that is not attacked by any other argument. This implies that $\texttt{cert}(\mathcal{I}''')$ does not have a ST extension. Consequently, A cannot be ST-sceptical-existent-IN in $\texttt{cert}(\mathcal{I}''')$.
- Alternatively, $U \notin \mathcal{A}'''$. In that case, $\mathcal{I}'''$ is also a partial completion of $\mathcal{I}$, so A cannot be ST-sceptical-existent-IN in $\texttt{cert}(\mathcal{I}''')$.

Since $\mathcal{I}'''$ was chosen arbitrarily from $\texttt{part}(\mathcal{I}')$, there can be no partial completion of $\mathcal{I}'''$ such that A is ST-sceptical-existent-IN in $\texttt{cert}(\mathcal{I}''')$. This implies that there is no ST-sceptical-existent-IN-relevant operation w.r.t. $\mathcal{I}'$. Therefore, $(\mathcal{I}', A, U)$ is a negative instance of ST-sceptical-existent-IN-RELEVANCE.

For ST-sceptical-existent-OUT-RELEVANCE, we transform an arbitrary instance of possible sceptical-existent acceptance under ST semantics $(\mathcal{I}, A)$ to an instance $(\mathcal{I}'', A, U)$ of ST-sceptical-existent-OUT-RELEVANCE, where U is a fresh uncertain argument that is not in $\mathcal{A} \cup \mathcal{A}^?$, B is a fresh argument that is not in $\mathcal{A} \cup \mathcal{A}^?$ and $\mathcal{I}'' = \langle \mathcal{A} \cup \{B\}, \mathcal{A}^? \cup \{U\}, \mathcal{C} \cup \{(B,A),(U,U)\}, \mathcal{C}^? \rangle$. Then $(\mathcal{I}, A)$ is a positive instance of possible sceptical-existent acceptance under ST semantics iff $(\mathcal{I}'', A, U)$ is a positive instance of ST-sceptical-existent-OUT-RELEVANCE. $\square$

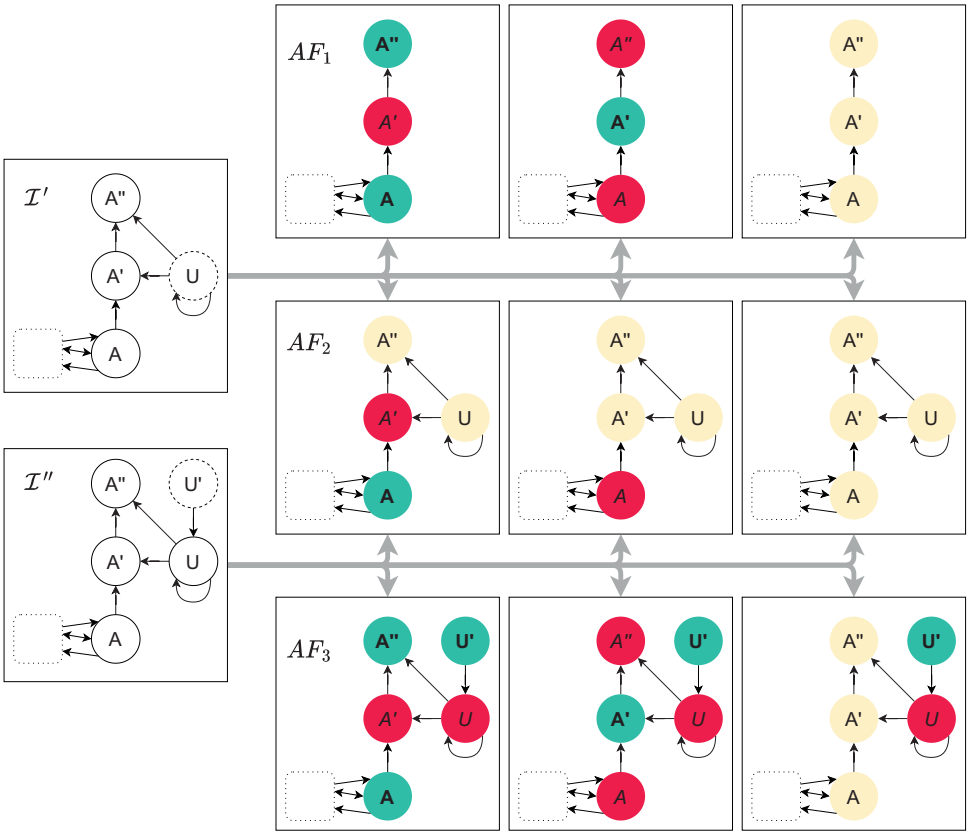


Figure A.3: Illustration of the IAF used to show Lemma 2.8 item 3. The IAFs given on the left are $\mathcal{I}'$ (upper) and $\mathcal{I}''$ (lower). The rounded rectangle with dotted borders represents the original IAF $\mathcal{I}$ (without A and in- and outgoing attacks). The grey arrows point to certain projections AF_1, AF_2 and AF_3 of partial completions. For each of these AFs, the possible justification statuses are colour-coded: green arguments with boldface font are IN, yellow arguments are UNDEC and red arguments with italic font are OUT. Note that, for a given justification status of A, there is only one possible justification status for each of the additional arguments in $\{A', A'', U, U'\}$.

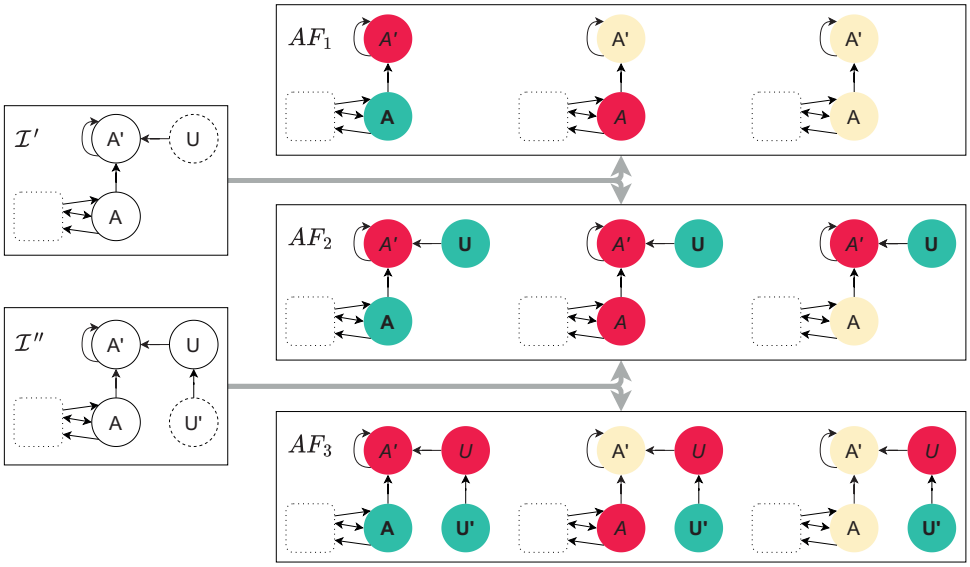


Figure A.4: Illustration of the IAF used to show Lemma 2.9 item 1. The IAFs given on the left are $\mathcal{I}'$ (upper) and $\mathcal{I}''$ (lower). The rounded rectangle with dotted borders represents the original IAF $\mathcal{I}$ (without A and in- and outgoing attacks). The grey arrows point to certain projections AF_1, AF_2 and AF_3 of partial completions. For each of these AFs, the possible justification statuses are colour-coded: green arguments with boldface font are IN, yellow arguments are UNDEC and red arguments with italic font are OUT. Note that, for a given justification status of A, there is only one possible justification status for each of the additional arguments in $\{A', U, U'\}$.

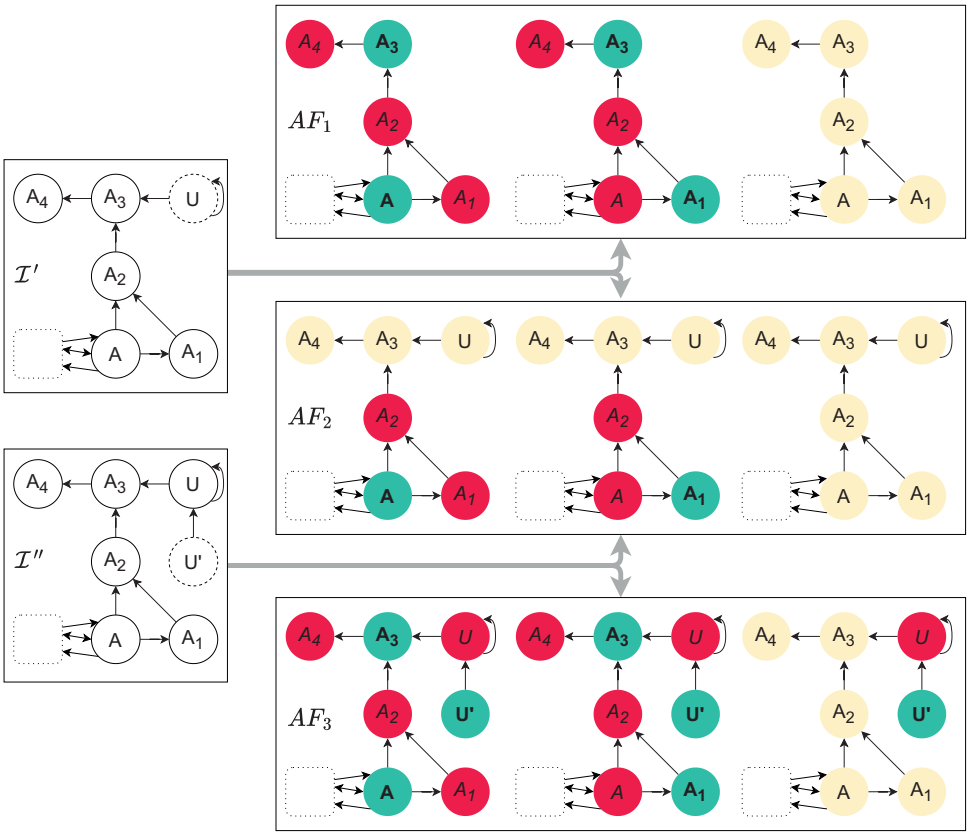


Figure A.5: Illustration of the IAF used to show Lemma 2.9 item 2. The IAFs given on the left are $\mathcal{I}'$ (upper) and $\mathcal{I}''$ (lower). The rounded rectangle with dotted borders represents the original IAF $\mathcal{I}$ (without A and in- and outgoing attacks). The grey arrows point to certain projections AF_1, AF_2 and AF_3 of partial completions. For each of these AFs, the possible justification statuses are colour-coded: green arguments with boldface font are IN, yellow arguments are UNDEC and red arguments with italic font are OUT. Note that, for a given justification status of A, there is only one possible justification status for each of the additional arguments in $\{A_1, A_2, A_3, A_4, U, U'\}$.

B

Appendix to Chapter 3

B.1. Proofs

Since the full proofs for lemmas and propositions in Chapter 3 are quite extensive, we collect them here in the appendix. In Sections B.1.1 and B.1.2, we give the full proof of the complexities of STABILITY and RELEVANCE. Section B.1.3 contains proofs for the time complexity of STABILITY-LABEL. In Sections B.1.4 and B.1.5, we give the full proofs of the soundness of the algorithms PREPROCESS and STABILITY-LABEL, respectively. Finally, we give the conditional completeness proof in Section B.1.6.

B.1.1. Complexity of stability

Theorem 3.2 (Complexity of deciding stability)**.** *Deciding whether a literal is stable-j in an AT is CoNP-complete for each justification status j in {unsatisfiable, defended, out, blocked}. Hardness holds even without preferences.*

Proof. Consider an argumentation theory $T = (AS, \mathcal{K})$ where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and $\mathcal{Q}$ is the set of queryable literals.

We will show that STABILITY is CoNP-complete by a CoNP-hardness proof and a proof that the STABILITY problem is in CoNP. In order to show that STABILITY is CoNP-hard, we will give a polynomial-time computable reduction f from the known CoNP-hard problem UNSAT, which is the decision problem of determining whether a given boolean expression in conjunctive normal form (CNF) has a truth assignment of its variables that makes the output true.

Formally: consider a boolean expression $\phi = (l_{11} \vee \ldots \vee l_{1k}) \wedge \ldots \wedge (l_{n1} \vee \ldots \vee l_{nm})$ in CNF; let $\mathcal{C}$ be the set containing all literals in ϕ and let v_I be the valuation function for propositional classical logic given the truth assignment function $I : \mathcal{C} \to \{\text{True}, \text{False}\}$. The goal of the *UNSAT problem* is to decide if $v_I(\phi)$ is False for every truth assignment I.

Next, we give a *reduction function* from UNSAT to the STABILITY problem. Let f be the function that converts an arbitrary formula $\phi = (l_{11} \vee \ldots \vee l_{1k}) \wedge \ldots \wedge (l_{n1} \vee \ldots \vee l_{nm})$ in CNF to an argumentation theory $T = (AS, \mathcal{K})$ with an argumentation system $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and a set of queryables $\mathcal{Q}$ (illustrated in Figure 3.9) such that:

- $\mathcal{L}$ consists of:

  (1) For each literal in $\mathcal{C}$: the literal and its negation; and

  (2) For each clause in ϕ: a clause-specific literal and its negation; and

  (3) A topic literal and its negation.

  The following formula expresses this formally, using numbers in superscript to refer to the three items identified above: $\mathcal{L} = \mathcal{C} \cup \{-l_{ij} \mid l_{ij} \in \mathcal{C}\}^{(1)} \cup \{c_i \mid \exists j : l_{ij} \in \mathcal{C}\} \cup \{-c_i \mid \exists j : l_{ij} \in \mathcal{C}\}^{(2)} \cup \{t, \neg t\}^{(3)}$;
- $\mathcal{R}$ consists of:

  (1) For each clause i and for each literal l_{ij} in this clause: $l_{ij} \Rightarrow l_i$; and

  (2) $(c_1, \ldots, c_n) \Rightarrow t$.

  Formally: $\mathcal{R} = \{l_{ij} \Rightarrow c_i \mid l_{ij} \in \mathcal{C}\}^{(1)} \cup \{(c_1, \ldots, c_n) \Rightarrow t\}^{(2)}$.
- $\bar{}$ corresponds to classical negation.
- $\mathcal{Q}$ contains all literals occurring in the CNF ϕ and their negations: $\mathcal{Q} = \{l_{ij} \mid l_{ij} \in \mathcal{C} \vee -l_{ij} \in \mathcal{C}\}$; and
- $\mathcal{K}$ is empty: $\mathcal{K} = \emptyset$.

We now examine the time needed for computing the reduction $f(\phi)$ for an arbitrary formula ϕ in CNF. For each literal l_{ij} occurring in ϕ, two literals are added to the language $\mathcal{L}$, one rule is added to $\mathcal{R}$ and two literals are added to $\mathcal{Q}$. Furthermore, for each clause two literals are added to $\mathcal{L}$. Finally, the topic literal and its negation are added to $\mathcal{L}$ and a single rule for the topic literal is added to $\mathcal{R}$. Since the number of clauses cannot exceed the number of literals occurring in ϕ, this reduction can be computed $\mathcal{O}(|\mathcal{C}|)$ operations; hence *the reduction f can be computed in polynomial time.*

Next, we prove that *there is no truth assignment I such that $v_I(\phi) = True$ iff t is stable in T w.r.t. $\mathcal{Q}$.*

- *Right to left*: we prove this by contraposition. Suppose that there exists a truth assignment I such that $v_I(\phi) = \text{True}$. $\mathcal{K}$ is empty, so $Arg_T = \emptyset$, hence there is no argument for t in Arg_T. Now let $\mathcal{K}' = \{l_{ij} \in \mathcal{Q} \mid v_I(l_{ij}) = \text{True}\}$ and let T' be $(AS, \mathcal{K}')$. $v_I(\phi) = \text{True}$, hence for each $i \in [1 \,..\, n]$, $v_I(l_{i1} \vee \ldots \vee l_{ik}) = \text{True}$. Consider an arbitrary $i \in [1 \,..\, n]$. $v_I(l_{i1} \vee \ldots \vee c_{ik}) = \text{True}$, so there is a $j \in [1 \,..\, k]$ such that $v_I(l_{ij}) = \text{True}$. Then by definition of $\mathcal{K}'$, $l_{ij} \in \mathcal{K}'$, which implies that there is an observation-based argument for c_{ij}. Furthermore, there is a rule $l_{ij} \Rightarrow c_i$ in $\mathcal{R}$; therefore there is a rule-based argument for c_i in $Arg_{T'}$. Since we chose i arbitrary, for each $i \in [1 \,..\, n]$, there is a rule-based argument for c_i in $Arg_{T'}$. Then there is an argument for t with top rule

$(c_1, \ldots, c_n) \Rightarrow t$ in $Arg_{T'}$. There is no argument for t in Arg_T and $T \sqsubseteq_Q T$, so by Definition 3.9, t is unsatisfiable in T. There is an argument for t in $Arg_{T'}$; this implies that t is not unsatisfiable in T'. Then by Definition 3.12, t is not stable in T w.r.t. $\mathcal{Q}$.

- *Left to right*: we prove this part by contraposition as well. Suppose that t is not stable in T w.r.t. $\mathcal{Q}$. Since there is no argument (for t) in Arg_T ($\mathcal{K} = \emptyset$), we have that t is unsatisfiable in T. By Definition 3.12, this implies that there exists an $T' = (AS, \mathcal{K}')$ such that $T \sqsubseteq_Q T'$ and there is an argument for t in $Arg_{T'}$. $t \notin \mathcal{Q}$, so the argument for t must be rule-based. The only rule for t in $\mathcal{R}$ is $(c_1, \ldots, c_n) \Rightarrow t$, so there is an argument for t with this top rule in $Arg_{T'}$. Then for each $i \in [1 \,..\, n]$, there is an argument for c_i in $Arg_{T'}$. Consider an arbitrary $i \in [1 \,..\, n]$. $c_i \notin \mathcal{Q}$, so there must be a rule-based argument for c_i. Let $\{r_{i1}, \ldots, r_{ik}\}$ be the rules for l_i in $\mathcal{R}$. Let $r_{ij} : l_{ij} \Rightarrow c_i$ (with $i \in [1 \,..\, k]$) be an arbitrary rule such that there is an argument having r_{ij} as its top rule in $Arg_{T'}$. Then there must be an argument for l_{ij} in $Arg_{T'}$. This argument must be observation-based, hence $l_{ij} \in \mathcal{K}'$. Let I be a truth assignment such that:

$$I(l_{ij}) = \begin{cases} \text{True} & \text{if } l_{ij} \in \mathcal{K}' \\ \text{False} & \text{if } l_{ij} \notin \mathcal{K}'. \end{cases}$$

  Given that $l_{ij} \in \mathcal{K}'$, we have that $v_I(l_{ij}) = \text{True}$ and therefore $v_I(l_{i1} \vee \ldots \vee l_{ik}) = \text{True}$ as well. Since we chose i arbitrary, this is the case for every i in $[1 \,..\, n]$. Then $v_I((l_{11} \vee \ldots \vee l_{1k}) \wedge \ldots \wedge (l_{n1} \vee \ldots \vee l_{nm})) = v_I(\phi) = \text{True}$. So there exists a truth assignment I such that $v_I(\phi) = \text{True}$.

We have shown that there exists a reduction function f from UNSAT to the STABILITY problem that can be computed in polynomial time, such that $v_I(\phi)$ is False for every truth assignment I iff $t \in \mathcal{L}$ is stable in $f(\phi) = T = (AS, \mathcal{K})$ w.r.t. $\mathcal{Q}$. This implies that UNSAT $\leq_p$ STABILITY. UNSAT is CoNP-hard, so *STABILITY is CoNP-hard.*

Now we prove that *STABILITY is in CoNP*. A decision problem is in CoNP iff for any no-instance x there is a "certificate" that a polynomial-time algorithm can use to verify that x is a no-instance. So STABILITY is in CoNP iff for any argumentation theory $T = (AS, \mathcal{K})$ where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$, for any set of queryables $\mathcal{Q}$ and for any $l \in \mathcal{L}$ such that l is not stable in T w.r.t. $\mathcal{Q}$, there is a polynomial-size certificate that a polynomial-time algorithm can use to verify that l is not stable in T w.r.t. $\mathcal{Q}$. A suitable certificate would be some knowledge base $\mathcal{K}' \subseteq \mathcal{Q}$ such that the justification status of l in $(AS, \mathcal{K}')$ differs from the justification status of l in T. Obviously $\mathcal{K}'$ is polynomial-size in the input since $\mathcal{K}' \subseteq \mathcal{Q}$. Furthermore, l is not stable in T w.r.t. $\mathcal{Q}$ iff there is some T' such that $T \sqsubseteq_Q T'$ and the justification status of l in T' differs from the justification status of l in T. Given the certificate $\mathcal{K}'$, the following procedure can be used to verify that l is not stable in T w.r.t. $\mathcal{Q}$:

1. Verify that $\mathcal{K}'$ is consistent and that $\mathcal{K} \subseteq \mathcal{K}' \subseteq \mathcal{Q}$. If this is the case, then $(AS, \mathcal{K})$ is a future argumentation theory of T. This step can be done in polynomial time.

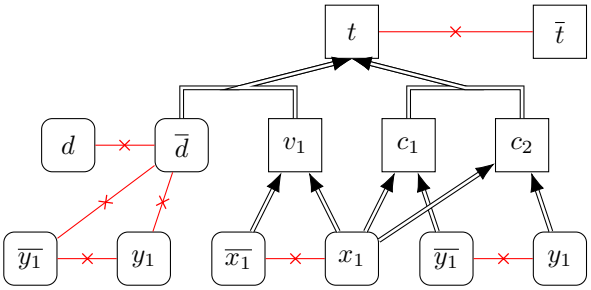


Figure B.1: Illustration of the reduction used in Theorem 3.3 for the formula $\phi = (x_1 \vee y_1) \wedge (x_i \vee \neg y_1)$. The queryables $\overline{y_1}$ and y_1 are displayed twice for readability.

2. Find the justification status of l in T. By Theorem 3.1, this can be done in polynomial time.

3. Find the justification status of l in T'. This can be done in polynomial time (Theorem 3.1).

4. Verify that the justification status of l in T differs from the justification status of l in T'. This step can be done in polynomial time as well.

To conclude, we have shown that the STABILITY problem is CoNP-hard and that it is in CoNP. This implies that *STABILITY is CoNP-complete.* □

B.1.2. Complexity of relevance

We show the remaining hardness results of Theorem 3.3 for the unsatisfiable, out, and blocked statuses. For readability, we split the proof into three proofs (one for each status).

All reductions are from the same Σ_2^P-complete problem (the same as for showing hardness for the defended status above). For the sake of readability, we recall the problem here again. In the Σ_2-SAT problem we are given a Boolean formula ϕ in conjunctive normal form (CNF), quantified over Boolean variables X and Y with $X = \{x_1, \ldots, x_n\}$ and $Y = \{y_1, \ldots, y_m\}$ are pairwise disjoint sets. An instance is a "yes" instance iff there is an assignment τ_X to variables in X such that for each assignment τ_Y to variables in Y the formula ϕ is refuted, i.e., $\phi[\tau_X, \tau_Y] = \text{False}$.

Proof of Theorem 3.3 for unsatisfiable status. To show Σ_2^P-hardness, we reduce from the Σ_2-SAT problem. Construct the following AT T and queryables $\mathcal{Q}$, with $C = \{c_1, \ldots, c_p\}$ the set of clauses in ϕ, and $\overline{X} = \{\overline{x} \mid x \in X\}$, $\overline{Y} = \{\overline{y} \mid y \in Y\}$ and $\overline{C} = \{\overline{c} \mid c \in C\}$. Let $V = \{v_i \mid x_i \in X\}$ and $\overline{V} = \{\overline{v_i} \mid x_i \in X\}$. This reduction is illustrated in Figure B.1.

$$
\begin{aligned}
\mathcal{Q} &= X \cup \overline{X} \cup Y \cup \overline{Y} \cup \{d, \overline{d}\} \\
\mathcal{L} &= \mathcal{Q} \cup C \cup \overline{C} \cup V \cup \overline{V} \cup \{t, \overline{t}\} \\
\overline{\ } &= \{(x, \overline{x}), (\overline{x}, x) \mid x \in X \cup Y \cup V \cup C \cup \{d, t\}\} \cup \\
&\quad \{(y, \overline{d}), (\overline{y}, \overline{d}), (\overline{d}, y), (\overline{d}, \overline{y}) \mid y \in Y\} \\
\mathcal{R} &= \{(\overline{d}, v_1, \ldots, v_n \Rightarrow t)\} \cup \\
&\quad \{(x \Rightarrow c) \mid x \in c\} \cup \{(\overline{x} \Rightarrow c) \mid \neg x \in c\} \cup \\
&\quad \{(y \Rightarrow c) \mid y \in c\} \cup \{(\overline{y} \Rightarrow c) \mid \neg y \in c\} \cup \\
&\quad \{(c_1, \ldots, c_p \Rightarrow t)\} \cup \\
&\quad \{(x_i \Rightarrow v_i), (\overline{x_i} \Rightarrow v_i) \mid x_i \in X\} \\
\mathcal{K} &= \emptyset
\end{aligned}
$$

With loss of generality, we assume that d, $\overline{d}$, t, and $\overline{t}$ do not occur in ϕ, i.e., are fresh atoms. It follows that $T = (AS, \mathcal{K})$ with $AS = (\mathcal{L}, \overline{\ }, \mathcal{R}, \leq)$ and $\mathcal{Q}$ can be constructed in polynomial time w.r.t. ϕ. We claim that (ϕ, X, Y) is a positive instance of Σ_2-SAT iff d is unsatisfiable-relevant for t w.r.t. T.

- From left to right: suppose that (ϕ, X, Y) is a positive instance of Σ_2-SAT. Then there is some assignment to variables of X such that for each assignment to variables of Y, $\phi[\tau_X, \tau_Y]$ is False. Let τ_X be this assignment and construct a knowledge base $\mathcal{K}' = \{x \in X \mid \tau_X[x] = \text{True}\} \cup \{\overline{x} \in X \mid \tau_X[x] = \text{False}\}$. Note that $\mathcal{K}'$ must be consistent, as no $x \in X$ can be assigned both True and False by τ_X. Therefore $T \sqsubseteq_Q (AS, \mathcal{K}')$. Then:

  - t is not stable-unsatisfiable w.r.t. $(AS, \mathcal{K}')$ and $\mathcal{Q}$, because t is not unsatisfiable w.r.t. $(AS, \mathcal{K}' \cup \{\overline{d}\})$: note that the contradictory of $\overline{d}$ is not in $\mathcal{K}'$; therefore $\mathcal{K}' \cup \{\overline{d}\}$ is a consistent knowledge base. Given that for each $x \in X$ either $x \in \mathcal{K}'$ or $\overline{x} \in \mathcal{K}'$, it must be that for each $x \in X$ either $x \in \mathcal{K}' \cup \{\overline{d}\}$ or $\overline{x} \in \mathcal{K}' \cup \{\overline{d}\}$. This implies that there is an argument for t with top rule $(\overline{d}, v_1, \ldots, v_n \Rightarrow t)$ in $Arg_{(AS, \mathcal{K}' \cup \{\overline{d}\})}$.
  - t is stable-unsatisfiable w.r.t. $(AS, \mathcal{K}' \cup \{d\})$ and $\mathcal{Q}$, as we show next. Let $T'' = (AS, \mathcal{K}' \cup \{d\} \cup \mathcal{K}'')$ be an arbitrary AT such that $(AS, \mathcal{K}' \cup \{d\}) \sqsubseteq_Q T''$. Note that $\mathcal{K}'' \subseteq Y \cup \overline{Y}$. Given that there is no assignment τ_Y to variables in Y such that $\phi[\tau_X, \tau_Y]$ is True, there can be no argument for t with top rule $(c_1, \ldots, c_p \Rightarrow t)$ in $Arg_{T''}$. Furthermore, given that d is in the knowledge base of T'', there can be no argument for t with top rule $(\overline{d}, v_1, \ldots, v_n \Rightarrow t)$, in $Arg_{T''}$. Since there are no other rules for t and t is not in $\mathcal{Q}$, t must be unsatisfiable w.r.t. T''. As T'' was chosen arbitrarily from all T''' such that $(AS, \mathcal{K}' \cup \{d\}) \sqsubseteq_Q T'''$, we derive that t is stable-unsatisfiable w.r.t. $(AS, \mathcal{K}' \cup \{d\})$ and $\mathcal{Q}$.

  So by Lemma 3.3, d is unsatisfiable-relevant for t w.r.t. T.

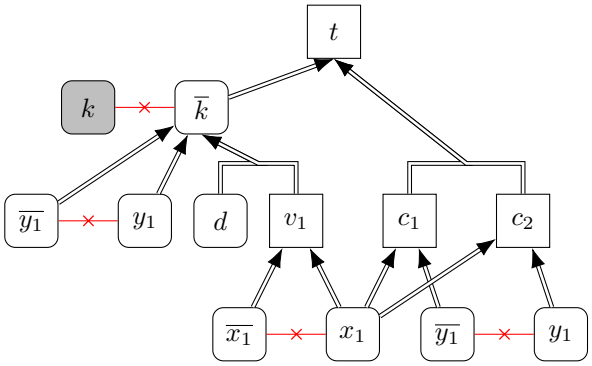


Figure B.2: Illustration of the reduction used in Theorem 3.3 for the out status for the formula $\phi = (x_1 \vee y_1) \wedge (x_i \vee \neg y_1)$. The queryables $\overline{y_1}$ and y_1 are displayed twice for readability.

- From right to left: suppose that d is unsatisfiable-relevant for t w.r.t. T. Then by Definition 3.14, there is some minimal stable-unsatisfiable future theory $T' = (AS, \mathcal{K}' \cup \{d\})$ w.r.t. T and $\mathcal{Q}$. Given that t is stable-unsatisfiable w.r.t. $(AS, \mathcal{K}' \cup \{d\})$ and $\mathcal{Q}$, t must be unsatisfiable w.r.t. $(AS, \mathcal{K}' \cup \{d\})$. Then there is no argument for t in $Arg_{(AS, \mathcal{K}' \cup \{d\})}$.

  In addition, by minimality of $(AS, \mathcal{K}' \cup \{d\})$, t cannot be stable-unsatisfiable w.r.t. $(AS, \mathcal{K}')$ and $\mathcal{Q}$. Then there must be some future argumentation theory of $(AS, \mathcal{K}')$ for which there is some argument for t having the observation-based argument for $\overline{d}$ as a subargument. This must be an argument with top rule $(\overline{d}, v_1, \ldots, v_n \Rightarrow t)$. Given that this argument exists, for each $x \in X$, either $x \in \mathcal{K}'$ or $\overline{x} \in \mathcal{K}'$. In addition, for each $y \in Y$: $y \notin \mathcal{K}'$ and $\overline{y} \notin \mathcal{K}'$ (as these are contradictories of $\overline{d}$). Now let τ_X be the assignment to variables in X corresponding to $\mathcal{K}'$: for each $x \in X$, $\tau_X[x] = \text{True}$ iff $x \in \mathcal{K}'$ and $\tau_X[x] = \text{False}$ iff $\overline{x} \in \mathcal{K}'$.

  Next, we prove that for each assigment τ_Y to variables in Y, $\phi[\tau_X, \tau_Y]$ must be False. Suppose, towards a contradiction, that there is some τ_Y such that $\phi[\tau_X, \tau_Y]$ is True. Let $\mathcal{K}' \cup \{d\} \cup \mathcal{K}''$ be the corresponding knowledge base: $\mathcal{K}'' = \{y \in Y \mid \tau_Y[y] = \text{True}\} \cup \{\overline{y} \in Y \mid \tau_Y[y] = \text{False}\}$. Then, given that $\phi[\tau_X, \tau_Y]$ is True, there is an argument for t in $Arg_{(AS, \mathcal{K}' \cup \{d\} \cup \mathcal{K}'')}$, which implies that t is not unsatisfiable w.r.t. $(AS, \mathcal{K}' \cup \{d\} \cup \mathcal{K}'')$, but then t was not stable-unsatisfiable w.r.t. $(AS, \mathcal{K}' \cup \{d\})$ and $\mathcal{Q}$; contradiction. Therefore for each assigment τ_Y to variables in Y, $\phi[\tau_X, \tau_Y]$ must be False. In other words: (ϕ, X, Y) is a positive instance of Σ_2-SAT. □

Proof of Theorem 3.3 for out status. For the Σ_2^P-hardness proof, we reduce from the Σ_2-SAT problem. Construct the following AT T and queryables $\mathcal{Q}$, with $C = \{c_1, \ldots, c_p\}$ the set of clauses in ϕ, and $\overline{X} = \{\overline{x} \mid x \in X\}$, $\overline{Y} = \{\overline{y} \mid y \in Y\}$ and $\overline{C} = \{\overline{c} \mid c \in C\}$. Let $V = \{v_i \mid x_i \in X\}$ and $\overline{V} = \{\overline{v_i} \mid x_i \in X\}$. An illustration can be found in Figure B.2.

$$
\begin{aligned}
\mathcal{Q} &= X \cup \overline{X} \cup Y \cup \overline{Y} \cup \{d, \overline{d}, k, \overline{k}\} \\
\mathcal{L} &= \mathcal{Q} \cup C \cup \overline{C} \cup V \cup \overline{V} \cup \{t, \overline{t}\} \\
\overline{\ } &= \{(x, \overline{x}), (\overline{x}, x) \mid x \in X \cup Y \cup V \cup C \cup \{d, t, k\}\} \\
\mathcal{R} &= \{(d, v_1, \ldots, v_n \Rightarrow \overline{k})\} \cup \\
&\quad \{(x \Rightarrow c) \mid x \in c\} \cup \{(\overline{x} \Rightarrow c) \mid \neg x \in c\} \cup \\
&\quad \{(y \Rightarrow c) \mid y \in c\} \cup \{(\overline{y} \Rightarrow c) \mid \neg y \in c\} \cup \\
&\quad \{(c_1, \ldots, c_p \Rightarrow t)\} \cup \\
&\quad \{(x_i \Rightarrow v_i), (\overline{x_i} \Rightarrow v_i) \mid x_i \in X\} \\
&\quad \{(y \Rightarrow \overline{k}), (\overline{y} \Rightarrow \overline{k}) \mid y \in Y\} \\
&\quad \{(\overline{k} \Rightarrow t)\} \\
\mathcal{K} &= \{k\}
\end{aligned}
$$

Similarly as for the other reductions, we assume that d, $\overline{d}$, k, and $\overline{k}$ are fresh atoms not occurring in ϕ. Then $T = (AS, \mathcal{K})$ where $AS = (\mathcal{L}, \overline{\ }, \mathcal{R}, \leq)$ and $\mathcal{Q}$ can be constructed in polynomial time w.r.t. ϕ.

We prove that (ϕ, X, Y) is a "yes" instance of Σ_2-SAT iff d is out-relevant for t w.r.t. T.

- From left to right: suppose that (ϕ, X, Y) is a positive instance of Σ_2-SAT. Then there is some assignment to variables of X such that for each assignment to variables of Y, $\phi[\tau_X, \tau_Y]$ is False. Let τ_X be this assignment and construct a knowledge base $\mathcal{K}' = \{k\} \cup \{x \in X \mid \tau_X[x] = \text{True}\} \cup \{\overline{x} \in X \mid \tau_X[x] = \text{False}\}$. Note that $\mathcal{K} \subseteq \mathcal{K}'$ and that $\mathcal{K}'$ must be consistent, as no $x \in X$ can be assigned both True and False by τ_X. Therefore $T \sqsubseteq_Q (AS, \mathcal{K}')$. Then:
  - t is not stable-out w.r.t. $(AS, \mathcal{K}')$ and $\mathcal{Q}$, because t is not out w.r.t. $(AS, \mathcal{K}')$: given that $d \notin \mathcal{K}'$ and for each $y \in Y$ both $y \notin \mathcal{K}'$ and $\overline{y} \notin \mathcal{K}'$, there is no argument for t in $Arg_{(AS, \mathcal{K}')}$.
  - t is stable-out w.r.t. $(AS, \mathcal{K}' \cup \{d\})$ and $\mathcal{Q}$, as we show next. Let $T'' = (AS, \mathcal{K}' \cup \{d\} \cup \mathcal{K}'')$ be an arbitrary AT such that $(AS, \mathcal{K}' \cup \{d\}) \sqsubseteq_Q T''$. Note that $\mathcal{K}'' \subseteq Y \cup \overline{Y}$. Given that there is no assignment τ_Y to variables in Y such that $\phi[\tau_X, \tau_Y]$ is True, there can be no argument for t with top rule $(c_1, \ldots, c_p \Rightarrow t)$ in $Arg_{T''}$. On the other hand, there is at least one argument for t, with top rule $(\overline{k} \Rightarrow t)$, in $Arg_{T''}$. In fact, every argument for t in $Arg_{T''}$ must have $(\overline{k} \Rightarrow t)$ as its top rule and is therefore defeated by the observation-based (undefeated) argument k, which must be in $G(T'')$. Given that there is an argument for t in Arg_T but every argument for t in Arg_T is defeated by an argument in $G(T'')$, t must be out w.r.t. T''. As T'' was chosen arbitrarily from all T''' such that $(AS, \mathcal{K}' \cup \{d\}) \sqsubseteq_Q T'''$, t is stable-out w.r.t. $(AS, \mathcal{K}' \cup \{d\})$ and $\mathcal{Q}$.

