Qualitative uncertainty and dynamics of argumentation through dynamic logic

Antonio Yuste-Ginel, Andreas Herzig

Qualitative Uncertainty and Dynamics of Argumentation through Dynamic Logic*

Antonio Yuste-Ginel[1] and Andreas Herzig[1]

[1]IRIT, CNRS, Toulouse, France, herzig@irit.fr,
antonioyusteginel@gmail.com

Abstract

Dynamics and uncertainty are essential features of real-life argumentation, and many recent studies have focused on integrating both aspects into Dung's well-known abstract Argumentation Frameworks (AFs). This paper proposes a combination of the two lines of research through a well-behaved logical tool: Dynamic Logic of Propositional Assignments (DL-PA). Our results show that the main reasoning tasks of virtually all existing formalisms qualitatively representing uncertainty about AFs are encodable in DL-PA. Moreover, the same tool is also useful for capturing dynamic structures, such as control argumentation frameworks, as well as for developing more refined forms of argumentative communication under uncertainty.

1 Introduction

Formal argumentation has been proved to be a successful approach to non-monotonic reasoning, among many other applications [17, 4, 55]. Within the studies directed to provide a formal model for argument-based inference, abstract models of argumentation play a crucial role, as they answer a rather fundamental question: how should a rational agent choose among a conflicting set of arguments those that are better justified? The adjective *abstract* stresses that these models disregard the nature and structure of arguments, in order to focus on the different semantics through which one could give a precise answer to the question above. The foremost abstract model of argumentation is the use of directed graphs, first proposed by Dung in [39] under the name of *argumentation frameworks* (AFs), where nodes stand for arguments and arrows stand for attacks among arguments.

*This is a preliminary version of the paper "Antonio Yuste-Ginel and Andreas Herzig, Qualitative uncertainty and dynamics of argumentation through dynamic logic, *Journal of Logic and Computation*, 2023; exac098, https://doi.org/10.1093/logcom/exac098". Please, consult the original publication and use the full reference for citation purposes.

While being an elegant and powerful tool, AFs have too limited modelling capabilities for many purposes. Consequently, many extensions of Dung's model were proposed in the literature, most prominently support relations [25], recursive forms of attacks [8], and preferences between arguments [3]. Two essential limitations of all these approaches are: (i) their static character; and (ii) the assumption that the formalized agent has perfect knowledge about the structure of the AF, that is, about the relevant arguments and attacks of the debate.

Regarding (i), an AF can be understood as a snapshot of a debate, and this has been shown useful to provide mathematically precise counterparts of many interesting argumentative notions. However, a fundamental aspect of argumentation is its dynamic character, since arguments, conflicts among them, and participants' opinions typically change during the development of an argumentative dialogue. It is then unsurprising that the dynamics of formal argumentation systems has been the center of attention of a recent research avenue within formal argumentation, with an important focus on abstract models; we refer to [38] and [11] for recent surveys.

As to (ii), it turns out to be a significant shortcoming in adversarial contexts where one typically wants to model the information (i.e., the part of an AF) that an agent thinks her opponent entertains, and thus uncertainty arises naturally. This assumption of perfect knowledge has been relaxed through the study of extensions of AFs that account for different forms of uncertainty, be it probabilistic or more qualitative; see [51] and [52] for recent surveys on the respective approaches. Among the second group of approaches, *incomplete argumentation frameworks* (IAFs) [13, 40, 14, 16, 15] and *control argumentation frameworks* (CAFs) [31, 58, 32] have recently received a lot of attention, resulting in a precise complexity map of the different associated reasoning tasks as well as some applications [32].

Concurrently, a considerable amount of work in formal argumentation has focused on building a suitable logical theory for reasoning about argumentation formalisms, with a special focus on AFs and their dynamics; see [18] for a recent survey. The *dynamic logic of propositional assignments* (DL-PA) [6] has been shown to be a useful tool for this enterprise [35, 37, 36, 48]. DL-PA is a well-behaved variant of *propositional dynamic logic* (PDL) [46], where atomic programs are restricted to assignments of propositional variables to either Truth or Falsity. It is expressive enough to capture all standard argumentation semantics. When compared to encodings in propositional logic, DL-PA can capture semantics that incorporate minimality or maximality criteria more succinctly. Moreover, its advantages over equally succinct languages such as *quantified Boolean formulas* have been highlighted [36].

This work pushes further the logical encoding of abstract AFs in DL-PA by pursuing three general aims: (1) to capture argumentation semantics that had not been captured before, some of them posing challenging encodings methods; (2) to integrate qualitative uncertainty about AFs and dynamics of argumentation in DL-PA by reducing reasoning tasks of different extensions of argumentation frameworks to DL-PA model checking problems; and (3) to show that the chosen logic is also a suitable tool for exploratory purposes, by developing

new forms of modelling argumentative communication under uncertainty that are directly inspired by our encodings.

After providing the essential background on AFs and DL-PA (Section 2), Section 3 provides polynomial encodings of a wide range of AF semantics in DL-PA. In Section 4 we present several formalisms for qualitatively representing uncertainty about AFs, as well as the reduction of their main reasoning tasks to DL-PA model checking problems. In Section 5, we discuss joint approaches to dynamics and qualitative uncertainty of AFs. Section 6 ends the paper with some discussion and challenges for future work. Proofs and proof sketches can be found in the Appendix.

2 Background

Throughout the paper we assume a fixed, finite, non-empty set of arguments $\mathcal{U}$ (the *universe*). We moreover assume that $\mathcal{U}$ is big enough to accommodate our examples. Sets of arguments (noted A, sometimes with a superscript) are supposed to be subsets of $\mathcal{U}$; and all conflict relations (noted R, sometimes with a superscript) are binary relations on $\mathcal{U}$, i.e., $R \subseteq \mathcal{U} \times \mathcal{U}$. Given $A \subseteq \mathcal{U}$ and $R \subseteq \mathcal{U} \times \mathcal{U}$, we use $R \restriction_A$ to abbreviate $R \cap (A \times A)$ (the **restriction of** R **to** A).

2.1 Abstract Argumentation Frameworks (AFs) and their Semantics

An **argumentation framework** (AF) is a directed graph (A, R) [39], where A stands for a set of arguments and R stands for a conflict-based relation among them (typically an attack relation).[1] We note $\mathcal{AF}$ the set of all argumentation frameworks (over $\mathcal{U}$). Argumentation semantics are meant to capture the informal notion of reasonable positions in a debate. The literature contains a large number of such semantics. They are typically presented either in extension-based terms or in labelling-based terms. For most of the existing semantics, both approaches (extensions and labellings) were proved equivalent. Here, we opt for an extension-based presentation and restrict our attention to a limited number of semantics, but the interested reader is referred to [7] for an overview.

