Computational Models of Argument
C. Reed et al. (Eds.)

doi:10.3233/FAIA240320

Finding Relevant Updates in Incomplete Argumentation Frameworks

Daphne ODEKERKEN [a,b,1]
[a] *Department of Information and Computing Sciences, Utrecht University*
[b] *National Police Lab AI, Netherlands Police*
ORCiD ID: Daphne Odekerken https://orcid.org/0000-0003-0285-0706

Abstract. Incomplete argumentation frameworks (IAFs) are abstract argumentation frameworks that encode qualitative uncertainty by distinguishing between certain and uncertain arguments and attacks. In a completion of an IAF, each uncertain argument or attack is either added (made certain) or removed. Given a completion, the acceptability of an argument is determined by its justification status. For arguments in an IAF that do not have the same justification status in each completion, it is interesting to study which uncertain arguments and attacks are relevant, in the sense that adding or removing them can lead to a different justification status. We propose algorithms based on Answer Set Programming for enumerating relevant arguments and attacks under grounded and complete semantics.

Keywords. relevance, incomplete argumentation frameworks, answer set programming

1. Introduction

Computational argumentation is an important research field in artificial intelligence, concerning reasoning with incomplete or inconsistent information [1]. A central concept are argumentation frameworks (AFs): a set of arguments and an attack relation between them [2]. Collectively acceptable sets of arguments within an AF, defined by semantics, are called extensions. Based on these extensions, each argument has at least one justification status, stating whether the argument should be accepted, rejected or otherwise.

In practice, however, argumentation is a dynamic process in which not all arguments may be known in advance. For instance, in an inquiry process where initially the presence of some arguments is uncertain, agents collaborate in collecting information, aiming to establish the justification status of some topic argument with certainty [3]. A second example is in negotiation procedures where the participating agents exchange arguments to support their preferred offer [4]. Third, participants in a persuasion setting may add arguments to influence the dialectical strength of other arguments (cf. [5]).

Incomplete argumentation frameworks (IAFs) are an extension to AFs that model this qualitative uncertainty by distinguishing between certain and uncertain arguments and attacks [6,7,4]. For the uncertain elements, it is unknown whether they are part of the argumentation framework. They may be added or removed in the future. An IAF can

[1] Corresponding Author: Daphne Odekerken, d.odekerken@uu.nl

be completed by deciding for all uncertain arguments and attacks whether or not they are present. Any completion of an IAF is an AF, for which the justification status can be computed. If the justification status of an argument is the same for all completions, this argument is stable. Otherwise, it is interesting to determine which uncertainties should be resolved to reach a point where the argument is stable. In other words: which uncertain arguments or attacks are *relevant* to add or remove to make some topic argument stable?

The identification of relevant updates is useful in inquiry as it enables efficient collection of information. In negotiation and persuasion settings, agents may choose updates that are relevant for accepting arguments that support their preferred offer or opinion.

Although the problem of relevance for IAFs has been studied from a complexity-theoretic perspective in [8], to the best of our knowledge no algorithms for relevance in IAFs have been proposed. In this paper, we therefore propose the first algorithms for 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 other formalisms, employing SAT- or ASP-solvers [9]. In this line of research, we propose ASP-based algorithms for enumerating relevant updates. Our empirical evaluation shows that the algorithms are promising, even on inputs with many (uncertain) arguments and attacks. All algorithms are available 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. Relevance in incomplete argumentation frameworks

In this section, we recall the notion of relevance for IAFs. IAFs are an extension to AFs, initially proposed as partial AFs in [6] and later studied as IAFs in e.g. [7,4]. In an IAF, the set of arguments and attacks is split into a certain part ($\mathcal{A}$ and $\mathcal{C}$) and an uncertain part ($\mathcal{A}^?$ and $\mathcal{C}^?$), where the uncertain elements may be added or removed in the future.

Definition 1 (IAF). *An IAF is a tuple* $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$ *s.t.* $\mathcal{A} \cap \mathcal{A}^? = \emptyset$, $\mathcal{C} \cap \mathcal{C}^? = \emptyset$ *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.*

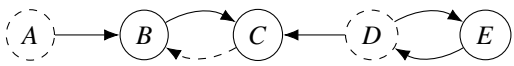

Figure 1. An example of an IAF. Certain arguments are depicted as circles with solid borders, whereas uncertain arguments are circles with dashed borders. Attacks are depicted as arrows, which have a solid line if they represent certain attacks and a dashed line if they represent uncertain attacks.