  So by Lemma 3.3, d is out-relevant for t w.r.t. T.

- From right to left: suppose that d is out-relevant for t w.r.t. T. Then by Definition 3.14, there is some minimal stable-out future theory $T' = (AS, \mathcal{K}' \cup \{d\})$ w.r.t. T and $\mathcal{Q}$. Given that t is stable-out w.r.t. $(AS, \mathcal{K}' \cup \{d\})$ and $\mathcal{Q}$, t must be out w.r.t. $(AS, \mathcal{K}' \cup \{d\})$. Then there must be some argument for t in $Arg_{T'}$ and each argument for t in $Arg_{T'}$ is defeated by an argument in $G(T')$. This implies that there is no argument with top rule $(c_1, \ldots, c_p \Rightarrow t)$, as this argument for t would have been undefeated.

  In addition, by minimality of $(AS, \mathcal{K}' \cup \{d\})$, t cannot be stable-out w.r.t. $(AS, \mathcal{K}')$ and $\mathcal{Q}$. So there is an AT $(AS, \mathcal{K}'')$ such that $(AS, \mathcal{K}') \sqsubseteq_{\mathcal{Q}} (AS, \mathcal{K}'')$ and t is not out w.r.t. $(AS, \mathcal{K}'')$. Note that there can be no argument with top rule $(c_1, \ldots, c_p \Rightarrow t)$ in $Arg_{(AS,\mathcal{K}'')}$, as that would imply that there would be an argument with top rule $(c_1, \ldots, c_p \Rightarrow t)$ in $Arg_{(AS,\mathcal{K}'' \cup \{d\})}$, while $(AS, \mathcal{K}' \cup \{d\}) \sqsubseteq_{\mathcal{Q}} (AS, \mathcal{K}'' \cup \{d\})$ and t is stable-out w.r.t. $(AS, \mathcal{K}' \cup \{d\})$. Therefore t is not defended w.r.t. $(AS, \mathcal{K}'')$. Atom t cannot be blocked w.r.t. $(AS, \mathcal{K}'')$ either, as there is no "equally strong" argument defeating any argument for t – the only way in which an argument for t can be defeated is by the observation-based argument k, which must be in the grounded extension. This implies that t must be unsatisfiable w.r.t. $(AS, \mathcal{K}'')$, which means that t was unsatisfiable w.r.t. $(AS, \mathcal{K}')$ as well. Then there is no argument for t, so for each $y \in Y$, there can be no argument with top rule $y \Rightarrow \overline{k}$ or $\overline{y} \Rightarrow \overline{k}$ in $Arg_{(AS,\mathcal{K}')}$. Consequently, for each $y \in Y$: $y \notin \mathcal{K}'$ and $\overline{y} \notin \mathcal{K}'$.

  Given that there is an argument for t in $Arg_{T'}$ and $T' = (AS, \mathcal{K}' \cup \{d\})$, the argument for t in $Arg_{(AS,\mathcal{K}' \cup \{d\})}$ must have been based on a set of rules including $(d, v_1, \ldots, v_n \Rightarrow \overline{k})$ and $(\overline{k} \Rightarrow t)$. This implies that for each $x \in X$, either $x \in \mathcal{K}'$ or $\overline{x} \in \mathcal{K}'$. Now let τ_X be the assignment to variables in X corresponding to $\mathcal{K}'$: for each $x \in X$, $\tau_X[x] = \text{True}$ iff $x \in \mathcal{K}'$ and $\tau_X[x] = \text{False}$ iff $\overline{x} \in \mathcal{K}'$.

  Next, we prove that for each assigment τ_Y to variables in Y, $\phi[\tau_X, \tau_Y]$ must be False. Suppose, towards a contradiction, that there is some τ_Y such that $\phi[\tau_X, \tau_Y]$ is True. Let $\mathcal{K}' \cup \{d\} \cup \mathcal{K}^*$ be the corresponding knowledge base: $\mathcal{K}^* = \{y \in Y \mid \tau_Y[y] = \text{True}\} \cup \{\overline{y} \in Y \mid \tau_Y[y] = \text{False}\}$. Then, given that $\phi[\tau_X, \tau_Y]$ is True, there is an argument for t in $Arg_{(AS,\mathcal{K}' \cup \{d\} \cup \mathcal{K}^*)}$, which implies that t is defended w.r.t. $(AS, \mathcal{K}' \cup \{d\} \cup \mathcal{K}^*)$, but then t was not stable-out w.r.t. $(AS, \mathcal{K}' \cup \{d\})$ and $\mathcal{Q}$; contradiction. Therefore for each assigment τ_Y to variables in Y, $\phi[\tau_X, \tau_Y]$ must be False. In other words: (ϕ, X, Y) is a positive instance of Σ_2-SAT. □

Proof of Theorem 3.3 for the blocked status. For the Σ_2^P-hardness proof, we reduce from the Σ_2-SAT problem. Construct the following AT T and queryables $\mathcal{Q}$, with $C = \{c_1, \ldots, c_p\}$ the set of clauses in ϕ, and $\overline{X} = \{\overline{x} \mid x \in X\}$, $\overline{Y} = \{\overline{y} \mid y \in Y\}$ and $\overline{C} = \{\overline{c} \mid c \in C\}$. Let $V = \{v_i \mid x_i \in X\}$ and $\overline{V} = \{\overline{v_i} \mid x_i \in X\}$. This reduction is also illustrated for a small Boolean formula in Figure B.3.

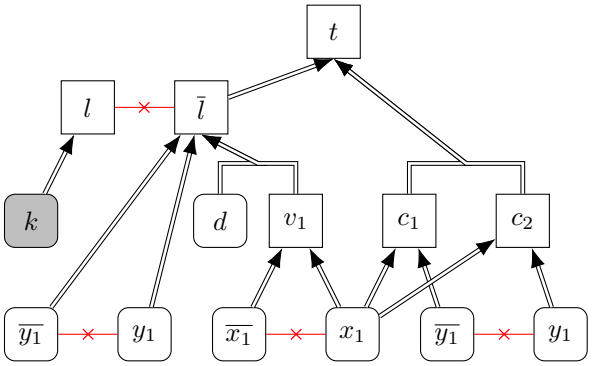


Figure B.3: Illustration of the reduction used in Theorem 3.3 for the blocked status for the formula $\phi = (x_1 \vee y_1) \wedge (x_i \vee \neg y_1)$. The queryables $\overline{y_1}$ and y_1 are displayed twice for readability.

$$
\begin{aligned}
\mathcal{Q} &= X \cup \overline{X} \cup Y \cup \overline{Y} \cup \{d, \overline{d}, k, \overline{k}\} \\
\mathcal{L} &= \mathcal{Q} \cup C \cup \overline{C} \cup V \cup \overline{V} \cup \{t, \bar{t}, l, \bar{l}\} \\
\overline{\ } &= \{(x, \overline{x}), (\overline{x}, x) \mid x \in X \cup Y \cup V \cup C \cup \{d, t, k, l\}\} \\
\mathcal{R} &= \{(d, v_1, \ldots, v_n \Rightarrow \bar{l})\} \cup \\
&\quad \{(x \Rightarrow c) \mid x \in c\} \cup \{(\overline{x} \Rightarrow c) \mid \neg x \in c\} \cup \\
&\quad \{(y \Rightarrow c) \mid y \in c\} \cup \{(\overline{y} \Rightarrow c) \mid \neg y \in c\} \cup \\
&\quad \{(c_1, \ldots, c_p \Rightarrow t)\} \cup \\
&\quad \{(x_i \Rightarrow v_i), (\overline{x_i} \Rightarrow v_i) \mid x_i \in X\} \cup \\
&\quad \{(y \Rightarrow \bar{l}), (\overline{y} \Rightarrow \bar{l}) \mid y \in Y\} \cup \\
&\quad \{(k \Rightarrow l), (\bar{l} \Rightarrow t)\} \\
\mathcal{K} &= \{k\}
\end{aligned}
$$

We require that d, $\overline{d}$, k, and $\overline{k}$ are fresh atoms not part of the vocabulary of ϕ. Then $T = (AS, \mathcal{K})$ where $AS = (\mathcal{L}, \overline{\ }, \mathcal{R}, \leq)$ and $\mathcal{Q}$ can be constructed in polynomial time w.r.t. ϕ. We prove that (ϕ, X, Y) is a positive instance of Σ_2-SAT iff d is blocked-relevant for t w.r.t. T.

- From left to right: suppose that (ϕ, X, Y) is a positive instance of Σ_2-SAT. Then there is some assignment to variables of X such that for each assignment to variables of Y, $\phi[\tau_X, \tau_Y]$ is False. Let τ_X be this assignment and construct a knowledge base $\mathcal{K}' = \{k\} \cup \{x \in X \mid \tau_X[x] = \text{True}\} \cup \{\overline{x} \in X \mid \tau_X[x] = \text{False}\}$. Note that $\mathcal{K} \subseteq \mathcal{K}'$ and that $\mathcal{K}'$ must be consistent, as no $x \in X$ can be assigned both True and False by τ_X. Therefore $T \sqsubseteq_{\mathcal{Q}} (AS, \mathcal{K}')$. Then:
  - t is not stable-blocked w.r.t. $(AS, \mathcal{K}')$ and $\mathcal{Q}$, because t is not blocked w.r.t. $(AS, \mathcal{K}')$: given that $d \notin \mathcal{K}'$ and for each $y \in Y$ both $y \notin \mathcal{K}'$ and $\overline{y} \notin \mathcal{K}'$, there is no argument for t in $Arg_{(AS, \mathcal{K}')}$.

– t is stable-blocked w.r.t. $(AS, \mathcal{K}' \cup \{d\})$ and $\mathcal{Q}$, as we show next. Let $T'' = (AS, \mathcal{K}' \cup \{d\} \cup \mathcal{K}'')$ be an arbitrary AT such that $(AS, \mathcal{K}' \cup \{d\}) \sqsubseteq_Q T''$. Note that $\mathcal{K}'' \subseteq Y \cup \overline{Y}$. Given that there is no assignment τ_Y to variables in Y such that $\phi[\tau_X, \tau_Y]$ is True, there can be no argument for t with top rule $(c_1, \ldots, c_p \Rightarrow t)$ in $Arg_{T''}$. On the other hand, there is at least one argument for t, with top rule $(\bar{l} \Rightarrow t)$, in $Arg_{T''}$. In fact, every argument for t in $Arg_{T''}$ must have the rule $(\bar{l} \Rightarrow t)$ as its top rule and is therefore defeated by the argument with top rule $(k \Rightarrow l)$, which is defeated by all arguments for $\bar{l}$. Given that there is an argument for t in Arg_T but every argument for t in Arg_T is defeated by an argument in $Arg_{T''}$ that is not in or defeated by any argument in $G(T'')$, t must be blocked w.r.t. T''. As T'' was chosen arbitrarily from all T''' such that $(AS, \mathcal{K}' \cup \{d\}) \sqsubseteq_Q T'''$, we derive that t is stable-blocked w.r.t. $(AS, \mathcal{K}' \cup \{d\})$ and $\mathcal{Q}$.

So by Lemma 3.3, d is blocked-relevant for t w.r.t. T.

- From right to left: suppose that d is blocked-relevant for t w.r.t. T. Then by Definition 3.14, there is some minimal stable-blocked future theory $T' = (AS, \mathcal{K}' \cup \{d\})$ w.r.t. T and $\mathcal{Q}$. Given that t is stable-blocked w.r.t. $(AS, \mathcal{K}' \cup \{d\})$ and $\mathcal{Q}$, t must be blocked w.r.t. $(AS, \mathcal{K}' \cup \{d\})$. Then there must be some argument for t in $Arg_{T'}$ and each argument for t in $Arg_{T'}$ is defeated by an argument in $Arg_{T'}$. This implies that there is no argument with top rule $(c_1, \ldots, c_p \Rightarrow t)$, as this argument for t would be undefeated.

  In addition, by minimality of $(AS, \mathcal{K}' \cup \{d\})$, t cannot be stable-blocked w.r.t. $(AS, \mathcal{K}')$ and $\mathcal{Q}$. So there is an AT $(AS, \mathcal{K}'')$ such that $(AS, \mathcal{K}') \sqsubseteq_Q (AS, \mathcal{K}'')$ and t is not blocked w.r.t. $(AS, \mathcal{K}'')$. Note that there can be no argument with top rule $(c_1, \ldots, c_p \Rightarrow t)$ in $Arg_{(AS, \mathcal{K}'')}$, as that would imply that there would be an argument with top rule $(c_1, \ldots, c_p \Rightarrow t)$ in $Arg_{(AS, \mathcal{K}'' \cup \{d\})}$, while $(AS, \mathcal{K}' \cup \{d\}) \sqsubseteq_Q (AS, \mathcal{K}'' \cup \{d\})$ and t is stable-blocked w.r.t. $(AS, \mathcal{K}' \cup \{d\})$. Therefore t is not defended w.r.t. $(AS, \mathcal{K}'')$. Atom t cannot be out w.r.t. $(AS, \mathcal{K}'')$ either, as there is no "stronger" argument defeating any argument for t. This implies that t must be unsatisfiable w.r.t. $(AS, \mathcal{K}'')$, which means that t is unsatisfiable w.r.t. $(AS, \mathcal{K}')$ as well. Then there is no argument for t, so for each $y \in Y$, there can be no argument with top rule $y \Rightarrow \bar{l}$ or $\overline{y} \Rightarrow \bar{l}$ in $Arg_{(AS, \mathcal{K}')}$. Consequently, for each $y \in Y$: $y \notin \mathcal{K}'$ and $\overline{y} \notin \mathcal{K}'$.

  Given that there is an argument for t in $Arg_{T'}$ and $T' = (AS, \mathcal{K}' \cup \{d\})$, the argument for t in $Arg_{(AS, \mathcal{K}' \cup \{d\})}$ must have been based on a set of rules including $(d, v_1, \ldots, v_n \Rightarrow \bar{l})$ and $(\bar{l} \Rightarrow t)$. This implies that for each $x \in X$, either $x \in \mathcal{K}'$ or $\overline{x} \in \mathcal{K}'$. Now let τ_X be the assignment to variables in X corresponding to $\mathcal{K}'$: for each $x \in X$, $\tau_X[x] = \text{True}$ iff $x \in \mathcal{K}'$ and $\tau_X[x] = \text{False}$ iff $\overline{x} \in \mathcal{K}'$.

  Next, we prove that for each assigment τ_Y to variables in Y, $\phi[\tau_X, \tau_Y]$ must be False. Suppose, towards a contradiction, that there is some τ_Y such that $\phi[\tau_X, \tau_Y]$ is True. Let $\mathcal{K}' \cup \{d\} \cup \mathcal{K}^*$ be the corresponding knowledge base: $\mathcal{K}^* = \{y \in Y \mid \tau_Y[y] = \text{True}\} \cup \{\overline{y} \in Y \mid \tau_Y[y] = \text{False}\}$. Then, given that $\phi[\tau_X, \tau_Y]$

is True, there is an argument for t in $Arg_{(AS,\mathcal{K}'\cup\{d\}\cup\mathcal{K}^*)}$, which implies that t is defended w.r.t. $(AS, \mathcal{K}' \cup \{d\} \cup \mathcal{K}^*)$, but then t was not stable-blocked w.r.t. $(AS, \mathcal{K}' \cup \{d\})$ and $\mathcal{Q}$; contradiction. Therefore for each assigment τ_Y to variables in Y, $\phi[\tau_X, \tau_Y]$ must be False. In other words: (ϕ, X, Y) is a positive instance of Σ_2-SAT. □

B.1.3. Time complexity of the heuristic algorithm for stability

In this section, we specify the time complexity of Stability-label in terms of its input (Proposition 3.10). Since the first step of Stability-label is running the Preprocess algorithm, we start with the time complexity of this algorithm in a separate lemma.

Lemma 3.6 (Time complexity Preprocess)**.** *The time complexity of Preprocess is* $\mathcal{O}(|\mathcal{L}|^2 + |\mathcal{L}| \cdot |\mathcal{R}|^2)$.

Proof. In the following proof, $\{c_1, \ldots, c_{14}\}$ is a set of positive constants.

- The Preprocess algorithm starts with a for loop in line 2–4, which is iterated $|\mathcal{L}|$ times. A single iteration of line 2 takes constant time c_1. A single iteration of line 3 requires for each literal $l \in \mathcal{Q}$ a check that no contradictory literal $l' \in \bar{l}$ is in $\mathcal{K}$. No literal can have more than $|\mathcal{L}|$ contradictories, so a single iteration of line 3 takes at most $c_2 \cdot |\mathcal{L}|$ operations. An iteration of line 4 can be done in constant time c_3. Then all iterations of line 2–4 require at most $(c_1 + |\mathcal{L}| \cdot c_2 + c_3) \cdot |\mathcal{L}|$ operations.
- Lines 5 and 6 are repeated $|\mathcal{R}|$ times and take constant time per iteration, which means that these lines take $(c_4 + c_5) \cdot |\mathcal{R}|$ steps in total.
- Line 7 takes constant time c_6 and is executed just once.
- Next we consider the while-loop (lines 8–14), which iterates until no label changed in the previous loop. Thanks to the check $L_p[r] = \langle 1, 0, 0, 0\rangle$ in line 11, each rule can change label at most once, hence the while-loop iterates at most $|\mathcal{R}|$ times.
  - Line 8 only requires a label check, which can be done in constant time c_7; the total time required for all iterations of line 8 is at most $c_7 \cdot |\mathcal{R}|$.
  - Similarly, the total time needed for all iterations of line 9 does not exceed $c_8 \cdot |\mathcal{R}|$.
  - The for-loop iterates $|\mathcal{R}|$ times for each iteration of the while-loop, so lines 10–13 are executed at most $|\mathcal{R}|^2$ times in total. A single execution of line 10 takes constant time c_9. Line 11 checks all antecedents of each rule, which requires at most $c_{10} \cdot |\mathcal{L}|$ checks per execution; a single execution of line 12, 13 and 14 take constant time $c_{11} + c_{12} + c_{13}$. So the total time required for all executions of lines 10–14 is at most $(c_9 + c_{10} \cdot |\mathcal{L}| + c_{11} + c_{12} + c_{13}) \cdot |\mathcal{R}|^2$.
- Finally, line 15 is executed just once and takes constant time c_{14}.

The total time required for Algorithm 3 is at most $(c_1 + |\mathcal{L}| \cdot c_2 + c_3) \cdot |\mathcal{L}| + (c_4 + c_5) \cdot |\mathcal{R}| + c_6 + (c_7 + c_8) \cdot |\mathcal{R}| + (c_9 + c_{10} \cdot |\mathcal{L}| + c_{11} + c_{12} + c_{13}) \cdot |\mathcal{R}|^2 + c_{14}$. As a result, the time complexity of the preprocessing step must be $\mathcal{O}(|\mathcal{L}|^2 + |\mathcal{L}| \cdot |\mathcal{R}|^2)$. □

Proposition 3.10 (Time complexity STABILITY-LABEL)**.** *The time complexity of* STABILITY-LABEL *is* $\mathcal{O}(|\mathcal{L}|^3 \cdot |\mathcal{R}| + |\mathcal{L}|^2 \cdot |\mathcal{R}|^2)$.

Proof. We will prove this by first showing the amount of time that is required for a *single* execution of a given line (also given in the second column of the table below). Next, we will consider the number of iterations of each line (third column), multiply them to get the total time required for each line (fourth column) and combine this into the big-O notation (final row). In the following, we will denote positive constants by c_i (with $i \in [1 \,..\, 29]$).

Line	Time single execution	Max iterations	Total time
2	$(c_1 + \lvert\mathcal{L}\rvert \cdot c_2 + c_3) \cdot \lvert\mathcal{L}\rvert + (c_4 + c_5) \cdot \lvert\mathcal{R}\rvert + c_6 + (c_7 + c_8) \cdot \lvert\mathcal{R}\rvert + (c_9 + c_{10} \cdot \lvert\mathcal{L}\rvert + c_{11} + c_{12} + c_{13}) \cdot \lvert\mathcal{R}\rvert^2 + c_{14}$	1	$(c_1 + \lvert\mathcal{L}\rvert \cdot c_2 + c_3) \cdot \lvert\mathcal{L}\rvert + (c_4 + c_5) \cdot \lvert\mathcal{R}\rvert + c_6 + (c_7 + c_8) \cdot \lvert\mathcal{R}\rvert + (c_9 + c_{10} \cdot \lvert\mathcal{L}\rvert + c_{11} + c_{12} + c_{13}) \cdot \lvert\mathcal{R}\rvert^2 + c_{14}$
3	c_{15}	1	c_{15}
4	c_{16}	$\lvert\mathcal{L}\rvert$	$c_{16} \cdot \lvert\mathcal{L}\rvert$
5	$c_{17} \cdot (\lvert\mathcal{L}\rvert + \lvert\mathcal{R}\rvert)$	$\lvert\mathcal{L}\rvert$	$c_{17} \cdot (\lvert\mathcal{L}\rvert^2 + \lvert\mathcal{L}\rvert \cdot \lvert\mathcal{R}\rvert)$
6	$c_{18} \cdot \lvert\mathcal{R}\rvert$	$\lvert\mathcal{L}\rvert$	$c_{18} \cdot \lvert\mathcal{L}\rvert \cdot \lvert\mathcal{R}\rvert$
7	c_{19}	$5 \cdot \lvert\mathcal{L}\rvert \cdot \lvert\mathcal{R}\rvert$	$5 \cdot c_{19} \cdot \lvert\mathcal{L}\rvert \cdot \lvert\mathcal{R}\rvert$
8	c_{20}	$5 \cdot \lvert\mathcal{L}\rvert \cdot \lvert\mathcal{R}\rvert$	$5 \cdot c_{20} \cdot \lvert\mathcal{L}\rvert \cdot \lvert\mathcal{R}\rvert$
9	$c_{21} \cdot \lvert\mathcal{L}\rvert$	$5 \cdot \lvert\mathcal{L}\rvert \cdot \lvert\mathcal{R}\rvert$	$5 \cdot c_{21} \cdot \lvert\mathcal{L}\rvert^2 \cdot \lvert\mathcal{R}\rvert$
10	c_{22}	$5 \cdot \lvert\mathcal{L}\rvert \cdot \lvert\mathcal{R}\rvert$	$5 \cdot c_{22} \cdot \lvert\mathcal{L}\rvert \cdot \lvert\mathcal{R}\rvert$
11	$c_{23} \cdot (\lvert\mathcal{L}\rvert + \lvert\mathcal{R}\rvert)$	$4 \cdot \lvert\mathcal{R}\rvert$	$4 \cdot c_{23} \cdot (\lvert\mathcal{L}\rvert \cdot \lvert\mathcal{R}\rvert + \lvert\mathcal{R}\rvert^2)$
12	c_{24}	$4 \cdot \lvert\mathcal{R}\rvert$	$4 \cdot c_{24} \cdot \lvert\mathcal{R}\rvert$
13	$c_{25} \cdot \lvert\mathcal{R}\rvert$	$4 \cdot \lvert\mathcal{L}\rvert$	$4 \cdot c_{25} \cdot \lvert\mathcal{L}\rvert \cdot \lvert\mathcal{R}\rvert$
14	c_{26}	$5 \cdot \lvert\mathcal{L}\rvert^2 \cdot \lvert\mathcal{R}\rvert$	$5 \cdot c_{26} \cdot \lvert\mathcal{L}\rvert^2 \cdot \lvert\mathcal{R}\rvert$
15	$c_{27} \cdot (\lvert\mathcal{L}\rvert + \lvert\mathcal{R}\rvert)$	$5 \cdot \lvert\mathcal{L}\rvert^2 \cdot \lvert\mathcal{R}\rvert$	$5 \cdot c_{27} \cdot (\lvert\mathcal{L}\rvert^3 \cdot \lvert\mathcal{R}\rvert + \lvert\mathcal{L}\rvert^2 \cdot \lvert\mathcal{R}\rvert^2)$
16	c_{28}	$5 \cdot \lvert\mathcal{L}\rvert^2 \cdot \lvert\mathcal{R}\rvert$	$5 \cdot c_{28} \cdot \lvert\mathcal{L}\rvert^2 \cdot \lvert\mathcal{R}\rvert$
17	$c_{29} \cdot \lvert\mathcal{R}\rvert$	$4 \cdot \lvert\mathcal{L}\rvert$	$4 \cdot c_{29} \cdot \lvert\mathcal{L}\rvert \cdot \lvert\mathcal{R}\rvert$
18	c_{30}	1	c_{30}
Total time required for all lines			$\mathcal{O}(\lvert\mathcal{L}\rvert^3 \cdot \lvert\mathcal{R}\rvert + \lvert\mathcal{L}\rvert^2 \cdot \lvert\mathcal{R}\rvert^2)$

We assume that for each literal $l \in \mathcal{L}$, the list of rules for that literal can be obtained in constant time, as well as the list of rules having that literal as an antecedent. As a result, checking if there are rules for this literal can be done in constant time, since we can check in constant time if the list of rules for the literal is empty. For any literal $l \in \mathcal{L}$, we can check in constant time if $l \in \mathcal{Q}$.

First we show *the amount of time that is required for a single execution of a given line.*

- Line 2 requires running PREPROCESS($\mathcal{L}, \mathcal{R}, \bar{}, \mathcal{Q}, \mathcal{K}$), which takes $(c_1 + |\mathcal{L}| \cdot c_2 + c_3) \cdot |\mathcal{L}| + (c_4 + c_5) \cdot |\mathcal{R}| + c_6 + (c_7 + c_8) \cdot |\mathcal{R}| + (c_9 + c_{10} \cdot |\mathcal{L}| + c_{11} + c_{12} + c_{13}) \cdot |\mathcal{R}|^2 + c_{14}$ steps, as shown in Lemma 3.6.
- Line 3 takes constant time c_{15}.
- Line 4 takes a new literal l from $\mathcal{L}$; a single execution of this line requires a check if $l \in \mathcal{Q}$, which, by assumption, takes constant time and, if $l \notin \mathcal{Q}$, requires an additional check if there is a rule for l in $\mathcal{R}$, which takes constant time as well. To conclude, a single execution of line 4 takes constant time c_{16}.
- A single execution of line 5 requires labelling a literal, which can be done in $c_{17} \cdot (|\mathcal{L}| + |\mathcal{R}|)$ time: in the worst case, it requires checking the presence of l and all its contradictories ($|\bar{l}| \leq |\mathcal{L}|$) in $\mathcal{Q}$ and $\mathcal{K}$, as well as the labels of all ($\max |\mathcal{R}|$) rules for that literal or a contradictory. Therefore, line 5 requires $c_{17} \cdot (|\mathcal{L}| + |\mathcal{R}|)$ time per execution.
- In a single execution of line 6, all rules having l as an antecedent are added to TODO-SET. There are at most $\mathcal{R}$ rules having l as an antecedent, so this takes at most $c_{18} \cdot |\mathcal{R}|$ time.
- Line 7 only needs to check if a set is empty, which can be done in constant time c_{19}. The next line only needs to pop an element from a set, which can be done in constant time as well, so line 8 needs c_{20} time.
- Line 9 relabels a rule, which requires checking the labels of all antecedents of this rule. Since a rule has at most $|\mathcal{L}|$ antecedents, this takes at most $c_{21} \cdot |\mathcal{L}|$ time per execution of line 9.
- Lines 10, 12 and 16 only check if the label of a rule or literal changed; this can be done in constant time c_{22}, c_{24} and c_{28}, respectively.
- Lines 11 and 15 relabel a literal. As explained before (for line 5), this requires checking the presence of a literal and all its contradictories in $\mathcal{Q}$ and $\mathcal{K}$, as well as checking the labels of all rules for that literal or one of its contradictories. A single execution therefore takes $c_{23} \cdot (|\mathcal{L}| + |\mathcal{R}|)$ time for line 11 and $c_{27} \cdot (|\mathcal{L}| + |\mathcal{R}|)$ time for line 15.
- Lines 13 and 17 both add at most $|\mathcal{R}|$ rules to TODO-SET, so a single execution of line 13 needs at most $c_{25} \cdot |\mathcal{R}|$ time and a single execution of line 17 needs at most $c_{29} \cdot |\mathcal{R}|$ time.
- A single execution of line 14 takes a literal from a list of contradictories, which can be done in constant time c_{26}.
- Finally, a single execution of line 18 takes constant time c_{30}.

Now we consider *the number of iterations of each line*; this is also represented in the third column of the table above.

- Lines 2, 3 and 18 are executed just once.
- Lines 4–6 are repeated for each literal in $\mathcal{Q}$ or for which there is no rule in $\mathcal{R}$. This implies that lines 4–6 are executed at most $|\mathcal{L}|$ times.

- The lines 7–10 are executed once in every iteration of the while loop. The total number of iterations of the while loop equals the number of times a rule is added to TODO-SET. A rule is only added to TODO-SET if it was not yet visited (line 6) or if the label of one of its antecedents changed after a relabelling (line 13 or line 17). Since the label of a literal can change at most four times (i.e. at most four booleans can be turned to False), each rule $r \in \mathcal{R}$ is relabelled at most $5 \cdot |\texttt{ants}(r)|$ times. There are $|\mathcal{R}|$ rules, so lines 7–10 are executed at most $5 \cdot |\mathcal{L}| \cdot |\mathcal{R}|$ times. Besides, we will see in Lemma B.2 that in practice, the maximum of changes is three, which means that these lines are executed at most $4 \cdot |\mathcal{L}| \cdot |\mathcal{R}|$ times.
- Next, we consider lines 11 and 12. These lines are only executed if the label of a rule changed. A label can only be changed by turning one of the four booleans to False. Therefore, for each rule, its label can be changed at most four times (again, we will see in Lemma B.2 that in practice, the maximum of changes is three). There are $|\mathcal{R}|$ rules in total, so lines 11 and 12 are executed at most $4 \cdot |\mathcal{R}|$ times.
- Lines 13 and 17 are only executed just after the label of a literal changed. This can happen at most four times for each literal, because at most four booleans can be turned to False (again, we will see in Lemma B.2 that the maximum of changes per label is in practice 3). There are $|\mathcal{L}|$ literals in total; therefore lines 13 and 17 are executed at most $4 \cdot |\mathcal{L}|$ times.
- Finally, lines 14–16 are executed at most $|\mathcal{L}|$ times (i.e. once for each $l' \in \overline{\texttt{conc}(r)}$) for each of the maximal $5 \cdot |\mathcal{L}| \cdot |\mathcal{R}|$ iterations of the while loop. This means that lines 14–16 are executed at most $5 \cdot |\mathcal{L}|^2 \cdot |\mathcal{R}|$ times.

An *upper bound on the total amount of time that is needed for all executions of a single line* can now be obtained by multiplying the maximum time required for a single execution by the number of executions of each line. We do this in the fourth column of our table. From these results, it becomes clear that the total running time of Algorithm 6 is dominated by the lines for relabeling, in particular of line 15. To conclude, the *total time complexity* of STABILITY-LABEL (Algorithm 6) is $\mathcal{O}(|\mathcal{L}|^3 \cdot |\mathcal{R}| + |\mathcal{L}|^2 \cdot |\mathcal{R}|^2)$. □

B.1.4. Soundness of the preprocessing algorithm

In this section, we show that PREPROCESS (Algorithm 3) is sound: if a literal is labelled $\langle 1, 0, 0, 0 \rangle$, then there is no future argumentation theory in which there is an argument for that literal. Before we can prove this in Section B.1.4, we first need some additional definitions in Section B.1.4.

Height of arguments

In this section, we introduce the notions of argument height and direct subarguments, since we will use them repeatedly in our induction proofs for the soundness of the preprocessing step (Section B.1.4) and the conditional completeness of the STABILITY-LABEL algorithm (Section B.1.6). Intuitively, the height of an argument

is the maximum number of inferences between a premise and the conclusion of the argument.

Definition B.1 (Height of arguments)**.** Let $T = (AS, \mathcal{K})$ be an argumentation theory and let $A \in Arg_T$ be an argument. The height $\mathtt{h}(A)$ of A is:

- if A is an *observation-based argument* then $\mathtt{h}(A) = 0$;
- if A is a *rule-based argument* of the form $A_1, \ldots, A_m \Rightarrow c$, then $\mathtt{h}(A) = 1 + \max(\mathtt{h}(A_1), \ldots, \mathtt{h}(A_m))$.

Given a rule-based argument, the arguments for the antecedents of the rule on which the arguments is based are the direct subarguments.

Definition B.2 (Direct subarguments)**.** Let $T = (AS, \mathcal{K})$ be an argumentation theory and let $A \in Arg_T$ be an argument. The direct subarguments $\mathtt{dirsub}(A)$ of A are:

- if A is an *observation-based argument* then $\mathtt{dirsub}(A) = \emptyset$.
- if A is a *rule-based argument* of the form $A_1, \ldots, A_m \Rightarrow c$ then the set of direct subarguments $\mathtt{dirsub}(A)$ of A is $\{A_1, \ldots, A_m\}$.

In the proofs that follow, we will repeatedly use the notion of argument height and direct subarguments to prove a certain property of an argument by induction, in the following way. As a base case, we prove the property for arguments with height of 0 or 1; consequently, we assume that the property holds for (direct) subarguments A' with $\mathtt{h}(A') \leq k$ and prove the property for argument A with height $\mathtt{h}(A) = k+1$. Note that Definition 3.4 on arguments enforces that all arguments have finite height, since the number of steps for constructing an argument is finite.

Soundness proof

In the next lemma, we prove the soundness of the preprocessing step: if a literal l is labelled $L_p[l] = \langle 1, 0, 0, 0\rangle$ by PREPROCESS, then l is stable-unsatisfiable in that argumentation theory with respect to the set of queryables. This property will also be useful in the soundness proofs for STABILITY-LABEL in Section B.1.5.

Lemma 3.4 (Soundness preprocessing step)**.** *Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and let $\mathcal{Q}$ be a set of queryables. Furthermore let L_p be the labelling obtained by PREPROCESS (Algorithm 3) on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. For each $l \in \mathcal{L}$: if $L_p[l] = \langle 1, 0, 0, 0\rangle$, then l is stable-unsatisfiable in T w.r.t. $\mathcal{Q}$.*

Proof. We prove this by contraposition. Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and let $\mathcal{Q}$ be a set of queryables. Further let $l \in \mathcal{L}$ be a literal and let L_p be the labelling after the preprocessing step.

Now suppose that l is not stable-unsatisfiable in T w.r.t. $\mathcal{Q}$. Then by Definition 3.12, there exists some future argumentation theory $T' = (AS, \mathcal{K}')$ of T such that there is an argument A for l in $Arg_{T'}$. We now prove by induction on the height of A that l is labelled $\langle 1, 1, 1, 1\rangle$ by Algorithm 3.

Base case: Suppose that $\mathtt{h}(A) = 0$. Then $l \in \mathcal{K}'$. This implies that $l \in \mathcal{Q}$ and for each $l' \in \bar{l}$: $l' \notin \mathcal{K}$, so l is labelled $\langle 1,1,1,1\rangle$ in Algorithm 3 line 3. Since there is no operation in Algorithm 3 that labels literals from $\langle 1,1,1,1\rangle$ to $\langle 1,0,0,0\rangle$, we have that $L_p[l] = \langle 1,1,1,1\rangle$.

Induction hypothesis: If A is an argument for l in $Arg_{T'}$ and $\mathtt{h}(A) \leq k$, then $L_p[l] = \langle 1,1,1,1\rangle$.

Induction step: Now suppose that $\mathtt{h}(A) = k+1$. Then A must have some top rule r and for each $A_i \in \mathtt{dirsub}(A)$: A_i in $Arg_{T'}$ and $\mathtt{h}(A_i) \leq k$. By the induction hypothesis, for each $a \in \mathtt{ants}(r)$: $L_p[a] = \langle 1,1,1,1\rangle$, so $L_p[l]$ is labelled $L_p[l] = \langle 1,1,1,1\rangle$ by Algorithm 3 line 13.

Since each argument A in $Arg_{T'}$ has a finite $\mathtt{h}(A)$, $L_p[l] = \langle 1,1,1,1\rangle$. As there is no operation that labels literals from $\langle 1,1,1,1\rangle$ to $\langle 1,0,0,0\rangle$, we have that $L_p[l] = \langle 1,1,1,1\rangle$ (so $L_p[l] \neq \langle 1,0,0,0\rangle$). By contraposition: if $L_p[l] = \langle 1,0,0,0\rangle$, then for each T' such that $T \sqsubseteq_{\mathcal{Q}} T'$: l is unsatisfiable in T'. In other words: l is stable-unsatisfiable in T w.r.t. $\mathcal{Q}$. □

B.1.5. Soundness of the stability algorithm

In this section, we show that Stability-label (Algorithm 6) is sound: if a literal $l \in \mathcal{L}$ is labelled stable in T w.r.t. $\mathcal{Q}$, then l is stable in T w.r.t. $\mathcal{Q}$. Before we can prove this in Section B.1.5, we first need some additional definitions in Section B.1.5.

Interim labelling

In order to prove the soundness of the Stability-label algorithm, we will repeatedly use the notion of interim label, that is: the label at some point before the labelling is finished.

Definition B.3 (Interim labelling)**.** Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and let $\mathcal{Q}$ be a set of queryables. Furthermore let L be the labelling obtained by Stability-label (Algorithm 6) on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. For any literal or rule $x \in \mathcal{L} \cup \mathcal{R}$, we denote the *interim labelling* state of x immediately after the i'th iteration of the while loop as $L_i[x]$.