Let us first define some useful concepts. Let (A, R) be an AF and let $E \subseteq A$. We define $E^+ = \{x \in A \mid \exists y \in E : (y, x) \in R\}$ (the **set of arguments attacked by** E), and $E^\oplus = E \cup E^+$ (the so-called **range of** E). A set of arguments $E \subseteq A$ is **conflict-free** iff $E \cap E^+ = \emptyset$. Moreover, E **defends** $a \in A$ iff for every $x \in A$: if $(x, a) \in R$, then $x \in E^+$. Finally, $E \subseteq A$ is **admissible** iff it is (i) conflict-free and (ii) self-defended (it defends all its members).

[1] As $A \subseteq \mathcal{U}$, we actually focus on *finite* AFs, as most of the literature does. This is an inherent limitation of our approach: our encodings use quantification over $\mathcal{U}$, which makes finiteness of $\mathcal{U}$ necessary. Capturing some more general argumentation semantics has turned out to require powerful logical languages, such as the modal μ-calculus for the grounded semantics [43].

In [39], Dung introduced four different semantics. A set of arguments $E \subseteq A$ is said to be:

- a **stable extension** iff (i) it is conflict-free, and (ii)$A \backslash E \subseteq E^+$ ('E attacks every argument outside itself');
- a **complete extension** iff (i) it is conflict-free; and (ii) it contains precisely the arguments of A that it defends;
- a **grounded extension** iff it is a minimal (w.r.t. set inclusion) complete extension;
- a **preferred extension** iff it is a maximal (w.r.t. set inclusion) complete extension.

It is well known that the existence of complete, grounded and preferred extensions is guaranteed for any AF. However, this does not hold for stable semantics: there exist frameworks lacking stable extensions. Moreover, the grounded semantics is the only one from the above list belonging to the so-called single-status approach: each AF has exactly one grounded extension. This is an advantage when, for instance, modelling the beliefs of an agent as the output of her argument-evaluation processes. More precisely, if a semantics admits AFs with several extensions then these extensions are usually logically incompatible when one works with structured arguments, and there is no clear way to choose among them.

Besides Dung's above four semantics we will take into account some others.

Semi-stable semantics was born to solve the problems caused by the absence of stable extensions under certain conditions. A set $E \subseteq A$ is a **semi-stable extension** of (A, R) iff E is a complete extension with maximal (w.r.t. set inclusion) range among complete extensions. More formally, E is a semi-stable extension iff

(i) E is a complete extension; and

(ii) there is no other complete extension E' such that $E^{\oplus} \subset E'^{\oplus}$.

Contrarily to what happens with stable extensions, there is at least one semi-stable extension in every finite AF. Moreover, when the set of stable extensions is nonempty, stable and semi-stable extensions coincide [23].

Although appealing because of its single-status approach, grounded semantics can be criticised as being too sceptical because it typically leaves many undecided arguments, i.e., arguments neither belonging to the grounded extension nor attacked by it. The idea of both the eager and the ideal semantics is to keep the advantage of returning a single extension while avoiding being overly sceptical.

Formally, a set $E \subseteq A$ is an **ideal set** of (A, R) iff it is admissible and it is contained in every preferred extension. The **ideal extension** of (A, R) is its maximal (w.r.t. set inclusion) ideal set.

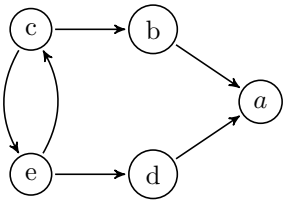
Moreover, a set $E \subseteq A$ is an **eager set** iff it is admissible and it is contained in every semi-stable extension. The **eager extension** of (A, R) is its maximal (w.r.t. set inclusion) eager set.

All the above semantics satisfy the so-called *admissibility principle*, meaning that all of their extensions are admissible sets. For some purposes, however, self-defence could be too strong a requirement, for instance when capturing human argument evaluation [45]. Alternative semantics selecting specific conflict-free sets were defined under the denomination *naivety-based* semantics (see e.g. [30]). The basis of all these semantics is the notion of naive extension. A **naive extension** of (A, R) is just a maximal (w.r.t. set inclusion) conflict-free set. A more elaborated naivety-based semantics, strongly inspired in the notion of semi-stability, is stage semantics. Formally, a **stage extension** of (A, R) is a conflict-free set with maximal range among conflict-free sets.

We abbreviate the name of each semantics by using the shorthands $\{st, co, gr, pr, se, id, ea, na, stg\}$ in the obvious way. For every $\sigma \in \{st, co, gr, pr, se, id, ea, na, stg\}$, we note $\sigma(A, R)$ the set of all σ-extensions of (A, R). An argument $x \in A$ is said to be credulously (resp. sceptically) σ-**accepted** iff it belongs to at least one (resp. every) σ-extension.

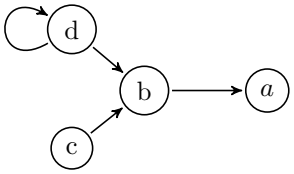
As an example, for the AF (A_0, R_0) represented in the picture below we have $\mathsf{st}(A_0, R_0) = \mathsf{pr}(A_0, R_0) = \mathsf{se}(A_0, R_0) = \mathsf{stg}(A_0, R_0) = \{\{b, e\}, \{c, d\}\}$; $\mathsf{gr}(A_0, R_0) = \mathsf{id}(A_0, R_0) = \mathsf{ea}(A_0, R_0) = \{\emptyset\}$; $\mathsf{co}(A_0, R_0) = \{\emptyset, \{b, e\}, \{c, d\}\}$ and $\mathsf{na}(A_0, R_0) = \{\{a, c\}, \{a, e\}, \{b, e\}, \{b, d\}, \{c, d\}\}$.

(A_0, R_0)



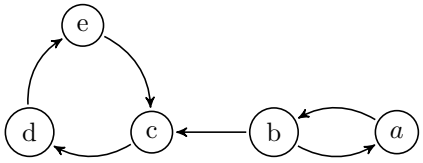
Moreover, (A_1, R_1), depicted below and borrowed from [23] illustrates the difference between stable and semi-stable semantics: the framework has no stable extension, while $\{c, a\}$ is a semi-stable extension (which is actually the only one).

(A_1, R_1)



Finally, to see the difference between (semi-)stable and preferred semantics consider (A_2, R_2), borrowed from [7] and depicted below. The set $\{a\}$ is a preferred extension, but it is not a (semi-)stable one. Moreover, the example also illustrates the difference between ideal and eager semantics, as the eager

extension is $\{b, d\}$, while the ideal one is empty. For more examples, the reader is referred to [7], as well as to the graphic on-line solver *ConArg* [20].