Example 1. *Figure 1 shows an example of an 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* $\mathcal{C}^? = \{(C,B)\}$.

An IAF can be *completed* by deciding for all uncertain arguments and attacks whether or not they are present, as defined below. Each completion of an IAF is an AF. An AF $\langle \mathcal{A}, \mathcal{C} \rangle$ consists of a finite set $\mathcal{A}$ of arguments and a binary attack relation $\mathcal{C} \subseteq \mathcal{A} \times \mathcal{A}$, where $(A, B) \in \mathcal{C}$ indicates that argument A attacks argument B.

Definition 2 (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 the restriction $\mathcal{C}'|_{\mathcal{A}'}$ of a set of (uncertain) attacks $\mathcal{C}' \subseteq \mathcal{C} \cup \mathcal{C}^?$ to a set of (uncertain) arguments $\mathcal{A}'$ is defined as $\mathcal{C}'|_{\mathcal{A}'} = \{(X, Y) \in \mathcal{C}' \mid X \in \mathcal{A}' \text{ and } Y \in \mathcal{A}'\}$.*

The evaluation of arguments is done using the semantics of [2].

Definition 3 (Extension-based semantics)**.** *Let $AF = \langle \mathcal{A}, \mathcal{C} \rangle$ be an AF and $S \subseteq \mathcal{A}$. Then: S is conflict-free iff for each $X, Y \in S : (X, Y) \notin \mathcal{C}$; $X \in \mathcal{A}$ is acceptable with respect to S 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 $X \in S$ implies that X is acceptable with respect to S; S is a **complete extension** (CP) iff S is admissible and for each X: if $X \in \mathcal{A}$ is acceptable with respect to S then $X \in S$; and S is the **grounded extension** (GR) iff it is the set inclusion minimal complete extension.*

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) 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) [10].

Definition 4 (Argument justification status)**.** *Let $AF = \langle \mathcal{A}, \mathcal{C} \rangle$ be an argumentation framework and σ some semantics in $\{\text{GR}, \text{CP}\}$. Let A be some argument in $\mathcal{A}$.*

- *A is σ-sceptical-*IN *(resp. σ-credulous-*IN*) iff A belongs to each (resp. some) σ-extension of AF;*
- *A is σ-sceptical-*OUT *(resp. σ-credulous-*OUT*) iff for each (resp. some) σ-extension S of AF, A is attacked by some argument in S;*
- *A is σ-sceptical-*UNDEC *(resp. σ-credulous-*UNDEC*) iff for each (resp. some) σ-extension of AF, A is not in S and not attacked by any argument in S.*

The justification statuses that we consider in this paper are $\{\text{GR}, \text{CP}\} \times \{\text{sceptical}, \text{credulous}\} \times \{\text{IN}, \text{OUT}, \text{UNDEC}\}$. If the justification status of an argument in the IAF is the same for all completions, we say that this argument is stable [8].

Definition 5 (Stability)**.** *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. *In the running example, E is stable-*CP*-credulous-*IN*. E is not stable-*GR*-credulous-*IN*, as there are completions (containing D) where E is not in the* GR *extension.*

In order to define which uncertainties are relevant to be resolved for obtaining some stability status, we need a notion of partial completions. 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 [8]. Partial completions without uncertain elements correspond to completions.

Definition 6 (Partial completion)**.** *Given an IAF $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$, a partial completion for $\mathcal{I}$ is an IAF $\mathcal{I}' = \langle \mathcal{A}', \mathcal{A}^{?\prime}, \mathcal{C}', \mathcal{C}^{?\prime} \rangle$, where $\mathcal{A} \subseteq \mathcal{A}' \subseteq \mathcal{A} \cup \mathcal{A}^?$, $\mathcal{C}|_{(\mathcal{A}' \cup \mathcal{A}^{?\prime})} \subseteq \mathcal{C}' \subseteq (\mathcal{C} \cup \mathcal{C}^?)|_{(\mathcal{A}' \cup \mathcal{A}^{?\prime})}$, $\mathcal{A}^{?\prime} \subseteq \mathcal{A}^?$ and $\mathcal{C}^{?\prime} \subseteq \mathcal{C}^?$. Since $\mathcal{I}'$ is an IAF, we have $\mathcal{A}' \cap \mathcal{A}^{?\prime} = \emptyset$; $\mathcal{C}' \cap \mathcal{C}^{?\prime} = \emptyset$; $\mathcal{C}' \subseteq (\mathcal{A}' \cup \mathcal{A}^{?\prime}) \times (\mathcal{A}' \cup \mathcal{A}^{?\prime})$ and $\mathcal{C}^{?\prime} \subseteq (\mathcal{A}' \cup \mathcal{A}^{?\prime}) \times (\mathcal{A}' \cup \mathcal{A}^{?\prime})$. We denote all partial completions for $\mathcal{I}$ by $\mathtt{part}(\mathcal{I})$.*