Given that the interim labelling L_i is the labelling state immediately after the i'th iteration of the while loop, L_0 is the labelling state just before the start of the first iteration of the while loop; to be precise: between line 6 and 7 of Algorithm 6. At this point, the preprocessing step has finished (line 2) and literals that are queryable or that are not the consequent of any rule are relabelled (line 5). Note that the labels of rules at this point are unchanged compared to the label obtained from preprocessing.

Remark 1 (No rule label change between preprocessing and while loop)**.** Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and let $\mathcal{Q}$ be a set of queryables. Let L_p be the labelling obtained by Preprocess (Algorithm 3) and let L be the labelling obtained by Stability-label (Algorithm 6) on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. For each rule $r \in \mathcal{R}$, $L_0[r] = L_p[r]$.

Proof. By Definition B.3, $L_0[r]$ is the interim labelling state of r immediately after the 0'th iteration of the while loop. Between the end of the preprocessing step (after Algorithm 6 line 2) and the start of the while loop (from line 7), there is no operation that changes the labelling state of any rule. Therefore, for each rule $r \in \mathcal{R}$, $L_p[r] = L_0[r]$. □

Another new concept that is related to the interim labelling is the number of the iteration of the while loop in which one or more booleans from u, d, o and b is turned from True to False. We will repeatedly use this in our induction proofs; for example, if we know that the d-boolean of literal l was relabelled for the last time in iteration i, then we know that l is labelled $\neg L_i[l].d$, but also $\neg L_{i+1}[l].d$, $\neg L_{i+2}[l].d$, etc. This means for example that each rule r that has l as an antecedent will be considered for relabelling in some iteration $j > i$ and can be labelled $\neg L_j[r].d$ at that point, because we know that l is an antecedent of r and l is labelled $\neg L_{j-1}[l].d$.

Definition B.4 (Last boolean flip iteration)**.** Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and let $\mathcal{Q}$ be a set of queryables. Let $\mathcal{J} = 2^{\{u,d,o,b\}}$ be the set of all subsets of the labelling booleans u, d, o and b. For each $J \in \mathcal{J}$ and for each literal or rule $x \in \mathcal{L} \cup \mathcal{R}$, we denote by $c_J(x)$ the number of the iteration of the while loop in Algorithm 6 (executed on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$) in which the last boolean $j \in J$ was turned from True to False – provided that each boolean $j \in J$ is False in the final labelling L. In case there is some $j \in J$ such that j is True in the final labelling L, $c_J(x) = \infty$.

Example B.1 (Last boolean flip iteration)**.** Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$. $\mathcal{L} = \{q, \neg q, a, \neg a\}$, $\mathcal{R} = \{q \Rightarrow a\}$ and $\bar{}$ corresponds to classical negation. The set of queryables is $\{q, \neg q\}$ and the knowledge base is $\mathcal{K} = \{q\}$.

Then $L_p[q] = L_p[a] = \langle 1, 1, 1, 1\rangle$ and $L_p[\neg q] = L_p[\neg a] = \langle 1, 0, 0, 0\rangle$. Before the first iteration of the while loop (i.e. Algorithm 6 line 7–17), q and $\neg q$ are considered for relabelling. q's label is changed into $L_0[q] = \langle 0, 1, 0, 0\rangle$, while $L_0[\neg q] = L_p[\neg q] = \langle 1, 0, 0, 0\rangle$. No rule in $\mathcal{R}$ is relabelled between the preprocessing step and the start of the while-loop, so $L_0[q \Rightarrow a] = L_p[q \Rightarrow a] = \langle 1, 1, 1, 1\rangle$. Then the rule $q \Rightarrow a$ is added to TODO-SET. In the first iteration of the while loop, this rule is popped from the TODO-SET and relabelled so that $L_1[q \Rightarrow a] = \langle 0, 1, 0, 0\rangle$. Since $L_1[q \Rightarrow a] \neq L_0[q \Rightarrow a]$, the consequent a is relabelled, so that $L_1[a] = \langle 0, 1, 0, 0\rangle$. After that, $\neg a$ is considered for relabelling as well, but its label does not change: $L_1[\neg a] = \langle 1, 0, 0, 0\rangle$. At that point, the TODO-SET is empty and the algorithm terminates – which implies that for each $x \in \mathcal{L} \cup \mathcal{R} : L[x] = L_1[x]$.

$c_{\{u,o,b\}}(a) = 1$ because $L[a] = L_1[a] = \langle 0, 1, 0, 0\rangle$ and in $L_0[a]$ not all booleans in $\{u, o, b\}$ are turned false yet. $c_{\{d,o,b\}}(\neg a) = c_{\{d,o\}}(\neg a) = c_{\{d,b\}}(\neg a) = c_{\{o,b\}}(\neg a) = c_{\{d\}}(\neg a) = c_{\{o\}}(\neg a) = c_{\{b\}}(\neg a) = 0$: none of these (combinations of) booleans changes in any iteration of the while loop. Finally, note that $c_{\{d\}}(a) = \infty$, because a is labelled $L[a].d$ in the final labelling L.

Soundness proof

We start this section by some general remarks that we will use throughout the proofs, sometimes implicitly.

First, note that PREPROCESS makes sure that all literals or rules have a true u-boolean, see the next remark. The reason for this is that all literals and rules initially get a true u label in line 3, 4 or 6, and there is no operation in PREPROCESS that turns these booleans to False.

Remark 2 (Each u-boolean is True after PREPROCESS)**.** Given an argumentation theory $T = (AS, \mathcal{K})$ where $AS = (\mathcal{L}, ^{-}, \mathcal{R}, \leq)$ and $\mathcal{Q}$ is a set of queryables, let L_p be the labelling obtained by PREPROCESS (Algorithm 3) on $\mathcal{L}$, $\mathcal{R}$, $^{-}$, $\mathcal{Q}$ and $\mathcal{K}$. For each literal or rule in $x \in \mathcal{L} \cup \mathcal{R}$ the u-boolean in $L_p[x]$ is True.

Second, note that there is no operation before the first iteration of the while loop in STABILITY-LABEL that changes the labels of rules, so these labels remain unaffected. From the previous remark and Remark 1, it directly follows that for each rule, its u-label before the first iteration of the while loop is True.

Remark 3 (Each u boolean for rules is True before STABILITY-LABEL line 7)**.** Given an argumentation theory $T = (AS, \mathcal{K})$ where $AS = (\mathcal{L}, ^{-}, \mathcal{R}, \leq)$ and $\mathcal{Q}$ is a set of queryables, let L be the labelling obtained by STABILITY-LABEL (Algorithm 6) on $\mathcal{L}$, $\mathcal{R}$, $^{-}$, $\mathcal{Q}$ and $\mathcal{K}$. For each $r \in \mathcal{R} : L_0[r].u$ is True.

These remarks will be used to specify implications of a given labelling for (arguments in) future argumentation theories. For starters, an implication of the labelling $\neg L[r].u$ to a given rule r implies that there is an argument for that rule in the current argumentation theory.

Lemma B.1 (Argument existence labelling)**.** *Given an argumentation theory $T = (AS, \mathcal{K})$ where $AS = (\mathcal{L}, ^{-}, \mathcal{R}, \leq)$ and $\mathcal{Q}$ is a set of queryables, let L be the labelling obtained by STABILITY-LABEL (Algorithm 6) on $\mathcal{L}$, $\mathcal{R}$, $^{-}$, $\mathcal{Q}$ and $\mathcal{K}$. For each $r \in \mathcal{R}$: if r is labelled $\neg L[r].u$ then there is an argument with top rule r in Arg_T.*

Proof. Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, ^{-}, \mathcal{R}, \leq)$ and let $\mathcal{Q}$ be a set of queryables. Let L be the labelling obtained by Algorithm 6 on $\mathcal{L}$, $\mathcal{R}$, $^{-}$, $\mathcal{Q}$ and $\mathcal{K}$. We proceed by induction on $c_{\{u\}}$.

Proposition ($P(n)$): For each rule $r \in \mathcal{R}$ such that r is labelled $\neg L[r].u$ and $c_{\{u\}}(r) \leq n$, there is an argument with top rule r in Arg_T.

Base case ($P(1)$): Let r be an arbitrary rule in $\mathcal{R}$ such that $c_{\{u\}}(r) \leq 1$. Note that r cannot be labelled $\neg L[r].u$ by the PREPROCESS algorithm (see Remark 2). So this must have happened in the first iteration of the while loop ($c_{\{u\}}(r) = 1$) by Algorithm 6 (STABILITY-LABEL) line 9, in which Algorithm 5 is applied. Then the fact that r is labelled $\neg L[r].u$ can only be caused by case R-U-a, so for each $a \in \texttt{ants}(r) : \neg L[a].u$ and $c_{\{u\}}(a) = 0$. Then each $a \in \texttt{ants}(r)$ must have been labelled $\neg L[a].u$ in Algorithm 6 line 5, by Algorithm 4. Case L-U-b does not apply (see Remark 3), so by case L-U-a, each $a \in \texttt{ants}(r)$ is in $\mathcal{K}$. Then there is an

argument in Arg_T for each $a \in \texttt{ants}(r)$; therefore, there is an argument with top rule r in Arg_T.

Induction hypothesis ($P(k)$): For each rule $r \in \mathcal{R}$ such that $c_{\{u\}}(r) \leq k$, there is an argument with top rule r in Arg_T.

Induction step ($P(k+1)$): Now let r be an arbitrary rule in $\mathcal{R}$ such that $c_{\{u\}}(r) = k+1$. Then r must have been relabelled to $\neg L[r].u$ by Algorithm 6 line 9, by case R-U-a: for each $a \in \texttt{ants}(r) : \neg L[a].u$ and $c_{\{u\}} \leq k$. Consider an arbitrary antecedent $a \in \texttt{ants}(r)$. Given that $\neg L[a].u$, either $a \in \mathcal{K}$ (case L-U-a) or for each r' for a: r' is labelled $\neg L[r'].u$ in or before the k'th iteration of the while loop (case L-U-b). If $a \in \mathcal{K}$, then there is an observation-based argument for a in Arg_T. Otherwise, by the induction hypothesis, there is an argument with top rule r', for a, in Arg_T. Because we picked a arbitrarily, for each $a \in \texttt{ants}(r)$, there is an argument for a in Arg_T. And hence, there is an argument with top rule r in Arg_T.

At this point, we have proven $P(n)$ for all natural numbers $n \in \mathbb{N}$. Since Algorithm 6 has a polynomial runtime (Proposition 3.10) in terms of finite $\mathcal{L}$ and $\mathcal{R}$, each $r \in \mathcal{R}$ such that $\neg L[r].u$ must have some non-negative $i < n$ such that $c_{\{u\}}(r) \leq i$ – hence there is an argument with top rule r in Arg_T. □

The following lemma guarantees that no literal or rule is labelled $\langle 0,0,0,0\rangle$ by Stability-label. This is an important property that we will use in many proofs related to soundness and conditional completeness, because it enables us to make statements like: "given that some rule r for l is labelled $\neg L[r].u$ and $\neg L[r].o$, it is impossible that each rule r for l is labelled $\neg L[r].d$ and $\neg L[r].b$".

Lemma B.2 (No 4x False label)**.** *Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and $\leq = \emptyset$, let $\mathcal{Q}$ be a set of queryables and let L be the labelling obtained by Stability-label (Algorithm 6) on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. For each $x \in \mathcal{L} \cup \mathcal{R}$: $L[x] \neq \langle 0,0,0,0\rangle$.*

Proof. Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and let $\mathcal{Q}$ be a set of queryables. Let L be the labelling obtained by Stability-label (Algorithm 6) on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. We proceed by induction on the interim labelling of literals and rules.

Proposition ($P(n)$): For each $x \in \mathcal{L} \cup \mathcal{R}$ and for each non-negative integer $i < n$: $L_i[l] \neq \langle 0,0,0,0\rangle$.

Base case ($P(1)$): We make a distinction between rules and literals.

- Let $r \in \mathcal{R}$ be an arbitrary rule. There are two options: $L_p[r] = \langle 1,0,0,0\rangle$ or $L_p[r] = \langle 1,1,1,1\rangle$.
  - First suppose that $L_p[r] = \langle 1,0,0,0\rangle$. This implies that there is an antecedent $a \in \texttt{ants}(r)$ such that $L_p[a] = \langle 1,0,0,0\rangle$: otherwise, the L_p label of r would have been changed to $\langle 1,1,1,1\rangle$ by Algorithm 3 line 12.

So by Lemma 3.4, there is an antecedent $a \in \texttt{ants}(r)$ such that for each T' such that $T \sqsubseteq_Q T'$: a is unsatisfiable in T'. Consequently, there is no argument for a in Arg_T, so there is no argument with top rule r in Arg_T. Then, by Lemma B.1, r is not labelled $\neg L[r].u$ and hence $L_0[r] \neq \langle 0,0,0,0\rangle$.

- Alternatively, suppose that $L_p[r] = \langle 1,1,1,1\rangle$. Then, by Remark 1, $L_0[r] = L_p[r] = \langle 1,1,1,1\rangle$, which means that $L_0[r] \neq \langle 0,0,0,0\rangle$.

Recall that we picked r as an arbitrary rule in $\mathcal{R}$. Therefore, for each $r \in \mathcal{R}$: $L_0[r] \neq \langle 0,0,0,0\rangle$.

- Now suppose that l is an arbitrary literal. Again, we consider both $L_p[l] = \langle 1,0,0,0\rangle$ and $L_p[l] = \langle 1,1,1,1\rangle$.
  - First suppose that $L_p[l] = \langle 1,0,0,0\rangle$. Then $l \notin \mathcal{K}$: if $l \in \mathcal{K}$, there would be an argument for l in Arg_T, which would imply that l is not stable-unsatisfiable in T (Definitions 3.11 and 3.12), a contradiction to the soundness of Preprocess (Lemma 3.4). Suppose, towards a contradiction, that $L_0 = \langle 0,0,0,0\rangle$. By Algorithm 4 case L-U-b, there is a rule r for l that is labelled $\neg L[r].u$ before the start of the while loop. This however contradicts Remark 3. Therefore, $L_0[l] \neq \langle 0,0,0,0\rangle$.
  - Alternatively, $L_p[l] = \langle 1,1,1,1\rangle$. We consider two cases.
    - First assume that $l \in \mathcal{Q}$ and for each $l' \in \bar{l} : l' \notin \mathcal{K}$. This implies that l is considered for relabelling in Algorithm 6 line 5. In this relabelling step, none of the labelling rules L-D-a, L-D-b or L-D-c from Algorithm 4 applies, so l is labelled $L[l].d$. This means that $L_0[l] \neq \langle 0,0,0,0\rangle$.
    - Alternatively, either $l \notin \mathcal{Q}$ or there is an $l' \in \bar{l}$ such that $l' \in \mathcal{K}$. In both cases, the fact that $L_p[l] = \langle 1,1,1,1\rangle$ must have been caused by the fact that there is a rule r for l such that $L_p[r] = \langle 1,1,1,1\rangle$ (Algorithm 6 line 9). Rules are not relabelled before the first iteration of the while loop (Remark 1), hence there is a rule r for l such that $L_0[r] = L_p[r] = \langle 1,1,1,1\rangle$. As a result, neither L-U-a nor L-U-b from Algorithm 4 applies; therefore l is labelled $L[l].u$. So $L_0[l] \neq \langle 0,0,0,0\rangle$.

Since we chose l arbitrarily from $\mathcal{L}$, we conclude: for each $l \in \mathcal{L}$: $L_0[l] \neq \langle 0,0,0,0\rangle$.

Induction hypothesis ($P(k)$): For each $x \in \mathcal{L} \cup \mathcal{R}$ and for each non-negative integer $i < k$: $L_i \neq \langle 0,0,0,0\rangle$.

Induction step ($P(k+1)$): We will show that for each $x \in \mathcal{L} \cup \mathcal{R}$: $L_k[x] \neq \langle 0,0,0,0\rangle$.

- Let $r \in \mathcal{R}$ be an arbitrary rule. Suppose, towards a contradiction, that $L_k[r] = \langle 0,0,0,0\rangle$. Then $\neg L[r].d$, which must be caused by Algorithm 5 case R-D-a, so there is an antecedent $a \in \texttt{ants}(r)$ that is labelled $\neg L[a].d$. Furthermore,

$\neg L[r].b$ can be caused by either R-B-a or R-B-b, but in both cases, there is an antecedent $a \in \texttt{ants}(r)$ such that $\neg L[a].d$ and $\neg L[a].b$. Since $\neg L[r].o$ must be caused by R-O-a, there is an antecedent $a \in \texttt{ants}(r)$ such that $\neg L[a].d$ and $\neg L[a].o$ and $\neg L[a].b$. Finally, $\neg L[r].u$ implies that $\neg L[a].u$ for each antecedent a of r. But then there is an antecedent $a \in \texttt{ants}(r)$ such that $L_{k-1}[a] = \langle 0,0,0,0\rangle$. This contradicts the induction hypothesis, *there is no rule $r \in \mathcal{R}$ such that $L_k[r] = \langle 0,0,0,0\rangle$.*

- Now let $l \in \mathcal{L}$ be an arbitrary literal. Suppose, towards a contradiction, that $L_k[l] = \langle 0,0,0,0\rangle$.

  First we show that $l \notin \mathcal{Q}$: suppose that $l \in \mathcal{Q}$. This implies that $\neg L[\bar{l}].d$ must have been caused by Algorithm 4 case L-D-a, hence there is an $l' \in \bar{l}$ that is in $\mathcal{K}$, which by consistency of $\mathcal{K}$ implies that $l \notin \mathcal{K}$, so $\neg L[l].u$ must have been caused by L-U-b. Furthermore, there is some $l' \in \bar{l}$ such that no $l'' \in \overline{l'}$ is in $\mathcal{K}$, so $\neg L[l].o$ must have been caused by L-O-f. But that implies that there exists a rule r for l such that $L_k[r] = \langle 0,0,0,0\rangle$, which contradicts our earlier conclusion that such a rule does not exist. Therefore $l \notin \mathcal{Q}$.

  Given that $l \notin \mathcal{Q}$, $\neg L[l].u$ must have been caused by case L-U-b. Then the fact that l is labelled $\neg L[l].o$ must be caused by either L-O-d or L-O-e; in both cases, there is a rule r for l with $\neg L[r].u$ and $\neg L[r].o$. Since there is no rule labelled $\langle 0,0,0,0\rangle$ by L_k, $\neg L[l].b$ must have been caused by L-B-c or L-B-d. In both cases, there is a rule r for l with $\neg L[r].u$, $\neg L[r].o$ and $\neg L[r].b$ and for each $l' \in \bar{l}$: for each rule r' for l': $\neg L[r'].d$ and $\neg L[r'].b$. Finally, $\neg L[l].d$ can be caused by either L-D-b or L-D-c. However, both cases contradict our earlier conclusion that there is no rule labelled $\langle 0,0,0,0\rangle$ by L_k. So for each $l \in \mathcal{L}$: $L_k[l] \neq \langle 0,0,0,0\rangle$.

At this point we have proven our proposition $P(n)$ for all natural numbers $n \in \mathbb{N}$: for each $x \in \mathcal{L} \cup \mathcal{R}$ and for each non-negative integer $i < n$: $L_i[x] \neq \langle 0,0,0,0\rangle$. Given that Algorithm 6 terminates (see Proposition 3.10), there is some finite i such that $L_i[x] = L[x]$ for all $x \in \mathcal{L} \cup \mathcal{R}$. Therefore for each $x \in \mathcal{L} \cup \mathcal{R}$: $\mathcal{L}[x] \neq \langle 0,0,0,0\rangle$. □

The next lemma shows that any argument in the current argumentation theory is retained in each future argumentation theory. We will repeatedly use this lemma in our soundness proofs.

Lemma B.3 (Argument persistence in future argumentation theories)**.** *Let $T = (AS, \mathcal{K})$ be an argumentation theory with argumentation system $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and let A be an argument in Arg_T. Then for each set of queryables $\mathcal{Q}$, for each T' such that $T \sqsubseteq_{\mathcal{Q}} T'$: $A \in Arg_{T'}$.*

Proof. Suppose that $T = (AS, \mathcal{K})$ is an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and let $\mathcal{Q}$ be a set of queryables. Let $T' = (AS, \mathcal{K}')$ be an arbitrary future argumentation theory of T w.r.t. $\mathcal{Q}$. By Definition 3.11 of future argumentation theories, $\mathcal{K} \subseteq \mathcal{K}'$. From the definition of arguments (Definition 3.4), it follows that arguments are constructed only based on their knowledge base and rule set, so given that A

can be constructed from $\mathcal{K}$ and $\mathcal{R}$, A can also be constructed from $\mathcal{K}'$ and $\mathcal{R}$. This implies that $A \in Arg_{T'}$. Since T' is an arbitrary argumentation theory of T w.r.t. $\mathcal{Q}$, this generalises to all future argumentation theories. □

Next, we discuss the situations in which the booleans representing the statuses for rules are turned to False. Due to the way the labelling rules are defined, there are many situations in which either the u- and o-boolean or the d- and b-boolean are turned False together. Lemma B.4 analyses the situation that both the unsatisfiable and the out boolean for a rule are turned to False. In this situation, there must be some argument with that top rule in each future argumentation theory T' that is not attacked *on a subargument* by an argument in the grounded extension $G(T')$. In the next definition, we formally define attacks on a subargument and their counterpart: attacks on a conclusion.

Definition B.5 (Attack on conclusion or on a subargument)**.** Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and let A and B be two arguments in Arg_T. If A attacks B on B, we say that A attacks B *on its conclusion*; otherwise (A attacks B on B' and $B' \neq B$), we say that A attacks B *on a subargument*.

Using Definition B.5 we can now prove Lemma B.4:

Lemma B.4 (Labelled not unsatisfiable or out)**.** *Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and $\leq = \emptyset$ and $\mathcal{Q}$ is a set of queryables; furthermore let L be the labelling obtained by* Stability-label *(Algorithm 6) on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. For each $r \in \mathcal{R}$: if r is labelled $\neg L[r].u$ and $\neg L[r].o$, then for each T' such that $T \sqsubseteq_Q T'$, there is an argument with top rule r in $Arg_{T'}$ that is not attacked on a subargument by any argument in $G(T')$.*

Proof. Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and let $\mathcal{Q}$ be a set of queryables. Let L be the labelling obtained by Stability-label (Algorithm 6) on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. We proceed by induction on $c_{\{u,o\}}$.

Proposition ($P(n)$): For each rule $r \in \mathcal{R}$ such that $c_{\{u,o\}}(r) \leq n$: if r is labelled $\neg L[r].u$ and $\neg L[r].o$, then for each T' such that $T \sqsubseteq_Q T'$, there is an argument with top rule r in $Arg_{T'}$ that is not attacked on a subargument by any argument in $G(T')$.

Base case ($P(1)$): Let r be an arbitrary rule in $\mathcal{R}$ such that $c_{\{u,o\}}(r) \leq 1$ and $\neg L[r].u$ and $\neg L[r].o$. Since $\neg L[r].u$ can neither be assigned by Preprocess nor before the first iteration of the while loop (Remark 3), $c_{\{u,o\}}(r) = 1$. Then $\neg L[r].u$ and $\neg L[r].o$ must have been caused by Algorithm 5 case R-U-a and R-O-a: each $a \in \texttt{ants}(r)$ is labelled $\neg L[a].u$ and $\neg L[a].o$ and $c_{\{u,o\}}(a) = 0$. Then each $a \in \texttt{ants}(r)$ is in $\mathcal{K}$: the $\neg L[a].u$ label before the first iteration of the while loop cannot be caused by Algorithm 4 case L-U-b (see Remark 3), and must hence have been caused by case L-U-a. Consequently, there is an observation-based argument for each of r's antecedents, hence there is an argument A with top rule r in Arg_T. Then, by

Definition 3.5 and Lemma B.3, for each T' such that $T \sqsubseteq_Q T'$, A is not attacked on a subargument by any argument in $G(T')$.

Induction hypothesis ($P(k)$): For each $r \in \mathcal{R}$ such that $c_{\{u,o\}}(r) \leq k$: if r is labelled $\neg L[r].u$ and $\neg L[r].o$, then for each T' such that $T \sqsubseteq_Q T'$, there is an argument with top rule r in $Arg_{T'}$ that is not attacked on a subargument by any argument in $G(T')$.

Induction step ($P(k+1)$): Now let r be an arbitrary rule in $\mathcal{R}$ such that $c_{\{u,o\}}(r) \leq k+1$ and $\neg L[r].u$ and $\neg L[r].o$. By Algorithm 5 case R-U-a and R-O-a, for each $a \in \texttt{ants}(r)$: $\neg L[a].u$, $\neg L[a].o$ and $c_{\{u,o\}}(a) \leq k$. Now let a be an arbitrary antecedent in $\texttt{ants}(r)$. We consider two cases:

- If $a \in \mathcal{K}$, then for each T' such that $T \sqsubseteq_Q T'$ there is an observation-based argument for a in $Arg_{T'}$ that, by Lemma 3.8, is not attacked by any argument in $G(T')$.
- Alternatively, $a \notin \mathcal{K}$. Then the label $\neg L[a].u$ must have been labelled by case L-U-b. This implies that the label $\neg L[a].o$ must have been by case L-O-b, L-O-c, L-O-d or L-O-e. In any case: (1) there is a rule r' for a that is labelled $\neg L[r'].u$ and $\neg L[r'].o$ and (2) if $a \in \mathcal{Q}$ then for each $a' \in \overline{a}$ there is some $a'' \in \overline{a}$ that is in $\mathcal{K}$. Given that $c_{\{u,o\}}(a) \leq k$, it follows that $c_{\{u,o\}}(r') \leq k$. So by the induction hypothesis and (1): for each T' such that $T \sqsubseteq_Q T'$, there is an argument for a (with top rule r') in $Arg_{T'}$ that is not attacked *on a subargument* by an argument in $G(T')$. Furthermore, by (2) there is no T' such that $T \sqsubseteq_Q T'$ such that there is an observation-based argument for any $a' \in \overline{a}$ in $Arg_{T'}$, so by Lemma 3.8 no argument for a in $Arg_{T'}$ for any T' such that $T \sqsubseteq_Q T'$ can be attacked *on its conclusion* by an argument in $G(T')$.

Given that for each T' such that $T \sqsubseteq_Q T'$, for each $a \in \texttt{ants}(r)$, there is an argument for a in $Arg_{T'}$ that is not attacked by any argument in $G(T')$, we can construct an argument with top rule r in $Arg_{T'}$ that is not attacked *on a subargument* by any argument in $G(T')$.

Finally, recall that $c_{\{u,o\}}(r)$ is finite for each $r \in \mathcal{R}$ that is labelled $\neg L[r].u$ and $\neg L[r].o$ (Definition B.4), which means that the proposition is valid for each $r \in \mathcal{R}$ in general. □

Lemma B.5 analyses the situation that a rule $r \in \mathcal{R}$ for l is labelled $\neg L[r].d$ and $\neg L[r].b$ and shows that l cannot be defended or blocked in any future argumentation theory of T *thanks to that rule*, that is: there is no future argumentation theory T' in which there is an argument with top rule r that is not attacked by an argument $G(T)$.

Lemma B.5 (Labelled not defended or blocked)**.** *Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and $\leq = \emptyset$ and let $\mathcal{Q}$ be a set of queryables. Let L be the labelling obtained by* Stability-label *(Algorithm 6) and L_p be the labelling obtained by* Preprocess *(Algorithm 3) on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. If $r \in \mathcal{R}$*

such that $L_p[r] = \langle 1,1,1,1\rangle$ *and* r *is labelled* $\neg L[r].d$ *and* $\neg L[r].b$*, then for each* T' *such that* $T \sqsubseteq_Q T'$*, any argument with top rule* r *is attacked by an argument in* $G(T')$*.*

Proof. Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and let $\mathcal{Q}$ be a set of queryables. Let L be the labelling obtained by Stability-label (Algorithm 6) and L_p be the labelling obtained by Preprocess (Algorithm 3) on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. We proceed by induction on $c_{\{d,b\}}$.

Proposition ($P(n)$): For each $r \in \mathcal{R}$ such that $c_{\{d,b\}}(r) \leq n$: if $L_p[r] = \langle 1,1,1,1\rangle$ and r is labelled $\neg L[r].d$ and $\neg L[r].b$, then for each T' such that $T \sqsubseteq_Q T'$, each argument with top rule r in $Arg_{T'}$ is attacked by an argument in $G(T')$.

Base case ($P(1)$): Let r be an arbitrary rule such that $c_{\{d,b\}}(r) \leq 1$ and $L_p[r] = \langle 1,1,1,1\rangle$ and suppose that r is labelled $\neg L[r].d$ and $\neg L[r].b$. Since $L_p[r] = \langle 1,1,1,1\rangle$, r must be labelled $\neg L[r].d$ and $\neg L[r].b$ by Algorithm 6 and by Remark 1 $c_{\{d,b\}}(r) = 1$.

The labelling of r as $\neg L[r].d$ and $\neg L[r].b$ must have happened in line 9, caused by Algorithm 5. Then by case R-D-a and case R-B-a or R-B-b, some $a \in \texttt{ants}(r)$ must have been labelled $\neg L[a].d$ and $\neg L[a].b$ and $c_{\{d,b\}}(a) = 0$.

Given that $c_{\{d,b\}}(a) = 0$, a must have been relabelled either in the preprocessing step or in Algorithm 6 line 5. However, if a was labelled $\neg L[a].d$ and $\neg L[a].b$ in the preprocessing step, then $L_p[a] = \langle 1,0,0,0\rangle$, which means that $L_p[r] = \langle 1,0,0,0\rangle$ (since Algorithm 3 line 11 would not have applied for r), which would contradict our assumption that $L_p[r] = \langle 1,1,1,1\rangle$. So a must have been labelled $\neg L[a].d$ and $\neg L[a].b$ in Algorithm 6 line 5. Then the condition in line 4 must have been true: either $a \in \mathcal{Q}$ or there is no rule for a in $\mathcal{R}$.

Next, we show by contradiction that $a \in \mathcal{Q}$: suppose that $a \notin \mathcal{Q}$. Then by the condition in Algorithm 6 line 4, there is no rule for a in $\mathcal{R}$. But in that case, a would have been labelled as $L_p[a] = \langle 1,0,0,0\rangle$ in the preprocessing step (line 4) and never relabelled to $\langle 1,1,1,1\rangle$ because the condition of line 7 would never apply (given that there is no rule for a in $\mathcal{R}$). This contradicts our earlier conclusion that $L_p[a] = \langle 1,1,1,1\rangle$. Therefore, $a \in \mathcal{Q}$.

Given that $a \in \mathcal{Q}$ and a is relabelled in Algorithm 6 line 5, by case L-D-a, there is some $a' \in \overline{a}$ such that $a' \in \mathcal{K}$. Then there is an observation-based argument for a' in Arg_T – and by Lemma B.3, for each T' such that $T \sqsubseteq_Q T'$: $a' \in Arg_{T'}$. So by Lemma 3.8, for each T' such that $T \sqsubseteq_Q T'$, each argument for a in $Arg_{T'}$ is attacked by an argument in $G(T')$.

Given that a is an antecedent of r, we can derive that *for each* T' *such that* $T \sqsubseteq_Q T'$*, each argument with top rule* r *in* $Arg_{T'}$ *is attacked by an argument in* $G(T')$. Since we chose r as an arbitrary rule in $\mathcal{R}$ such that $c_{\{d,b\}}(r) \leq 1$, we can generalise this to all rules that are labelled $\langle 1,1,1,1\rangle$ by L_p.

Induction hypothesis ($P(k)$): For each $r \in \mathcal{R}$ such that $c_{\{d,b\}}(r) \leq k$, if $L_p[r] = \langle 1,1,1,1\rangle$ and r is labelled $\neg L[r].d$ and $\neg L[r].b$ then for each T' such that $T \sqsubseteq_Q T'$, each argument with top rule $r \in Arg_{T'}$ is attacked by an argument in $G(T')$.

Induction step ($P(k+1)$): Let r be an arbitrary rule in $\mathcal{R}$ such that $c_{\{d,b\}}(r) = k+1$ and assume that $L_p[r] = \langle 1,1,1,1\rangle$ and r is labelled $\neg L[r].d$ and $\neg L[r].b$. By Algorithm 5 case R-D-a and either R-B-a or R-B-b, there is an antecedent a of r that is labelled $\neg L[a].d$ and $\neg L[a].b$ and $c_{\{d,b\}}(a) \leq k$. We distinguish two cases: $a \in \mathcal{Q}$ and $a \notin \mathcal{Q}$.

- If $a \in \mathcal{Q}$, then by case L-D-a there is some $a' \in \overline{a}$ such that $a' \in \mathcal{K}$. So there is an observation-based argument A for a' that is in $Arg_{T'}$ for every T' such that $T \sqsubseteq_Q T'$. Being observation-based, A cannot be attacked and therefore $A \in G(T')$ for every T' such that $T \sqsubseteq_Q T'$. So *for each T' such that $T \sqsubseteq_Q T'$: each argument for a in $Arg_{T'}$ is attacked by an argument in $G(T')$*.
- Now suppose that $a \notin \mathcal{Q}$. Then by the labelling rules of Algorithm 5, *each rule r' for a is labelled $\neg L[r'].d$ and $\neg L[r'].b$ and $c_{\{d,b\}}(r') \leq k$*: if L-D-c caused $\neg L[a].d$, then by Lemma B.2, a must have been labelled $\neg L[a].b$ by case L-B-b; if, alternatively, L-D-b caused $\neg L[a].d$, then either L-B-b or L-B-c caused $\neg L[a].b$.

  Consider an arbitrary rule r' for a. Either $L_p[r'] = \langle 1,0,0,0\rangle$ or $L_p[r'] = \langle 1,1,1,1\rangle$. If $L_p[r'] = \langle 1,0,0,0\rangle$, some $a' \in \texttt{ants}(r')$ is labelled $L_p[a'] = \langle 1,0,0,0\rangle$, so by Lemma 3.4 there is no argument with top rule r' in any $Arg_{T'}$ where $T \sqsubseteq_Q T'$. If, alternatively, $L_p[r'] = \langle 1,1,1,1\rangle$ (which must be the case for at least one rule for a, since $L_p[a] = \langle 1,1,1,1\rangle$), we apply the induction hypothesis: for each T' such that $T \sqsubseteq_Q T'$: each argument with top rule r' in $Arg_{T'}$ is attacked by an argument in $G(T')$. Since $a \notin \mathcal{Q}$, there are no observation-based arguments for a either, so: *for each T' such that $T \sqsubseteq_Q T'$, each argument for a in $Arg_{T'}$ is attacked by an argument in $G(T')$*.

By generalising r, we can now derive that for each rule $r \in \mathcal{R}$ such that $c_{\{d,b\}} \leq k+1$: if r is labelled $\neg L[r].d$ and $\neg L[r].b$, then for each T' such that $T \sqsubseteq_Q T'$, each argument with top rule r in $Arg_{T'}$ is attacked by an argument in $G(T')$. Finally, remember that for each rule $r \in \mathcal{R}$ that is labelled $\neg L[r].d$ and $\neg L[r].b$, $c_{\{d,b\}}(r)$ is finite (Definition B.4), which means that the proposition is valid for each $r \in \mathcal{R}$ in general. □

In the next lemma, we show that no literal $l \in \mathcal{L}$ can be labelled $L[l] = \langle 1,0,0,0\rangle$ *after* the preprocessing step, that is: if $L_p[l] \neq \langle 1,0,0,0\rangle$. We will use this lemma for proving soundness of stable-unsatisfiable labelling.

Lemma B.6 (Stable-unsatisfiable labelling only in preprocessing)**.** *Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$, let $\mathcal{Q}$ be a set of queryable literals and let L be the labelling obtained by* Stability-label *(Algorithm 6) and L_p the labelling obtained by* Preprocess *(Algorithm 3) on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. For each literal $l \in \mathcal{L}$: if $L[l] = \langle 1,0,0,0\rangle$ then $L_p[l] = \langle 1,0,0,0\rangle$.*

Proof. Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and let $\mathcal{Q}$ be a set of queryables. Let L be the labelling obtained by STABILITY-LABEL (Algorithm 6) on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. We will, towards a proof by contraposition, first show that for each $l \in \mathcal{L}$ such that $L_p[l] \neq \langle 1,0,0,0\rangle$ it must be that $L[l] \neq \langle 1,0,0,0\rangle$. We prove this by induction on the interim labelling of literals.

Proposition $P(n)$: For each $l \in \mathcal{L}$: if $L_p[l] = \langle 1,1,1,1\rangle$ then for each non-negative integer $i < n$: $L_i[l] \neq \langle 1,0,0,0\rangle$.

Base case ($P(1)$): Let $l \in \mathcal{L}$ be an arbitrary literal such that $L_p[l] = \langle 1,1,1,1\rangle$ and suppose that $L_0[l] = \langle 1,0,0,0\rangle$.

Then l must have been relabelled by Algorithm 6 line 5, using Algorithm 4. Given that l is labelled $\neg L[l].d$, either $l \notin \mathcal{Q}$ or there is some $l' \in \bar{l}$ such that $l' \in \mathcal{K}$. Therefore, $L_p[l] = \langle 1,1,1,1\rangle$ cannot be caused by Algorithm 3 line 3 and hence must be caused by Algorithm 3 line 13. This means that Algorithm 3 line 12 must have been executed as well, so there is some rule r for l such that $L_p[r] = \langle 1,1,1,1\rangle$. Then by Remark 1 *there is some rule* r *for* l *such that* $L_0[r] = \langle 1,1,1,1\rangle$.