(A_2, R_2) 

2.2 Dynamic Logic of Propositional Assignments (DL-PA)

We use DL-PA as the general logical framework of this paper. The language of DL-PA is built from a countably infinite set of propositional variables $\mathtt{Prp} = \{p_1, p_2, \ldots\}$. We suppose that $\mathtt{Prp}$ contains several kinds of propositional variables capturing statuses of arguments and relations between them. First, to every set of arguments $A \subseteq \mathcal{U}$ we associate the set of **awareness variables** $\mathsf{AW}_A = \{\mathtt{aw}_x \mid x \in A\}$ and the set of **acceptance variables** $\mathsf{IN}_A = \{\mathtt{in}_x \mid x \in A\}$. Second, to every relation $R \subseteq \mathcal{U} \times \mathcal{U}$ we associate the set of **attack variables** $\mathsf{ATT}_R = \{\mathtt{r}_{x,y} \mid (x, y) \in R\}$. The set of propositional variables of our logic therefore contains

$$\mathtt{Prp}_{\mathcal{U}} = \mathsf{AW}_{\mathcal{U}} \cup \mathsf{IN}_{\mathcal{U}} \cup \mathsf{ATT}_{\mathcal{U}\times\mathcal{U}}.$$

As $\mathtt{Prp}_{\mathcal{U}}$ is finite, the countably infinite $\mathtt{Prp}$ provides a reservoir of auxiliary variables that are going to help us to encode e.g. semi-stable and stage semantics. Formulas and programs of DL-PA are defined by mutual recursion:

$$\varphi ::= p \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid [\pi]\varphi,$$
$$\pi ::= +p \mid -p \mid \varphi? \mid (\pi;\pi) \mid (\pi \cup \pi) \mid \pi^{\smile},$$

where p ranges over $\mathtt{Prp}$. The formula $[\pi]\varphi$ reads "φ is true after every possible execution of π". The program $+p$ makes p true and $-p$ makes p false. The program $\varphi?$ tests that φ is true and fails when it is false. The program $\pi_1;\pi_2$ is the sequential composition of π_1 and π_2; and $\pi_1 \cup \pi_2$ is their nondeterministic composition. Finally, $\pi^{\smile}$ is the execution of π 'the other way round'; for example, the program $+p^{\smile}$ undoes the assignment of p to true: when p is false then it fails, and when p is true then it nondeterministically either does nothing or makes p false.

Here are some more examples. The formula $[-p]\neg p$ is going to be valid: there is only one way of executing $-p$, and p is false afterwards. In contrast, $[+p]\neg p$ is going to be unsatisfiable. Moreover, $[+p]q$ is equivalent to q for syntactically different p and q. The formula $[\varphi?]\psi$ says that ψ is true after every possible execution of the test $\varphi?$. There is at most one such execution, namely when φ is true, and it does not change anything; when φ is false then the test fails and $[\varphi?]\psi$ is vacuously true. Therefore $[\varphi?]\psi$ has to be equivalent to $\neg\varphi \vee \psi$. The formula $[\pi_1;\pi_2]\varphi$ is equivalent to $[\pi_1][\pi_2]\varphi$ and $[\pi_1 \cup \pi_2]\varphi$ is going to be equivalent to $[\pi_1]\varphi \wedge [\pi_2]\varphi$. Finally, the formulas $[+p \cup -p]\neg p$ and $[-p^{\smile}]p$ are both going

to be unsatisfiable. The former is the case because there is a nondeterministic choice (namely that of $+p$) after which p is true; the latter is the case because there is an execution of the nondeterministic $-p^\smile$ after which p is still false.

Here are some abbreviations of formulas and programs that are going to be useful in the rest of the paper. The program $\top?$ is abbreviated as skip: it always succeeds and does not change anything. The program $(\varphi?;\alpha)\cup(\neg\varphi?;\beta)$ abbreviates $\mathsf{if}\,\varphi\,\mathsf{then}\,\alpha\,\mathsf{else}\,\beta$. A special case of the latter is when β is skip, where we just write $\mathsf{if}\,\varphi\,\mathsf{then}\,\pi$. (Observe that this is not the same as $\varphi?;\alpha$: when φ is false then the latter fails while the former succeeds and does nothing.) As to formulas, the missing Boolean connectives are defined as usual. Moreover, the formula $\langle\pi\rangle\varphi$ abbreviates $\neg[\pi]\neg\varphi$. It therefore reads "φ is true after some possible execution of π". In particular, $\langle\pi\rangle\top$ has to be read "π is executable". For example, the formula $\varphi\to[\pi]\langle\pi^\smile\rangle\varphi$ expresses that every successful execution of π can be reversed; it is going to be valid. (Observe that the diamond cannot be replaced by a box, as illustrated by the invalid $p\to[+p][+p^\smile]p$.)

Our models are classical propositional valuations over $\mathtt{Prp}$, i.e., they are subsets of $\mathtt{Prp}$. We use v, v', v'' to denote valuations. Formulas φ are interpreted in a way similar to dynamic logic, and programs π are interpreted as binary relations on valuations. Just as the syntax, the semantics of DL-PA is defined by mutual recursion. The interpretation of formulas is:

$$\begin{array}{lll} v\models p & \text{if} & p\in v,\\ v\models[\pi]\varphi & \text{if} & (v,v')\in||\pi||\text{ implies }v'\models\varphi,\end{array}$$

and as usual for the Boolean connectives; and the interpretation of programs is:

$$\begin{aligned} ||+p|| &= \{(v,v')\mid v'=v\cup\{p\}\},\\ ||-p|| &= \{(v,v')\mid v'=v\setminus\{p\}\},\\ ||\varphi?|| &= \{(v,v)\mid v\models\varphi\},\\ ||\pi;\pi'|| &= ||\pi||\circ||\pi'||,\\ ||\pi\cup\pi'|| &= ||\pi||\cup||\pi'||,\\ ||\pi^\smile|| &= ||\pi||^{-1}.\end{aligned}$$

The interpretation of $+p$ is the relation that makes p true while not changing anything else; and similarly for $-p$. That of the test $\varphi?$ relates every valuation where φ is true with itself; for example, $||\top?|| = \{(v,v)\mid v\subseteq\mathtt{Prp}\}$. Sequential composition $\pi_1;\pi_2$ is naturally interpreted as relation composition and nondeterministic composition $\pi_1\cup\pi_2$ as set union of the two relations $||\pi_1||$ and $||\pi_2||$. The interpretation of the converse $\pi^\smile$ is the inverse of the relation $||\pi||$ and relates a valuation v to all those valuations where π is executable and may lead to v. For example, $||+p^\smile|| = \{(v',v)\mid v'=v\cup\{p\}\}$.

A formula φ is DL-PA satisfiable if $v\models\varphi$ for some v, and it is DL-PA valid if $v\models\varphi$ for every v. For example, $\langle+p\rangle\top$ and $\langle-p\rangle\top$ are both valid, while $\langle-p^\smile\rangle p$ and $\langle-p^\smile\rangle\neg p$ are satisfiable but not valid. It is known that satisfiability, validity, and model checking are all PSPACE complete decision problems [5].