Example 3. *Returning to the running example, the following IAFs are some (but not all) examples of partial completions in $\mathtt{part}(\mathcal{I})$:*

- *$\mathcal{I}_1 = \langle \mathcal{A} \cup \mathcal{A}^?, \emptyset, \mathcal{C} \cup \mathcal{C}^?, \emptyset \rangle$: all uncertain arguments and attacks become certain.*
- *$\mathcal{I}_2 = \langle \mathcal{A}, \emptyset, \{(B,C)\}, \emptyset \rangle$: only certain arguments and attacks in $\mathcal{C}|_{\mathcal{A}}$ are left.*
- *$\mathcal{I}_3 = \langle \mathcal{A} \cup \{A\}, \{D\}, \mathcal{C}, \mathcal{C}^? \rangle$: the argument A is moved from the uncertain to the certain part. The argument D and the attack (C,B) are still uncertain.*

Before proceeding to a formal definition of relevance, we define the notion of minimal stable partial completions. Intuitively, the minimal stable-j partial completion for A is a partial completion in which A is stable-j, while in any partial completion with more uncertain elements, A is not stable-j [8].

Definition 7 (Minimal stable-j partial completion)**.** *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, a minimal stable-j partial completion for A w.r.t. $\mathcal{I}$ is an $\mathcal{I}' \in \mathtt{part}(\mathcal{I})$ such that A is stable-j in $\mathcal{I}'$ and there is no $\mathcal{I}''$ in $\mathtt{part}(\mathcal{I})$ such that A is stable-j in $\mathcal{I}''$, $\mathcal{I}'' \neq \mathcal{I}'$ and $\mathcal{I}' \in \mathtt{part}(\mathcal{I}'')$.*

We can now define j-relevance in terms of minimal stable-j partial completions. In 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 [8].

Definition 8 (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 $U \notin \mathcal{A}' \cup \mathcal{A}^{?\prime} \cup \mathcal{C}' \cup \mathcal{C}^{?\prime}$.*

Example 4. *Suppose that we need 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 A was uncertain in $\mathcal{I}$ and is certain 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 uncertain in $\mathcal{I}$ and is no longer present in $\mathcal{I}_4$, the removal of D is* GR*-sceptical-*IN*-relevant for C w.r.t. $\mathcal{I}$.*

For any justification status j, the j-RELEVANCE problem is the problem of determining if the addition and/or removal of some uncertain argument or attack is j-relevant for some argument w.r.t. an IAF. The complexity of this problem has been studied in [8]. For grounded semantics, all variants of the RELEVANCE problem are NP-complete. For complete semantics, CP-sceptical-IN-, CP-sceptical-OUT- and CP-credulous-UNDEC-

RELEVANCE are NP-complete (as these coincide with their GR counterparts), while CP-credulous-IN-, CP-credulous-OUT- and CP-sceptical-UNDEC-RELEVANCE are Σ_2^p-complete.

3. Representing AFs and IAFs as an ASP program

In this section, we show how IAFs, AFs and their semantics can be encoded into ASP programs. First, we give a brief introduction to ASP. A normal ASP program π is a set of rules of the form $\mathbf{b_0} \leftarrow \mathbf{b_1}, \ldots, \mathbf{b_k}, \mathbf{not\ b_{k+1}}, \ldots, \mathbf{not\ b_m}.$, 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 $\mathbf{a} \leftarrow \mathbf{b_1}, \ldots, \mathbf{b_k}, \mathbf{not\ b_{k+1}}, \ldots, \mathbf{not\ b_m}, \mathbf{not\ a}.$ 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 $\mathbf{r} = \mathbf{b_0} \leftarrow \mathbf{b_1}, \ldots, \mathbf{b_k}.$ iff 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 $\pi^I = \{\mathbf{b_0} \leftarrow \mathbf{b_1}, \ldots, \mathbf{b_k}. \mid \mathbf{b_0} \leftarrow \mathbf{b_1}, \ldots, \mathbf{b_k}, \mathbf{not\ b_{k+1}}, \ldots, \mathbf{not\ b_m}. \in \pi, \{\mathbf{b_{k+1}}, \ldots, \mathbf{b_m}\} \cap \mathbf{I} = \emptyset\}$ is called the reduct. 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.