Furthermore, given that l is relabelled by Algorithm 6 line 5, the condition in Algorithm 6 line 4 must have applied, which implies that $l \in \mathcal{Q}$. Consequently, there is an $l' \in \bar{l}$ such that $l' \in \mathcal{K}$. But then no labelling rule in Algorithm 4 derives $\neg L[l].o$. This contradicts our assumption that $L_0[l] = \langle 1,0,0,0\rangle$. Therefore, *for each* $l \in \mathcal{L}$ *such that* $L_p[l] = \langle 1,1,1,1\rangle$*, for each non-negative integer* $i < 1$*:* $L_i[l] \neq \langle 1,0,0,0\rangle$.

Induction hypothesis ($P(k)$): For each $l \in \mathcal{L}$: if $L_p[l] = \langle 1,1,1,1\rangle$ then for each non-negative integer $i < k$: $L_i[l] \neq \langle 1,0,0,0\rangle$.

Induction step ($P(k+1)$): Let l be an arbitrary literal such that $L_p[l] = \langle 1,1,1,1\rangle$. Suppose, towards a contradiction, that $L_k[l] = \langle 1,0,0,0\rangle$. Then each rule r for l is labelled $L_i[r] = \langle 1,0,0,0\rangle$, as we will show next.

- First suppose that $l \in \mathcal{Q}$. Then there is some $l' \in \bar{l}$ such that $l' \in \mathcal{K}$ by case L-D-a, so $l \notin \mathcal{K}$. Then case L-O-f must have applied, so *each rule* r *for* l *is labelled* $L_k[r] = \langle 1,0,0,0\rangle$.
- Alternatively, suppose that $l \notin \mathcal{Q}$. We next show by contradiction that each rule r for l is labelled $\neg L[r].d$: suppose that there is some rule r for l that is labelled $L[r].d$. Then the fact that l is labelled $\neg L[l].d$ must have been caused by case L-D-c and, by Lemma B.2, $\neg L[l].b$ must be caused by case L-B-b: each rule r for l is labelled $\neg L[r].d$ and $\neg L[r].b$. But that contradicts our assumption, so each rule r for l is labelled $\neg L[r].d$.

  Given that each rule r for l is labelled $\neg L[r].d$, the fact that l is labelled $\neg L[l].b$ must be caused by case L-B-b or L-B-c; as a result, the fact that l is labelled $\neg L[l].o$ must be caused by case L-O-f or L-O-d. In both cases, by Lemma B.2, *each rule* r *for* l *is labelled* $L_k[r] = \langle 1,0,0,0\rangle$.

Now let r be an arbitrary rule for l; we know that $L_k[r] = \langle 1,0,0,0\rangle$. The fact that r is labelled $\neg L[r].d$, $\neg L[r].o$ and $\neg L[r].b$ must have been caused by Algorithm 5 case R-

D-a, case R-B-a or R-B-b and case R-O-a: there is some $a \in \texttt{ants}(r)$ that is labelled $\neg L[a].d$ and $\neg L[a].o$ and $\neg L[a].b$. Note that a must have been labelled as such before the $(k-1)$'th iteration of the while loop. By Lemma B.2, $L_{k-1}[a] = \langle 1,0,0,0\rangle$.

But then by the induction hypothesis, a was already labelled stable-unsatisfiable by L_p. Given that $L_p[a] = \langle 1,0,0,0\rangle$, the condition in Algorithm 3 line 11 would not have applied for r, so r would have been labelled $L_p[r] = \langle 1,0,0,0\rangle$. Since we chose r arbitrary, we know that $L_p[r] = \langle 1,0,0,0\rangle$ for each rule r for l. But that implies that l is labelled stable-unsatisfiable in Algorithm 3 line 4 and is not re-labelled to $\langle 1,1,1,1\rangle$ in Algorithm 3 line 13. So $L_p[l] = \langle 1,0,0,0\rangle$. This contradicts our assumption that $L_p[l] = \langle 1,1,1,1\rangle$. Hence *for each* $l \in \mathcal{L}$ *such that* $L_p[l] = \langle 1,1,1,1\rangle$, *for each non-negative integer* $i < k+1$*:* $L_i[l] \neq \langle 1,0,0,0\rangle$.

At this point, we have proven $P(i)$ for each finite non-negative $i < n$ and for each $l \in \mathcal{L}$. Since Algorithm 6 has a polynomial runtime (Proposition 3.10), there must be some finite non-negative $i < n$ such that for each $l \in \mathcal{L}$: $L[l] = L_i[l]$. By contraposition, this implies for each $l \in \mathcal{L}$: *if* $L[l] = \langle 1,0,0,0\rangle$ *then* $L_p[l] \neq \langle 1,1,1,1\rangle$ and therefore $L_p[l] = \langle 1,0,0,0\rangle$. □

Having shown that literals can only be labelled as unsatisfiable in the prepocessing step, it is a small step to prove that Stability-label is sound for the stable-unsatisfiable cases.

Lemma B.7 (Soundness stable-unsatisfiable labelling)**.** *Let* $T = (AS, \mathcal{K})$ *be an argumentation theory where* $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$*, let* $\mathcal{Q}$ *be the set of queryable literals and let* L *be the labelling obtained by* Stability-label *(Algorithm 6) on* $\mathcal{L}$*,* $\mathcal{R}$*,* $\bar{}$*,* $\mathcal{Q}$ *and* $\mathcal{K}$*. For each* $l \in \mathcal{L}$*: if* $L[l] = \langle 1,0,0,0\rangle$ *then* l *is stable-unsatisfiable in* T *w.r.t.* $\mathcal{Q}$*.*

Proof. Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and let $\mathcal{Q}$ be a set of queryables. Let L be the labelling obtained by Stability-label on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. Each literal $l \in \mathcal{L}$ that is labelled $L[l] = \langle 1,0,0,0\rangle$ was already labelled $L_p[l] = \langle 1,0,0,0\rangle$ by Lemma B.6, which by Lemma 3.4 implies that l is stable-unsatisfiable in T w.r.t. $\mathcal{Q}$. □

Next, we show that Stability-label is sound for literals that are labelled stable-defended.

Lemma B.8 (Soundness stable-defended labelling)**.** *Let* $T = (AS, \mathcal{K})$ *be an argumentation theory where* $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ *and* $\leq = \emptyset$*, let* $\mathcal{Q}$ *be the set of queryable literals and let* L *be the labelling obtained by* Stability-label *(Algorithm 6) on* $\mathcal{L}$*,* $\mathcal{R}$*,* $\bar{}$*,* $\mathcal{Q}$ *and* $\mathcal{K}$*. For each literal* $l \in \mathcal{L}$*: if* $L[l] = \langle 0,1,0,0\rangle$ *then* l *is stable-defended in* T *w.r.t.* $\mathcal{Q}$*.*

Proof. Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and let $\mathcal{Q}$ be a set of queryables. Let L be the labelling obtained by Stability-label (Algorithm 6) on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. We proceed by induction on $c_{\{u,o,b\}}(l)$.

Proposition ($P(n)$): For each $l \in \mathcal{L}$ such that $c_{\{u,o,b\}}(l) \leq n$: if $L[l] = \langle 0,1,0,0\rangle$ then l is stable-defended in T w.r.t. $\mathcal{Q}$.

Base case ($P(0)$): Let l be an arbitrary literal in $\mathcal{L}$ such that $c_{\{u,o,b\}}(l) \leq 0$ and suppose that $L[l] = \langle 0,1,0,0\rangle$. Note that $L[l].d$ implies that $L_p[l] = \langle 1,1,1,1\rangle$. It follows that l must have been relabelled to $\langle 0,1,0,0\rangle$ by Algorithm 6 line 5. At that moment, all rules $r \in \mathcal{R}$ are labelled $L[r].u$ (Remark 3) hence the $\neg L[l].u$ label cannot be caused by Algorithm 4 L-U-b. So case L-U-a must have applied: $l \in \mathcal{K}$. Then there is an observation-based argument for l in each T' such that $T \sqsubseteq_Q T'$ (Definition 3.11) that cannot be attacked (so it cannot be defeated either) and therefore is in the grounded extension. Consequently, l is stable-defended in T w.r.t. $\mathcal{Q}$.

Induction hypothesis ($P(k)$): For each $l \in \mathcal{L}$ such that $c_{\{u,o,b\}}(l) \leq k$: if $L[l] = \langle 0,1,0,0\rangle$ then l is stable-defended in T w.r.t. $\mathcal{Q}$.

Induction step ($P(k+1)$): Now consider an arbitrary literal $l \in \mathcal{L}$ such that $c_{\{u,o,b\}}(l) = k+1$ and suppose that $L[l] = \langle 0,1,0,0\rangle$.

First suppose towards a contradiction that $l \in \mathcal{Q}$. Note that $l \in \mathcal{K}$, since otherwise there would be a rule r for l with $\neg L[r].u$ (L-U-b) and for each rule r for l: $\neg L[r].d$, $\neg L[r].o$ and $\neg L[r].b$ (L-O-f), a contradiction with Lemma B.2. However, this means that l would be labelled stable-defended before the start of the while loop, which contradicts our assumption that $c_{\{u,o,b\}}(l) = k+1$ for positive k. Consequently, $l \notin \mathcal{Q}$.

Given that $l \notin \mathcal{Q}$, the label $\neg L[l].u$ must have been caused by Algorithm 4 case L-U-b. Then $\neg L[l].o$ must have been caused by either L-O-d or L-O-e and $\neg L[l].b$ must have been caused by either L-B-c or L-B-d (L-B-a contradicts $l \notin \mathcal{Q}$; L-B-b contradicts Lemma B.2). It follows that there is *a rule r for l with $\neg L[r].u$, $\neg L[r].o$ and* $\neg L[r].b$ and *for each $l' \in \bar{l}$, for each rule r' for l': $\neg L[r'].d$ and* $\neg L[r'].b$. Consider both conclusions:

- Let r be the rule for l such that $\neg L[r].u$, $\neg L[r].o$ and $\neg L[r].b$. The fact that r is labelled $\neg L[r].u$ must have been caused by R-U-a; then $\neg L[r].o$ has to be caused by R-O-a and $\neg L[r].b$ must have been caused by R-B-a, so each antecedent a of r is labelled $\neg L[a].u$, $\neg L[a].o$ and $\neg L[a].b$. By Lemma B.2, the d-boolean must have been positive: for each $a \in \texttt{ants}(r) : L_k[a] = \langle 0,1,0,0\rangle$. Then by the induction hypothesis, each antecedent $a \in \texttt{ants}(r)$ is stable-defended in T w.r.t. $\mathcal{Q}$. So by Definition 3.12, for each $a \in \texttt{ants}(r)$, for each T' such that $T \sqsubseteq_Q T'$, there is an argument for a in $G(T')$. By Lemma 3.7, each argument attacking this argument is attacked by an observation-based argument (which is unattacked and therefore in the grounded extension). To conclude, for each T' such that $T \sqsubseteq_Q T'$, there is an argument A with top rule r in $Arg_{T'}$ such that each argument B attacking A *on a subargument* in T' is attacked by an observation-based argument in $G(T')$.
- Now consider that each rule r' for some $l' \in \bar{l}$ is labelled $\neg L[r'].d$ and $\neg L[r'].b$.

By Lemma B.5, this implies that for each T' such that $T \sqsubseteq_Q T'$, for each $l' \in \bar{l}$, for each rule r' for l', each argument with top rule r in $Arg_{T'}$ is attacked by an argument in $G(T')$. Thus for each T' such that $T \sqsubseteq_Q T'$, each argument attacking an argument for l *on its conclusion* in T' is attacked by an argument in the grounded extension $Arg_{T'}$.

To summarize, for each T' such that $T \sqsubseteq_Q T'$, there is some argument A (with top rule r, for l) in $Arg_{T'}$ such that each argument B in $Arg_{T'}$ attacking A (either on a subargument or on its conclusion l) is attacked by an argument in the grounded extension Arg_T. Given that $\leq = \emptyset$, all attacks succeed as defeats. Then by Definitions 2.1, 3.9 and 3.12, l *is stable-defended in* T *w.r.t.* $\mathcal{Q}$. Since we chose l arbitrarily, this concludes the induction step.

At this point, we have shown that for each $n \in \mathbb{N}$: for each $l \in \mathcal{L}$: if $c_{\{u,o,b\}}(l) \leq n$ and $L[l] = \langle 0,1,0,0\rangle$ then l is stable-defended in T w.r.t. $\mathcal{Q}$. Given that for each $l \in \mathcal{L}$ such that $L[l] = \langle 0,1,0,0\rangle$, $c_{\{u,o,b\}}(l)$ is finite (Definition B.4), we derive: *for each* $l \in \mathcal{L}$*: if* $L[l] = \langle 0,1,0,0\rangle$ *then* l *is stable-defended in* T *w.r.t.* $\mathcal{Q}$. □

We continue with showing soundness for literals that are labelled stable-out by Stability-label.

Lemma B.9 (Soundness stable-out labelling)**.** *Let* $T = (AS, \mathcal{K})$ *be an argumentation theory where* $AS = (\mathcal{L}, {}^{-}, \mathcal{R}, \leq)$ *and* $\leq = \emptyset$*, let* $\mathcal{Q}$ *be the set of queryable literals and let* L *be the labelling obtained by* Stability-label *(Algorithm 6) on* $\mathcal{L}$*,* $\mathcal{R}$*,* $^{-}$*,* $\mathcal{Q}$ *and* $\mathcal{K}$*. For each* $l \in \mathcal{L}$*: if* $L[l] = \langle 0,0,1,0\rangle$ *then* l *is stable-out in* T *w.r.t.* $\mathcal{Q}$*.*

Proof. Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, {}^{-}, \mathcal{R}, \leq)$ and let $\mathcal{Q}$ be a set of queryables. Let L be the labelling obtained by Stability-label (Algorithm 6) on $\mathcal{L}$, $\mathcal{R}$, $^{-}$, $\mathcal{Q}$ and $\mathcal{K}$. Suppose that $l \in \mathcal{L}$ is labelled $L[l] = \langle 0,0,1,0\rangle$ by Stability-label (Algorithm 6). Note that this implies that $L_p[l] = \langle 1,1,1,1\rangle$, so the fact that l is labelled $\neg L[l].u$, $\neg L[l].d$ and $\neg L[l].b$ must have been caused by Algorithm 4. We consider two cases: $l \in \mathcal{Q}$ and $l \notin \mathcal{Q}$.

- First suppose that $l \in \mathcal{Q}$. Then the fact that l is labelled $\neg L[l].d$ must have been caused by Algorithm 4 case L-D-a, so there is some $l' \in \bar{l}$ such that $l' \in \mathcal{K}$. Knowledge bases are consistent, therefore $l \notin \mathcal{K}$. This means that $\neg L[l].u$ cannot be caused by Algorithm 4 case L-U-a, hence must have been caused by case L-U-b: there is a rule r for l with $\neg L[r].u$. Then by Lemma B.1, there is an argument for l with top rule r in Arg_T. However, since there is some $l' \in \bar{l}$ such that $l' \in \mathcal{K}$, each argument for l is attacked by the observation-based argument for l'. Hence, by Lemmas B.3 and 3.8, *for each* T' *such that* $T \sqsubseteq_Q T'$*,* l *is out in* T'.
- Now suppose that $l \notin \mathcal{Q}$. Then $l \notin \mathcal{K}$, so label $\neg L[l].u$ must have been caused by case L-U-b: there is a rule r for l with $\neg L[r].u$. Then by Lemma B.1, there is an argument with top rule r in Arg_T. Furthermore, since l is labelled $L[l].o$, we know that there is no rule r for l with $\neg L[r].u$ and $\neg L[r].o$ (otherwise

Algorithm 4 case L-O-e would apply). As a result, L-B-d cannot apply, so the fact that l is labelled $\neg L[l].b$ must be caused by either L-B-b or L-B-c. In both cases, for each rule r for l: $\neg L[r].d$ and $\neg L[r].b$. Now, by Lemma B.5: for each T' such that $T \sqsubseteq_Q T'$, each argument for l is attacked (and, given that $\leq = \emptyset$, defeated) by the grounded extension. Therefore, by Definition 3.9, *l is out in T' for each T' such that $T \sqsubseteq_Q T'$.*

To conclude, *for each $l \in \mathcal{L}$ that is labelled $L[l] = \langle 0, 0, 1, 0\rangle$: l is stable-out in T w.r.t. $\mathcal{Q}$.* □

In order to prove soundness for stable-blocked labeled literals, we first need the following additional lemma: if a literal l is labelled $\neg L[l].d$ by Stability-label, then there is no future argumentation theory such that there is an argument for l in the grounded extension.

Lemma B.10 (Labelled not defended)**.** *Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and $\leq = \emptyset$, let $\mathcal{Q}$ be the set of queryable literals and let L be the labelling obtained by Stability-label (Algorithm 6) on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. For each $l \in \mathcal{L}$: if l is labelled $\neg L[l].d$ then there is no argument for l in $G(T')$ for any T' such that $T \sqsubseteq_Q T'$.*

Proof. Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and let $\mathcal{Q}$ be a set of queryables. Let L be the labelling obtained by Stability-label (Algorithm 6) on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. We proceed by induction on $c_{\{d\}}$.

Proposition ($P(n)$): For each $l \in \mathcal{L}$ such that $c_{\{d\}}(l) \leq n$: if l is labelled $\neg L[l].d$, then there is no argument for l in $G(T')$ for any T' such that $T \sqsubseteq_Q T'$.

Base case ($P(0)$): Let l be an arbitrary literal from $\mathcal{L}$ such that $c_{\{d\}}(l) \leq 0$. We consider two cases:

- First suppose that $L_p[l] = \langle 1, 0, 0, 0\rangle$. By Lemma 3.4, for each T' such that $T \sqsubseteq_Q T'$, there is no argument for l in $Arg_{T'}$. As a consequence, *there is no argument for l in $G(T')$ for any T' such that $T \sqsubseteq_Q T'$.*
- Alternatively, suppose that $L_p[l] = \langle 1, 1, 1, 1\rangle$. We will first show that $l \in \mathcal{Q}$: suppose that $l \notin \mathcal{Q}$. We know that l was labelled $\neg L[l].d$ in Algorithm 6 line 5, so the condition in line 4 must have applied. By assumption $l \notin \mathcal{Q}$, hence we have that there is no rule for l in $\mathcal{R}$. However, that implies that l would have been labelled $\langle 1, 0, 0, 0\rangle$ in the preprocessing step (Algorithm 3 line 4), and this label would not change: Algorithm 3 line 13 would not be executed for l since the condition in line 11 does not apply. A contradiction with our assumption that $L_p[l] = \langle 1, 1, 1, 1\rangle$. Therefore, $l \in \mathcal{Q}$.

  Given that $l \in \mathcal{Q}$, the label $\neg L[l].d$ must be caused by Algorithm 4 case L-D-a: there is some $l' \in \bar{l}$ such that $l' \in \mathcal{K}$. This means that there is an observation-based argument for l' in Arg_T which, by Lemma B.3 and Definitions B.5 and 2.1, is in the grounded extension of every T' such that $T \sqsubseteq_Q T'$. Therefore *there is no argument for l in $G(T')$ for any T' such that $T \sqsubseteq_Q T'$.*

Induction hypothesis ($P(k)$): For each $l \in \mathcal{L}$ such that $c_{\{d\}}(l) \leq k$: if l is labelled $\neg L[l].d$, then there is no argument for l in $G(T')$ for any future argumentation theory T' such that $T \sqsubseteq_Q T'$.

Induction step: Let l be an arbitrary literal in $\mathcal{L}$ such that $c_{\{d\}}(l) = k+1$. First, we show that $l \notin \mathcal{Q}$: if l would be in $\mathcal{Q}$, then the label $\neg L[l].d$ must have been caused by Algorithm 4 case L-D-a: there is an $l' \in \bar{l}$ such that $l' \in \mathcal{K}$. However, that would imply that $c_{\{d\}}(l) = 0$, which contradicts our assumption that $c_{\{d\}}(l) = k+1$ for positive k.

Given that $l \notin \mathcal{Q}$, $\neg L[l].d$ must have been caused by case L-D-b or L-D-c. Next, we consider these two cases:

- Suppose that each rule r for l is labelled $\neg L[r].d$ (L-D-b is applied). $c_{\{d\}}(l) = k+1$, so there exists at least one rule for l. We take an arbitrary rule r for l. Then, by R-D-a, there is an $a \in \texttt{ants}(r)$ with $\neg L[a].d$ and $c_{\{d\}}(a) \leq k$. By the induction hypothesis, for each T' such that $T \sqsubseteq_Q T'$, there is no argument for a in $G(T')$. As a result, for each T' such that $T \sqsubseteq_Q T'$, each argument for a would be defeated by some argument that is not defeated by any argument in $G(T')$, which implies that each argument with top rule r would be attacked by some argument that is not attacked by any argument in $G(T')$. Since we chose r arbitrarily, we know that there is no rule-based argument for l in $G(T')$ for any T' such that $T \sqsubseteq_Q T'$. Furthermore, the fact that $l \notin \mathcal{Q}$ implies that there is no observation-based argument either, so *there is no argument for l in $G(T')$ for any T' such that $T \sqsubseteq_Q T'$.*
- Now suppose that there is a rule r for l that is labelled $L[r].d$ (L-D-c is applied). Then $l \notin \mathcal{Q}$ and there is some $l' \in \bar{l}$ such that there is a rule r' for l' with $\neg L[r'].u$ and $\neg L[r'].o$. By Lemma B.4, for each future argumentation theory T' such that $T \sqsubseteq_Q T'$ there exists an argument for l' with top rule r' that is not attacked by any argument in $G(T')$. So for every T' such that $T \sqsubseteq_Q T'$, *if an argument for l exists, then it cannot be in $G(T')$.*

At this point, we have shown for each non-negative integer $n \in \mathbb{N}$: for each $l \in \mathcal{L}$ such that $c_{\{d\}}(l) \leq k$: if l is labelled $\neg L[l].d$, then there is no argument for l in $G(T')$ for any T' such that $T \sqsubseteq_Q T'$. Given that for each $l \in \mathcal{L}$ such that l is labelled $\neg L[l].d$: $c_{\{d\}}(l)$ is finite (Definition B.4), we derive: *for each $l \in \mathcal{L}$ that is labelled $\neg L[l].d$, then there is no argument for l in $G(T')$ for any T' such that $T \sqsubseteq_Q T'$.* □

We now use Lemma B.10 to prove Lemma B.11, which is the soundness proof for the fourth and final justification status: blocked.

Lemma B.11 (Soundness stable-blocked labelling)**.** *Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and $\leq = \emptyset$, let $\mathcal{Q}$ be the set of queryable literals and let L be the labelling obtained by* Stability-label *(Algorithm 6) on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. For each $l \in \mathcal{L}$: if $L[l] = \langle 0, 0, 0, 1 \rangle$ then l is stable-blocked in T w.r.t. $\mathcal{Q}$.*

Proof. Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and let $\mathcal{Q}$ be a set of queryables. Let L be the labelling obtained by Stability-label (Algorithm 6) on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. Let $l \in \mathcal{L}$ be an arbitrary literal that is labelled $L[l] = \langle 0, 0, 0, 1\rangle$ by Stability-label (Algorithm 6). Note that $L[l].b$ implies that $L_p[l] = \langle 1, 1, 1, 1\rangle$, so the fact that l is labelled $\neg L[l].u$, $\neg L[l].d$ and $\neg L[l].o$ must have been caused by Algorithm 4.

The fact that l is labelled $L[l].b$, although l is considered for labelling by Algorithm 4, implies that $l \notin \mathcal{Q}$ (case L-B-a did not apply). Then $l \notin \mathcal{K}$, so label $\neg L[l].u$ must have been caused by case L-U-b. Then the label $\neg L[l].o$ must be caused by either L-O-d or L-O-e. In both cases, some rule r for l is labelled $\neg L[r].u$ and $\neg L[r].o$. By Lemma B.4, for each T' such that $T \sqsubseteq_Q T'$, there is an argument with top rule r in $Arg_{T'}$ that is not attacked (and thus not defeated) by an argument in $G(T')$. Furthermore, the fact that l is labelled $\neg L[l].d$ implies that for every future argumentation theory T' such that $T \sqsubseteq_Q T'$ there is no argument for l in the grounded extension $G(T')$ (Lemma B.10). To conclude, for each T' such that $T \sqsubseteq_Q T'$, l is blocked in T' (Definition 3.9), so l is stable-blocked in T w.r.t. $\mathcal{Q}$ (Definition 3.12). □

Finally, we combine these lemmas to show that Stability-label is sound: if some literal l is labelled as stable in some argumentation theory T w.r.t. the set of queryables $\mathcal{Q}$, then l is stable in T w.r.t. $\mathcal{Q}$.

Proposition 3.8 (Soundness stability labelling)**.** *Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and $\leq = \emptyset$, let $\mathcal{Q}$ be the set of queryable literals and let L be the labelling obtained by Stability-label (Algorithm 6) on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$.*

- *If $L[l] = \langle 1, 0, 0, 0\rangle$ then l is stable-unsatisfiable in T w.r.t. $\mathcal{Q}$;*
- *if $L[l] = \langle 0, 1, 0, 0\rangle$, then l is stable-defended in T w.r.t. $\mathcal{Q}$;*
- *if $L[l] = \langle 0, 0, 1, 0\rangle$, then l is stable-out in T w.r.t. $\mathcal{Q}$; and*
- *if $L[l] = \langle 0, 0, 0, 1\rangle$, then l is stable-blocked in T w.r.t. $\mathcal{Q}$.*

Proof. Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and let $\mathcal{Q}$ be a set of queryables. Let L be the labelling obtained by Stability-label (Algorithm 6) on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. Suppose that a literal $l \in \mathcal{L}$ is labelled stable by Stability-label (Algorithm 6) and let L be the obtained labelling. Then $L[l] = \langle 1, 0, 0, 0\rangle$, $L[l] = \langle 0, 1, 0, 0\rangle$, $L[l] = \langle 0, 0, 1, 0\rangle$ or $L[l] = \langle 0, 0, 0, 1\rangle$. That l is stable follows from:

- If $L[l] = \langle 1, 0, 0, 0\rangle$ then l is stable-unsatisfiable in T w.r.t. $\mathcal{Q}$ by Lemma B.7.
- If $L[l] = \langle 0, 1, 0, 0\rangle$ then l is stable-defended in T w.r.t. $\mathcal{Q}$ by Lemma B.8.
- If $L[l] = \langle 0, 0, 1, 0\rangle$ then l is stable-out in T w.r.t. $\mathcal{Q}$ by Lemma B.9.
- If $L[l] = \langle 0, 0, 0, 1\rangle$ then l is stable-blocked in T w.r.t. $\mathcal{Q}$ by Lemma B.11. □

B.1.6. Conditional completeness of the stability algorithm

In this section, we prove Proposition 3.9, which specifies conditions under which Stability-label is complete. Similar to Section B.1.4, we first define height of potential arguments in Section B.1.6 before proceeding to the actual proofs in Section B.1.6.

Height of potential arguments

Recall that the completeness cases from Proposition 3.9 are defined in terms of potential arguments (Definition 3.21). In order to prove this proposition by induction, we need the notion of height of potential arguments. Similar to the height of arguments, the height of a potential argument is the maximum number of inferences between a premise and the conclusion of the potential argument.

Definition B.6 (Height of potential arguments)**.** Let $T = (AS, \mathcal{K})$ be an argumentation theory and let $A^p \in P_{\mathcal{Q}}(T)$ be an argument. The height $\mathtt{h}(A^p)$ of A^p is:

- if A^p is an *observation-based potential argument* then $\mathtt{h}(A^p) = 0$;
- if A is a *rule-based argument* of the form $A_1, \ldots, A_m \Rightarrow c$, then $\mathtt{h}(A) = 1 + \max(\mathtt{h}(A_1), \ldots, \mathtt{h}(A_m))$.

Note that Definition 3.21 on potential arguments enforces that all potential arguments have finite height, since the number of steps for constructing a potential argument is finite.

Conditional completeness proof

In this section, we prove Proposition 3.9, which specifies conditions under which Stability-label is complete. In order to prove this, we first need Lemma 3.5 (see Section 3.7.5), which consists of six items and shows in which situations Stability-label turns one or more booleans (u, d, o and/or b) of the labelling L to False. In the following lemmas, we will prove each of these items. First, we prove that Preprocess (Algorithm 3) labels literals/rules as $\langle 1, 0, 0, 0\rangle$ if and only if there exists no potential argument for them or having them as a top rule. This is Item 1 of Lemma 3.5.

Lemma B.12 (Soundness and completeness Algorithm 3 for potential arguments)**.** *Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and $\leq = \emptyset$ and let L_p be the labelling after executing Algorithm 3. For each rule $r \in \mathcal{R}$: $L_p[r] = \langle 1, 0, 0, 0\rangle$ iff there is no $A^p \in P_{\mathcal{Q}}(T)$ such that A^p has top rule r. For each literal $l \in \mathcal{L}$: $L_p[l] = \langle 1, 0, 0, 0\rangle$ iff there is no $A^p \in P_{\mathcal{Q}}(T)$ such that A^p is a potential argument for l.*

Proof. Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and let $\mathcal{Q}$ be a set of queryables. Let L_p be the labelling obtained by Preprocess (Algorithm 3) on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$.

- *Left to right (soundness)*: We proceed by induction:

Proposition ($P(n)$): Each $l \in \mathcal{L}$ for which there is a potential argument A^p in $P_{\mathcal{Q}}(T)$ with $\mathtt{h}(A) \leq n$ is labelled $L_p[l] = \langle 1,1,1,1\rangle$; each $r \in \mathcal{R}$ such that there is a potential argument A^p with top rule r in $P_{\mathcal{Q}}(T)$ with $\mathtt{h}(A^p) \leq n+1$ is labelled $L_p[r] = \langle 1,1,1,1\rangle$.

Base case ($P(0)$):

- For each l in $\mathcal{L}$ such that there is some A^p for l in $P_{\mathcal{Q}}(T)$ and $\mathtt{h}(A^p) = 0$, then by Definition B.6 $l \in \mathcal{Q}$ and for each $l' \in \bar{l}$: $l' \notin \mathcal{K}$, so by Algorithm 3 line 3, $L_0^p[l] = \langle 1,1,1,1\rangle$.
- For each $r \in \mathcal{R}$ such that there is some A^p with top rule r in $P_{\mathcal{Q}}(T)$ with $\mathtt{h}(A^p) = 0$, there must be an observation-based potential argument for each $a \in \mathtt{ants}(r)$, so for each $a \in \mathtt{ants}(r)$: $L_p[a] = \langle 1,1,1,1\rangle$. This label must have been assigned by Algorithm 3 line 3, after which r was labelled $L_p[r] = \langle 1,1,1,1\rangle$ in Algorithm 3 line 12.

Induction hypothesis ($P(k)$): Each $l \in \mathcal{L}$ for which there is a potential argument A^p in $P_{\mathcal{Q}}(T)$ with $\mathtt{h}(A) \leq k$ is labelled $L_p[l] = \langle 1,1,1,1\rangle$; each $r \in \mathcal{R}$ such that there is a potential argument A^p with top rule r in $P_{\mathcal{Q}}(T)$ with $\mathtt{h}(A^p) \leq k+1$ is labelled $L_p[r] = \langle 1,1,1,1\rangle$.

Induction step ($P(k+1)$):

- For each l in $\mathcal{L}$ such that there is some A^p for l in $P_{\mathcal{Q}}(T)$ and $\mathtt{h}(A^p) = k+1$, there must be some rule r on which A^p is based; by the induction hypothesis, $L_p[r] = \langle 1,1,1,1\rangle$. This must have happened in line 12 of Algorithm 3, after which l was labelled $L_p[l] = \langle 1,1,1,1\rangle$ in line 13.
- For each $r \in \mathcal{R}$ such that there is some A^p with top rule r in $P_{\mathcal{Q}}(T)$ with $\mathtt{h}(A^p) = k+2$, there must be potential arguments for each $a \in \mathtt{ants}(r)$ with a height of at most $k+1$, so by the induction hypothesis, each $a \in \mathtt{ants}(r)$: $L_p[a] = \langle 1,1,1,1\rangle$. At least one of these labels must have been assigned by Algorithm 3 line 13, after which r was labelled $L_p[r] = \langle 1,1,1,1\rangle$ in Algorithm 3 line 12 in a later iteration of the while loop.

Finally note that all potential arguments have finite height, so $P(n)$ can be generalised to all l in $\mathcal{L}$. If $L_p[l] = \langle 1,0,0,0\rangle$ then $L_p[l] = \langle 1,1,1,1\rangle$, so there is no A^p for l in $P_{\mathcal{Q}}(T)$. Similarly, for each $r \in \mathcal{R}$ labelled $L_p[r] = \langle 1,0,0,0\rangle$, there is no A^p with top rule r in $P_{\mathcal{Q}}(T)$.

- *Right to left (completeness)*: First, we introduce some notation, which is similar to the interim labelling from Definition B.3. Let $x \in \mathcal{L} \cup \mathcal{R}$ be a literal or rule; we denote by $L_0^p[x]$ the label given to x by the preprocessing algorithm (Algorithm 3) between line 7 and 8. Furthermore, $L_k^p[x]$ is the label given to x by the preprocessing algorithm just after the k'th iteration of the for loop (line 10–14). Using this notation, we proceed by induction.

Proposition ($P(n)$): For each $l \in \mathcal{L}$ such that $L_n^p[l] = \langle 1,1,1,1\rangle$, there is a potential argument for l in $P_{\mathcal{Q}}(T)$; for each $r \in \mathcal{R}$ such that $L_n^p[r] = \langle 1,1,1,1\rangle$, there is a potential argument with top rule r in $P_{\mathcal{Q}}(T)$.

Base case ($P(0)$):

- For each $l \in \mathcal{L}$ such that $L_0^p[l] = \langle 1,1,1,1\rangle$, the condition in Algorithm 3 line 2 must have applied ($l \in \mathcal{Q}$ and for each $l' \in \bar{l}$: $l' \notin \mathcal{K}$), so by Definition 3.21, there is an observation-based potential argument for l in $P_{\mathcal{Q}}(T)$.
- No rule $r \in \mathcal{R}$ is labelled $L_n^p[r] = \langle 1,1,1,1\rangle$, so for each $r \in \mathcal{R}$ such that $L_n^p[r] = \langle 1,1,1,1\rangle$, there is a potential argument with top rule r in $P_{\mathcal{Q}}(T)$.

Induction hypothesis ($P(k)$): For each $l \in \mathcal{L}$ such that $L_k^p[l] = \langle 1,1,1,1\rangle$, there is a potential argument for l in $P_{\mathcal{Q}}(T)$; for each $r \in \mathcal{R}$ such that $L_k^p[r] = \langle 1,1,1,1\rangle$, there is a potential argument with top rule r in $P_{\mathcal{Q}}(T)$.

Induction step ($P(k+1)$):

- For each $r \in \mathcal{R}$ such that $L_{k+1}^p[r] = \langle 1,1,1,1\rangle$, it must be that for each $a \in \texttt{ants}(r)$: $L_k^p[a] = \langle 1,1,1,1\rangle$. Then by the induction hypothesis, for each $a \in \texttt{ants}(r)$, there is a potential argument for a in $P_{\mathcal{Q}}(T)$, which by Definition 3.21 implies that there is a potential argument with top rule r in $P_{\mathcal{Q}}(T)$.
- For each $l \in \mathcal{L}$ such that $L_{k+1}^p[l] = \langle 1,1,1,1\rangle$, it must be that there is some rule that is a top rule of some potential argument in $P_{\mathcal{Q}}(T)$. Consequently, there is a potential argument for l in $P_{\mathcal{Q}}(T)$.

Finally note that Algorithm 3 terminates, given the running time is polynomial in the input (Lemma 3.6) and the language and rule set are finite. Consequently, for each $l \in \mathcal{L}$: if $L_p[l] = \langle 1,1,1,1\rangle$ then there is a potential argument for l in $P_{\mathcal{Q}}(T)$. Then by contraposition: if there is no potential argument for l in $P_{\mathcal{Q}}(T)$, then $L_p[l] \neq \langle 1,1,1,1\rangle$, so $L_p[l] = \langle 1,0,0,0\rangle$. Similarly, each $r \in \mathcal{R}$ that is not a top rule of any potential argument in $P_{\mathcal{Q}}(T)$ is labelled $L_p[r] = \langle 1,0,0,0\rangle$. □

Next, we specify the condition under which a literal l is labelled $\neg L[l].u$ or a rule r is labelled $\neg L[r].u$. This is Item 2 of Lemma 3.5. Note that this is the reversed version of Lemma B.1.

Lemma B.13 (Argument existence labelling)**.** *Given an argumentation theory $T = (AS, \mathcal{K})$ where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and $\leq = \emptyset$ and $\mathcal{Q}$ is a set of queryables, let L be the labelling obtained by* Stability-label *(Algorithm 6) on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. For each literal $l \in \mathcal{L}$: if there is an argument for l in Arg_T, then $\neg L[l].u$. For each $r \in \mathcal{R}$: if there is an argument with top rule r in Arg_T then $\neg L[r].u$.*

Proof. Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and let $\mathcal{Q}$ be a set of queryables. Let L be the labelling obtained by STABILITY-LABEL (Algorithm 6) on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. We proceed by induction on the height of arguments for l/with top rule r.

Proposition ($P(n)$): Each $l \in \mathcal{L}$ for which there is an argument A with $\mathtt{h}(A) \leq n$ is labelled $\neg L[a].u$ and each $r \in \mathcal{R}$ that is the top rule of some argument A with $\mathtt{h}(A) \leq n+1$ is labelled $\neg L[r].u$.