Let us now introduce some DL-PA programs that will be useful later on. Let $\mathsf{P} = \{p_1, \ldots, p_n\} \subseteq \mathtt{Prp}$ be a finite set of propositional variables. First of all, we define

$$\mathbin{;}_{p\in\mathsf{P}}\pi_p = \pi_{p_1}; \ldots; \pi_{p_n}.$$

By convention, we assume that the abbreviation amounts to skip when $\mathsf{P} = \emptyset$. We adopt the same convention for nondeterministic union, i.e., $\bigcup_{p\in\emptyset} \pi_p = \mathsf{skip}$. In principle the order of the elements of P matters, but each time we are going to use this notation we will make sure that the programs π_{p_i} are such that this is not the case. This will in particular hold for the following abbreviations:

$$\begin{aligned}
\mathsf{mkTrueOne}(\mathsf{P}) &= \bigcup_{p\in\mathsf{P}} (\neg p?; +p) = (\neg p_1?; +p_1) \cup \ldots \cup (\neg p_n?; +p_n),\\
\mathsf{mkFalseOne}(\mathsf{P}) &= \bigcup_{p\in\mathsf{P}} (p?; -p) = (p_1?; -p_1) \cup \ldots \cup (p_n?; -p_n),\\
\mathsf{mkTrueSome}(\mathsf{P}) &= \mathbin{;}_{p\in\mathsf{P}}(+p \cup \mathsf{skip}) = (+p_1 \cup \mathsf{skip}); \ldots; (+p_n \cup \mathsf{skip}),\\
\mathsf{mkFalseSome}(\mathsf{P}) &= \mathbin{;}_{p\in\mathsf{P}}(-p \cup \mathsf{skip}) = (-p_1 \cup \mathsf{skip}); \ldots; (-p_n \cup \mathsf{skip}),\\
\mathsf{vary}(\mathsf{P}) &= \mathbin{;}_{p\in\mathsf{P}}(+p \cup -p) = \big(+p_1 \cup -p_1\big); \ldots; \big(+p_n \cup -p_n\big).
\end{aligned}$$

The program $\mathsf{mkTrueOne}(\mathsf{P})$ chooses an element of P, checks that it is false and makes it true, while $\mathsf{mkTrueSome}(\mathsf{P})$ makes true some elements of P that were false before. The programs $\mathsf{mkTrueOne}(\mathsf{P})$ and $\mathsf{mkFalseOne}(\mathsf{P})$ are the converse of each other; same for $\mathsf{mkTrueSome}(\mathsf{P})$ and $\mathsf{mkFalseSome}(\mathsf{P})$. The sequential composition $\mathsf{mkTrueOne}(\mathsf{P}); \mathsf{mkTrueSome}(\mathsf{P})$ makes true at least one element of P that was false before (possibly more). The last program—i.e., $\mathsf{vary}(\mathsf{P})$—has the same interpretation as the sequential compositions $\mathsf{mkTrueSome}(\mathsf{P}); \mathsf{mkFalseSome}(\mathsf{P})$ and $\mathsf{mkFalseSome}(\mathsf{P}); \mathsf{mkTrueSome}(\mathsf{P})$.

Let us state formally the meaning of these programs:[2]

Proposition 1. *We have:*

$$\begin{aligned}
||\mathsf{mkTrueOne}(\mathsf{P})|| &= \{(v, v') \mid v' = v \cup \{p\} \textit{ for some } p \in \mathsf{P}\backslash v\},\\
||\mathsf{mkFalseOne}(\mathsf{P})|| &= \{(v, v') \mid v' = v \setminus \{p\} \textit{ for some } p \in \mathsf{P}\cap v\},\\
||\mathsf{mkTrueSome}(\mathsf{P})|| &= \{(v, v') \mid v' = v \cup \mathsf{P}' \textit{ for some } \mathsf{P}' \subseteq \mathsf{P}\},\\
||\mathsf{mkFalseSome}(\mathsf{P})|| &= \{(v, v') \mid v' = v \setminus \mathsf{P}' \textit{ for some } \mathsf{P}' \subseteq \mathsf{P}\},\\
||\mathsf{vary}(\mathsf{P})|| &= \{(v, v') \mid v \setminus v' \subseteq \mathsf{P} \textit{ and } v' \setminus v \subseteq \mathsf{P}\}.
\end{aligned}$$

From valuations to AFs and backward. Thanks to our hypothesis that $\mathtt{Prp}$ contains $\mathtt{Prp}_{\mathcal{U}}$, each valuation $v \subseteq \mathtt{Prp}$ represents the AF (A_v, R_v) defined by:

$$\begin{aligned}
A_v &= \{x \in \mathcal{U} \mid \mathtt{aw}_x \in v\},\\
R_v &= \{(x, y) \in \mathcal{U} \times \mathcal{U} \mid \mathtt{r}_{x,y} \in v\} \restriction_{A_v}\\
&= \{(x, y) \in A_v \times A_v \mid \mathtt{r}_{x,y} \in v\}.
\end{aligned}$$

[2]The following proposition is a slight correction of [36, Lemma 1].

The other way round, each AF (A, R) is represented by the valuation

$$v_{(A,R)} = \{\mathtt{aw}_x \mid x \in A\} \cup \{\mathtt{r}_{x,y} \mid (x, y) \in R\}.$$

Note that the valuation $v_{(A,R)}$ is well defined for any set $A \subseteq \mathcal{U}$ and relation $R \subseteq \mathcal{U} \times \mathcal{U}$, even when (A, R) is not an AF. (This is the case as soon as R contains pairs $(x, y) \in \mathcal{U} \times \mathcal{U}$ that are not in $A \times A$.) Moreover, notice that if we start with a valuation v' then $v_{(A_{v'}, R_{v'})} = v'$ does not generally hold because a valuation can contain an attack variable $\mathtt{r}_{a,b}$ without containing $\mathtt{aw}_a$ and $\mathtt{aw}_b$. If we, however, start with an AF (A', R') then $(A_{v_{(A',R')}}, R_{v_{(A',R')}}) = (A', R')$ is always the case. Finally, for each valuation v we define the **extension associated to** v by:

$$\mathsf{E}_v = \{x \in \mathcal{U} \mid \mathtt{in}_x \in v\}.$$

3 Argumentation Semantics in DL-PA

We now show how to capture argumentation semantics in DL-PA. The starting point is to adopt the encoding of AFs in propositional logic as introduced in [19]. It consists in associating to each semantics σ a formula φ_σ such that $v \models \varphi_\sigma$ if and only if E_v is a σ-extension of (A_v, R_v). This approach was pushed further in [35, 37, 36, 48], where it was proposed to go beyond the characterisation of extensions and exploit DL-PA programs to *describe the computation of extensions*. The most basic way to do so is a 'generate and test' approach: the generic program

$$\mathsf{makeExt}^\sigma = \mathsf{vary}(\mathsf{IN}_\mathcal{U}); \varphi_\sigma?$$

nondeterministically builds all possible σ-extensions by first varying the values of the acceptance variables and then checking that a σ-valuation has been obtained. As worked out in [36], other, more efficient extension building algorithms can also be captured as DL-PA programs and can be proved to be equivalent to $\mathsf{makeExt}^\sigma$.

Due to our hypothesis of a background universe of arguments $\mathcal{U}$ we need an encoding of argumentation semantics that takes awareness variables $\mathtt{aw}_x$ into account. This was done by [37] for stable semantics.[3] Here we extend the encoding to the rest of the semantics presented in Section 2.1. The correctness of all encodings is formally stated at the end of this section.