3.1. 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 in [11]. 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 [12].

$$\pi_{\langle \mathcal{A},\mathcal{C} \rangle} = \{\mathbf{arg}(\mathbf{a}) \mid a \in \mathcal{A}\} \cup \{\mathbf{attack}(\mathbf{a},\mathbf{b}) \mid (a,b) \in \mathcal{C}\}.$$

$$\begin{aligned}
\pi_{lab} = {} & \{\mathbf{lab}(\mathbf{out},\mathbf{Y}) \leftarrow \mathbf{lab}(\mathbf{in},\mathbf{X}),\ \mathbf{attack}(\mathbf{X},\mathbf{Y}).\} \\
& \cup \{\mathbf{lab}(\mathbf{undec},\mathbf{X}) \leftarrow \mathbf{arg}(\mathbf{X}),\ \mathbf{not\ lab}(\mathbf{in},\mathbf{X}),\ \mathbf{not\ lab}(\mathbf{out},\mathbf{X}).\} \\
\pi_{cp} = {} & \{\mathbf{lab}(\mathbf{in},\mathbf{X}) \leftarrow \mathbf{not\ not_in}(\mathbf{X}),\ \mathbf{arg}(\mathbf{X}).\} \\
& \cup \{\mathbf{not_in}(\mathbf{X}) \leftarrow \mathbf{not\ lab}(\mathbf{in},\mathbf{X}),\ \mathbf{arg}(\mathbf{X}).\} \\
& \cup \{\leftarrow \mathbf{lab}(\mathbf{in},\mathbf{X}),\ \mathbf{lab}(\mathbf{in},\mathbf{Y}),\ \mathbf{attack}(\mathbf{X},\mathbf{Y}).\} \\
& \cup \{\mathbf{undefended}(\mathbf{Y}) \leftarrow \mathbf{attack}(\mathbf{X},\mathbf{Y}),\ \mathbf{not\ lab}(\mathbf{out},\mathbf{X}).\} \\
& \cup\ \{\leftarrow \mathbf{lab}(\mathbf{in},\mathbf{X}),\ \mathbf{undefended}(\mathbf{X}).\} \\
& \cup \{\leftarrow \mathbf{not_in}(\mathbf{X}),\ \mathbf{not\ undefended}(\mathbf{X}).\} \\
\pi_{gr} = {} & \{\mathbf{lab}(\mathbf{in},\mathbf{X}) \leftarrow \mathbf{arg}(\mathbf{X}),\ \mathbf{lab}(\mathbf{out},\mathbf{Y})\ :\ \mathbf{attack}(\mathbf{Y},\mathbf{X}).\}
\end{aligned}$$

Example 5. *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)**.

3.2. 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.

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

Example 6. *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* $\{\leftarrow \mathbf{not}\ \mathbf{lab}(\mathbf{in}, \mathbf{b}).\}$ *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* $\{\leftarrow \mathbf{lab}(\mathbf{in}, \mathbf{b}).\}$, *we obtain 13 answer sets, corresponding to* CP *extensions of completions where B is not* IN.

Example 6 indicates a correspondence between answer sets and the possible justification status in the IAF's completions. Proposition 1 states this correspondence formally.

Proposition 1. *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* $\sigma \in \{\text{CP}, \text{GR}\}$ *and a label* $l \in \{\text{IN}, \text{OUT}, \text{UNDEC}\}$.

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

4. Encodings for relevance under grounded semantics

In this section, we propose ASP encodings for relevance under grounded semantics. Given that all variants of the relevance problem for grounded semantics are NP-complete [8], they can be solved in a single call of an ASP-solver. The crux of the complexity proof is the following property: 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.