Base case ($P(0)$): For each $l \in \mathcal{L}$ such that there is an argument A for l with $\mathtt{h}(A) = 0$, it must be that $l \in \mathcal{K}$, hence $l \in \mathcal{Q}$. This means that the condition for the if-statement in Algorithm 6 line 4 applies and l is labelled $\neg L[l].u$ in Algorithm 6 line 5 by Algorithm 4 case L-U-a.

For each $r \in \mathcal{R}$ such that there is an argument A with top rule r with $\mathtt{h}(A) = 1$, it must be that for each $a \in \mathtt{ants}(r)$ there is an argument A' for a with $\mathtt{h}(A) = 0$, so a is labelled $\neg L[a].u$ in Algorithm 6 line 5; consequently r is added to TODO-SET in line 6. When popped from this set, r is labelled $\neg L[r].u$ by case R-U-a.

Induction hypothesis ($P(k)$): Each $l \in \mathcal{L}$ for which there is an argument A with $\mathtt{h}(A) \leq k$ is labelled $\neg L[l].u$ and each $r \in \mathcal{R}$ that is the top rule of some argument A with $\mathtt{h}(A) \leq k+1$ is labelled $\neg L[r].u$.

Induction step ($P(k+1)$): For each $l \in \mathcal{L}$ such that there is an argument A for l with $\mathtt{h}(A) = k+1$, it must be that $\mathtt{top\text{-}rule}(A)$ is labelled $\neg L[\mathtt{top\text{-}rule}(A)].u$ (induction hypothesis). After the top rule's label change, which must have happened in Algorithm 6 line 9, its consequent l is relabelled in line 11 as $\neg L[l].u$ (by case L-U-b).

For each $r \in \mathcal{R}$ such that there is an argument A with top rule r with $\mathtt{h}(A) = k+2$, it must be that for each $a \in \mathtt{ants}(r)$ there is an argument A' for a with $\mathtt{h}(A) \leq k+1$, so by the induction hypothesis (applicable if $\mathtt{h}(A') \leq k$) and above conclusion (applicable if $\mathtt{h}(A') = k+1$), a is labelled $\neg L[a].u$. The last of these antecedents must have been relabelled in Algorithm 6 line 11 or line 15; in both cases, r is added to TODO-SET immediately afterwards and, when popped, labelled $\neg L[r].u$ by case R-U-a.

Finally, recall from Definition B.1 that all arguments have finite height. This concludes the proof. □

The next lemma shows under which conditions a literal l is labelled $\neg L[l].d$ and $\neg L[l].b$. This corresponds to Lemma 3.5 Item 3.

Lemma B.14 (Conditions for $\neg L[l].d$ and $\neg L[l].b$)**.** *Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and $\leq = \emptyset$, let $\mathcal{Q}$ be the set of queryable literals and let L be the labelling obtained by STABILITY-LABEL (Algorithm 6) on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. For each literal $l \in \mathcal{L}$: if each potential argument for l in $P_{\mathcal{Q}}(T)$ is p-attacked by an observation-based argument in Arg_T, then $\neg L[l].d$ and $\neg L[l].b$.*

Proof. Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and let $\mathcal{Q}$ be a set of queryables. Let L be the labelling obtained by STABILITY-LABEL (Algorithm 6) on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$.

Let $l \in \mathcal{L}$ be a literal such that each potential argument for l in $P_{\mathcal{Q}}(T)$ is p-attacked by an observation-based argument in Arg_T. We proceed by induction on the height of potential arguments for a literal.

Proposition ($P(n)$): For each $l \in \mathcal{L}$: if for each potential argument A^p for l in $P_{\mathcal{Q}}(T)$: [$\mathtt{h}(A^p) \leq n$ and A^p is p-attacked by an observation-based argument in Arg_T] then l is labelled $\neg L[l].d$ and $\neg L[l].b$.

Base case ($P(1)$): Let $l \in \mathcal{L}$ be an arbitrary literal such that each potential argument A^p for l in $P_{\mathcal{Q}}(T)$ has $\mathtt{h}(A^p) \leq 1$ and A^p is p-attacked by an observation-based argument in Arg_T. Note the following:

- *There is no potential argument A^p for l in $P_{\mathcal{Q}}(T)$ such that $\mathtt{h}(A^p) = 0$*: if such a potential argument would exist, then it would be observation-based (by Definition 3.21: $l \in \mathcal{Q}$ and for each $l' \in \bar{l}$: $l' \notin \mathcal{K}$), which means that it could not be p-attacked by an observation-based argument in Arg_T.
- $l \in \mathcal{Q}$: otherwise, the attack on A^p by some observation-based argument in Arg_T must have been *on a subargument*, but that is impossible: since $A^p \in P_{\mathcal{Q}}(T)$ and $\mathtt{h}(A^p) = 1$, it must be that for each $a \in \mathtt{ants}(\mathtt{top\text{-}rule}(A^p))$, there is no $a' \in \overline{a}$ such that $a' \in \mathcal{K}$.
- *There is some $l' \in \bar{l}$ such that $l' \in \mathcal{K}$*: if none of $\bar{l}$ would be in $\mathcal{K}$, then there would be an observation-based potential argument for l in $P_{\mathcal{Q}}(T)$ that is not p-attacked by any observation-based argument in Arg_T, a contradiction.

Given that $l \in \mathcal{Q}$, l is considered for relabelling by Algorithm 6 line 5 and labelled $\neg L[l].d$ *and* $\neg L[l].b$ by Algorithm 4 case L-D-a and L-B-a. As l is chosen arbitrarily, we can generalise this result.

Induction hypothesis ($P(k)$): For each $l \in \mathcal{L}$: if for each potential argument A^p for l in $P_{\mathcal{Q}}(T)$: [$\mathtt{h}(A^p) \leq k$ and A^p is p-attacked by an observation-based argument in Arg_T] then l is labelled $\neg L[l].d$ and $\neg L[l].b$.

Induction step ($P(k+1)$): Now let $l \in \mathcal{L}$ be an arbitrary literal such that each potential argument A^p for l has $\mathtt{h}(A^p) \leq k+1$ and is p-attacked by an observation-based argument in Arg_T. We consider two cases: either $l \in \mathcal{Q}$ or $l \notin \mathcal{Q}$.

- First suppose that $l \in \mathcal{Q}$. Then *there is some $l' \in \bar{l}$ such that $l' \in \mathcal{K}$*: if for each $l' \in \bar{l}$: $l' \notin \mathcal{K}$, then there would be an observation-based potential argument for l in $P_{\mathcal{Q}}(T)$ that is not p-attacked by any observation-based argument in Arg_T (Definition 3.21); a contradiction. Given that $l \in \mathcal{Q}$, l is considered for relabelling in Algorithm 6 line 5 and labelled $\neg L[l].d$ *and* $\neg L[l].b$ by Algorithm 4 case L-D-a and L-B-a.

- Alternatively suppose that $l \notin \mathcal{Q}$. Then each potential argument for l in $P_{\mathcal{Q}}(T)$ must be rule-based. Let $r \in \mathcal{R}$ be an arbitrary rule for l. We consider two cases and show that $\neg L[r].d$ and $\neg L[r].b$:
  - If $L_p[r] = \langle 1,0,0,0\rangle$ then $L[r] = \langle 1,0,0,0\rangle$ since there is no operation in Algorithm 6 that can turn booleans from False to True. This implies that r is labelled $\neg L[r].d$ *and* $\neg L[r].b$.
  - Alternatively, $L_p[r] = \langle 1,1,1,1\rangle$, so by Lemma B.12, there is an A^p with top rule r in $P_{\mathcal{Q}}(T)$. Since A^p is p-attacked by an observation-based argument in Arg_T and $l \notin \mathcal{Q}$, it must be p-attacked *on a subargument*.

    Next we prove that *there is some* $a \in \texttt{ants}(r)$ *such that each* A_i^p *for* a *is p-attacked by an observation-based argument in* Arg_T: if this would not be the case, we could construct a potential argument $A_l^p = A_1^p, \ldots, A_n^p \Rightarrow l$ for l that is not p-attacked by an observation-based argument in Arg_T; a contradiction.

    Then by the induction hypothesis $\neg L[a].d$ *and* $\neg L[a].b$. This label cannot be caused by the preprocessing step: if $L_p[a]$ would be $\langle 1,0,0,0\rangle$, then r would be labelled $\langle 1,0,0,0\rangle$ in Algorithm 3 line 6, and never be relabelled to $\langle 1,1,1,1\rangle$ by line 12; a contradiction. So $L_p[a] = \langle 1,1,1,1\rangle$. As a consequence, a must have been labelled $\neg L[a].d$ and $\neg L[a].b$ in Algorithm 6 line 5, line 11 or line 15. In all cases, r is added to TODO-SET immediately afterwards and relabelled in a later iteration of the while loop (Algorithm 6 line 9) as $\neg L[r].d$ *and* $\neg L[r].b$ by case R-D-a and R-B-b.

  Thus in both cases, r *is labelled* $\neg L[r].d$ *and* $\neg L[r].b$. Since r is an arbitrary rule for l, we can generalise this to all rules for l in $\mathcal{R}$. Finally, to show that l is labelled $\neg L[l].d$ and $\neg L[l].b$, we consider two possibilities:
  - If each rule r for l is labelled $L_p[r] = \langle 1,0,0,0\rangle$ then l is initially labelled $\langle 1,0,0,0\rangle$ by Algorithm 3 line 4. Since line 12 was never reached for any rule r for l, line 13 was not reached for l, hence $L_p[l]$ remains $\langle 1,0,0,0\rangle$. No operation in Algorithm 6 turns booleans to True, so $\neg L[l].d$ *and* $\neg L[l].b$.
  - Alternatively, there is a rule r for l that is labelled $L_p[r] = \langle 1,1,1,1\rangle$. Since r is a rule for l, it must be labelled $\neg L[r].d$ and $\neg L[r].b$. This must have been caused by Algorithm 6 line 9; after this change, l has been considered for relabelling by line 11 and labelled $\neg L[l].d$ *and* $\neg L[l].b$ by case L-D-b and L-B-b.

At this point, we have proven $P(n)$ for each non-negative integer $n \in \mathbb{N}$. Given that all potential arguments have finite height, we have shown for each $l \in \mathcal{L}$: if each potential argument for l in $P_{\mathcal{Q}}(T)$ is p-attacked by an observation-based argument in Arg_T, then l is labelled $\neg L[l].d$ and $\neg L[l].b$. □

Lemma B.15 shows under which conditions a literal l is labelled $\neg L[l].u$ and $\neg L[l].o$. This corresponds to the left-to-right part of Lemma 3.5 Item 4.

Lemma B.15 (Conditions for $\neg L[l].u$ and $\neg L[l].o$)**.** *Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and $\leq = \emptyset$, let $\mathcal{Q}$ be the set of queryable literals and let L be the labelling obtained by* Stability-label *(Algorithm 6) on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. For each literal $l \in \mathcal{L}$: if there is an argument A for l in Arg_T and there is no observation-based potential argument in $P_{\mathcal{Q}}(T)$ that p-attacks A, then $\neg L[l].u$ and $\neg L[l].o$.*

Proof. Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$, let $\mathcal{Q}$ be a set of queryables and let L be the labelling obtained by Stability-label on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. We proceed by induction:

Proposition ($P(n)$): If there is an argument A for l in Arg_T such that $\mathtt{h}(A) \leq n$ and there is no observation-based potential argument in $P_{\mathcal{Q}}(T)$ that p-attacks A, then l is labelled $\neg L[l].u$ and $\neg L[l].o$.

Base case ($P(0)$): Suppose that there is an argument A for l in Arg_T such that $\mathtt{h}(A) \leq 0$. Then $l \in \mathcal{K}$ (Definition B.1), so l is relabelled in Algorithm 6 line 5 as $\neg L[l].u$ and $\neg L[l].o$ by Algorithm 4 case L-U-a and L-O-a.

Induction hypothesis ($P(k)$): If there is an argument A for l in Arg_T such that $\mathtt{h}(A) \leq k$ and there is no observation-based potential argument in $P_{\mathcal{Q}}(T)$ that p-attacks A, then l is labelled $\neg L[l].u$ and $\neg L[l].o$.

Induction step ($P(k+1)$): Now suppose that there is an argument A for l in Arg_T such that $\mathtt{h}(A) = k + 1$ and there is no observation-based potential argument in $P_{\mathcal{Q}}(T)$ that p-attacks A. Then by Definition B.1 there is a rule r for l such that for each antecedent $a \in \mathtt{ants}(r)$, there is an argument A_i for a in Arg_T, such that A_i is not p-attacked by any observation-based potential argument in $P_{\mathcal{Q}}(T)$ and $\mathtt{h}(A_i) \leq k$. Then by the induction hypothesis, each $a \in \mathtt{ants}(r)$ is labelled $\neg L[a].u$ and $\neg L[a].o$. This label change must have happened in line 5, 11 or 15 of Algorithm 6, but in all cases, r is added to todo-set immediately afterwards (line 6/13/17) and considered for relabelling in line 9 of a later iteration of the while loop. By case R-U-a and R-O-a, r is labelled $\neg L[r].u$ and $\neg L[r].o$. Subsequently, l is considered for relabelling in line 11 and labelled $\neg L[l].u$ by case L-U-b. Furthermore, l is labelled $\neg L[l].o$ by either case L-O-e (if $l \notin \mathcal{Q}$) or L-O-c (if $l \in Q$, because then no observation-based potential argument for some $l' \in \bar{l}$ would p-attack A, which means that for each $l' \in \bar{l}$, some $l'' \in \overline{l'}$ must be in $\mathcal{K}$).

We have proven $P(n)$ for each $n \in \mathbb{N}$. Recall that all arguments have a finite height. This means that for each $l \in \mathcal{L}$: if there is an argument A for l in Arg_T and there is no observation-based potential argument in $P_{\mathcal{Q}}(T)$ that p-attacks A, then l is labelled $\neg L[l].u$ and $\neg L[l].o$. □

Lemma B.16 shows the consequences of the label $\neg L[l].u$ and $\neg L[l].o$ of a literal l: if it is labelled as such, we can derive that there is an argument for l in Arg_T that is not p-attacked by any observation-based potential argument in $P_{\mathcal{Q}}(T)$. This corresponds to the right-to-left part of Lemma 3.5 Item 4.

Lemma B.16 (Consequences of $\neg L[l].u$ and $\neg L[l].o$)**.** *Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and $\leq = \emptyset$, let $\mathcal{Q}$ be the set of queryable literals and let L be the labelling obtained by* Stability-label *(Algorithm 6) on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. For each literal $l \in \mathcal{L}$: if $\neg L[l].u$ and $\neg L[l].o$ then there is an argument A for l in Arg_T and there is no observation-based potential argument in $P_{\mathcal{Q}}(T)$ that p-attacks A.*

Proof. Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and let $\mathcal{Q}$ be a set of queryables. Let L be the labelling obtained by Stability-label on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. We proceed by induction.

Proposition ($P(n)$): For each $l \in \mathcal{L}$ that is labelled $\neg L[l].u$ and $\neg L[l].o$ such that $c_{\{u,o\}}(l) \leq n$, there is an argument for l in Arg_T that is not p-attacked by any observation-based potential argument in $P_{\mathcal{Q}}(T)$.

Base case ($P(0)$): Let $l \in \mathcal{L}$ be an arbitrary literal labelled $\neg L[l].u$ and $\neg L[l].o$ such that $c_{\{u,o\}}(l) \leq 0$. The label $\neg L[l].u$ cannot be assigned by Preprocess, so l must have been relabelled in Algorithm 6 line 5. At that point, all rules for l are still labelled $L[r].u$, so $\neg L[l].u$ cannot be caused by Algorithm 4 case L-U-b hence must be caused by L-U-a: $l \in \mathcal{K}$. Then there is an observation-based argument for l in Arg_T, which by Definition 3.22 cannot be p-attacked by any observation-based potential argument in $P_{\mathcal{Q}}(T)$.

Induction hypothesis ($P(k)$): For each $l \in \mathcal{L}$ that is labelled $\neg L[l].u$ and $\neg L[l].o$ such that $c_{\{u,o\}}(l) \leq k$, there is an argument for l in Arg_T that is not p-attacked by any observation-based potential argument in $P_{\mathcal{Q}}(T)$.

Induction step ($P(k+1)$): Let $l \in \mathcal{L}$ be an arbitrary literal labelled $\neg L[l].u$ and $\neg L[l].o$ such that $c_{\{u,o\}}(l) = k{+}1$. This means that $l \notin \mathcal{K}$ (if $l \in \mathcal{K}$ then $c_{\{u,o\}}(l) = 0$). So the label $\neg L[l].u$ must have been caused by case L-U-b: there is a rule r for l labelled $\neg L[r].u$. Then by Lemma B.2, the label $\neg L[l].o$ cannot be caused by case L-O-f and therefore must be caused by case L-O-b, L-O-c, L-O-d or L-O-e. In any of these four cases: there is a rule r for l labelled $\neg L[r].u$ and $\neg L[r].o$ ($c_{\{u,o\}}(r) = k{+}1$) and if $l \in \mathcal{Q}$ then for each $l' \in \bar{l}$ there is an $l'' \in \bar{l}$ such that $l'' \in \mathcal{K}$.

Given that there is a rule r for l labelled $\neg L[r].u$ and $\neg L[r].o$, each antecedent a of l must be labelled $\neg L[a].u$ and $\neg L[a].o$ by case R-U-a and R-O-a and the fact that $c_{\{u,o\}}(a) \leq k$. Then by the induction hypothesis, for each $a \in \texttt{ants}(r)$ there is an argument A for a in Arg_T that is not p-attacked by any observation-based potential argument in $P_{\mathcal{Q}}(T)$ *on a subargument.*

Finally, since either $l \notin \mathcal{Q}$ or for each $l' \in \bar{l}$ there is some $l'' \in \overline{l'}$ such that $l'' \in \mathcal{K}$, there is no observation-based potential argument for any $l' \in \bar{l}$, p-attacking A *on its conclusion.* To conclude, there is an argument for l that is not p-attacked by any observation-based potential argument in $P_{\mathcal{Q}}(T)$.

We have proven $P(i)$ for each non-negative integer $i < n$. Since Algorithm 6 has a polynomial runtime (Proposition 3.10), there must be some finite non-negative

$i < n$ such that for each $l \in \mathcal{L}$: if $\neg L[l].u$ and $\neg L[l].o$ then there is an argument for l in Arg_T that is not p-attacked by any observation-based potential argument in $P_{\mathcal{Q}}(T)$. □

The next lemma shows in which situations a literal l is labelled $\neg L[l].d$ by Stability-label. This corresponds to Lemma 3.5 Item 5.

Lemma B.17 (Conditions for $\neg L[l].d$)**.** *Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and $\leq = \emptyset$, let $\mathcal{Q}$ be the set of queryable literals and let L be the labelling obtained by Stability-label (Algorithm 6) on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. For each literal $l \in \mathcal{L}$: if each potential argument A^p for l in $P_{\mathcal{Q}}(T)$ is p-attacked by an argument B in Arg_T and there is no observation-based potential argument in $P_{\mathcal{Q}}(T)$ that p-attacks B, then $\neg L[l].d$.*

Proof. Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and let $\mathcal{Q}$ be a set of queryables. Let L be the labelling obtained by Stability-label on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. Let l be an arbitrary literal in $\mathcal{L}$ and suppose that each potential argument A^p for l in $P_{\mathcal{Q}}(T)$ is p-attacked by an argument B in Arg_T that is not p-attacked by any observation-based potential argument in $P_{\mathcal{Q}}(T)$. We proceed by induction on the height of potential arguments.

Proposition ($P(n)$):

- For each $l \in \mathcal{L}$, if each potential argument A^p for l has a height $\mathtt{h}(A^p) \leq n$ and is p-attacked by an argument in Arg_T that is not p-attacked by any observation-based potential argument in $P_{\mathcal{Q}}(T)$, then l is labelled $\neg L[l].d$; and
- For each $r \in \mathcal{R}$, if each potential argument A^p with top rule r has a height $\mathtt{h}(A^p) \leq n + 1$ and is p-attacked on a subargument by an argument in Arg_T that is not p-attacked by any observation-based potential argument in $P_{\mathcal{Q}}(T)$, then r is labelled $\neg L[r].d$.

Base case ($P(0)$):

- Let $l \in \mathcal{L}$ be an arbitrary literal such that each potential argument for l has a height of 0 and is p-attacked by an argument that is not p-attacked by any observation-based potential argument in $P_{\mathcal{Q}}(T)$.
  - If $l \in \mathcal{Q}$ then there is some $l' \in \bar{l}$ that is in $\mathcal{K}$: otherwise, by Definition 3.21 there would be some observation-based potential argument for l that p-attacks each argument in Arg_T by which it is p-attacked; a contradiction. Then l is labelled in Algorithm 6 line 5 as $\neg L[l].d$ by Algorithm 4 case L-D-a.
  - If $l \notin \mathcal{Q}$ then there is no potential argument for l in $P_{\mathcal{Q}}(T)$, so by Lemma B.12 $L_p[l] = \langle 1, 0, 0, 0 \rangle$, hence $\neg L[l].d$.
- Let $r \in \mathcal{R}$ be an arbitrary rule such that each potential argument with top rule r has a height of at most 1 and is p-attacked on a subargument by an

argument that is not p-attacked any observation-based potential argument in $P_{\mathcal{Q}}(T)$.

Then *there is no potential argument with top rule r in $P_{\mathcal{Q}}(T)$*: suppose, towards a contradiction, that such a potential argument A^p would exist, then A^p would be p-attacked on a subargument A' by an argument B in Arg_T. Given that $\mathtt{h}(A^p) = 1$, A' must be observation-based, so there is some $a \in \mathtt{ants}(r)$ such that $a \in \mathcal{Q}$ and for each $a' \in \overline{a}$: $a' \notin \mathcal{K}$. However, that would mean that B's conclusion is not in $\mathcal{K}$, so (the observation-based potential argument) A' p-attacks B; a contradiction.

Consequently, r is labelled $L_p[r] = \langle 1, 0, 0, 0\rangle$ (Lemma B.12), which implies $\neg L[r].d$.

Induction hypothesis ($P(k)$):

- For each $l \in \mathcal{L}$, if each potential argument A^p for l has a height $\mathtt{h}(A^p) \leq k$ and is p-attacked by an argument in Arg_T that is not p-attacked by any observation-based potential argument in $P_{\mathcal{Q}}(T)$, then l is labelled $\neg L[l].d$; and
- For each $r \in \mathcal{R}$, if each potential argument A^p with top rule r has a height $\mathtt{h}(A^p) \leq k + 1$ and is p-attacked on a subargument by an argument in Arg_T that is not p-attacked by any observation-based potential argument in $P_{\mathcal{Q}}(T)$, then r is labelled $\neg L[r].d$.

Induction step ($P(k+1)$):

- Let $l \in \mathcal{L}$ be an arbitrary literal such that each potential argument for l has a height of at most $(k + 1)$ and is p-attacked by an argument that is not p-attacked by any observation-based potential argument in $P_{\mathcal{Q}}(T)$.
  - If $l \in \mathcal{Q}$ then there is some $l' \in \overline{l}$ such that $l' \in \mathcal{K}$ (see base case) so l is labelled $\neg L[l].d$ by L-D-a.
  - Alternatively, $l \notin \mathcal{Q}$. We distinguish two options:
    - First suppose that *there is an argument B for some $l' \in \overline{l}$ in Arg_T that is not p-attacked by any observation-based potential argument in $P_{\mathcal{Q}}(T)$*. Given $l' \notin \mathcal{Q}$, B must be rule-based: there is a rule r' for l' such that for each $a' \in \mathtt{ants}(r')$, there is an argument for a' in Arg_T that is not p-attacked by any observation-based potential argument in $P_{\mathcal{Q}}(T)$. Then by Lemma B.15, each $a' \in \mathtt{ants}(r')$ is labelled $\neg L[a'].u$ and $\neg L[a'].o$. Given that $L_p[a'] = \langle 1, 1, 1, 1\rangle$ for each $a' \in \mathtt{ants}(r)$ (Lemma B.2), these labels must have been assigned in Algorithm 6 line 5, 11 or 15. In all cases, r' is added to TODO-SET immediately afterwards, considered for relabelling in a later iteration of the while loop (Algorithm 6 line 9) and labelled $\neg L[r'].u$ and $\neg L[r'].o$ by case R-U-a and R-O-a. Subsequently, l is relabelled in Algorithm 6 line 15 (recall that $l' \in \overline{l}$ implies $l \in \overline{l'}$), as $\neg L[l].d$ by case L-D-c.

  - Alternatively, *for each $l' \in \bar{l}$, each argument for l' in Arg_T is p-attacked by some observation-based potential argument in $P_{\mathcal{Q}}(T)$.* Then each potential argument for l in $P_{\mathcal{Q}}(T)$ must be rule-based. So for each rule r for l in $\mathcal{R}$: each potential argument with top rule r has $\mathtt{h}(A^p) \leq k+1$ and is p-attacked by an argument in Arg_T that is not p-attacked by any observation-based potential argument in $P_{\mathcal{Q}}(T)$. By the induction hypothesis, each rule r for l in $\mathcal{R}$ is labelled $\neg L[r].d$.
    - If this happened in preprocessing, Algorithm 3 line 12 was never reached for any r for l, so line 13 was not reached either, so $L_p[l] = \langle 1,0,0,0\rangle$, which implies $\neg L[l].d$.
    - Alternatively, some rule for l that was relabelled, must have been considered by Algorithm 6 line 9, which means that l was relabelled in line 11 as $\neg L[l].d$ by case L-D-b.

- Let $r \in \mathcal{R}$ be an arbitrary rule such that each potential argument A^p with top rule r has $\mathtt{h}(A^p) \leq (k+2)$ and is p-attacked on a subargument by an argument B in Arg_T that is not p-attacked by any observation-based potential argument in $P_{\mathcal{Q}}(T)$. Then there is at least one $a \in \mathtt{ants}(r)$ such that each potential argument A' for a has $\mathtt{h}(A') \leq k+1$ and is p-attacked by an argument B in Arg_T and there is no observation-based potential argument in $P_{\mathcal{Q}}(T)$ that p-attacks B, which implies that a is labelled $\neg L[a].d$ (see the item above).
  - If this happened in preprocessing, then Algorithm 3 line 11 never applied for r so line 8 was never reached, hence $L_p[r] = \langle 1,0,0,0\rangle$, therefore, $\neg L[r].d$.
  - Alternatively, a was labelled $\neg L[a].d$ by Algorithm 6 line 5, 11 or 15; in all cases, r was added to TODO-SET immediately afterwards and, when being popped from this set, labelled $\neg L[r].d$ by R-D-a in line 9.

Finally, recall that each potential argument has finite height, so from $P(n)$ we can generalise to each $l \in \mathcal{L}$. □

Finally, Lemma B.18 specifies in which situations $L[l] = \langle 0,1,0,0\rangle$. This corresponds to Item 6 of Lemma 3.5.

Lemma B.18 (Conditions for $\underline{L}[l] = \langle 0,1,0,0\rangle$)**.** *Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, {}^{-}, \mathcal{R}, \leq)$ and $\leq = \emptyset$, let $\mathcal{Q}$ be the set of queryable literals and let L be the labelling obtained by* Stability-label *(Algorithm 6) on $\mathcal{L}$, $\mathcal{R}$, ${}^{-}$, $\mathcal{Q}$ and $\mathcal{K}$. For each literal $l \in \mathcal{L}$: if there is an argument A for l in Arg_T and each potential argument B^p in $P_{\mathcal{Q}}(T)$ that p-attacks A is p-attacked by an observation-based argument in Arg_T, then $L[l] = \langle 0,1,0,0\rangle$.*

Proof. Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, {}^{-}, \mathcal{R}, \leq)$ and let $\mathcal{Q}$ be a set of queryables. Let L be the labelling obtained by Stability-label

(Algorithm 6) on $\mathcal{L}$, $\mathcal{R}$, $^{-}$, $\mathcal{Q}$ and $\mathcal{K}$. We proceed by induction; in order to prove the lemma for literals, we will also prove an auxiliary statement for rules.

Proposition ($P(n)$):

- For each $l \in \mathcal{L}$ for which there is an argument $A \in Arg_T$ for l such that $\mathtt{h}(A) \leq n$ and each potential argument in $P_{\mathcal{Q}}(T)$ attacking it is attacked by an observation-based argument in Arg_T, $L[l] = \langle 0, 1, 0, 0 \rangle$.
- For each $r \in \mathcal{R}$ that is the top rule of some argument $A \in Arg_T$ such that $\mathtt{h}(A) \leq n+1$ and each potential argument in $P_{\mathcal{Q}}(T)$ attacking it on a subargument is attacked by an observation-based argument in Arg_T, $L[r] = \langle 0, 1, 0, 0 \rangle$.

Base case ($P(0)$):

- For each $l \in \mathcal{L}$ for which there is an argument $A \in Arg_T$ for l such that $\mathtt{h}(A) = 0$, $l \in \mathcal{K}$. Then l is labelled in Algorithm 6 line 5 as $L[l] = \langle 0, 1, 0, 0 \rangle$ by Algorithm 4 case L-U-a, L-O-a and L-B-a.
- For each $r \in \mathcal{R}$ that is the top rule of some argument $A \in Arg_T$ such that $\mathtt{h}(A) \leq 1$, each $a \in \mathtt{ants}(r)$ is in $\mathcal{K}$ and therefore labelled $L[a] = \langle 0, 1, 0, 0 \rangle$. Afterwards, r is added to TODO-SET in line 6 and relabelled in line 9 as $L[r] = \langle 0, 1, 0, 0 \rangle$ by case R-U-a, R-O-a and R-B-a.

Induction hypothesis ($P(k)$):

- For each $l \in \mathcal{L}$ for which there is an argument $A \in Arg_T$ for l such that $\mathtt{h}(A) \leq k$ and each potential argument in $P_{\mathcal{Q}}(T)$ attacking it is attacked by an observation-based argument in Arg_T, $L[l] = \langle 0, 1, 0, 0 \rangle$.
- For each $r \in \mathcal{R}$ that is the top rule of some argument $A \in Arg_T$ such that $\mathtt{h}(A) \leq k+1$ and each potential argument in $P_{\mathcal{Q}}(T)$ attacking it on a subargument is attacked by an observation-based argument in Arg_T, $L[r] = \langle 0, 1, 0, 0 \rangle$.

Induction step ($P(k+1)$):

- Let l be an arbitrary literal from $\mathcal{L}$ for which there is an argument $A \in Arg_T$ such that $\mathtt{h}(A) = k+1$ and each potential argument in $P_{\mathcal{Q}}(T)$ attacking it is attacked by an observation-based argument in Arg_T.
  - If $l \in \mathcal{K}$ then l is labelled $L[l] = \langle 0, 1, 0, 0 \rangle$ in line 5 by case L-U-a, L-O-a and L-B-a.
  - Alternatively, suppose that $l \notin \mathcal{K}$.

    First we show that each rule r' for each $l' \in \bar{l}$ (if any) is labelled $\neg L[r'].d$ and $\neg L[r'].b$. Let $r' \in \mathcal{R}$ be an arbitrary rule for some $l' \in \bar{l}$.

    * If there is a potential argument with top rule r' in $P_{\mathcal{Q}}(T)$, then there is a potential argument for each $a' \in \mathtt{ants}(r')$ in $P_{\mathcal{Q}}(T)$ (Definition 3.21).

Note that there must be some $a' \in \texttt{ants}(r')$ such that each potential argument for a' is p-attacked by an observation-based argument in Arg_T: suppose, towards a contradiction, that this is not the case. Then we could construct a potential argument B^p with top rule r' (hence p-attacking A on its conclusion) from the potential arguments for $a' \in \texttt{ants}(r)$ that is not p-attacked *on a subargument* by any observation-based argument in Arg_T. Since $l' \notin \mathcal{Q}$, B^p cannot be p-attacked *on its conclusion* either by any observation-based argument in Arg_T; a contradiction.

Then by Lemma B.14, there must be some antecedent $a' \in \texttt{ants}(r')$ that is labelled $\neg L[a'].d$ and $\neg L[a'].b$. This implies that r' is labelled $\neg L[r'].d$ and $\neg L[r'].b$ (Algorithm 5 case R-D-a and R-B-b).

- Otherwise $L[r'] = \langle 1, 0, 0, 0\rangle$ by Lemma B.12, which implies that $\neg L[r'].d$ and $\neg L[r'].b$.

Since r' was chosen arbitrarily, each rule r' for each $l' \in \bar{l}$ must be labelled $\neg L[r'].d$ and $\neg L[r'].b$.

Recall that A is rule-based because $\texttt{h}(A) = k + 1$. Let r be the top rule of A. By the induction hypothesis, $L[r] = \langle 0, 1, 0, 0\rangle$.

Also note that either $l \in \mathcal{Q}$ or for each $l' \in \bar{l}$, there is some $l'' \in \overline{l'}$ in $\mathcal{K}$: if this would not be the case, then there would be an (observation-based) potential argument for some $l' \in \bar{l}$ in $P_{\mathcal{Q}}(()T)$ (p-attacking A) that is not attacked by any observation-based argument in Arg_T.

After relabelling each r' for each $l' \in \bar{l}$ and r this way in Algorithm 6 line 9, l is relabelled in line 11 or 15 as $L[l] = \langle 0, 1, 0, 0\rangle$ by case L-U-b, L-O-c or L-O-e and L-B-d.

- Let r be an arbitrary rule from $\mathcal{R}$ that is the top rule of some argument $A \in Arg_T$ such that $\texttt{h}(A) \leq k + 2$ and each potential argument in $P_{\mathcal{Q}}(T)$ attacking it is attacked by an observation-based argument in Arg_T. Then by the induction hypothesis, each $a \in \texttt{ants}(r)$ is labelled $L[a] = \langle 0, 1, 0, 0\rangle$; this must have happened in Algorithm 6 line 5, 11 or 15; in all cases, r is added to TODO-SET immediately afterwards and labelled in a later iteration of line 9 as $L[r] = \langle 0, 1, 0, 0\rangle$ by case R-U-a, R-O-a and R-B-a.

At this point, we have proven $P(n)$ for each non-negative integer $n \in \mathbb{N}$. Since each argument has finite height, we can generalise to each $l \in \mathcal{L}$ and each $r \in \mathcal{R}$, which concludes our proof. □

We have now formally proven all items in Lemma 3.5:

Lemma 3.5 (Conditions for labelling)**.** *Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and $\leq = \emptyset$ and let L be the labelling after executing the* Stability-label *algorithm on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. Let L_p be the labelling after executing* Preprocess *on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. Let $l \in \mathcal{L}$ be a literal. Then:*

1. *$L_p[l] = \langle 1,0,0,0\rangle$ iff there is no potential argument A^p for l in $P_{\mathcal{Q}}(T)$.*
2. *If there is an argument for l in Arg_T, then $\neg L[l].u$.*
3. *If each potential argument for l in $P_{\mathcal{Q}}(T)$ is p-attacked by an observation-based argument in Arg_T, then $\neg L[l].d$ and $\neg L[l].b$.*
4. *There is an argument A for l in Arg_T and there is no observation-based potential argument in $P_{\mathcal{Q}}(T)$ that p-attacks A iff $\neg L[l].u$ and $\neg L[l].o$.*
5. *If each potential argument A^p for l in $P_{\mathcal{Q}}(T)$ is p-attacked by an argument B in Arg_T and there is no observation-based potential argument in $P_{\mathcal{Q}}(T)$ that p-attacks B, then $\neg L[l].d$.*
6. *If there is an argument A for l in Arg_T and each potential argument B^p in $P_{\mathcal{Q}}(T)$ that p-attacks A is p-attacked by an observation-based argument in Arg_T, then $L[l] = \langle 0,1,0,0\rangle$.*

Proof. We have proven these items in the Lemmas B.12, B.13, B.14, B.15, B.17 and B.18, respectively. □

From the above result, we can derive in which situations literals are stable in the argumentation theory, but not labelled as such. First, Lemma B.19 specifies the situations in which a literal l is labelled $L[l].u$ or $L[l].o$, while l is not unsatisfiable or out in any future argumentation theory (hence the label $\neg L[l].u$ and $\neg L[l].o$ would be correct).

Lemma B.19 (Incorrect labelling $L[l].u$ or $L[l].o$)**.** *Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and $\leq = \emptyset$ and let L be the labelling after executing the* Stability-label *algorithm on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. Given a literal $l \in \mathcal{L}$, if for each T' such that $T \sqsubseteq_{\mathcal{Q}} T'$, l is not unsatisfiable or out in T', but l is labelled $L[l].u$ or $L[l].o$, then either l is observation-unattackable in T w.r.t. $\mathcal{Q}$ or some argument for l in Arg_T is p-attacked by an observation-based potential argument B^p such that the introduction of B^p in T forces a new argument for l.*

Proof. Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and let $\mathcal{Q}$ be a set of queryables. Let L be the labelling obtained by Stability-label (Algorithm 6) on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. Let $l \in \mathcal{L}$ be a literal such that for each T' such that $T \sqsubseteq_{\mathcal{Q}} T'$, l is not unsatisfiable or out in T' and l is labelled $L[l].u$ or $L[l].o$.

Given that l is labelled $L[l].u$ or $L[l].o$, each argument for l in Arg_T is p-attacked by an observation-based potential argument in $P_{\mathcal{Q}}(T)$ (Lemma B.15). Since l is not unsatisfiable or out in any T' such that $T \sqsubseteq_{\mathcal{Q}} T'$, for each T' such that $T \sqsubseteq_{\mathcal{Q}} T'$ there is at least one argument for l in $Arg_{T'}$ that is not p-attacked by an observation-based argument in $Arg_{T'}$ (Lemma 3.8).