We start by defining some formulas that allow us to capture the different semantics in a compact way.

3.1 Useful DL-PA Formulas

The following DL-PA formula expresses that the arguments identified by acceptance variables are indeed arguments entertained by the formalised agent

[3] In [37], the term *enablement* and the notation En_x are used instead of *awareness* and $\mathtt{aw}_x$.

(arguments she is aware of):

$$\mathsf{Well} = \bigwedge_{x \in \mathcal{U}} (\mathtt{in}_x \rightarrow \mathtt{aw}_x).$$

This abbreviation allows us to express conflict-freeness and admissibility:

$$\mathsf{ConFree} = \mathsf{Well} \wedge \bigwedge_{x \in \mathcal{U}} \bigwedge_{y \in \mathcal{U}} \neg(\mathtt{in}_x \wedge \mathtt{in}_y \wedge \mathtt{r}_{x,y}),$$

$$\mathsf{Admissible} = \mathsf{ConFree} \wedge \bigwedge_{x \in \mathcal{U}} \Big(\mathtt{in}_x \rightarrow \bigwedge_{y \in \mathcal{U}} \big((\mathtt{aw}_y \wedge \mathtt{r}_{y,x}) \rightarrow \bigvee_{z \in \mathcal{U}} (\mathtt{in}_z \wedge \mathtt{r}_{z,y})\big)\Big).$$

Our characterisation of semi-stable and stage extensions makes use of fresh copies $\mathtt{in}'_x$ of the variables $\mathtt{in}_x$, one per $x \in \mathcal{U}$ (which are available because $\mathtt{Prp}$ is countably infinite while $\mathcal{U}$ is finite). For these auxiliary variables we define a program that copies the values of the $\mathsf{IN}_{\mathcal{U}}$ variables:

$$\mathsf{copy}(\mathsf{IN}_{\mathcal{U}}) = \mathbin{;}_{x \in \mathcal{U}}((\mathtt{in}_x?; +\mathtt{in}'_x) \cup (\neg\mathtt{in}_x?; -\mathtt{in}'_x)).$$

Furthermore, the following two formulas characterise whether the range of the extension E_v represented by v, in symbols $\mathsf{E}_v^{\oplus}$, is included in the range of the extension represented by the copies; and vice versa:

$$\mathsf{IncludedInCp} = \bigwedge_{x \in \mathcal{U}} \left[\left(\mathtt{in}_x \vee \left(\mathtt{aw}_x \wedge \bigvee_{y \in \mathcal{U}} (\mathtt{in}_y \wedge \mathtt{r}_{y,x}) \right) \right) \rightarrow \left(\mathtt{in}'_x \vee \left(\mathtt{aw}_x \wedge \bigvee_{y \in \mathcal{U}} (\mathtt{in}'_y \wedge \mathtt{r}_{y,x}) \right) \right) \right],$$

$$\mathsf{IncludesCp} = \bigwedge_{x \in \mathcal{U}} \left[\left(\mathtt{in}'_x \vee \left(\mathtt{aw}_x \wedge \bigvee_{y \in \mathcal{U}} (\mathtt{in}'_y \wedge \mathtt{r}_{y,x}) \right) \right) \rightarrow \left(\mathtt{in}_x \vee \left(\mathtt{aw}_x \wedge \bigvee_{y \in \mathcal{U}} (\mathtt{in}_y \wedge \mathtt{r}_{y,x}) \right) \right) \right].$$

Finally, to capture ideal and eager semantics we need to ensure that the entertained set is admissible and belongs to every preferred extension (for the case of ideal semantics), or to every semi-stable extension (for the case of eager semantics). This can be done in a compact way by means of the extension-building programs $\mathsf{makeExt}^{\sigma}$:

$$\mathsf{IdealSet} = \mathsf{Admissible} \wedge \bigwedge_{x \in \mathcal{U}} (\mathtt{in}_x \rightarrow [\mathsf{makeExt}^{pr}]\mathtt{in}_x),$$

$$\mathsf{EagerSet} = \mathsf{Admissible} \wedge \bigwedge_{x \in \mathcal{U}} (\mathtt{in}_x \rightarrow [\mathsf{makeExt}^{se}]\mathtt{in}_x).$$

$$\mathsf{Stable} = \mathsf{Well} \wedge \bigwedge_{x \in \mathcal{U}} \Big(\mathtt{aw}_x \rightarrow \big(\mathtt{in}_x \leftrightarrow \neg \bigvee_{y \in \mathcal{U}} (\mathtt{in}_y \wedge \mathtt{r}_{y,x}) \big) \Big),$$

$$\mathsf{Complete} = \mathsf{ConFree} \wedge \bigwedge_{x \in \mathcal{U}} \Big(\mathtt{in}_x \leftrightarrow \bigwedge_{y \in \mathcal{U}} \big((\mathtt{aw}_y \wedge \mathtt{r}_{y,x}) \rightarrow \bigvee_{z \in \mathcal{U}} (\mathtt{in}_z \wedge \mathtt{r}_{z,y}) \big) \Big),$$

$$\mathsf{Grounded} = \mathsf{Complete} \wedge [\mathsf{mkFalseOne}(\mathsf{IN}_{\mathcal{U}}); \mathsf{mkFalseSome}(\mathsf{IN}_{\mathcal{U}})] \neg \mathsf{Complete},$$

$$\mathsf{Preferred} = \mathsf{Admissible} \wedge [\mathsf{mkTrueOne}(\mathsf{IN}_{\mathcal{U}}); \mathsf{mkTrueSome}(\mathsf{IN}_{\mathcal{U}})] \neg \mathsf{Admissible},$$

$$\mathsf{Naive} = \mathsf{ConFree} \wedge [\mathsf{mkTrueOne}(\mathsf{IN}_{\mathcal{U}})] \neg \mathsf{ConFree},$$

$$\mathsf{SemiStable} = \mathsf{Complete} \wedge [\mathsf{copy}(\mathsf{IN}_{\mathcal{U}}); \mathsf{makeExt}^{co}] \left(\mathsf{IncludesCp} \rightarrow \mathsf{IncludedInCp} \right),$$

$$\mathsf{Stage} = \mathsf{ConFree} \wedge [\mathsf{copy}(\mathsf{IN}_{\mathcal{U}}); \mathsf{vary}(\mathsf{IN}_{\mathcal{U}}); \mathsf{ConFree?}] \left(\mathsf{IncludesCp} \rightarrow \mathsf{IncludedInCp} \right),$$

$$\mathsf{Ideal} = \mathsf{IdealSet} \wedge [\mathsf{mkTrueOne}(\mathsf{IN}_{\mathcal{U}}); \mathsf{mkTrueSome}(\mathsf{IN}_{\mathcal{U}})] \neg \mathsf{IdealSet},$$

$$\mathsf{Eager} = \mathsf{EagerSet} \wedge [\mathsf{mkTrueOne}(\mathsf{IN}_{\mathcal{U}}); \mathsf{mkTrueSome}(\mathsf{IN}_{\mathcal{U}})] \neg \mathsf{EagerSet}.$$

Table 1: Encoding the Semantics of Section 2.1 by DL-PA formulas

3.2 Encoding the Semantics of Section 2.1 in DL-PA

Table 1 lists all the encodings. That of stable and complete semantics slightly simplifies that of [37, 48]. Our encoding of grounded, complete, and preferred semantics straightforwardly adapts those of [36] for computing minimality and maximality criteria. The first four encodings are essentially a combination of those developed in [37] and [36], with some slight improvements and adaptations. Among the semantics that have not been captured in DL-PA before, our encoding of naive semantics simplifies the program for checking set maximality w.r.t. other semantics such as preferred semantics because no superset of a set containing conflicts can be conflict-free.