Lemma 1 ([8], Lemma 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 $\langle \mathcal{A}', \mathcal{C}'|_{\mathcal{A}'} \rangle$, while A is j in $\langle \mathcal{A}' \cup \{U\}, \mathcal{C}'|_{\mathcal{A}' \cup \{U\}} \rangle$.*
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 $\langle \mathcal{A}', \mathcal{C}'|_{\mathcal{A}'} \rangle$, while A is j in $\langle \mathcal{A}', (\mathcal{C}' \cup \{U\})|_{\mathcal{A}'} \rangle$.*
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 $\langle \mathcal{A}', \mathcal{C}'|_{\mathcal{A}'} \rangle$, while A is **not** j in $\langle \mathcal{A}' \cup \{U\}, \mathcal{C}'|_{\mathcal{A}' \cup \{U\}} \rangle$.*
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 $\langle \mathcal{A}', \mathcal{C}'|_{\mathcal{A}'} \rangle$, while A is **not** j in $\langle \mathcal{A}', (\mathcal{C}' \cup \{U\})|_{\mathcal{A}'} \rangle$.*

The program π_{arg_rel} in Listing 1 uses the property from Lemma 1 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 1 based on the label of the topic and the **topic**'s label in the original completion.

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) ← attack(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).
```

Example 7. *In Example 4, 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* **add_query_relevant_for(in,a,c)**.

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. *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* $l \in \{\text{IN}, \text{OUT}, \text{UNDEC}\}$. *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* $(a,b) \in \mathcal{C}^?$:

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

From Proposition 2 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 (inquiry, negotiation or persuasion) 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.

Proposition 3. *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* $l \in \{\text{IN}, \text{OUT}, \text{UNDEC}\}$. *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* $(a,b) \in \mathcal{C}^?$:

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).
```

- *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 containing* **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 containing* **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 containing* **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 containing* **remove_att_relevant_for(l,a,b,t)**.

All relevant operations for t w.r.t. $\mathcal{I}$ 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 [13].

5. CEGAR-based approach for relevance in other semantics

While deciding relevance for GR is NP-complete [8] and can therefore be solved by a single ASP call, this does not hold in general for other semantics. In particular, for CP semantics, deciding CP-credulous-IN-, CP-credulous-OUT- and CP-sceptical-UNDEC-relevance is Σ_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 [13]. Specifically, we propose ASP-based counterexample-guided abstraction refinement (CEGAR) algorithms [14]. 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 retrieving all queryables that are credulous-CP-IN-relevant to add and/or remove is given in Algorithm 1. This algorithm uses subprocedures COMPLETE (Lines 13 and 19) to specify one completion and REFINE (Line 23) to exclude a completion in further search. These are defined as follows.

Definition 9. *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 $\mathcal{C}^* \subseteq \mathcal{C} \cup \mathcal{C}^?$.* $\textsc{Parts}(\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle, \langle \mathcal{A}^*, \mathcal{C}^* \rangle) = \{\mathbf{arg(a_i)} \mid a_i \in \mathcal{A}^? \cap \mathcal{A}^*\} \cup \{\mathbf{not\ arg(a'_i)} \mid a'_i \in \mathcal{A}^? \setminus \mathcal{A}^*\} \cup \{\mathbf{att(c_i)} \mid c_i \in \mathcal{C}^? \cap \mathcal{C}^*\} \cup \{\mathbf{not\ att(c'_i)} \mid c'_i \in \mathcal{C}^? \setminus \mathcal{C}^*\}$. $\textsc{Complete}(\mathcal{I}, AF) = \{\boldsymbol{r}. \mid r \in \textsc{Parts}(\mathcal{I}, AF)\}$ *and* $\textsc{Refine}(\mathcal{I}, AF) = \{\leftarrow\ \mathbf{r_1}, \ldots, \mathbf{r_n}. \mid \{r_1, \ldots, r_n\} = \textsc{Parts}(\mathcal{I}, AF)\}$.