So each argument A for l in Arg_T that is not attacked by an observation-based argument in Arg_T must be p-attacked by some observation-based potential argument B^p in $P_{\mathcal{Q}}(T)$. Let S be the set of all arguments for l in Arg_T that are not attacked by any observation-based argument in Arg_T. We consider two possibilities:

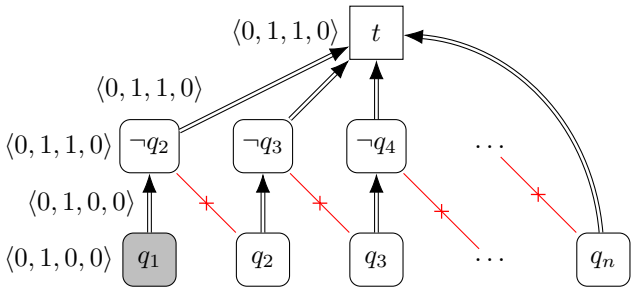


Figure B.4: t is stable-defended in T. However, t is labelled $L[t].o$ because the argument $[q_1 \Rightarrow \neg q_2] \Rightarrow t$ is p-attacked by the observation-based potential argument q_2. The introduction of this potential argument in T forces a new argument for t, i.e. $[q_2 \Rightarrow \neg q_3] \Rightarrow t$. *(Case 2b)*

- If there exists some *consistent* $\mathcal{K}'$ such that each argument in S is attacked by an observation-based argument in $Arg_{(AS,\mathcal{K}\cup\mathcal{K}')}$,[1] then for $T' = (AS, \mathcal{K} \cup \mathcal{K}')$ it must be that $T \sqsubseteq_{\mathcal{Q}} T'$. Let $\mathcal{K}'$ be a minimal set with this property; this implies that each $k \in \mathcal{K}'$ is the conclusion of some observation-based potential argument in $P_{\mathcal{Q}}(T)$ that p-attacks an argument in S. Given that l is not unsatisfiable or out in T', there must be some argument B for l in $Arg_{T'}$ that is not attacked by an observation-based argument and therefore cannot be in S, so $B \notin Arg_T$. Then by Definition 3.24, some argument $A \in Arg_T$ for l is p-attacked by at least one observation-based potential argument B^p such that the introduction of B^p in T *forces a new argument* for l.[2]
- Alternatively, there is no consistent $\mathcal{K}'$ such that each argument in S is attacked by an observation-based argument in $Arg_{(AS,\mathcal{K}\cup\mathcal{K}')}$; then there is no consistent set of observation-based potential arguments $T^p \subseteq P_{\mathcal{Q}}(T)$ such that each argument for l in Arg_T is p-attacked by some potential argument in T^p.[3] Then by Definition 3.23, l is *observation-unattackable* in T w.r.t. $\mathcal{Q}$. □

Finally, we use Lemmas 3.5 and B.19 to specify the other situations in which Stability-label does not detect stability in Proposition 3.9.

Proposition 3.9 (Conditional completeness stability labelling)**.** *Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and $\leq = \emptyset$ and let L be the labelling after executing the Stability-label algorithm on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. Given a literal $l \in \mathcal{L}$, if l is stable in T but l is not labelled stable by L, then some of the following five cases applies:*

[1] See Figure 3.21 for an example: let S be all arguments for t in Arg_T: $S = \{[q_1 \Rightarrow q_2] \Rightarrow t\}$. It is possible to attack all these arguments by adding $\mathcal{K}' = \{\neg q_2\}$ to the knowledge base.

[2] Note that this new argument is not attacked by an observation-based argument in $Arg_{T'}$, but may be attacked by an observation-based argument in some $Arg_{T''}$ where $T' \sqsubseteq_{\mathcal{Q}} T''$ Figure B.4 is an example of this situation: adding q_2 to the knowledge base results in a new argument $[q_2 \Rightarrow \neg q_3] \Rightarrow t$ for t, but this argument can be attacked in a future argumentation theory by adding q_3 to the knowledge base.

[3] See Figure 3.20 for an example.

1. *l is stable-unsatisfiable in T and each potential argument for l in $P_{\mathcal{Q}}(T)$ is inconsistent.*
2. *l is stable-defended or stable-blocked in T and:*
   (a) *l is observation-unattackable in T w.r.t. $\mathcal{Q}$;*[4] *or*
   (b) *some argument A for l in Arg_T is p-attacked by an observation-based potential argument B^p such that the introduction of B^p in T forces a new argument for l.*
3. *l is stable-defended in T and there is an argument A for l in Arg_T that is p-attacked by a potential argument in $P_{\mathcal{Q}}(T)$ that is not p-attacked by an argument in Arg_T and:*
   (a) *each potential argument in $P_{\mathcal{Q}}(T)$ p-attacking A that is not p-attacked by an argument in Arg_T is inconsistent; or*
   (b) *there is a consistent potential argument B^p in $P_{\mathcal{Q}}(T)$ p-attacking A that is not p-attacked by an argument in Arg_T, but the introduction of B^p in T forces a new argument for l; or*
   (c) *l is unattackable in T w.r.t. $\mathcal{Q}$.*
4. *l is stable-out in T and each potential argument for l in $P_{\mathcal{Q}}(T)$ that is not p-attacked by an observation-based argument in Arg_T has inconsistent premises; or*
5. *l is stable-blocked in T and there is a potential argument A^p for l in $P_{\mathcal{Q}}(T)$ such that each argument in Arg_T p-attacking A^p is p-attacked by an observation-based potential argument C^p in $P_{\mathcal{Q}}(T)$ and:*
   (a) *A^p is inconsistent; or*
   (b) *the introduction of A^p in T forces an argument attacking A^p; or*
   (c) *the introduction of C^p in T forces an argument that p-attacks A^p.*

Proof. Let $T = (AS, \mathcal{K})$ be an argumentation theory where $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ and $\mathcal{Q}$ is a set of queryables and let L be the labelling after executing the StabilityLabel algorithm on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$. Let L_p be the labelling after executing Preprocess on $\mathcal{L}$, $\mathcal{R}$, $\bar{}$, $\mathcal{Q}$ and $\mathcal{K}$ and let $l \in \mathcal{L}$ be a literal.

Unsatisfiable Suppose that l is stable-unsatisfiable in T and $L[l] \neq \langle 1,0,0,0\rangle$.[5] Then $L_p[l] \neq \langle 1,0,0,0\rangle$, so by Lemma B.12, there is a potential argument for l in $P_{\mathcal{Q}}(T)$. Each potential argument for l in $P_{\mathcal{Q}}(T)$ is inconsistent, as we show next by contradiction. If there would be some consistent A^p for l in $P_{\mathcal{Q}}(T)$, then $\mathcal{K}' = \mathcal{K} \cup \texttt{prem}(A^p)$ is consistent, since $\texttt{prem}(A^p)$ is required to be consistent with $\mathcal{K}$ by Definition 3.21. Then $T' = (AS, \mathcal{K}')$ is an argumentation theory such that $T \sqsubseteq_Q T'$ and $A^p \in Arg_{T'}$; contradiction. Thus there is a

[4] This condition is specifically required for argumentation systems that do not have classical negation as a contrariness function.

[5] See f in Example 3.21.

potential argument for l in $P_{\mathcal{Q}}(T)$ but each potential argument for l in $P_{\mathcal{Q}}(T)$ is inconsistent *(Case 1)*.

Out Suppose that l is stable-out in T and $L[l] \neq \langle 0,0,1,0\rangle$.[6] $T \sqsubseteq_{\mathcal{Q}} T$, so l is out and therefore not unsatisfiable in T. Then by Lemma B.13, l is labelled $\neg L[l].u$. So l must be labelled $L[l].d$ or $L[l].b$, while for each T' such that $T \sqsubseteq_{\mathcal{Q}} T'$, l is not defended or blocked in T'. Then by Lemma B.14, there is a potential argument for l in $P_{\mathcal{Q}}(T)$ that is not p-attacked by any observation-based argument in Arg_T. Let A^p be an arbitrary potential argument with this property. Next, we show by contradiction that A^p is inconsistent.

Suppose that A^p is consistent; then $\mathcal{K} \cup \texttt{prem}(A^p)$ is consistent. This means that $T' = (AS, \mathcal{K} \cup \texttt{prem}(A^p))$ is an argumentation theory such that $T \sqsubseteq_{\mathcal{Q}} T'$ and that $A \in Arg_{T'}$. Since l is not defended and blocked in any future argumentation theory of T, it must be unsatisfiable or out in T', which by Definition 3.9 and Lemma 3.8 means that each argument for l in $Arg_{T'}$ must be attacked by an observation-based argument in $Arg_{T'}$. Given that A^p is not p-attacked by any observation-based argument in Arg_T, A^p cannot be attacked by any observation-based argument in Arg_T. Hence A^p must be attacked by an observation-based argument in $Arg_{T'} \setminus Arg_T$, which means that it has its only premise in $\texttt{prem}(A^p)$. This however contradicts with the assumed consistency of $\texttt{prem}(A^p)$.

To conclude, each potential argument for l in $P_{\mathcal{Q}}(T)$ that is not p-attacked by an observation-based argument in Arg_T is inconsistent. *(Case 4)*

Defended Suppose that l is stable-defended in T and $L[l] \neq \langle 0,1,0,0\rangle$. We consider two cases.

- First suppose that l is labelled $L[l].u$ or $L[l].o$. Since l is defended in T' for each T' such that $T \sqsubseteq_{\mathcal{Q}} T'$, we know that for each T' such that $T \sqsubseteq_{\mathcal{Q}} T'$, l is not unsatisfiable or out in T'. Then by Lemma B.19, either l is observation-unattackable in T w.r.t. $\mathcal{Q}$ or some argument for l in Arg_T is p-attacked by an observation-based potential argument B^p such that the introduction of B^p in T forces a new argument for l *(Case 2)*.
- Alternatively, l is labelled $\neg L[l].u$ and $\neg L[l].o$. By Lemma B.18, the fact that $L[l] \neq \langle 0,1,0,0\rangle$ implies that each argument for l in Arg_T is p-attacked by some potential argument in $P_{\mathcal{Q}}(T)$ that is not p-attacked by any observation-based argument in Arg_T. Given that l is labelled $\neg L[l].u$ and $\neg L[l].o$, by Lemma B.16 there is an argument A for l in Arg_T that is not p-attacked by any observation-based potential argument in $P_{\mathcal{Q}}(T)$. A is however p-attacked by some potential argument in $P_{\mathcal{Q}}(T)$ (that is not p-attacked by any observation-based argument in Arg_T), which must then be rule-based. We consider two possibilities:
  - One possibility would be that each potential argument in $P_{\mathcal{Q}}(T)$ p-attacking A that is not p-attacked by any observation-based argu-

[6] See for example Figure B.8: t is stable-out in T but not labelled as such.

ment in Arg_T is inconsistent[7] *(Case 3a)*.

– Alternatively, there exists some *consistent* potential argument B^p in $P_{\mathcal{Q}}(T)$ that p-attacks A and is not p-attacked by any observation-based argument in Arg_T. Let $b = \texttt{conc}(B^p)$ and consider $T' = (AS, \mathcal{K} \cup \{b\})$. There are two possibilities:

∗ The introduction of B^p in T forces a new argument for l[8]. This corresponds to *Case 3b*.

∗ Alternatively, the arguments for t in $Arg_{T'}$ are exactly the same as the arguments for t in Arg_T. Then it cannot be possible to attack all arguments for t at the same time by arguments that are not attacked by an observation-based argument. So by Definition 3.23, l is unattackable in T w.r.t. $\mathcal{Q}$[9] *(Case 3c)*.

Blocked Suppose that for each T' such that $T \sqsubseteq_Q T'$, l is blocked in T' and $L[l] \neq \langle 0, 0, 0, 1\rangle$. We consider two cases.

- First suppose that l is labelled $L[l].u$ or $L[l].o$. Since l is blocked in each T' such that $T \sqsubseteq_Q T'$, it cannot be unsatisfiable or out. Then by Lemma B.19, either l is observation-unattackable in T w.r.t. $\mathcal{Q}$ or some argument for l in Arg_T is p-attacked by an observation-based potential argument B^p such that the introduction of B^p in T forces a new argument for l *(Case 2)*.
- Alternatively, l is labelled $\neg L[l].u$ and $\neg L[l].o$ but still $L \neq \langle 0, 0, 0, 1\rangle$. Then l should be labelled $L[l].d$. By Lemma B.17 there is a potential argument $A^p \in P_{\mathcal{Q}}(T)$ for l such that each argument B in Arg_T that p-attacks A^p is p-attacked by some observation-based potential argument $C^p \in P_{\mathcal{Q}}(T)$. Given that l is stable-blocked in T, $l \notin \mathcal{Q}$ (otherwise there would be some future argumentation theory in which either l or some of its contradictories would be in the knowledge base). So each potential argument for l in $P_{\mathcal{Q}}(T)$ must be rule-based.

  – One possibility is that $\texttt{prem}(A^p)$ is inconsistent.[10] This corresponds to *Case 5a*.

  – Alternatively, suppose that $\texttt{prem}(A^p)$ is consistent. Then A^p is an argument in $Arg_{T'}$ where $T' = (AS, \mathcal{K} \cup \texttt{prem}(A^p))$. Given that l is stable-blocked in T, it must be blocked in T', so there must be some argument B in $Arg_{T'}$ that attacks A^p.

[7] See Figure B.5 for an example. The only argument for t in Arg_T is p-attacked by an inconsistent potential argument.

[8] See Figure B.6: the introduction of each potential argument p-attacking the argument $[q_1 \Rightarrow a] \Rightarrow t$ in T forces a new argument for t.

[9] See Figure B.7 for an example: there are two arguments for t in Arg_T, but it is not possible to attack them both because the set of potential attackers attacking arguments for t in Arg_T is inconsistent.

[10] In Figure B.9 there is an inconsistent potential argument for t that is not p-attacked by any argument in Arg_T.

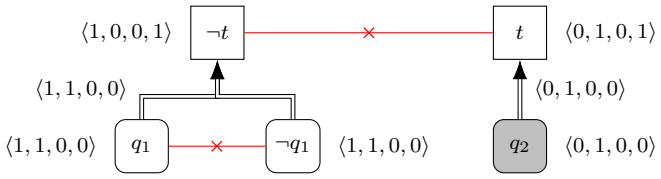


Figure B.5: Although t is stable-defended in T, it is labelled $L[t].b$ because the only argument for t in Arg_T (i.e. $q_2 \Rightarrow t$) is p-attacked by the inconsistent potential argument $q_1, \neg q_1 \Rightarrow t$. *(Case 3a)*

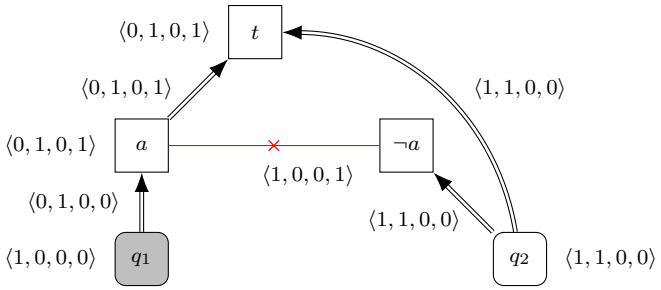


Figure B.6: t is defended in T. However, t is labelled $L[t].b$. The reason for this is that for each argument A for t in Arg_T (in this case only $[q_1 \Rightarrow a] \Rightarrow t$), there is some potential argument B^p (in this case $q_2 \Rightarrow \neg a$) p-attacking A that is not p-attacked by any observation-based argument. However, the introduction of B^p in T forces a new argument ($q_2 \Rightarrow t$). *(Case 3b)*

- First suppose that $B \notin Arg_T$. Then there is some $b \in \texttt{prem}(B)$ such that $b \in \texttt{prem}(A^p)$.[11] This implies that the introduction of A^p in T forces an argument attacking A^p, which corresponds to *Case 5b*.
- Alternatively, $B \in Arg_T$. However, given that l is labelled $L[l].d$, there must be some observation-based potential argument C^p in $P_{\mathcal{Q}}(T)$ that p-attacks B.[12] Still, $C^p \notin Arg_{T''}$ for any T'' such that $T' \sqsubseteq_Q T''$ (since l is blocked and not defended in T'). Let $c = \texttt{conc}(C^p)$. So for each $T'' = (AS, \mathcal{K}'')$ such that $T' \sqsubseteq_Q T''$ there is some $c' \in \overline{c}$ in $\mathcal{K}$. This implies that there is some $c' \in \overline{c}$ in $\mathcal{K}' = \mathcal{K} \cup \texttt{prem}(A^p)$. Given that $C^p \in P_{\mathcal{Q}}(T)$, no $c' \in \overline{c}$ was in $\mathcal{K}$. So there is some $c' \in \overline{c}$ in $\texttt{prem}(A^p)$. By Definition 3.24, this implies that the introduction of C^p in T forces an argument that attacks A^p *(Case 5c)*.

To conclude, each situation in which l is stable in T but not labelled as such by L matches one of the five cases. □

[11] See Figure B.10 for an example. Whereas t is stable-blocked in T, it is labelled $L[t].d$ because there is a potential argument for t of which the introduction in T forces an argument attacking itself.

[12] See $\neg t$ in Figure 3.21 for an example.

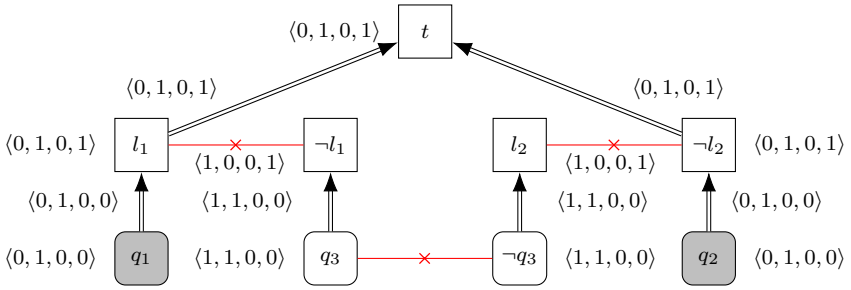


Figure B.7: t is stable-defended in T, but it is labelled $L[t].b$ because each argument for t in Arg_T is p-attacked by a potential argument in $P_{\mathcal{Q}}(T)$ that is not p-attacked by an observation-based argument in Arg_T. However, there is no T' such that $T \sqsubseteq_Q T'$ in which all arguments for t are attacked by an argument in $Arg_{T'}$ that is not attacked by an observation-based argument. *(Case 3c)*

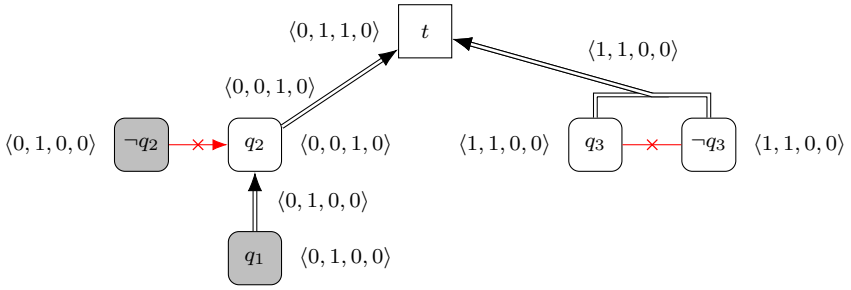


Figure B.8: t is stable-out in T. However, t is labelled $L[l].d$ because each potential argument for l in $P_{\mathcal{Q}}(T)$ that is not p-attacked by an observation-based argument in Arg_T (in this case only $q_3, \neg q_3 \Rightarrow t$) is inconsistent. *(Case 4)*

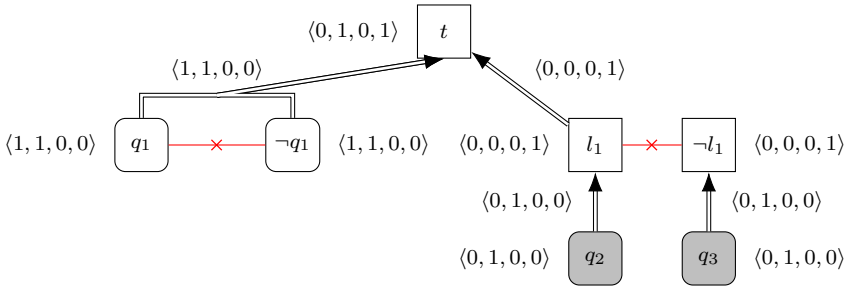


Figure B.9: Although t is stable-blocked in T, it is labelled $L[t].d$ because there is an inconsistent potential argument $q_1, \neg q_1 \Rightarrow t$ for t in $P_{\mathcal{Q}}(T)$ that is not p-attacked by any argument in Arg_T. *(Case 5a)*

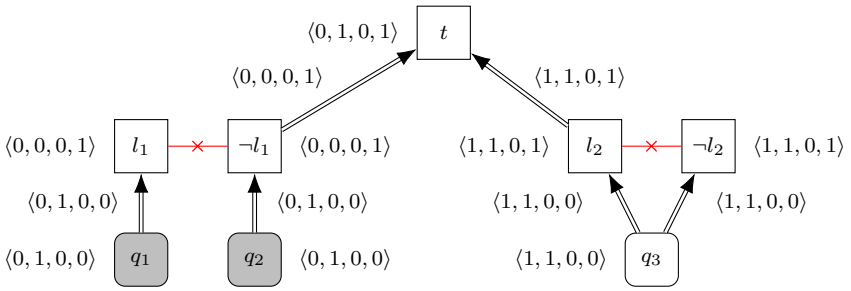


Figure B.10: Although t is stable-blocked in T, it is labelled $L[t].d$ because there is a potential argument $A^p : [q_3 \Rightarrow l_2] \Rightarrow t$ for t in $P_{\mathcal{Q}}(T)$ of which the introduction forces an argument attacking A^p. *(Case 5b)*

B.2. ASP-based algorithms

Lemma 3.7 (Specification "in" grounded extension)**.** *Let $T = (AS, \mathcal{K})$ be an argumentation theory with argumentation system $AS = (\mathcal{L}, \bar{}, \mathcal{R}, \leq)$ where $\leq = \emptyset$. An argument $A \in Arg_T$ is in the grounded extension $G(T)$ iff each argument defeating A is defeated by an observation-based argument.*

Proof. The proof from *right to left* is trivial: observation-based arguments cannot be defeated (Definition 3.6), so each observation-based argument is in $G(T)$. If each argument defeating A is defeated by an observation-based argument, then A is defended by $G(T)$, so $A \in G(T)$.

We now prove the *left to right* part by contradiction. Suppose that $A \in G(T)$ and that there is an argument B defeating A, and B is not defeated by an observation-based argument. We will prove that there is a strict subset of $G(T)$ that is complete, which contradicts the assumption that $G(T)$ is a grounded extension. We construct this set S as follows:

$$S = \{C \in G(T) \mid \text{ there is no } C' \in \mathtt{sub}(C) \text{ such that } C' \text{ defeats } B\}$$

First, we show that S is a *strict subset* of $G(T)$. Since A is defeated by B, there must be some $A' \in \mathtt{sub}(A)$ such that $\mathtt{conc}(B) \in \overline{\mathtt{conc}(A')}$ and $\mathtt{conc}(A') \notin \mathcal{K}$. Given that B cannot be observation-based (otherwise B would be in $G(T)$, which contradicts the conflict-freeness of $G(T)$) and $\mathtt{conc}(A') \in \overline{\mathtt{conc}(B)}$ (by the symmetry of $\bar{}$), A' defeats B. This implies that A cannot be in S, hence $S \subset G(T)$. Next, we prove that S is complete.

- $S \subset G(T)$ and $G(T)$ is conflict-free, so S *is conflict-free.*
- S *is an admissible set*: suppose, towards a contradiction, that there is some $D \in S$ such that some argument E defeats D and each argument defeating E is not in S. E defeats D, so there is an argument $D' \in \mathtt{sub}(D)$ such that $\mathtt{conc}(E) \in \overline{\mathtt{conc}(D')}$. Since $D \in G(T)$, which is complete, there must

be some argument in Arg_T that defeats E, which means that $\texttt{conc}(E) \notin \mathcal{K}$. Furthermore, since the contradiction is symmetric, $\texttt{conc}(D') \in \overline{\texttt{conc}(E)}$. This implies that D' defeats E and therefore $D' \notin S$. However note that $D' \in G(T)$: $D \in G(T)$, so each argument defeating D is defeated by some argument in $G(T)$; each argument defeating D' also defeats D and therefore must be defeated by some argument in $G(T)$ so $D' \in G(T)$. By definition of S, D' has a subargument that defeats B. But then D must have the same subargument defeating B, so $D \notin S$; a contradiction. As a consequence, S must be an admissible set.

- *Each argument that is acceptable w.r.t. S is in S*: suppose, towards a contradiction, that there exists an argument $D \in Arg_T$ that is acceptable w.r.t. S and $D \notin S$. If D is acceptable w.r.t. S, then D is acceptable w.r.t. $G(T)$; therefore $D \in G(T)$. $D \notin S$, so by definition of S there is a subargument $D' \in \texttt{sub}(D)$ such that D' defeats B. Let $B' \in \texttt{sub}(B)$ be the subargument on which D' defeats B: $\texttt{conc}(D') \in \overline{\texttt{conc}(B')}$. By an earlier assumption, B is not defeated by an observation-based argument, so $\texttt{conc}(D') \notin \mathcal{K}$. Furthermore, by symmetry of contradiction, $\texttt{conc}(B') \in \overline{\texttt{conc}(D')}$, which means that B' defeats D' and therefore also defeats D. Since D is acceptable w.r.t. S, there must be an argument E in S defeating B'. But then E would defeat B as well, hence $E \notin S$, a contradiction. As a result, each argument that is acceptable w.r.t. S is in S.

To conclude, there is a set $S \subset G(T)$ such that S is complete. This contradicts our assumption that $G(T)$ is the grounded extension since the grounded extension is minimal w.r.t. set inclusion. So if $A \in G(T)$, then each argument defeating A is defeated by an observation-based argument. □

Lemma 3.8 (Specification "out" arguments)**.** *Let $T = (AS, \mathcal{K})$ be an argumentation theory with argumentation system $AS = (\mathcal{L}, {}^{-}, \mathcal{R}, \leq)$ where $\leq = \emptyset$. An argument $A \in Arg_T$ is defeated by an argument in the grounded extension $G(T)$ (A is "out"), iff A is defeated by an observation-based argument.*

Proof. We prove this in both directions:

- *Right to left*: if an argument $A \in Arg_T$ is defeated by an observation-based argument B, then $B \in G(T)$, since B cannot be defeated. So A is defeated by an argument in the grounded extension.
- *Left to right*: let $A \in Arg_T$ be an argument that is defeated by some $B \in G(T)$ and suppose, towards a contradiction, that A is not defeated by any observation-based argument. Then there is a subargument $A' \in \texttt{sub}(A)$ such that $\texttt{conc}(B) \in \overline{\texttt{conc}(A')}$ and therefore, by the symmetry of ${}^{-}$, $\texttt{conc}(A') \in \overline{\texttt{conc}(B)}$, $\texttt{conc}(A') \notin \mathcal{K}$ and $\texttt{conc}(B) \notin \mathcal{K}$, hence A' defeats B. $B \in G(T)$, so by Lemma 3.7, A' must be defeated by an observation-based argument. However, this observation-based argument would defeat A as well; a contradiction. To conclude, A is defeated by an observation-based argument. □

C

Appendix to Chapter 4

This appendix provides full proofs for the propositions stated in Chapter 4.

C.1. Maintaining consistency

Proposition 4.1. *Let $\mathcal{C}$ be a consistent complete case base and c a complete fact situation.*

1. *If c is PRO w.r.t. $\mathcal{C}$ then $\mathcal{C} \cup \{(c, PRO)\}$ is consistent.*
2. *If c is PRO w.r.t. $\mathcal{C}$ then $\mathcal{C} \cup \{(c, CONTRA)\}$ is inconsistent.*
3. *If c is CONTRA w.r.t. $\mathcal{C}$ then $\mathcal{C} \cup \{(c, PRO)\}$ is inconsistent.*
4. *If c is CONTRA w.r.t. $\mathcal{C}$ then $\mathcal{C} \cup \{(c, CONTRA)\}$ is consistent.*
5. *If c is UNDECIDED w.r.t. $\mathcal{C}$ then $\mathcal{C} \cup \{(c, PRO)\}$ is consistent.*
6. *If c is UNDECIDED w.r.t. $\mathcal{C}$ then $\mathcal{C} \cup \{(c, CONTRA)\}$ is consistent.*

Proof. Let $\mathcal{C}$ be a consistent complete case base and let c be a complete fact situation. We prove the items separately.

1. Suppose that c is PRO w.r.t. $\mathcal{C}$. By Definition 4.8, there is some $(c', \text{PRO}) \in \mathcal{C}$ such that $c \geq_{\texttt{PRO}} c'$. Suppose, towards a contradiction, that $\mathcal{C} \cup \{(c, \text{PRO})\}$ is not consistent. Then there is some $(c'', \text{CONTRA}) \in \mathcal{C}$ such that $c \geq_{\texttt{CONTRA}} c''$, which equals $c'' \geq_{\texttt{PRO}} c$. But that would imply (by transitivity of $\geq_{\texttt{PRO}}$) that $c'' \geq_{\texttt{PRO}} c'$ while $(c', \text{PRO}) \in \mathcal{C}$, therefore $\mathcal{C}$ would be inconsistent; contradiction. To conclude, $\mathcal{C} \cup \{(c, \text{PRO})\}$ is consistent.
2. If c is PRO w.r.t. $\mathcal{C}$ then by Definition 4.8, there is some $(c', \text{PRO}) \in \mathcal{C}$ such that $c \geq_{\texttt{PRO}} c'$. Given that the complete case base $\mathcal{C} \cup \{(c, \text{CONTRA})\}$ contains both complete cases with c and c' while $c \geq_{\texttt{PRO}} c'$, it is inconsistent by Definition 4.7.

3. The proof is analogous to the proof of Item 2.
4. The proof is analogous to the proof of Item 1.
5. Suppose that c is UNDECIDED w.r.t. $\mathcal{C}$. Suppose, towards a contradiction, that $\mathcal{C} \cup \{(c, \text{PRO})\}$ is not consistent. Then there is some $(c'', \text{CONTRA}) \in \mathcal{C} \cup \{(c, \text{PRO})\}$ such that $c'' \geq_{\texttt{PRO}} c$, hence $c \geq_{\texttt{CONTRA}} c''$. But then $(c'', \text{CONTRA}) \in \mathcal{C}$, which by Definition 4.8 implies that c is CONTRA w.r.t. $\mathcal{C}$; contradiction. So $\mathcal{C} \cup \{(c, \text{PRO})\}$ is consistent.
6. The proof is analogous to the proof of Item 5. □

Proposition 4.2 (Impossibility of inconsistency)**.** *Let $\mathcal{C}$ be a consistent incomplete case base and c an incomplete fact situation. For each $o \in \{PRO, CONTRA\}$: if c is Stable-o w.r.t. $\mathcal{C}$ then $\mathcal{C} \cup \{(c, o)\}$ is consistent.*

Proof. Suppose that c is Stable-PRO w.r.t. $\mathcal{C}$; then for each completion c' of c, there is some $(c^*, \text{PRO}) \in \mathcal{C}$ that has some completion $c^{*\prime}$ such that $c' \geq_{\texttt{PRO}} c^{*\prime}$. Given that $\mathcal{C}$ is consistent, there are no (c_1, CONTRA) and (c_2, PRO) in $\mathcal{C}$ such that c_1 has a completion c_1' and c_2 has a completion c_2' for which $c_1' \geq_{\texttt{PRO}} c_2'$. Thus, assuming towards a contradiction that $\mathcal{C} \cup \{(c, \text{PRO})\}$ is inconsistent, the inconsistency must have been introduced by adding (c, PRO): there must be some $(c^\star, \text{CONTRA})$ in $\mathcal{C}$ and some completion c' of c such that $c^\star$ has a completion $c^{\star\prime}$ and $c^{\star\prime} \geq_{\texttt{PRO}} c'$. But then, by transitivity of $\geq_{\texttt{PRO}}$, $c^{\star\prime} \geq_{\texttt{PRO}} c^{*\prime}$, which would imply that $\mathcal{C}$ was inconsistent. From this contradiction we derive that $\mathcal{C} \cup \{(c, \text{PRO})\}$ is consistent. □

C.2. Properties of relevance

Proposition 4.3. *For any incomplete case base $\mathcal{C}$, incomplete fact situation c and dimension d in domain $\mathcal{D}$ such that $d = (\{v_1, v_2\}, \leq_{PRO})$ and $\leq_{PRO} = \{(v_1, v_1), (v_1, v_2), (v_2, v_2)\}$, the removal of some possible value v for d cannot be both PRO- and CONTRA-relevant w.r.t. $\mathcal{C}$ and c.*

Proof. Suppose that we are given an incomplete case base $\mathcal{C}$, incomplete fact situation c and dimension d such that $d = (\{v_1, v_2\}, \leq_{\texttt{PRO}})$ with $\leq_{\texttt{PRO}} = \{(v_1, v_1), (v_1, v_2), (v_2, v_2)\}$ and value $v \in \{v_1, v_2\}$. In the following, we will show that only the removal of v_1 for d can be PRO-relevant, while only the removal of v_2 for d can be CONTRA-relevant w.r.t. $\mathcal{C}$ and c.

1. Suppose that removal of some $v \in \{v_1, v_2\}$ for d is PRO-relevant w.r.t. $\mathcal{C}$ and c. First, we show that $v = v_1$: suppose, towards a contradiction, that $v = v_2$. Then by Definition 4.18 there is a minimal stable-PRO partial completion c_1 w.r.t. $\mathcal{C}$ and c such that $v_2 \notin V_p(d, c_1)$. Then $V_p(d, c_1) = \{v_1\}$ and for each completion c_2 of c_1, there is some incomplete case $(c', \text{PRO}) \in C$ having some completion c'' such that $c_2 \geq_{\texttt{PRO}} c''$. Now construct c_3 that equals c_1, except for the possible values for d: $V_p(d, c_3) = \{v_1, v_2\}$. Note that c_3 must be in $F(c)$, as c_1 is a partial completion of c and c_3 is almost the same, except that the possible values for d also include v_2, which must have been in $V_p(d, c)$ as we

assumed that the removal of v_2 is PRO-relevant for c. Given that $v_2 \geq_{\texttt{PRO}} v_1$, it must be that for each completion c_4 of c_3, there is some completion c_2 of c_1 such that $c_4 \geq_{\texttt{PRO}} c_2$, hence there is some incomplete case $(c''', \text{PRO}) \in C$ having some completion c'''' such that $c_4 \geq_{\texttt{PRO}} c''''$. But then c_1 was not minimal, as c_1 is a partial completion of c_3; contradiction.

2. Now suppose that removal of some $v \in \{v_1, v_2\}$ for d is CONTRA-relevant w.r.t. $\mathcal{C}$ and c. We will show that $v = v_2$ by contradiction: suppose that $v = v_1$. Then there is a minimal stable-CONTRA partial completion c_1 w.r.t. $\mathcal{C}$ and c such that $v_1 \notin V_p(d, c_1)$. Consequently, $V_p(d, c_1) = \{v_2\}$ and for each completion c_2 of c_1, there is some incomplete case $(c', \text{CONTRA}) \in C$ having some completion c'' such that $c_2 \geq_{\texttt{CONTRA}} c''$. Construct c_3 that equals c_1, except for the possible values for d: $V_p(d, c_3) = \{v_1, v_2\}$. Note that c_3 must be in $F(c)$, as c_1 is a partial completion of c and c_3 is almost the same, except that the possible values for d also include v_1, which must have been in $V_p(d, c)$ as we assumed that the removal of v_1 is CONTRA-relevant for c. Given that $v_1 \geq_{\texttt{CONTRA}} v_2$, it must be that for each completion c_4 of c_3, there is some completion c_2 of c_1 such that $c_4 \geq_{\texttt{CONTRA}} c_2$, hence there is some incomplete case $(c''', \text{CONTRA}) \in C$ having some completion c'''' such that $c_4 \geq_{\texttt{CONTRA}} c''''$. But then c_1 was not minimal, as c_1 is a partial completion of c_3; contradiction.

Given that only the removal of v_1 for d can be PRO-relevant, while only the removal of v_2 for d can be CONTRA-relevant w.r.t. $\mathcal{C}$ and c and $V_p(d, c) = \{v_1, v_2\}$, there can be no v such that the removal of v is both PRO- and CONTRA-relevant w.r.t. $\mathcal{C}$ and c. □

Proposition 4.4. *For any incomplete case base $\mathcal{C}$, incomplete fact situation c and justification status j, let ϕ_j be an arbitrary function from $F(c)$ to $F(c)$ such that, for a given $c' \in F(c)$:*

1. *if there is some d for which there is some v such that removing v is j-relevant w.r.t. $\mathcal{C}$ and c', then $\phi_j(c')$ returns a partial completion c'' of c' such that $V_p(d, c'') = V_p(d, c') \setminus \{v\}$ while for all d' such that $d' \neq d$: $V_p(d', c'') = V_p(d', c')$.*
2. *otherwise, $\phi_j(c') = c'$.*

Then for any partial completion $c' \in F(c)$ that has a completion that is j w.r.t. $\mathcal{C}$, there is some minimal stable-j partial completion c'' that can be obtained within a finite number n of applications of ϕ_j (so $\phi_j^n(\phi_j^{n-1}(\ldots \phi_j^1(c') \ldots)) = c''$).