Theorem 1. *Let $\sigma \in \{st, co, gr, pr, se, id, ea, na, stg\}$. Let $v \subseteq \mathtt{Prp}$. Let (A, R) be an AF. Then:*

- $v \models \varphi_\sigma$ *iff* $\mathsf{E}_v \in \sigma(A_v, R_v)$*;*
- $\sigma(A, R) = \big\{ \mathsf{E}_v \mid (v_{(A,R)}, v) \in ||\mathsf{makeExt}^\sigma|| \big\}$.

The proof can be found in the Appendix, just as the proofs or proof sketches of all other results.

4 Qualitative Uncertainty in Abstract Argumentation through DL-PA

In this section, we review existing formalisms for representing uncertainty about AFs. We restrict our attention to *qualitative* forms of uncertainty, that is, representations neither using probabilities nor any other kind of numeric device.

In particular, we cover: *incomplete argumentation frameworks* [13], their *enriched version* [53], *constrained incomplete argumentation frameworks* [48, 54], and *incomplete argumentation frameworks with dependencies* [41, 42]. The main motivation for the study of these formalisms is that there are several sources of uncertainty in real-life argumentation. For instance, arguments can be so complex that the reasoning agent is not sure whether they are to be taken into account or whether they attack other arguments. Perhaps more frequently, uncertainty appears in argumentation when an agent reasons about *her opponent's argumentative situation*. Due to the lack of total knowledge about her adversary, the agent might doubt whether the latter entertains some of the arguments or sees some of the attacks. And this is in turn crucial for choosing the right arguments to convince her opponent. We keep this latter intuition in mind as a guideline for the rest of the paper.

All the formalisms of the present section share the idea of representing uncertainty through the notion of *completion*. A completion is a hypothetical removal of uncertainty, such that the formalised agent reasons under the assumption that her opponent's AF is such-and-such. In epistemic logic terms, this amounts to the notion of *possible world*, as mentioned in [14, 15], and studied in detail in [59, 50]. For a more elaborated comparison among the formalisms presented in this section and epistemic logic, the interested reader is referred to Section 6.

After introducing each formalism we explain how the main associated reasoning tasks can be reduced to DL-PA model checking problems. We conclude by providing a comparison of the different approaches.

4.1 Incomplete AFs

An **incomplete AF** [13] (IAF), is a pair $\mathsf{IAF} = (F, U)$, where $F = (A^F, R^F)$ is called the *fixed part*, $U = (A^?, R^?)$ is called the *uncertain part*, $R, R^? \subseteq (A^F \cup A^?) \times (A^F \cup A^?)$, $A^F \cap A^? = \emptyset$ and $R^F \cap R^? = \emptyset$. Hence an IAF is basically an AF where arguments and attacks have been split into two disjoint sets. We sometimes omit internal parentheses when talking about IAFs, that is, we write $(A^F, R^F, A^?, R^?)$ instead of $((A^F, R^F), (A^?, R^?))$. Note that, by definition, there can be fixed attacks among uncertain arguments (sometimes called *conditionally definite attacks* [14]). We can intuitively think about these as attacks the agent thinks her opponent entertains whenever she thinks that her opponent is aware of the involved arguments.

A **completion** of an $\mathsf{IAF} = (A^F, R^F, A^?, R^?)$ is any AF (A^*, R^*) such that:

- $A^F \subseteq A^* \subseteq A^F \cup A^?$; and
- $R^F \restriction_{A^*} \subseteq R^* \subseteq (R^F \cup R^?) \restriction_{A^*}$.

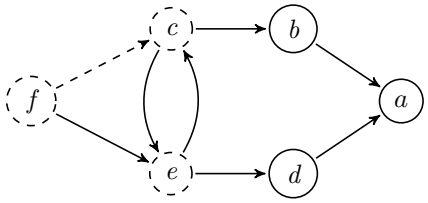
Given an IAF IAF, we note $\mathsf{completions}(\mathsf{IAF})$ the set of all its completions.

A standard AF (A, R) can be identified with the IAF $(A, R, \emptyset, \emptyset)$, which is the unique completion of itself. Two subclasses of IAFs are well-studied in the literature, namely **attack-incomplete AFs** (att-IAFs, for short),[4] which are

[4]This subclass was previously studied under the name of *partial AFs* [24, 28].

IAFs with empty $A^?$; and **argument-incomplete** AFs (arg-IAFs, for short), which are IAFs with empty $R^?$.

Example 1. *Let us consider* $\mathsf{IAF}_0 = (A_0^F, R_0^F, A_0^?, R_0^?)$*, where* $A_0^F = \{a, b, d\}$*,* $R_0^F = \{(b,a),(d,a),(c,b),(e,d),(c,e),(e,c),(f,e)\}$*,* $A_0^? = \{c, e, f\}$ *and* $R_0^? = \{(f,c)\}$*, graphically represented below. The set of completions of* IAF_0 *is the one depicted in Table 2 except for the cells* ***B2****,* ***C2****,* ***B4****,* ***C4****,* ***B5*** *and* ***C5****.*



Classic reasoning tasks such as extension enumeration or argument acceptance have been generalized from AFs to IAFs. We here focus on acceptance queries such as the following:

σ-Necessary-Credulous-Acceptance (σ-NCA)
Given: An IAF $\mathsf{IAF} = (A^F, R^F, A^?, R^?)$ and an argument $a \in A^F$.
Question: Is it true that for every $(A^*, R^*) \in \mathsf{completions}(\mathsf{IAF})$ there is an $E \in \sigma(A^*, R^*)$ such that $a \in E$?

We can switch quantifiers in the definition above in order to obtain different variants of the problem, resulting in possible and sceptical variants. Note that the only difference between these reasoning tasks and standard acceptance problems in AFs is an added quantification layer, namely quantification over completions.

Our aim now is to reduce these acceptance problems to DL-PA model checking problems. As we already have programs for building the extensions of AFs, the fundamental step in this reduction consists in designing a DL-PA program, $\mathsf{makeComp}^{\mathsf{IAF}}$, that computes all the completions of IAF.