Algorithm 1 Listing queryables that are credulous-relevant for t

1: **procedure** GET-CP-CREDULOUS-RELEVANT($\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle, l, t$)
2: $R^+ = \emptyset, R^- = \emptyset$
3: $\pi_c = \pi_{\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle} \cup \pi_{guess} \cup \pi_{val\_comp} \cup \pi_{lab} \cup \pi_{cp} \cup \{\leftarrow \textbf{not lab}(\mathbf{l}, \mathbf{t}).\}$
4: $\pi_v = \pi_{\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle} \cup \pi_{val\_comp} \cup \pi_{lab} \cup \pi_{cp} \cup \{\leftarrow \textbf{not lab}(\mathbf{l}, \mathbf{t}).\}$
5: **while** True **do**
6: $(sat, model) = \text{SOLVE}(\pi_c)$
7: **if** *sat* **then**
8: $\mathcal{A}^* = \{a \in \mathcal{A} \cup \mathcal{A}^? \mid \textbf{arg}(\mathbf{a}) \in model\}$
9: $\mathcal{C}^* = \{(a,b) \in \mathcal{C} \cup \mathcal{C}^? \mid \textbf{att}(\mathbf{a}, \mathbf{b}) \in model\}$
10: **for** $q \in \mathcal{A}^? \cup \mathcal{C}^?$ **do**
11: **if** $q \in \mathcal{A}^* \cup \mathcal{C}^*$ **then**
12: **if** $q \in \mathcal{A}^*$ **then** $F^- = \langle \mathcal{A}^* \setminus \{q\}, \mathcal{C}^* \rangle$ **else** $F^- = \langle \mathcal{A}^*, \mathcal{C}^* \setminus \{q\} \rangle$
13: $\pi_{completion} = \pi_v \cup \text{COMPLETE}(F^-)$
14: $(sat', model') = \text{SOLVE}(\pi_{completion})$
15: **if not** sat' **then**
16: $R^+ = R^+ \cup \{q\}$
17: **else**
18: **if** $q \in \mathcal{A}^*$ **then** $F^+ = \langle \mathcal{A}^* \cup \{q\}, \mathcal{C}^* \rangle$ **else** $F^+ = \langle \mathcal{A}^*, \mathcal{C}^* \cup \{q\} \rangle$
19: $\pi_{completion} = \pi_v \cup \text{COMPLETE}(F^+)$
20: $(sat', model') = \text{SOLVE}(\pi_{completion})$
21: **if not** sat' **then**
22: $R^- = R^- \cup \{q\}$
23: $\pi_c = \pi_c \cup \text{REFINE}(\langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle, \langle \mathcal{A}^*, \mathcal{C}^* \rangle)$
24: **else**
25: **return** (R^+, R^-)

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 w.r.t. AF (Proposition 1). 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, REFINEMENT 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 4. *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 $q \in R^-$.*

6. Reducing the input IAF in a preprocessing step

Each of the relevance 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 relevant updates, as these semantics satisfy directionality [15].

Definition 10. *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 $\mathcal{I}' = \langle \mathcal{A}', \mathcal{A}^{?\prime}, \mathcal{C}', \mathcal{C}^{?\prime} \rangle = \text{REDUCE}(\mathcal{I}, t) = \langle \mathcal{A} \cap R, \mathcal{A}^? \cap R, \mathcal{C}|_{\mathcal{A} \cap R}, \mathcal{C}^?|_{\mathcal{A} \cap R} \rangle$ is the IAF reduced to $R = \{a \in \mathcal{C} \cup \mathcal{C}^? \mid \langle (t, a_i), (a_i, a_{i+1}), \ldots, (a_{j-1}, a_j), (a_j, a) \rangle$ in $\mathcal{C} \cup \mathcal{C}^?\}$.*

Proposition 5. *For any IAF $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{C}, \mathcal{C}^? \rangle$, justification status j, uncertain argument or rule $q \in \mathcal{A}^? \cup \mathcal{C}^?$ and argument $t \in \mathcal{A}$, addition (resp. removal) for q is j-relevant for t w.r.t. $\mathcal{I}$ iff addition (resp. removal) for q is j-relevant for t w.r.t.* $\text{REDUCE}(\mathcal{I}, t)$.

7. Empirical evaluation

Finally, we empirically evaluate the runtime of the proposed algorithms on synthetically generated benchmarks, using clingo 5.4.1 for ASP solving [13] on a Intel(R) Core(TM) i7-7820HQ CPU 2.90 GHz 16 GB RAM machine. For generating the benchmarks for the grounded relevance algorithm, 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 which we measured the runtime of the algorithm for enumerating all grounded-relevant operations, in two variations: without preprocessing step (no prep.) and with preprocessing step (with prep.) – see the left part of Table 1. 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 1 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 10) and the relevance procedure is applied on the reduced IAF. The reduction in runtime is present in particular for large IAFs with high p.