Proof. For any incomplete case base $\mathcal{C}$ in domain $\mathcal{D}$, incomplete fact situation c and justification status j, let ϕ_j be an arbitrary function as specified above and suppose that $c' \in F(c)$ has a completion that is j w.r.t. $\mathcal{C}$. An upper bound on the number of iterations to reach a fixed point is $T = \Sigma_{d \in \mathcal{D}}(|V_p(d, c')| - 1)$: after T iterations, no value can be removed from any dimension, as there is only one possible value for each dimension. We prove by induction that, after the i'th iteration, $\phi_j^i(c')$ has at least one completion that is j w.r.t. $\mathcal{C}$. For $i = 0$, this follows from the

assumption that c' has a completion that is j w.r.t. $\mathcal{C}$. Assuming the induction hypothesis for i, the proof for $(i+1)$ follows from the definition of ϕ_j: given that $c_i = \phi_j^i(\phi_j^{i-1}(\ldots\phi_j^1(c')\ldots))$ has a completion that is j, there must be some minimal stable-j partial completion w.r.t. $\mathcal{C}$ and c'. Then there are two possibilities:

- c_i itself is a minimal stable-j partial completion w.r.t. $\mathcal{C}$ and c. Then there is no relevant update, so $\phi_j^{i+1}(c_i) = c_i$. From the induction hypothesis, it follows that $\phi_j^{i+1}(c_i)$ has a completion that is j w.r.t. $\mathcal{C}$.
- Alternatively, there are some d and v such that removal of v for d is j-relevant w.r.t. $\mathcal{C}$ and c and $\phi_j^{i+1}(c_i)$ is obtained by removing v from the possible values for d. Given that removing v is j-relevant, there is some minimal stable-j partial completion of c_i that does not have v in its possible values. Then this partial completion is still a partial completion of $\phi_j(c_i)$, so $\phi_j^{i+1}(c_i)$ has a completion that is j w.r.t. $\mathcal{C}$.

After T iterations, $\phi_j^T(c_i)$ has a completion that is j w.r.t. $\mathcal{C}$ and there can be no value that is j-relevant to remove for any dimension. Then $\phi_j(c_i)$ is a minimal stable-j completion w.r.t. $\mathcal{C}$. □

C.3. Possibility

Proposition 4.5. *[Maintaining consistency] Let $\mathcal{C}$ be an incomplete case base that is consistent and let c be an incomplete fact situation.*

1. *$\mathcal{C} \cup (c, PRO)$ is consistent iff c is **not** Possible-CONTRA given $\mathcal{C}$; and*
2. *$\mathcal{C} \cup (c, CONTRA)$ is consistent iff c is **not** Possible-PRO given $\mathcal{C}$.*

Proof. We give the proof for the first item here; the proof for the second item is analogous. Let $\mathcal{C}$ be a consistent incomplete case base and let c be an incomplete fact situation.

Left to right Suppose that $\mathcal{C} \cup (c, \text{PRO})$ is consistent. Then by Definition 4.13 there is no incomplete case (c', CONTRA) in $\mathcal{C} \cup (c, \text{PRO})$ such that c' has a completion c'' and c has a completion c^* for which $c'' \geq_{\texttt{PRO}} c^*$. Then there is no incomplete case (c', CONTRA) in $\mathcal{C}$ such that c' has a completion c'' and c has a completion c^* for which $c^* \geq_{\texttt{CONTRA}} c''$. Consequently, by Definition 4.19, c is not Possible-CONTRA given $\mathcal{C}$.

Right to left Suppose that c is not Possible-CONTRA given $\mathcal{C}$. Then by Definition 4.19 there is no completion of c that is CONTRA given $\mathcal{C}$. Now suppose towards a contradiction that $\mathcal{C} \cup (c, \text{PRO})$ is inconsistent. Then by Definition 4.13 there is some incomplete case (c', CONTRA) in $\mathcal{C} \cup (c, \text{PRO})$ such that c' has a completion c'' and c has a completion c^* for which $c'' \geq_{\texttt{PRO}} c^*$. But then $c^* \geq_{\texttt{CONTRA}} c''$. Since it must be that (c', CONTRA) was already in $\mathcal{C}$, this implies that c must be Possible-CONTRA given $\mathcal{C}$; contradiction. To conclude, $\mathcal{C} \cup (c, \text{PRO})$ must be consistent. □

C.4. Computational aspects of justification

Proposition 4.6. *[Reformulation of the a fortiori constraint for incomplete case bases] Let $\mathcal{C}$ be a consistent incomplete case base in domain $\mathcal{D}$ and c a complete fact situation. Then for outcome o (PRO or CONTRA): c is forced towards outcome o by $\mathcal{C}$ iff there is some incomplete (c', o) in $\mathcal{C}$ where c' is such that for each $d \in \mathcal{D}$ there is some $v' \in V_p(d, c')$ such that $v(d, c) \geq_o v'$.*

Proof. Let $\mathcal{C}$ be a consistent incomplete case base, let c be a fact situation and let o be an outcome in $\{\text{PRO}, \text{CONTRA}\}$. We proceed by proving both directions.

- Suppose that c is forced towards outcome o by $\mathcal{C}$. Then by Definition 4.12 there is some incomplete (c', o) in $\mathcal{C}$ such that c' has at least one completion c'' such that $c \geq_o c''$. Then by Definition 4.5 for all $d \in \mathcal{D}$: $v(d, c) \geq_o v(d, c'')$. Given that c'' is a completion of c', it must be that $v(d, c'') \in V_p(d, c')$. So for each $d \in \mathcal{D}$, there is some v' $(v' = v(d, c''))$ in $V_p(d, c')$ such that $v(d, c) \geq_o v'$.
- Suppose that there is some incomplete (c', o) in $\mathcal{C}$ where c' is such that for each $d \in \mathcal{D}$ there is some $v' \in V_p(d, c')$ such that $v(d, c) \geq_o v'$. Now construct the fact situation c'' by selecting for each dimension $d \in \mathcal{D}$ one value v' from $V_p(d, c')$ such that $v(d, c) \geq_o v'$. Given that, for each dimension $d \in \mathcal{D}$, $v(d, c'') \in V_p(d, c')$, by Definition 4.10, c'' is a completion of c'. In addition, $c \geq_o c''$ by Definition 4.5. We thus conclude that c is forced towards outcome o by $\mathcal{C}$ (Definition 4.12). □

Proposition 4.7. *Deciding if a complete fact situation is PRO/CONTRA/UNDECIDED w.r.t. some consistent incomplete case base is in P.*

Proof. Algorithm 8 computes the justification status of a (complete) fact situation c given an incomplete case base $\mathcal{C}$ in domain $\mathcal{D}$. Correctness of the algorithm follows directly from Proposition 4.6. The for-loop in Line 2–4 is iterated at most $|\mathcal{C}|$ times. Let *maxval* be the maximum number of possible values V_p over all incomplete cases in $\mathcal{C}$ and dimensions in $\mathcal{D}$; note that this must be a finite constant. Within this loop, at most $c_1 + c_2 \cdot |\mathcal{D}| \cdot \mathit{maxval}$ operations are performed in Line 3, where c_1 and c_2 are positive constants. A single iteration of Line 4 or 5 takes constant time. Therefore, the runtime of the algorithm is $\mathcal{O}(|\mathcal{C}| \cdot |\mathcal{D}| \cdot \mathit{maxval})$, which is polynomial in the input. □

C.5. Computational aspects of possibility

Proposition 4.8. *Let $\mathcal{C}$ be a consistent incomplete case base in domain $\mathcal{D}$ and c an incomplete fact situation. Then for outcome o (where o is PRO or CONTRA): c is Possible-o w.r.t. $\mathcal{C}$ iff there is some incomplete (c', o) in $\mathcal{C}$ where c' is such that for each $d \in \mathcal{D}$ there is some $v \in V_p(d, c)$ such that there is some $v' \in V_p(d, c')$ such that $v \geq_o v'$.*

Proof. We will prove this for the outcome PRO; for CONTRA, the proof is analogous. Let $\mathcal{C}$ be a consistent incomplete case base in domain $\mathcal{D}$ and let c be an incomplete fact situation.

Left to right Suppose that c is Possible-PRO w.r.t. $\mathcal{C}$. Then by Definition 4.19, there is some completion c^* of c such that there is some incomplete case $(c', \text{PRO}) \in \mathcal{C}$ that has a completion c'' and $c^* \geq_{\texttt{PRO}} c''$ (so by Definition 4.5 for each $d \in \mathcal{D}$: $v(d, c^*) \geq_{\texttt{PRO}} v(d, c'')$). Given that c^* is a completion of c, $v(d, c^*) \in V_p(d, c)$. Similarly, as c'' is a completion of c', $v(d, c'') \in V_p(d, c')$. Therefore, for each $d \in \mathcal{D}$, there is some $v \in V_p(d, c)$ (i.e., $v(d, c^*)$) such that there is some $v' \in V_p(d, c')$ (i.e., $v(d, c'')$) such that $v \geq_{\texttt{PRO}} v'$.

Right to left Suppose that there is some $(c', o) \in \mathcal{C}$ where c' is such that for each $d \in \mathcal{D}$ there is some $v \in V_p(d, c)$ such that there is some $v' \in V_p(d, c')$ such that $v \geq_{\texttt{PRO}} v'$. Now construct the completions c^* of c and c'' of c' by collecting for each $d \in \mathcal{D}$ exactly one value $v \in V_p(d, c)$ such that there is some $v' \in V_p(d, c')$ such that $v \geq_{\texttt{PRO}} v'$ and assigning $v(d, c^*) = v$ and $v(d, c'') = v'$. By Definition 4.5 $c^* \geq_{\texttt{PRO}} c''$, therefore c is Possible-PRO w.r.t. $\mathcal{C}$. □

Proposition 4.9. *Deciding if an incomplete fact situation is Possible-PRO and/or Possible-CONTRA w.r.t. some incomplete case base is in P.*

Proof. Algorithm 9 checks if an incomplete fact situation c is Possible-PRO or Possible-CONTRA given an incomplete case base $\mathcal{C}$ in domain $\mathcal{D}$. Correctness of the algorithm follows directly from Proposition 4.8. The for-loop in Line 2–5 is iterated at most $|\mathcal{C}|$ times. Let *maxval* be the maximum number of possible values V_p over all incomplete cases in $\mathcal{C}$ and dimensions in $\mathcal{D}$; note that this must be a finite constant. Within this loop, at most $c_1 + c_2 \cdot |\mathcal{D}| \cdot \mathit{maxval} \cdot \mathit{maxval}$ operations are preformed in Line 4, where c_1 and c_2 are positive constants. A single iteration of Line 3, 5 or 6 takes constant time. Therefore, the runtime of the algorithm is $\mathcal{O}(|\mathcal{C}| \cdot |\mathcal{D}| \cdot \mathit{maxval}^2)$, which is polynomial in the input. □

C.6. Computational aspects of stability

Proposition 4.10. *Given an incomplete case base $\mathcal{C}$ and an incomplete fact situation c:*

1. *c is Stable-PRO given $\mathcal{C}$ iff each MC(c) is PRO given $\mathcal{C}$;*
2. *c is Stable-CONTRA given $\mathcal{C}$ iff each MP(c) is CONTRA given $\mathcal{C}$; and*
3. *c is Unstable given $\mathcal{C}$ iff the justification status of some MP(c) does not equal the justification status of some MC(c) given $\mathcal{C}$.*

Proof. The proof for items 1 and 2 from left to right follow directly from Definition 4.15 for stability: for a given justification status j (PRO or CONTRA) such that c is Stable-j given $\mathcal{C}$, it must be that each completion, including each completion in MC(c) and MP(c), has the status j given $\mathcal{C}$.

For item 1 from right to left, suppose that each c^* in $\text{MC}(c)$ is PRO given $\mathcal{C}$ and let c' be an arbitrary completion of c. By Definition 4.20, there must be some $c^* \in \text{MC}(c)$ such that $c^* \geq_{\texttt{CONTRA}} c'$, so $c' \geq_{\texttt{PRO}} c^*$. Since c^* is PRO given $\mathcal{C}$, there must be some $(c'', \text{PRO}) \in C$ such that $c^* \geq_{\texttt{PRO}} c''$. By transitivity of $\geq_{\texttt{PRO}}$, $c' \geq_{\texttt{PRO}} c''$, so c' is PRO given $\mathcal{C}$. As c' was chosen arbitrarily, c must be Stable-PRO given $\mathcal{C}$.

The proof for item 2 from right to left is analogous: suppose that each c^* in $\text{MP}(c)$ is CONTRA given $\mathcal{C}$ and consider an arbitrary completion c' of c. By Definition 4.20, there must be some $c^* \in \text{MP}(c)$ such that $c^* \geq_{\texttt{PRO}} c'$, so $c' \geq_{\texttt{CONTRA}} c^*$. Given that c^* is CONTRA given $\mathcal{C}$, there must be some $(c'', \text{CONTRA}) \in C$ such that $c^* \geq_{\texttt{CONTRA}} c''$. By transitivity of $\geq_{\texttt{CONTRA}}$, $c' \geq_{\texttt{CONTRA}} c''$, so c' is CONTRA given $\mathcal{C}$. Since c' was chosen arbitrarily, c must be Stable-CONTRA given $\mathcal{C}$.

Finally consider item 3. The right to left part follows directly from Definition 4.15 for stability. For the proof from left to right, suppose that c is Unstable given $\mathcal{C}$. Then at least one of the following is true:

- There is some completion c_1 of $\mathcal{C}$ such that c_1 is PRO given $\mathcal{C}$ and there is some completion c_2 of $\mathcal{C}$ such that c_2 is not PRO given $\mathcal{C}$. As there must be some $c^* \in \text{MP}(c)$ such that $c^* \geq_{\texttt{PRO}} c_1$, this c^* must be PRO given $\mathcal{C}$. Given that c_2 is not PRO and there must be some $c^- \in \text{MC}(c)$ such that $c_2 \geq_{\texttt{PRO}} c^-$, this c^- cannot be PRO given $\mathcal{C}$. So the justification statuses of $c^* \in \text{MP}(c)$ and $c^- \in \text{MC}(c)$ are not equal.
- There is some completion c_1 of $\mathcal{C}$ such that c_1 is CONTRA given $\mathcal{C}$ and there is some completion c_2 of $\mathcal{C}$ such that c_2 is not CONTRA given $\mathcal{C}$. Analogously, there must be some $c^- \in \text{MC}(c)$ such that $c^- \geq_{\texttt{CONTRA}} c_1$, which must be CONTRA and some $c^* \in \text{MP}(c)$ such that $c_2 \geq_{\texttt{CONTRA}} c^*$, which cannot be CONTRA given $\mathcal{C}$. So $c^* \in \text{MP}(c)$ and $c^- \in \text{MC}(c)$ do not have equal justification statuses. □

Proposition 4.11. *Given an incomplete case base $\mathcal{C}$ in domain $\mathcal{D}$ and some incomplete fact situation c, if for each $d \in \mathcal{D}$ the set $V_p(d, c)$ has a single minimum and a single maximum value, then* Check-Stable-PRO/Check-Stable-CONTRA *runs in polynomial time for $\mathcal{D}$, $\mathcal{C}$ and c.*

Proof. Suppose that for each $d \in \mathcal{D}$, the set $V_p(d, c)$ has a single minimum and a single maximum value. Then c has a single element in $\text{MP}(c)$ that is obtained by selecting the maximum possible value for each dimension. Similarly, $\text{MC}(c)$ has a single element, obtained by selecting the minimum possible values. Since $|\text{MP}(c)| + |\text{MC}(c)| = 2$, at most two calls to the polynomial algorithm for justification are required, so the stability algorithm based on extreme completions runs in polynomial time for c. □

Proposition 4.12. *Deciding whether an incomplete fact situation is Stable-PRO or Stable-CONTRA w.r.t. some consistent incomplete case base is CoNP-complete.*

Proof. Membership in CoNP follows from the fact that a negative instance can be verified in polynomial time, given a completion with a different justification status as a certificate (specifically: a completion that is not PRO as a counterexample for Stable-PRO and a completion that is not CONTRA as a counterexample for Stable-CONTRA).

Next, we prove CoNP-hardness for deciding if an incomplete fact situation is Stable-PRO; the proof for Stable-CONTRA is analogous. We reduce from the CoNP-complete problem TAUTOLOGY. Let (ϕ, X) be an instance of this problem, where ϕ is a formula in DNF on the variables in X.

First check that ϕ is satisfiable, i.e.: there is an assignment τ_X to the values of X such that $\phi[\tau_X]$ is True. Given that ϕ is in DNF, this can be done in polynomial time (by checking if there exists at least one clause that is *consistent*, in the sense that it does not contain both x_i and $\neg x_i$ for some $x_i \in X$). If ϕ is not satisfiable, then it is not a tautology. In the remainder of this proof, we consider the alternative case that ϕ is satisfiable.

Now let $(\mathcal{D}, C, c)$ be the result of a transformation $T(\phi, X)$ that is constructed as follows: for each $x_i \in X$, construct a dimension d_i with three values: $\{v_{\mathtt{p}}, v_{\mathtt{n}}, v_{\mathtt{u}}\}$ and ordering $\leq_{\texttt{PRO}} = \{(v_{\mathtt{u}}, v_{\mathtt{u}}), (v_{\mathtt{u}}, v_{\mathtt{p}}), (v_{\mathtt{u}}, v_{\mathtt{n}}), (v_{\mathtt{p}}, v_{\mathtt{p}}), (v_{\mathtt{n}}, v_{\mathtt{n}})\}$ and let $\mathcal{D}$ be the set of all dimensions. Construct an incomplete fact situation c such that for each $d_i \in \mathcal{D}$: $V_p(d_i, c) = \{v_{\mathtt{p}}, v_{\mathtt{n}}\}$. Finally, construct an incomplete case base $\mathcal{C}$ that has an incomplete case (c^*, PRO) for each *consistent* clause c' in ϕ (i.e. there is no $x_i \in X$ such that both $x_i \in c'$ and $\neg x_i \in c'$), where $V_p(d_i, c^*) = \{v_{\mathtt{p}}\}$ if $x_i \in c'$; $V_p(d_i, c^*) = \{v_{\mathtt{n}}\}$ if $\neg x_i \in c'$ and $V_p(d_i, c^*) = \{v_{\mathtt{u}}\}$ otherwise. $T(\phi, X)$ can be constructed in polynomial time. Next, we show that (ϕ, X) is a positive instance of TAUTOLOGY iff c is Stable-PRO w.r.t. $\mathcal{C}$:

- If (ϕ, X) is a positive instance of TAUTOLOGY, then for each assignment τ_X to the values of X, $\phi[\tau_X]$ is True. Now construct $(\mathcal{D}, C, c) = T(\phi, X)$. Let c'' be an arbitrary completion of c; then the corresponding assignment to the values x_i in X will be x_i = True if $v(d, c'') = v_{\mathtt{p}}$ and x_i = False if $v(d, c'') = v_{\mathtt{n}}$. We know that $\phi[\tau_X]$ is True and $\phi[\tau_X]$ is in DNF, so at least one of the clauses in $\phi[\tau_X]$ is True. Let (c^*, PRO) be the incomplete case in $\mathcal{C}$ that corresponds to this clause. Then for each $d_i \in \mathcal{D}$, either $V_p(d_i, c^*) = \{v_{\mathtt{u}}\}$ or $V_p(d_i, c^*) = \{v(d_i, c'')\}$, so $c'' \geq_{\texttt{PRO}} c^{*\prime}$, where $c^{*\prime}$ is the only completion of c^*. This implies that c'' is PRO given $\mathcal{C}$. As c'' was chosen arbitrarily from the completions of c, c must be Stable-PRO w.r.t. $\mathcal{D}$ and $\mathcal{C}$.
- Alternatively, suppose that (ϕ, X) is a negative instance of TAUTOLOGY, then for some assignment τ_X to the values of X, $\phi[\tau_X]$ is False. Construct $(\mathcal{D}, C, c) = T(\phi, X)$ and let c'' be the completion corresponding to $\phi[\tau_X]$ (so for each $d_i \in \mathcal{D}$: $v(d_i, c'') = v_{\mathtt{p}}$ iff x_i is True and $v(d_i, c'') = v_{\mathtt{n}}$ iff x_i is False). Then each clause in $\phi[\tau_X]$ is False, so for each incomplete case (c^*, PRO) in $\mathcal{C}$ where $c^{*\prime}$ is the only completion of c^*, it must be that $c'' \not\geq_{\texttt{PRO}} c^{*\prime}$. Therefore c'' is UNDECIDED given $\mathcal{C}$, hence c cannot be Stable-PRO w.r.t. $\mathcal{D}$ and $\mathcal{C}$.

Given that (ϕ, X) is a positive instance of TAUTOLOGY iff c is Stable-PRO w.r.t. $\mathcal{C}$ and TAUTOLOGY is CoNP-hard, it follows that deciding if an incomplete fact situation is Stable-PRO is CoNP-hard as well. Together with CoNP-membership, this implies that deciding if an incomplete fact situation is Stable-PRO w.r.t. an incomplete case base is CoNP-complete. This even holds if each incomplete case in the incomplete case base has a single completion. □

Proposition 4.13. *Deciding whether an incomplete fact situation is Unstable w.r.t. some consistent incomplete case base is NP-complete.*

Proof. Analogous to the proof of Proposition 4.12, membership in NP follows from the fact that a positive instance can be verified in polynomial time given two completions with different justification statuses as certificates.

For the NP-hardness proof, the transformation from Proposition 4.12 can be used to reduce from the NP-complete problem NON-TAUTOLOGY, the co-problem of TAUTOLOGY. Let (ϕ, X) be an instance of this problem, where ϕ is a formula in DNF on the variables in X. First check that ϕ is satisfiable, i.e.: there is an assignment τ_X to the values of X such that $\phi[\tau_X]$ is True. Given that ϕ is in DNF, this can be done in polynomial time (by checking if there exists at least one clause that is *consistent*, in the sense that it does not contain both x_i and $\neg x_i$ for some $x_i \in X$). If ϕ is not satisfiable, then it is a non-tautology. In the remainder of this proof, we consider the alternative case that ϕ is satisfiable.

Now let $(\mathcal{D}, C, c)$ be the result of a transformation $T(\phi, X)$ that is constructed as follows: for each $x_i \in X$, construct a dimension d_i with three values: $\{v_{\mathtt{p}}, v_{\mathtt{n}}, v_{\mathtt{u}}\}$ and ordering $\leq_{\mathtt{PRO}} = \{(v_{\mathtt{u}}, v_{\mathtt{u}}), (v_{\mathtt{u}}, v_{\mathtt{p}}), (v_{\mathtt{u}}, v_{\mathtt{n}}), (v_{\mathtt{p}}, v_{\mathtt{p}}), (v_{\mathtt{n}}, v_{\mathtt{n}})\}$ and let $\mathcal{D}$ be the set of all dimensions. Construct an incomplete case c such that for each $d_i \in \mathcal{D}$: $V_p(d_i, c) = \{v_{\mathtt{p}}, v_{\mathtt{n}}\}$. Finally, construct an incomplete case base $\mathcal{C}$ that has an incomplete case (c^*, PRO) for each *consistent* clause c' in ϕ (i.e. there is no $x_i \in X$ such that both $x_i \in c'$ and $\neg x_i \in c'$), where $V_p(d_i, c^*) = \{v_{\mathtt{p}}\}$ if $x_i \in c'$; $V_p(d_i, c^*) = \{v_{\mathtt{n}}\}$ if $\neg x_i \in c'$ and $V_p(d_i, c^*) = \{v_{\mathtt{u}}\}$ otherwise. $T(\phi, X)$ can be constructed in polynomial time. Next, we show that (ϕ, X) is a positive instance of NON-TAUTOLOGY iff (ϕ, X) is a negative instance of TAUTOLOGY iff c is Stable-PRO w.r.t. $\mathcal{C}$:

- If (ϕ, X) is a positive instance of NON-TAUTOLOGY then it is negative instance of TAUTOLOGY: for some assignment τ_X to the values of X, $\phi[\tau_X]$ is False. Construct $(\mathcal{D}, C, c) = T(\phi, X)$ and let c'' be the completion corresponding to $\phi[\tau_X]$ (so for each $d_i \in \mathcal{D}$: $v(d_i, c'') = v_{\mathtt{p}}$ iff x_i is True and $v(d_i, c'') = v_{\mathtt{n}}$ iff x_i is False). Then each clause in $\phi[\tau_X]$ is False, so for each incomplete case (c^*, PRO) in $\mathcal{C}$ it must be that $c'' \not\geq_{\mathtt{PRO}} c^{*\prime}$ where $c^{*\prime}$ is the only completion of c^*. Therefore c'' is UNDECIDED given $\mathcal{C}$. On the other hand, given that ϕ is satisfiable, there is at least one clause c' in ϕ that is consistent; therefore $\mathcal{C}$ contains at least one incomplete case (c^*, PRO) such that the only completion of c^* is $c^{*\prime}$ and there is at least one completion c''' corresponding to c' such that $c''' \geq_{\mathtt{PRO}} c^{*\prime}$; therefore c''' is PRO (and not UNDECIDED) given $\mathcal{C}$. To conclude, c is Unstable w.r.t. $\mathcal{D}$ and $\mathcal{C}$.

- Alternatively, suppose that (ϕ, X) is a negative instance of NON-TAUTOLOGY. Then (ϕ, X) is a positive instance of TAUTOLOGY, so for each assignment τ_X to the values of X, $\phi[\tau_X]$ is True. Now construct $(\mathcal{D}, C, c) = T(\phi, X)$. Let c'' be an arbitrary completion of c; then the corresponding assignment to the values x_i in X will be $x_i =$ True if $v(d, c'') = v_{\mathtt{p}}$ and $x_i =$ False if $v(d, c'') = v_{\mathtt{n}}$. We know that $\phi[\tau_X]$ is True and $\phi[\tau_X]$ is in DNF, so at least one of the clauses in $\phi[\tau_X]$ is True. Let (c^*, PRO) be the incomplete case in $\mathcal{C}$ that corresponds to this clause. Let $c^{*\prime}$ be the only completion of c^*. Then for each $d_i \in \mathcal{D}$, either $v(d_i, c^{*\prime}) = v_{\mathtt{u}}$ or $v(d_i, c^{*\prime}) = v(d_i, c'')$, so $c'' \geq_{\texttt{PRO}} c^{*\prime}$. This implies that c'' is PRO given $\mathcal{C}$. As c'' was chosen arbitrarily from the completions of c, c must be Stable-PRO w.r.t. $\mathcal{D}$ and $\mathcal{C}$. In other words, c is not Unstable w.r.t. $\mathcal{D}$ and $\mathcal{C}$.

Given that (ϕ, X) is a positive instance of TAUTOLOGY iff c is Stable-PRO w.r.t. $\mathcal{C}$ and TAUTOLOGY is CoNP-hard, it follows that deciding if an incomplete fact situation is Stable-PRO is CoNP-hard as well. Together with CoNP-membership, this implies that deciding if an incomplete fact situation is Stable-PRO w.r.t. an incomplete case base is CoNP-complete. This even holds if each incomplete case in the incomplete case base has a single completion. □

C.7. Computational aspects of relevance

Proposition 4.14. *Let $\mathcal{C}$ be an incomplete case base in domain $\mathcal{D}$ and let c be an incomplete fact situation. Let j be a justification status (PRO, CONTRA or UNDECIDED). Then for a given dimension $d \in \mathcal{D}$, the removal of possible value $v \in V_p(d, c)$ is j-relevant w.r.t. $\mathcal{C}$ and c iff there is some completion c' of c such that:*

- *c' is j w.r.t. $\mathcal{C}$; and*
- *c'' is **not** j w.r.t. $\mathcal{C}$ where c'' is equal to c' except for the value of d, which is v. Formally: $c'' = \{(d', v(d', c')) \mid d' \in \mathcal{D} \setminus \{d\}\} \cup \{(d, v)\}$.*

Proof. First we prove the proposition from left to right:

1. Suppose that the removal of v is j-relevant w.r.t. $\mathcal{C}$ and c.
2. Then there is a minimal stable-j partial completion c^* w.r.t. $\mathcal{C}$ and c such that $v \notin V_p(d, c^*)$.
3. Now construct the partial completion $c^{*\prime}$ from c^* by including v to the possible values of d: $c^{*\prime} = \{(d', V_p(d', c^*)) \mid d' \in \mathcal{D} \setminus \{d\}\} \cup \{(d, V_p(d, c^*) \cup \{v\})\}$. Given that c^* was a minimal stable-j partial completion, that $c^* \neq c^{*\prime}$ and that $c^* \in F(c^{*\prime})$, $c^{*\prime}$ cannot be stable-j w.r.t. $\mathcal{C}$.
4. So there is some completion c'' of $c^{*\prime}$ that is **not** j w.r.t. $\mathcal{C}$. Then it must be that $v(d, c'') = v$: otherwise, c'' would also be a completion of $c^{*\prime}$ (and have the justification status j).
5. Finally construct c', which equals c'' except for the value of d. Formally: $c' = \{(d', v(d', c'')) \mid d' \in \mathcal{D} \setminus \{d\}\} \cup \{(d, v')\}$, where v' is an arbitrary value

from $V_p(d, c^*)$. Then c' is a completion of c^*, so c' is j w.r.t. $\mathcal{C}$.

Then c' (in item 5) and c'' (in item 4) fulfill the left-to-right part of the proposition. Next, we prove the proposition from right to left:

1. Suppose that there is some completion c' of c such that c' is j w.r.t. $\mathcal{C}$ and $c'' = \{(d', v(d', c')) \mid d' \in \mathcal{D} \setminus \{d\}\} \cup \{(d, v)\}$ is **not** j w.r.t. $\mathcal{C}$.
2. Let $c^{*\prime}$ be an incomplete fact situation for which c' is the only completion as each dimension has one possible value: the actual value of that dimension in c'. Formally: $c^{*\prime} = \{(d, \{v(d, c')\}) \mid d \in \mathcal{D}\}$. Then $c^{*\prime}$ is a partial completion of c (as c' is a completion of c) and it is stable-j w.r.t. $\mathcal{C}$ and $\mathcal{D}$ (as there is a single completion which is j).
3. This implies that there must be some minimal stable-j partial completion c^* (possibly $c^{*\prime}$ itself) such that $c^{*\prime} \in F(c^*)$. Note that $v \notin V_p(d, c^*)$: if $v \in V_p(d, c^*)$ then c'' would be a completion of c^* and therefore be j w.r.t. $\mathcal{C}$, which would contradict our assumption. Then by Definition 4.18, removal of v is j-relevant w.r.t. $\mathcal{C}$ and c. □

Proposition 4.15. *For each justification status j (PRO, CONTRA or UNDECIDED), deciding whether the removal of a possible value for a given dimension of an incomplete fact situation is j-relevant w.r.t. some incomplete case base is NP-complete.*

Proof. Membership in NP follows from Proposition 4.14: in order to verify that the removal of a value for a dimension d of an incomplete fact situation is j-relevant (where j is PRO, CONTRA or UNDECIDED) w.r.t. some incomplete case base (i.e. a positive instance of the problem), a suitable certificate would be some completion c' such that c' is j w.r.t. the incomplete case base, while replacing d's value by v results in a completion c'' that is not j. This can be validated in polynomial time using Algorithm 8 for justification on both c' and c''.

In order to prove NP-hardness, we give a reduction from the NP-complete problem NON-TAUTOLOGY (the complementary problem of TAUTOLOGY, where an instance is a NON-TAUTOLOGY iff it is not a TAUTOLOGY). Let (ϕ, X) be an instance of this problem, where ϕ is a formula in DNF on the variables in X.

First check that ϕ is satisfiable, i.e.: there is an assignment τ_X to the values of X such that $\phi[\tau_X]$ is True. Given that ϕ is in DNF, this can be done in polynomial time (by checking if there exists at least one clause that is *consistent*, in the sense that it does not contain both x_i and $\neg x_i$ for some $x_i \in X$). If ϕ is not satisfiable, then it is a non-tautology. In the remainder of this proof, we consider the alternative case that ϕ is satisfiable.

Now let $(\mathcal{D}, \mathcal{C}, c, d)$ the result of the transformation $T(\phi, X)$ that is constructed as follows: for each $x_i \in X$, construct a dimension d_i with three values: $\{v_{\mathtt{p}}, v_{\mathtt{n}}, v_{\mathtt{u}}\}$ and ordering $\leq_{\mathtt{PRO}} = \{(v_{\mathtt{u}}, v_{\mathtt{u}}), (v_{\mathtt{u}}, v_{\mathtt{p}}), (v_{\mathtt{u}}, v_{\mathtt{n}}), (v_{\mathtt{p}}, v_{\mathtt{p}}), (v_{\mathtt{n}}, v_{\mathtt{n}})\}$ and let $\mathcal{D}'$ be the set of all of these dimensions. In addition, let d be a dimension with two values: $\{0, 1\}$

and ordering $\leq_{\texttt{PRO}} = \leq$ and let $\mathcal{D} = \mathcal{D}' \cup \{d\}$. Construct an incomplete fact situation c such that for each $d_i \in \mathcal{D}'$: $V_p(d_i, c) = \{v_{\texttt{p}}, v_{\texttt{n}}\}$ and $V_p(d, c) = \{0, 1\}$. Finally, construct an incomplete case base $\mathcal{C}$ that has an incomplete case (c_0^*, PRO) as well as an incomplete case (c_1^*, PRO) for each *consistent* clause c' in ϕ (i.e. there is no $x_i \in X$ such that both $x_i \in c'$ and $\neg x_i \in c'$), where:

- For each $d_i \in \mathcal{D}'$:
  - $V_p(d_i, c_0^*) = \{v_{\texttt{p}}\}$ if $x_i \in c'$; $V_p(d_i, c_0^*) = \{v_{\texttt{n}}\}$ if $\neg x_i \in c'$ and $V_p(d_i, c_0^*) = \{v_{\texttt{u}}\}$ otherwise; and
  - $V_p(d_i, c_1^*) = \{v_{\texttt{p}}\}$ if $x_i \in c'$; $V_p(d_i, c_1^*) = \{v_{\texttt{n}}\}$ if $\neg x_i \in c'$ and $V_p(d_i, c_1^*) = \{v_{\texttt{u}}\}$ otherwise; and
- $v(d, c_0^*) = \{0\}$ and $v(d, c_1^*) = \{1\}$.

In addition, add an incomplete case $(c^\star, \text{PRO})$ in the incomplete case base such that for each $d_i \in \mathcal{D}'$: $V_p(d_i, c^\star) = \{v_{\texttt{u}}\}$ and $V_p(d, c^\star) = \{1\}$. This transformation can be constructed in polynomial time. Then the following statements are equivalent:

1. (ϕ, X) is a positive instance of NON-TAUTOLOGY;
2. $(\mathcal{D}, \mathcal{C}, c, d, 0)$ is a positive instance of PRO-relevance; and
3. $(\mathcal{D}, \mathcal{C}, c, d, 1)$ is a positive instance of UNDECIDED-relevance.

We prove this in the following:

(1) → (2) and (3) If (ϕ, X) is a positive instance of NON-TAUTOLOGY (1) then there is some assignment τ_X to the values of X such that $\phi[\tau_X]$ is False. Construct $(\mathcal{D}, \mathcal{C}, c, d) = T(\phi, X)$ and the corresponding completions c_0 and c_1, defined as follows:

- let c_0 be the completion corresponding to $\phi[\tau_X]$ with $v(d, c_0) = 0$ (so for each $d_i \in \mathcal{D}'$: $v(d_i, c_0) = v_{\texttt{p}}$ iff x_i is True and $v(d_i, c_0) = v_{\texttt{n}}$ iff x_i is False); and
- let c_1 be the completion corresponding to $\phi[\tau_X]$ with $v(d, c_1) = 1$ (and having $v(d', c_1) = v(d', c_0)$ for all $d' \in \mathcal{D}'$).

Given that each clause in $\phi[\tau_X]$ is False, for each incomplete case (c^*, PRO) in $\mathcal{C}$ it must be that $c_0 \not\geq_{\texttt{PRO}} c^{*\prime}$, where $c^{*\prime}$ is the only completion of c^*. Therefore c_0 is UNDECIDED given $\mathcal{C}$. Now let $c^{\star\prime}$ be the only completion of $c^\star$. Given that $c_1 \geq_{\texttt{PRO}} c^{\star\prime}$, c_1 is PRO given $\mathcal{C}$. Then by Proposition 4.14, for the dimension d the removal of possible value 0 is PRO-relevant w.r.t. $\mathcal{C}$ and c (2). Similarly, by Proposition 4.14, for the dimension d the removal of possible value 1 is UNDECIDED-relevant w.r.t. $\mathcal{C}$ and c (3).

(2) → (1) Suppose that the removal of possible value 0 from the dimension d is PRO-relevant w.r.t. $\mathcal{C}$ and c (2). Then by Proposition 4.14, there is some completion c_1' of c such that $v(d, c_1') = 0$ and c_1' is **not** PRO w.r.t. $\mathcal{C}$. Given that $\mathcal{C}$ does not contain CONTRA incomplete cases, c_1' is UNDECIDED w.r.t. $\mathcal{C}$. Therefore no incomplete case can have matched: for each incomplete case

(c^*, PRO) in $\mathcal{C}$ with as its only completion $c^{*\prime}$ it must be that $c'_1 \not\geq_{\texttt{PRO}} c^{*\prime}$. Now construct an assignment τ_X that corresponds to c'_1, defined in the following way: for each $x_i \in X$, x_i is assigned True by τ_X iff $v(d_i, c'_1) = v_p$ and x_i is assigned False by τ_X iff $v(d_i, c'_1) = v_n$ (note that there is no dimension in c'_1 having v_u as value, as v_u is not in the possible values of c). Given this assignment τ_X, consider the formula $\phi[\tau_X]$: each of its clauses must be False, so $\phi[\tau_X]$ is False. In other words: (ϕ, X) is a positive instance of NON-TAUTOLOGY (1).