First, the **valuation associated to** IAF is determined by its fixed part:

$$\begin{aligned} v_{\mathsf{IAF}} &= v_{(A^F, R^F)} \\ &= \mathsf{AW}_{A^F} \cup \mathsf{ATT}_{R^F} \\ &= \{\mathtt{aw}_x \mid x \in A^F\} \cup \{\mathtt{r}_{x,y} \mid (x,y) \in R^F\}. \end{aligned}$$

Note that $(A_{v_{\mathsf{IAF}}}, R_{v_{\mathsf{IAF}}})$ is already a completion of IAF: it is the smallest one, where only fixed arguments and fixed attacks between them are considered. In order to compute all the completions of IAF we make true subsets of propositional variables representing arguments in $A^?$ and attacks in $R^?$:

$$\mathsf{makeComp}^{\mathsf{IAF}} = \mathsf{mkTrueSome}(\mathsf{AW}_{A^?}); \mathsf{mkTrueSome}(\mathsf{ATT}_{R^?}).$$

The next proposition shows that our original target is reached.

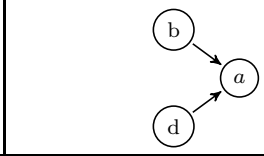
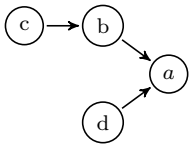
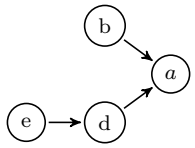
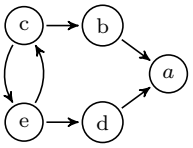
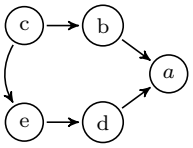
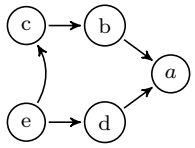
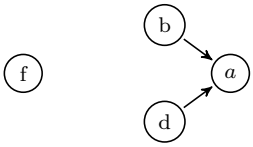
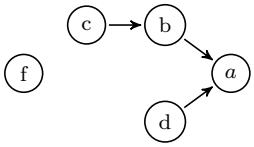
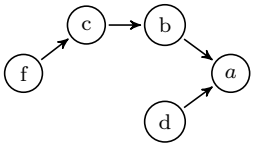
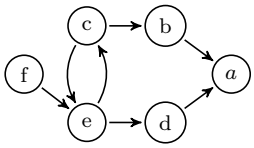
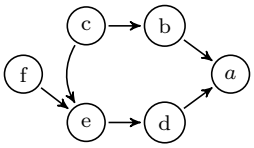
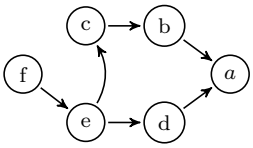
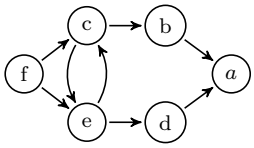
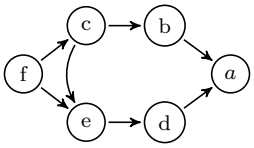
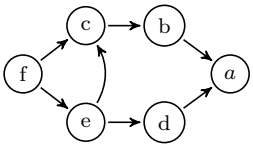
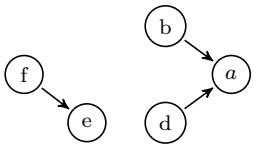
	A	**B**	**C**
1			
2			
3			
4			
5			
6			

Table 2: Completions of CAF_0. The column [**1**, **2**,..., **6**] and the row [**A**, **B**, **C**] are just included for numbering purposes. (Empty cells do *not* represent the empty completion $(\emptyset, \emptyset)$.)

Proposition 2. *Let* $\mathsf{IAF} = (A^F, R^F, A^?, R^?)$. *Then:*

- *If* $(v_{\mathsf{IAF}}, v) \in ||\mathsf{makeComp}^{\mathsf{IAF}}||$, *then* $(A_v, R_v) \in \mathsf{completions}(\mathsf{IAF})$.
- *If* $(A^*, R^*) \in \mathsf{completions}(\mathsf{IAF})$, *then* $(v_{\mathsf{IAF}}, v_{(A^*,R^*)}) \in ||\mathsf{makeComp}^{\mathsf{IAF}}||$.

Using this result together with the general technique to compute extensions provided in Section 3, we can reduce reasoning problems in IAFs to model checking problems in DL-PA.

Proposition 3. *Let* $\mathsf{IAF} = (F, U)$, $\sigma \in \{st, co, gr, pr, se, id, ea, na, stg\}$, *and* $a \in A^F$. *Then:*

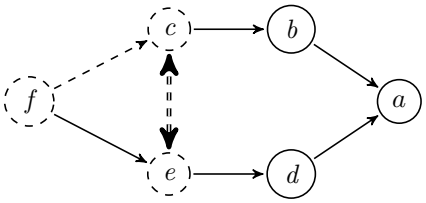
- *The answer to* σ*-NSA with input* IAF *and* a *is yes iff* $v_{\mathsf{IAF}} \models [\mathsf{makeComp}^{\mathsf{IAF}}; \mathsf{makeExt}^{\sigma}]\mathtt{in}_a$.
- *The answer to* σ*-NCA with input* IAF *and* a *is yes iff* $v_{\mathsf{IAF}} \models [\mathsf{makeComp}^{\mathsf{IAF}}]\langle\mathsf{makeExt}^{\sigma}\rangle\mathtt{in}_a$.
- *The answer to* σ*-PCA with input* IAF *and* a *is yes iff* $v_{\mathsf{IAF}} \models \langle\mathsf{makeComp}^{\mathsf{IAF}}; \mathsf{makeExt}^{\sigma}\rangle\mathtt{in}_a$.
- *The answer to* σ*-PSA with input* IAF *and* a *is yes iff* $v_{\mathsf{IAF}} \models \langle\mathsf{makeComp}^{\mathsf{IAF}}\rangle[\mathsf{makeExt}^{\sigma}]\mathtt{in}_a$.

4.2 Rich Incomplete AFs

A **rich incomplete AF** (rIAF) [53] extends an IAF in its uncertain part U by adding a new (symmetric and irreflexive) uncertainty relation $R^{\leftrightarrow} \subseteq (A^F \cup A^?) \times (A^F \cup A^?)$ such that $R^{\leftrightarrow} \cap R^F = \emptyset$ and $R^{\leftrightarrow} \cap R^? = \emptyset$. We sometimes omit internal brackets when talking about rIAFs and note them $(A^F, R^F, A^?, R^?, R^{\leftrightarrow})$. The new component, $R^{\leftrightarrow}$, is informally understood as a set of attacks whose existence is known, but whose direction is unknown. The introduction of $R^{\leftrightarrow}$ can be motivated by pointing out that attacks have two essential properties: their existence and their direction. Thus, while $R^?$ captures uncertainty about the former, $R^{\leftrightarrow}$ captures uncertainty about the latter. Note that any IAF can be understood as a rIAF with empty $R^{\leftrightarrow}$. The notion of completion is easily adapted to rIAFs, capturing the intuitions about $R^{\leftrightarrow}$ that we have just mentioned. A **completion** of $\mathsf{rIAF} = (A^F, R^F, A^?, R^?, R^{\leftrightarrow})$ is any AF (A^*, R^*) such that:

- $A^F \subseteq A^* \subseteq (A^F \cup A^?)$;
- $R^F \restriction_{A^*} \subseteq R^* \subseteq (R^F \cup R^? \cup R^{\leftrightarrow}) \restriction_{A^*}$;
- for every $x, y \in A^*$: $(x, y) \in R^{\leftrightarrow}$ implies $(x, y) \in R^*$ or $(y, x) \in R^*$.