To evaluate the runtime of Algorithm 1 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 algorithm with preprocessing step for the task of enumerating all CP-credulous-IN-relevant updates. As shown in the right-hand part of Table 1, the 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.

$\lvert\mathcal{A} \cup \mathcal{A}?\rvert$	$\lvert\mathcal{C} \cup \mathcal{C}^?\rvert$	**No prep.**	**With prep.**
50	75	100/100 (0.52)	100/100 (0.16)
100	150	96/100 (1.24)	98/100 (0.69)
150	225	88/100 (3.17)	96/100 (2.13)
200	300	79/100 (3.63)	84/100 (1.51)
250	375	80/100 (4.55)	86/100 (1.93)

$\lvert\mathcal{A} \cup \mathcal{A}?\rvert$	$\lvert\mathcal{C} \cup \mathcal{C}^?\rvert$	**With prep.**
5	7	100/100 (0.02)
10	15	100/100 (0.06)
15	22	100/100 (0.21)
20	30	96/100 (1.31)
25	37	89/100 (1.11)

Table 1. Number of finished instances and average runtime in seconds for the algorithms for grounded relevance (on the left, without and with preprocessing step) and complete-credulous-IN-relevance (on the right).

8. Conclusion

The contribution of this paper consists of providing the first algorithms for 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. In future work, this approach could be extended to other semantics. Furthermore, it would be interesting to develop strategies for pruning even more candidates from the solution space. Finally, we would like to investigate if our algorithms can be adapted for relevance in structured notions of argumentation, where completions are only considered if they have an instantiation in the structured argumentation framework.

References

[1] Atkinson K, Baroni P, Giacomin M, Hunter A, Prakken H, Reed C, et al. Towards artificial argumentation. AI magazine. 2017;38(3):25-36.
[2] Dung PM. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. Artificial Intelligence. 1995;77:321-57.
[3] Odekerken D, Bex F, Borg A, Testerink B. Approximating stability for applied argument-based inquiry. Intelligent Systems with Applications. 2022;16:200110.
[4] Mailly JG, Rossit J. Stability in abstract argumentation. In: NMR 2020 Workshop Notes; 2020. p. 93-9.
[5] Prakken H. Formalising an aspect of argument strength: degrees of attackability. In: Computational Models of Argument: Proceedings of COMMA 2022; 2022. p. 296-307.
[6] Cayrol C, Devred C, Lagasquie-Schiex MC. Handling ignorance in argumentation: semantics of partial argumentation frameworks. In: European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty. Springer; 2007. p. 259-70.
[7] Baumeister D, Järvisalo M, Neugebauer D, Niskanen A, Rothe J. Acceptance in incomplete argumentation frameworks. Artificial Intelligence. 2021;295:103470.
[8] Odekerken D, Borg A, Bex F. Justification, stability and relevance in incomplete argumentation frameworks. Argument & Computation. 2023;Pre-press(Pre-press):1-58.
[9] Cerutti F, Gaggl SA, Thimm M, Wallner J. Foundations of implementations for formal argumentation. IfCoLog Journal of Logics and their Applications. 2017;4(8):2623-705.
[10] Caminada M. On the issue of reinstatement in argumentation. In: European Workshop on Logics in Artificial Intelligence. Springer; 2006. p. 111-23.
[11] Egly U, Gaggl SA, Woltran S. Answer-set programming encodings for argumentation frameworks. Argument & Computation. 2010;1(2):147-77.
[12] Dvořák W, Rapberger A, Wallner JP, Woltran S. ASPARTIX-V19-an answer-set programming based system for abstract argumentation. In: International Symposium on Foundations of Information and Knowledge Systems. Springer; 2020. p. 79-89.
[13] Gebser M, Kaminski R, Kaufmann B, Schaub T. Multi-shot ASP solving with clingo. Theory and Practice of Logic Programming. 2019;19(1):27-82.
[14] Clarke EM, Grumberg O, Jha S, Lu Y, Veith H. Counterexample-guided abstraction refinement for symbolic model checking. Journal of the ACM. 2003;50(5):752-94.
[15] Baroni P, Giacomin M. On principle-based evaluation of extension-based argumentation semantics. Artificial Intelligence. 2007;171(10-15):675-700.