(3) → (1) Suppose that the removal of possible value 1 from the dimension d is UNDECIDED-relevant w.r.t. $\mathcal{C}$ and c (3). Then by Proposition 4.14, there is some completion c'_1 of c such that $v(d, c'_1) \neq 1$ (so $v(d, c'_1) = 0$) and c'_1 is UNDECIDED w.r.t. $\mathcal{C}$. Similar to the reasoning for our proof for (2) → (1), we can conclude that (ϕ, X) is a positive instance of NON-TAUTOLOGY (1).

This equivalence concludes the proof that the problems PRO- and UNDECIDED-relevance are NP-hard (and, in combination with the NP-membership result, NP-complete). For CONTRA-relevance, the proof is analogous, but then all incomplete cases in the incomplete case base should have the outcome CONTRA rather than PRO. Finally, note that this even holds if each incomplete case in the incomplete case base has a single completion. □

Proposition 4.16. *Within some domain $\mathcal{D}$, let $\mathcal{C}$ be an incomplete case base and let c be an incomplete focus case. For any dimension $d \in \mathcal{D}$, $(d, v, o) \in$ LIST-RELEVANT-UPDATES$(\mathcal{D}, \mathcal{C}, c)$ iff v is o-relevant to remove for dimension d w.r.t. c, $\mathcal{D}$ and $\mathcal{C}$.*

Proof. Let $\mathcal{C}$ be an incomplete case base within domain $\mathcal{D}$ and let c be an incomplete fact situation within this domain. Let $\mathcal{C}^*$ be the alternative case base obtained by applying Algorithm 12: $\mathcal{C}^* =$ GET-ALTERNATIVE-CASE-BASE$(\mathcal{D}, \mathcal{C}, c)$.

First, we show that for each completion c' of c, the justification status of c' given (the original incomplete case base) $\mathcal{C}$ is the same as the justification status of c' given (the alternative complete case base) $\mathcal{C}^*$, as well as the justification status of c' given the minimised complete case base $\mathcal{C}' = \texttt{min}(\mathcal{C}^*)$ (by Algorithm 13 Lines 3-6). Let c' be an arbitrary completion of c. We distinguish three cases.

1. First, suppose that c' is PRO w.r.t. $\mathcal{C}$. Then by Proposition 4.6 there is some $(c'', \text{PRO}) \in \mathcal{C}$ such that for each $d \in \mathcal{D}$ there is some $v' \in V_p(d, c'')$ such that $v(d, c') \geq_{\texttt{PRO}} v'$. Then by Definition 4.21 of minimal matching values V_m, for each $d \in \mathcal{D}$ there is some v'' in $V_m(d, (c'', \text{PRO}))$ such that $v(d, c') \geq_{\texttt{PRO}} v''$. Given that the set of minimal matching completions $C_m(c'', \text{PRO})$ contains all combinations of minimal matching values for each dimension, there must be some $c^* \in C_m(c'', \text{PRO})$ such that $c' \geq_{\texttt{PRO}} c^*$. Finally, the alternative (complete) case base $\mathcal{C}^*$ must contain (c^*, PRO). So by Definition 4.8 of justification w.r.t. complete case bases, c' is PRO w.r.t. $\mathcal{C}^*$.

Then there is at least one, possibly multiple, $(c^*, \text{PRO}) \in \mathcal{C}^*$ such that $c' \geq_{\texttt{PRO}} c^*$. Let $(c^{*\prime}, \text{PRO})$ be a complete case in $\mathcal{C}^*$ such that $c' \geq_{\texttt{PRO}} c^{*\prime}$ and there is no $(c'', \text{PRO}) \in \mathcal{C}^*$ such that $c'' <_{\texttt{PRO}} c^{*\prime}$. Then Line 6 of Algorithm 13 was reached for $(c^{*\prime}, \text{PRO})$ as Line 5 evaluated to True. Accordingly, $(c^{*\prime}, \text{PRO}) \in \mathcal{C}'$, so c' is PRO w.r.t. $\mathcal{C}'$.

2. If c' is CONTRA w.r.t. $\mathcal{C}$ then c' is CONTRA w.r.t. $\mathcal{C}^*$ and $\mathcal{C}'$. The proof is analogous to the proof of item 1.
3. Finally suppose that c' is UNDECIDED w.r.t. $\mathcal{C}$. Then by Proposition 4.6 for all $o \in \{\text{PRO}, \text{CONTRA}\}$ there is no $(c'', o) \in \mathcal{C}$ such that for each $d \in \mathcal{D}$ there is some $v' \in V_p(d, c'')$ such that $v(d, c') \geq_o v'$. Let (c'', o) be an arbitrary incomplete case in $\mathcal{C}$. Then there is some $d \in \mathcal{D}$ such that there is no $v' \in V_p(d, c'')$ such that $v(d, c') \geq_o v'$. As a consequence, $V_m(d, (c'', o))$ as defined in Definition 4.21 must be empty. Then also $C_m(c'', o) = \emptyset$. As (c'', o) was chosen arbitrarily, this holds for each incomplete case in $\mathcal{C}$. So the alternative case base $\mathcal{C}^*$ does not contain any case. As $\mathcal{C}' \subseteq \mathcal{C}^*$, $\mathcal{C}'$ must be empty as well. Accordingly, c' is UNDECIDED w.r.t. $\mathcal{C}^*$ and $\mathcal{C}'$.

As c' was chosen arbitrarily, it holds that for each completion of c the justification status w.r.t. $\mathcal{C}$ equals its justification status w.r.t. $\mathcal{C}^*$ as well as its justification status w.r.t. $\mathcal{C}'$. We will use this property later in the proof.

By Proposition 4.14, for any given dimension $d \in \mathcal{D}$ and $j \in \{\text{PRO}, \text{CONTRA}\}$, the removal of possible value $v \in V_p(d, c)$ is j-relevant w.r.t. $\mathcal{C}$ and c iff there is some completion c' of c such that c' is j w.r.t. $\mathcal{C}$ and $c'' = \{(d', v(d', c')) \mid d' \in \mathcal{D} \setminus \{d\}\} \cup \{(d, v)\}$ is not j w.r.t. $\mathcal{C}$. Lines 7-15 in Algorithm 13 find these completions efficiently. Next, we will prove that such completion exists iff there is some $(c^*, j) \in \mathcal{C}'$ such that $c^{*\prime} = \{(d', v(d', c^*)) \mid d' \in \mathcal{D} \setminus \{d\}\} \cup \{(d, v)\}$ is not j w.r.t. $\mathcal{C}'$:

- We first prove this from left to right. Suppose that there is some completion c' of c such that c' is j w.r.t. $\mathcal{C}$ and $c'' = \{(d', v(d', c')) \mid d' \in \mathcal{D} \setminus \{d\}\} \cup \{(d, v)\}$ is not j w.r.t. $\mathcal{C}$. Then, as we have shown before, given that c' is j w.r.t. $\mathcal{C}$, c' must also be j w.r.t. $\mathcal{C}'$. Hence there must be some $(c^*, j) \in \mathcal{C}'$ such that $c' \geq_j c^*$. Note that this implies that c^* itself is also j w.r.t. $\mathcal{C}'$. Now construct $c^{*\prime}$ by replacing the old value of d by v: $c^{*\prime} = \{(d', v(d', c^*)) \mid d' \in \mathcal{D} \setminus \{d\}\} \cup \{(d, v)\}$. Note the following:
  - For each $d' \in \mathcal{D} \setminus \{d\}$ we have $v(d', c'') = v(d', c')$ (by construction of c'') and $v(d', c^*) = v(d', c^{*\prime})$ (by construction of $c^{*\prime}$). Furthermore, given that $c' \geq_j c^*$, for each $d' \in \mathcal{D} \setminus \{d\}$ it must hold that $v(d', c') \geq_j v(d, c^*)$. Therefore, for each $d' \in \mathcal{D} \setminus \{d\}$ it must be that $v(d', c'') \geq_j v(d', c^{*\prime})$.
  - For d it holds that $v(d, c'') = v(d, c^{*\prime}) = v$, so $v(d, c'') \geq_j v(d, c^{*\prime})$.

  To conclude, it must be that $c'' \geq_j c^{*\prime}$. Given that c'' is not j w.r.t. $\mathcal{C}$, it cannot be that c'' is j w.r.t. $\mathcal{C}'$. Therefore, there is no $(c^\star, j) \in \mathcal{C}'$ such that $c'' \geq_j c^\star$. Then also there can be no $(c^\star, j) \in \mathcal{C}'$ such that $c^{*\prime} \geq_j c^\star$ (if there would be such a $c^\star$ then by transitivity of $\geq_j$ it would hold that $c'' \geq_j c^\star$, which would contradict c'' not being j w.r.t. $\mathcal{C}'$). Thus $c^{*\prime}$ is not j w.r.t. $\mathcal{C}'$.

- From right to left this follows immediately, as c^* must be a completion of c.

We have shown that any possible value v is j-relevant to remove for d w.r.t. $\mathcal{C}$ and c if and only if there is some $(c^*, j) \in \mathcal{C}'$ such that $c^{*\prime} = \{(d', v(d', c^*)) \mid d' \in \mathcal{D} \setminus \{d\}\} \cup \{(d, v)\}$ is not j w.r.t. $\mathcal{C}'$ This is checked in Algorithm 13 Line 13 and only if this is the case, (d, v, o) is added to R. To conclude, all j-relevant updates are returned in R. □

Proposition 4.17. *Within some domain $\mathcal{D}$, let $\mathcal{C}$ be an incomplete case base and let c be an incomplete focus case. If for each dimension $d = (V, \leq_{PRO})$ in $\mathcal{D}$, $\leq_{PRO}$ is a total order on V then* LIST-RELEVANT-UPDATES *runs in polynomial time.*

Proof. The only step that does not take polynomial time in general is the step of generating the alternative case base $\mathcal{C}^*$ of minimal matching completions (Algorithm 12, Lines 16–23): if the size of this case base is polynomial compared to the input problem, then the remaining steps can be done in polynomial time. Now suppose that, for each $d = (V, \leq_{\texttt{PRO}})$ in $\mathcal{D}$, $\leq_{\texttt{PRO}}$ is a total order on V. Then for any dimension $d \in \mathcal{D}$, incomplete case $(c', o) \in \mathcal{C}$ and incomplete focus case c, the set of values in $V_p(d, c)$ such that there is some $v' \in V_p(d, c')$ with $v \geq_o$ must have a single minimum value. As a result, there is at most one minimal matching value $\mathrm{Vm}[d, c', o]$ for each dimension for any incomplete case in $\mathcal{C}$. Therefore each incomplete case in $\mathcal{C}$ has a single minimal matching completion $\mathrm{Cm}[c', o]$. This implies that the size of $\mathcal{C}^*$ cannot exceed the size of $\mathcal{C}$. As a consequence, the algorithm runs in polynomial time. □

D

Summary

Artificial intelligence (AI) is developing fast, promising quality and efficiency benefits in various processes where (repetitive) tasks are outsourced from human analysts to machines. At the Dutch police, two examples of tasks in which AI can be helpful are the intake of complaints on online trade fraud and the classification of suspect web shops.

For decision-making tasks in law enforcement, it is essential that domain experts understand machine-made advice and have the possibility to correct possible mistakes. Therefore, AI systems used in this domain should be transparent. Furthermore, given that a wrong decision has serious consequences, the AI system should have the possibility to abstain from proposing a decision. As an additional challenge, often not all information that is required to make a decision is known in advance. We therefore need AI systems that are cautious in proposing a decision, can explain those decisions and are able to reason with incomplete information.

To optimise transparency, we focus on AI techniques that reason in a similar way as humans do when performing a given task. The two aforementioned use cases require two different types of reasoning. In order to decide whether or not a citizen is a victim of fraud, the analyst checks whether a legal rule (and any of its exceptions) applies. This type of reasoning is rule-based and can be modeled by computational argumentation. Conversely, when an analyst decides that a web shop is mala fide, they do this after a comparison with earlier (precedent) web shops for which they already made a decision. This is an example of precedent-based reasoning.

Existing formalisms for argumentation- and precedent-based AI fit our transparency and cautiousness requirements very well for situations in which all information is available in advance. Most existing research in these areas has been devoted to the "static" problems in these formalisms, such as: given the current information regarding arguments or precedents, which conclusions can we draw *at this moment*?

We refer to this problem as the task of determining the *justification* status, which can be seen as giving an initial advice regarding a specific topic, thereby only considering information that is currently available. However, these formalisms are not yet equipped for reasoning with incomplete information. In this thesis, we aim to solve this issue by extending argumentation- and precedent-based formalisms so that they are able to efficiently reason with incomplete information in real-life applications. Towards reaching this goal, we answer four research questions.

Research question 1. How can argumentation- and precedent-based formalisms be extended to enable reasoning with incomplete information?

- A How can incomplete information be represented?
- B In which situations does new information not update the justification status?
- C Which new information can still change the justification status?

As a first step towards enabling reasoning with incomplete information, research question 1A must be answered. In other words, argumentation- and precedent-based formalisms must be extended to *represent incomplete information*. For abstract approaches to argumentation, such an extension already exists in the notion of Incomplete Argumentation Frameworks, which explicitly include uncertain arguments and attacks. For structured approaches to argumentation, an extension to represent incomplete information only exists in a restricted fragment of ASPIC$^+$. In the field of precedent-based reasoning, no such extension exists. In this thesis we therefore extend (a more general fragment of) ASPIC$^+$, as well as the model of precedential constraint so that they explicitly represent qualitative uncertainty. Each of these extended formalisms encodes incomplete information and thus effectively represents all possible ways in which the uncertainty can be resolved.

In order to answer research questions 1B and 1C, the (theoretical) notions of stability and relevance are defined for all three extended formalisms. The goal of the notion of *stability* is to identify situations in which a decision can be made, even though there is still some uncertainty about the information. In an application, this means that the corresponding advice will not change, regardless of information updates. In situations where the topic is not stable, it is not clear which decision should be made as some decisive information is missing. It is therefore necessary to identify *relevant* information updates, in that investigation into the presence of this information possibly leads to a stable topic. A comparison between the notions of stability and relevance in abstract approaches to these notions in structured approaches demonstrates that the definitions of stability and relevance on a structured level are more suitable for rule-based reasoning with incomplete information.

For each of the studied formalisms, the extension for modeling incomplete information represents a space of multiple possible situations that grows exponentially with the amount of uncertain information. This raises concerns related to the computation of stability and relevance, as the investigation of an exponential number of

possible situations may require an algorithm that takes exponential time. To investigate whether these concerns are valid, the following research question is answered.

Research question 2. In which complexity classes are the problems of justification, stability and relevance in the context of argumentation- and precedent-based formalisms situated?

This research question is answered by complexity analyses for abstract and structured approaches to argumentation, as well as for the result model of precedential constraint. We prove that many of these problems are NP-complete, CoNP-complete or even harder. These answers are interesting from a theoretical perspective, as they give insight into the complexity of reasoning with incomplete information. From a practical perspective, their high complexity implies (under the common assumption that $P \neq NP$) that it is impossible that any exact algorithm can solve these problems in polynomial time. Thus the question arises to which extent the stability status and relevant updates can still be computed within reasonable time.

Research question 3. How can the problems of justification, stability and relevance in the context of argumentation- and precedent-based formalisms be solved efficiently?

This question is answered for the three different representations. For abstract approaches to argumentation we propose algorithms based on answer set programming (ASP). The algorithms for relevance are shown to be sufficiently fast for instances with hundreds of (certain or uncertain) arguments under grounded semantics and tens of (certain or uncertain) arguments under complete semantics. In line with the lower complexity of the stability problem, the algorithms for stability are faster.

For structured approaches to argumentation, both a polynomial-time heuristic algorithm and a worst-case exponential exact ASP-based algorithm for stability are discussed. Even though the former is a heuristic algorithm and therefore not exact in general, it is proven to be exact under specific conditions on the input. Both algorithms were sufficiently fast to be used in practice. Surprisingly, the ASP-based algorithm is even faster than the polynomial heuristic algorithm. As a solution to the relevance problem, an exact ASP-based algorithm is proposed. The empirical evaluation of their runtimes show promising scalability.

For precedent-based reasoning, exact algorithms for the problems of stability and relevance are proposed. These algorithms run in polynomial time under conditions of the ordering of the dimension values. In particular, if the dimension values are in a total ordering, all algorithms run in polynomial time. For other orderings, the algorithms can be exponential. This is in line with the complexity results.

The extension of formalisms, complexity results and algorithms form the theoretical contribution of this thesis. The answers to these research questions enable the representation of uncertain information in both argument- and precedent-based reasoning and provide sufficiently fast algorithms for the (generally complex) problems

of stability and relevance. The only remaining research question asks how to apply these algorithms in practical decision support systems at the Dutch police.

Research question 4. How can the algorithms for stability and relevance be applied in practical decision support systems at the police?

As an answer to this research question, we report on our experience for the use cases of fraud intake and web shop classification. The system for fraud intake was launched in September 2019 and has been used by hundreds of thousands of citizens, while the system for web shop classification is used by analysts at the police. This shows that the algorithms for stability and relevance are useful in real-life applications, in combination with other techniques for natural language processing and web scraping. In addition, the argumentation-based algorithms have been implemented in the open source package PyArg, which is freely available to the research community.

E

Samenvatting in het Nederlands

Jaarlijks krijgt de politie tienduizenden aangiftes van online handelsfraude. Om deze aangiftes te verwerken, vindt er onder andere een intake met de burger plaats waarbij er bepaald wordt of het zinvol is om aangifte te doen. Daarnaast bepaalt de politie welke verdachte webwinkels malafide zijn. Hoewel het grotendeels om routinewerk gaat, is het soms noodzakelijk om eerst handmatig onderzoek te doen voordat een besluit kan worden genomen. Noch mensen, noch automatische procedures kunnen deze taken snel uitvoeren. Daarom is het interessant om te onderzoeken of ze elkaar kunnen aanvullen. Kan kunstmatige intelligentie (KI) mensen ondersteunen in dit soort besluitvorming?

Bij dit soort taken in het politiedomein is het belangrijk dat domeinexperts in staat zijn om het advies van de machine te begrijpen en, indien nodig, fouten te corrigeren. Daarom moeten KI-systemen bij de politie transparant zijn. Daarnaast kan een foute beslissing grote gevolgen hebben, dus een KI-systeem moet ook de mogelijkheid hebben om van een beslissing af te zien. Een bijkomende uitdaging is dat in veel gevallen niet alle benodigde informatie bij voorbaat beschikbaar is. KI-systemen moeten hiermee om kunnen gaan. Kortom, KI-systemen die ingezet worden voor het ondersteunen van besluitvorming bij de politie moeten terughoudend zijn in het nemen van beslissingen, hun beslissingen kunnen uitleggen en in staat zijn om te redeneren met onvolledige informatie.

Ten behoeve van de uitlegbaarheid richten we ons op KI-technieken die op een vergelijkbare manier redeneren als mensen wanneer ze een bepaalde taak uitvoeren. In de twee eerdergenoemde taken is er sprake van twee verschillende soorten redeneren. Om te besluiten of een burger slachtoffer is van fraude, gaat de analist na of een wettelijke regel (of uitzonderingen daarop) van toepassing is (of zijn). Dit type redeneren is regelgebaseerd en kan gemodelleerd worden met computationële

argumentatie. Wanneer een analist besluit dat een webwinkel malafide is, gebeurt dit echter op basis van een vergelijking met eerdere webwinkels waarover eerder al besloten is. Dit is een voorbeeld van redeneren met precedenten.

Bestaande formalismen voor argumentatie- en precedentgebaseerde KI voldoen aan onze eisen op het gebied van transparant redeneren en terughoudendheid in besluitvorming, maar zijn niet in staat om te redeneren met onvolledige informatie. Om dit probleem op te lossen, laat ik in dit proefschift zien hoe deze formalismen zodanig uitgebreid kunnen worden dat ze efficiënt kunnen redeneren met onvolledige informatie in praktijktoepassingen. Daarbij zal ik vier onderzoeksvragen beantwoorden.

Onderzoeksvraag 1. Hoe kunnen formalismen voor argumentatie en precedentgebaseerd redeneren zodanig worden uitgebreid dat ze kunnen redeneren met onvolledige informatie?

- A Hoe kan onvolledige informatie worden gerepresenteerd?
- B Onder welke voorwaarden verandert nieuwe informatie de rechtvaardigingsstatus niet?
- C Welke informatie kan de rechtvaardigingsstatus nog veranderen?

Om redeneren met onvolledige informatie mogelijk te maken, dient eerst onderzoeksvraag 1A te worden beantwoord. In andere woorden, formalismen voor redeneren op basis van argumentatie en precedenten moeten worden uitgebreid zodat ze onvolledige informatie kunnen modelleren. Voor abstracte benaderingen tot argumentatie bestaat een dergelijke uitbreiding al. Het gaat om zogenaamde incomplete argumentatieraamwerken, waarin expliciet een onderscheid wordt gemaakt tussen zekere en onzekere argumenten en aanvallen. Voor gestructureerde benaderingen tot argumentatie bestaat dit type uitbreiding alleen voor een beperkt fragment van ASPIC$^+$. Binnen het onderzoeksgebied van redeneren met precedenten bestaat de uitbreiding nog niet. Daarom breiden we in dit proefschrift zowel (een algemener fragment van) ASPIC$^+$ als een model voor redeneren met precedenten zodanig uit, dat het mogelijk wordt om kwalitatieve onzekerheid met betrekking tot informatie uit te drukken. In feite worden in deze uitgebreide formalismen dus alle mogelijke manieren uitgedrukt waarin deze onzekerheid kan worden ingevuld.

Om onderzoeksvragen 1B en 1C te beantwoorden, worden de (theoretische) begrippen stabiliteit en relevantie gedefinieerd voor alledrie de uitgebreide formalismen. Het idee van *stabiliteit* is dat het aangeeft in welke situatie een beslissing genomen kan worden, zelfs als nog niet alle informatie beschikbaar is. Dit betekent dat het advies niet meer zal veranderen, ook niet als de beschikbare informatie aangepast wordt. In situaties waarin het onderwerp niet stabiel is, is het nog niet duidelijk welke beslissing er genomen moet worden, omdat er nog doorslaggevende informatie ontbreekt. Het is dan noodzakelijk om vast te stellen welke informatiewijzigingen nog *relevant* zijn, in de zin dat onderzoek naar de aanwezigheid van deze informatie het onderwerp mogelijk nog stabiel kan maken. Een vergelijking tussen de begrippen van stabiliteit en relevantie in abstracte benaderingen ten opzichte van

gestructureerde benaderingen tot argumentatie laat zien dat de definities van stabiliteit en relevantie op een gestructureerd niveau geschikter zijn voor regelgebaseerd redeneren met onvolledige informatie dan de abstracte benadering.

Voor elk van de drie bestudeerde formalismen is de genoemde uitbreiding een manier om een veelheid van mogelijke toekomstscenario's uit te drukken die exponentieel groeit met de hoeveelheid aan onvolledige informatie. Deze exponentiële groei is verontrustend voor het berekenen van stabiliteit en relevantie, omdat het doet vermoeden dat het onderzoeken van alle toekomstscenario's een algoritme vereist dat een exponentiële hoeveelheid tijd nodig heeft. Om te onderzoeken of dit vermoeden klopt, dient de volgende onderzoeksvraag te worden beantwoord.

Onderzoeksvraag 2. In welke complexiteitsklassen bevinden zich de problemen van rechtvaardiging, stabiliteit en relevantie op het gebied van argumentatie- en precedentgebaseerd redeneren?

Deze onderzoeksvraag wordt beantwoord door middel van complexiteitsanalyses. We bewijzen dat veel van deze problemen NP-compleet, CoNP-compleet of zelfs nog moeilijker zijn. Deze resultaten zijn interessant vanuit een theoretisch oogpunt, omdat ze inzicht geven in de complexiteit van het redeneren met onvolledige informatie. Vanuit een praktisch oogpunt duidt hun hoge complexiteit er (onder de algemeen aanvaarde aanname dat $P \neq NP$) op dat er geen exact algoritme kan bestaan dat deze problemen in polynomiale tijd oplost. Zo doet de vraag zich voor of stabiliteit en relevante wijzigingen nog wel binnen redelijke tijd berekend kunnen worden.

Onderzoeksvraag 3. Hoe kunnen de problemen van rechtvaardiging, stabiliteit en relevantie efficiënt worden opgelost in de context van argumentatie- en precedentgebaseerd redeneren?

Deze vraag wordt beantwoord voor elk van de drie verschillende representaties. Voor abstracte benaderingen tot argumentatie ontwikkelen we algoritmen die gebaseerd zijn op *answer set programming* (ASP). We laten zien dat de algoritmen voor relevantie snel genoeg zijn voor instanties met tientallen tot honderden (zekere of onzekere) argumenten. De algoritmen voor stabiliteit zijn nog sneller. Dit is overeenkomstig met de lagere complexiteit van het stabiliteitsprobleem.

Voor gestructureerde benaderingen tot argumentatie bespreken we zowel een benaderingsalgoritme voor stabiliteit dat in polynomiale tijd loopt als een exact algoritme op basis van ASP dat, in het slechtste geval, exponentiële tijd nodig heeft. Hoewel het eerste algoritme een benadering is en daarom niet in het algemeen exact is, bewijzen we dat het exact is onder specifieke voorwaarden met betrekking tot de invoer. Beide algoritmen zijn snel genoeg om in de praktijk gebruikt te worden. Verrassend genoeg is het op ASP gebaseerde algoritme zelfs sneller dan het polynomiale benaderingsalgoritme. Om het relevantieprobleem op te lossen, hebben we exacte algoritmen gebaseerd op ASP ontwikkeld. De empirische evaluatie van de rekentijd laat een veelbelovende schaalbaarheid zien.

Voor redeneren op basis van precedenten hebben we exacte algoritmen ontwikkeld om stabiliteit en relevantie te bepalen. Deze algoritmen hebben polynomiale tijd nodig onder de voorwaarde dat dimensiewaarden totaal geordend zijn. Voor andere ordeningen hebben de algoritmen, in het slechtste geval, een exponentiële hoeveelheid tijd nodig. Dit resultaat is in lijn met de complexiteitsresultaten.

De uitbreiding van formalismen, complexiteitsresultaten en algoritmen vormen de theoretische bijdrage van dit proefschrift. De antwoorden op deze onderzoeksvragen maken het mogelijk om onzekere informatie te representeren in zowel argument- als precedentgebaseerd redeneren en leveren algoritmen die snel genoeg zijn om de (complexe) problemen van stabiliteit en relevantie op te lossen. De enige overgebleven onderzoeksvraag is hoe deze algoritmen kunnen worden toegepast in de praktijk voor beslissingsondersteunende systemen bij de Nederlandse politie.

Onderzoeksvraag 4. Hoe kunnen de algoritmen voor stabiliteit en relevantie in de praktijk worden toegepast in beslissingsondersteunende systemen bij de politie?

Om deze onderzoeksvraag te beantwoorden doen we verslag van onze ervaringen met de toepassingen op het gebied van de intake van aangiftes van fraude en webwinkelclassificatie. Het intakesysteem is sinds september 2019 toegankelijk via de politiewebsite en is sindsdien gebruikt door honderdduizenden burgers. Het systeem voor webwinkelclassificatie wordt gebruikt door analisten bij de politie. Hieruit blijkt dat de algoritmen voor stabiliteit en relevantie bruikbaar zijn in de praktijk. Daarnaast zijn de algoritmen voor argumentatie geïmplementeerd in het open-source softwarepakket PyArg, dat gratis beschikbaar is voor de onderzoeksgemeenschap.

F

Publications

Journal papers

Odekerken, Daphne, Floris Bex, AnneMarie Borg, and Bas Testerink (2022a). "Approximating stability for applied argument-based inquiry". In: *Intelligent Systems with Applications* 16, p. 200110. DOI: 10.1016/j.iswa.2022.200110

Odekerken, Daphne, AnneMarie Borg, and Floris Bex (2023e). "Justification, stability and relevance in incomplete argumentation frameworks". In: *Argument & Computation* 15.3, pp. 1–58. DOI: 10.3233/AAC-230002

Odekerken, Daphne, Floris Bex, and Henry Prakken (2024). "Precedent-based reasoning with incomplete information for human-in-the-loop decision support". Accepted for Artificial Intelligence and Law

Araszkiewicz, Michał, Ilaria Angela Amantea, Saurabh Chakravarty, Robert van Doesburg, Maria Dymitruk, Marie Garin, Leilani Gilpin, Daphne Odekerken, and Seyedeh Sajedeh Salehi (2020). "ICAIL doctoral consortium, Montreal 2019". In: *Artificial Intelligence and Law* 28, pp. 267–280. DOI: 10.1007/s10506-020-09267-z

Conference proceedings

Odekerken, Daphne, AnneMarie Borg, and Floris Bex (2020). "Estimating stability for efficient argument-based inquiry". In: *Computational Models of Argument. Proceedings of COMMA 2020*. Ed. by Henry Prakken, Stefano Bistarelli, Francesco Santini, and Carlo Taticchi. IOS Press, pp. 307–318. DOI: 10.3233/FAIA200514

Odekerken, Daphne and Floris Bex (2020). "Towards transparent human-in-the-loop classification of fraudulent web shops". In: *Legal Knowledge and Information Systems. Proceedings of JURIX 2020*. Ed. by Serena Villata, Jakub Harašta, and Petr Křemen. IOS Press, pp. 239–242. DOI: 10.3233/FAIA200873

Odekerken, Daphne, AnneMarie Borg, and Floris Bex (2022b). "Stability and relevance in incomplete argumentation frameworks". In: *Computational Models of Argument. Proceedings of COMMA 2022*. Ed. by Francesca Toni, Sylwia Polberg, Richard Booth, Martin Caminada, and Hiroyuki Kido. IOS Press, pp. 272–283. DOI: 10.3233/FAIA220159

Odekerken, Daphne, Floris Bex, and Henry Prakken (2023a). "Justification, stability and relevance for case-based reasoning with incomplete focus cases". In: *Proceedings of the 19th International Conference on Artificial Intelligence and Law*. Ed. by Matthias Grabmair, Francisco Andrade, and Paulo Novais, pp. 177–186. DOI: 10.1145/3594536.3595136

Odekerken, Daphne, Tuomo Lehtonen, AnneMarie Borg, Johannes P. Wallner, and Matti Järvisalo (2023f). "Argumentative reasoning in ASPIC+ under incomplete information". In: *Proceedings of the 20th International Conference on Principles of Knowledge Representation and Reasoning*. Ed. by Pierre Marquis, Tran Cao Son, and Gabriele Kern-Isberner, pp. 531–541. DOI: 10.24963/kr.2023/52

Odekerken, Daphne, Floris Bex, and Henry Prakken (2023b). "Precedent-based reasoning with incomplete cases". In: *Legal Knowledge and Information Systems. Proceedings of JURIX 2023*. Ed. by Giovanni Sileno, Jerry Spanakis, and Gijs van Dijck. IOS Press, pp. 33–42. DOI: 10.3233/FAIA230943

Odekerken, Daphne (2024). "Finding relevant updates in incomplete argumentation frameworks". In: *Computational Models of Argument. Proceedings of COMMA 2024*. Ed. by Chris Reed, Matthias Thimm, and Tjitze Rienstra. IOS Press, pp. 181–192. DOI: 10.3233/FAIA240320

Lehtonen, Tuomo, Daphne Odekerken, Johannes P. Wallner, and Matti Järvisalo (2024). "Complexity results and algorithms for preferential argumentative reasoning in ASPIC+". In: *Proceedings of the 21th International Conference on Principles of Knowledge Representation and Reasoning*. Ed. by Pierre Marquis, Magdalena Ortiz, and Maurice Pagnucco, pp. 520–530. DOI: 10.24963/kr.2024/49

Demo papers and handbook chapters

Borg, AnneMarie and Daphne Odekerken (2022). "PyArg for solving and explaining argumentation in Python: Demonstration". In: *Computational Models of Argument. Proceedings of COMMA 2022*. Ed. by Francesca Toni, Sylwia Polberg, Richard Booth, Martin Caminada, and Hiroyuki Kido. IOS Press, pp. 349–350. DOI: 10.3233/FAIA220167

Odekerken, Daphne (2022). "Justification, stability and relevance for transparent and efficient human-in-the-loop decision support". In: *Online Handbook of Argumentation for AI* 3, pp. 28–32. DOI: 10.48550/ARXIV.2212.07996

Berthold, Matti, Matthias Knorr, and Daphne Odekerken (2023a). "ForgettingWeb". In: *Technical Communications of the 39th International Conference on Logic Programming*, pp. 321–323. DOI: 10.48550/arXiv.2308.14898

Odekerken, Daphne, AnneMarie Borg, and Matti Berthold (2023c). "Accessible algorithms for applied argumentation". In: *First International Workshop on Argumentation and Applications*, pp. 92–98. URL: https://www.ceur-ws.org/Vol-3472/short3.pdf

Odekerken, Daphne, AnneMarie Borg, and Matti Berthold (2023d). "Demonstrating PyArg 2.0". In: *Seventh Workshop on Advances in Argumentation in Artificial Intelligence*. URL: https://ceur-ws.org/Vol-3546/paper14.pdf

Odekerken, Daphne (2023). "Argumentative reasoning with incomplete information in law enforcement". In: *Online Handbook of Argumentation for AI* 4. DOI: 10.48550/arXiv.2401.09444

Xia, Yilin, Daphne Odekerken, Shawn Bowers, and Bertram Ludäscher (2024). "Layered visualization of argumentation frameworks". In: *Computational Models of Argument. Proceedings of COMMA 2024*. Ed. by Chris Reed, Matthias Thimm, and Tjitze Rienstra. IOS Press, pp. 373–374. DOI: 10.3233/FAIA240346

Currently under review

Odekerken, Daphne, Tuomo Lehtonen, Johannes P. Wallner, and Matti Järvisalo (n.d.). "Argumentative reasoning in ASPIC+ under incomplete information". Submitted to Journal of Artificial Intelligence Research

G

Curriculum vitae

1993	Born in Stein, The Netherlands

Education

2014–2018	Masters in Computing Science (cum laude) Utrecht University
2011–2014	Bachelors in Computer Science (cum laude) Utrecht University
2005–2011	Gymnasium (cum laude) Trevianum Scholengroep

Work

2018–present	AI scientist / PhD student Dutch police
2013–2018	Teaching assistant Utrecht University

Volunteering work

2015–2019	Secretary Limburgs Fanfare Orkest
2016–2017	Chairperson Utrechtsch Studenten Concert
2014–2015	Orchestra manager Utrechtsch Studenten Concert

Acknowledgements

While pursuing a PhD is typically a solitary journey, my experience at the National Police Lab AI has been quite the opposite. In the past six years, I have enjoyed the company of and collaboration with so many nice colleagues and friends.

First, I would like to thank my PhD supervisors. Floris, thank you for grounding the Police Lab and for immediately involving me in research, even before my PhD officially started. Thanks for randomly dropping into the lab and starting discussions on some new AI technique or police task. And thanks for teaching me how to introduce and motivate my research. AnneMarie, when we first met, I hardly had any experience in writing proofs. Thank you for your patience in reading and correcting them over and over again. Compared to the difficulties I had in the first year, proving is now a piece of cake. Speaking of which, thanks for all those home-made cakes as well! Henry, thank you for thoroughly proofreading my papers and thesis. Your critical and detailed feedback have significantly improved them. Besides, it was a pleasure to see you in the audience of my concerts.

Bas, I always enjoy discussing new ideas with you at the whiteboard. You are so creative in combining techniques towards responsible AI solutions. I feel like having spent six years in developing the ideas that you shared with me in 2018. Now I am very much looking forward to collaborating on future AI projects.

I am very grateful to the Dutch police and Utrecht University for initiating their collaboration. For me this was a perfect match, providing a PhD project that enabled me to do theoretical research with practical applications in an inspiring environment. With so many colleagues of different backgrounds, there was always a suitable person to discuss work-related matters. And, even more importantly, you made me feel welcome and appreciated. There was always someone who had time for a coffee break, to discuss travel plans or for a walk in the botanic gardens. Speaking of foliage, whenever I returned from a conference or holiday, my mint plant somehow

always seemed to be better cared for than before I left. A special thanks to Elize and Michiel.

Speaking of conferences, I would not have stayed motivated for doing research if I would not have met so many nice people in the argumentation and AI&Law communities. It was a pleasure to meet you, to organise a workshop and a tutorial and to do research together. We will meet again!

There is a time for work and there is a time to relax. The best time for relaxing is for sure on Tuesday and Wednesday evening. Cu4+ and Atletische Eters: thank you for the music, for the great food and for keeping me sane during the pandemic.

Papa, mama en Luuk: och, noe bèn ich toch nog acadeem gewoore. Mer ger mot toch auch toegaeve dat ger dat zelf auch ein bietje sjawd zeet. Thoes in Stein waor altied een plaats om te ontpanne, (staum aaf) te blaoze en lekker te aete. En noe geer mien ongerzeuk in vlekkeloos plat kènt reproducere, zeet geer zelf auch academisch bezig.