Example 2. *Let* $\mathsf{rIAF}_0 = (A_0^F, A_0^?, R_0^F, R_0^?, R_0^{\leftrightarrow})$ *where* $A_0^F = \{a, b, d\}$, $A_0^? = \{c, e, f\}$, $R_0^F = \{(b, a), (d, a), (c, b), (e, d), (f, e)\}$, $R_0^? = \{(f, c)\}$, *and* $R_0^{\leftrightarrow} = \{(c, e), (e, c)\}$. *We represent* rIAF_0 *graphically as follows:*



The set of completions of rIAF_0 *is depicted in Table 2.*

The computation of the completions of a rich IAF in DL-PA gets slightly more complicated since the program mkTrueSome does not suffice to deal with the symmetric attacks of $R^{\leftrightarrow}$. We can, however, define a specific program for this purpose.

First of all, given $\mathsf{rIAF} = (A^F, R^F, A^?, R^?, R^{\leftrightarrow})$, the **valuation associated to** rIAF is determined by its fixed part as before:

$$\begin{aligned} v_{\mathsf{rIAF}} &= v_{(A^F, R^F)} \\ &= \mathsf{AW}_{A^F} \cup \mathsf{ATT}_{R^F} \\ &= \{\mathtt{aw}_x \mid x \in A^F\} \cup \{\mathtt{r}_{x,y} \mid (x,y) \in R^F\}. \end{aligned}$$

Note that, contrarily to what happened with IAFs, $(A_{v_{\mathsf{rIAF}}}, R_{v_{\mathsf{rIAF}}})$ is *not* always a completion of rIAF: this fails to be the case as soon as $R^{\leftrightarrow} \cap (A^F \times A^F)$ is nonempty. Let us now define the program that integrates the elements of $R^{\leftrightarrow}$ into each completion.

Let $\mathsf{ATT}_R = \{\mathtt{r}_{x_1,y_1}, ..., \mathtt{r}_{x_n,y_n}\}$ [5] be a set of attack variables, and define the program

$$\mathsf{dis}(\mathsf{ATT}_R) = (+\mathtt{r}_{x_1,y_1} \cup +\mathtt{r}_{y_1,x_1})\,; \ldots ;(+\mathtt{r}_{x_n,y_n} \cup +\mathtt{r}_{y_n,x_n})\,.$$

Intuitively, $\mathsf{dis}(\mathsf{ATT}_R)$ makes true at least one of the variables from the set $\{\mathtt{r}_{x,y}, \mathtt{r}_{y,x}\}$, for each $(x,y) \in R$. Moreover, when applied to a symmetric relation $R^{\leftrightarrow}$, dis makes true either $\mathtt{r}_{x,y}$, or $\mathtt{r}_{y,x}$, or both, for every $(x,y) \in R^{\leftrightarrow}$.

We are now ready to define the program makeComp in its version for rIAFs. Given $\mathsf{rIAF} = (A^F, R^F, A^?, R^?, R^{\leftrightarrow})$, let

$$\mathsf{makeComp}^{\mathsf{rIAF}} = \mathsf{mkTrueSome}(\mathsf{AW}_{A^?})\,; \mathsf{mkTrueSome}(\mathsf{ATT}_{R^?})\,; \mathsf{dis}(\mathsf{ATT}_{R^{\leftrightarrow}}).$$

The following proposition states that the above program is correct.

Proposition 4. *Let* $\mathsf{rIAF} = (A^F, R^F, A^?, R^?, R^{\leftrightarrow})$*, then:*

- *If* $(v_{\mathsf{rIAF}}, v) \in ||\mathsf{makeComp}^{\mathsf{rIAF}}||$*, then* $(A_v, R_v) \in \mathsf{completions}(\mathsf{rIAF})$.
- *If* $(A^*, R^*) \in \mathsf{completions}(\mathsf{rIAF})$*, then* $(v_{\mathsf{rIAF}}, v_{(A^*,R^*)}) \in ||\mathsf{makeComp}^{\mathsf{rIAF}}||$.

[5] Remember that $\mathsf{ATT}_R = \{\mathtt{r}_{x,y} \mid (x,y) \in R\}$, and that ATT_R is a subset of the set of propositional variables $\mathsf{ATT}_{\mathcal{U}\times\mathcal{U}}$.

Again, acceptance problems can be reduced to DL-PA model checking problems. Note that the definition of acceptance problems for rIAFs is just as for IAFs (we only have to change the input). Let us just state the reduction result we are after:

Proposition 5. *Let* $\sigma \in \{st, co, gr, pr, se, id, ea, na, stg\}$. *Let* $\mathsf{rIAF} = (A^F, R^F, A^?, R^?, R^{\leftrightarrow})$ *and* $a \in A^F$. *Then:*

- *The answer to* σ*-NSA with input* rIAF *and* a *is yes iff* $v_{\mathsf{rIAF}} \models [\mathsf{makeComp}^{\mathsf{rIAF}}; \mathsf{makeExt}^{\sigma}]\mathtt{in}_a$.
- *The answer to* σ*-NCA with input* rIAF *and* a *is yes iff* $v_{\mathsf{rIAF}} \models [\mathsf{makeComp}^{\mathsf{rIAF}}]\langle\mathsf{makeExt}^{\sigma}\rangle\mathtt{in}_a$.
- *The answer to* σ*-PCA with input* rIAF *and* a *is yes iff* $v_{\mathsf{rIAF}} \models \langle\mathsf{makeComp}^{\mathsf{rIAF}}; \mathsf{makeExt}^{\sigma}\rangle\mathtt{in}_a$.
- *The answer to* σ*-PSA with input* rIAF *and* a *is yes iff* $v_{\mathsf{rIAF}} \models \langle\mathsf{makeComp}^{\mathsf{rIAF}}\rangle[\mathsf{makeExt}^{\sigma}]\mathtt{in}_a$.

4.3 Shrinking the Set of Completions

Incomplete AFs (and their enriched version) deal with uncertainty about argumentative situations in a simple and intuitive manner. However, the kind of situations that we can model with them is rather limited (as we will discuss in detail later on). This is the main motivation for the development of more expressive formalisms, and it actually led to concurrent proposals during the last year, either under the name of *constrained incomplete argumentation frameworks* [48, 54] or *incomplete argumentation frameworks with dependencies* [41, 42]. We start by presenting our version of constrained incomplete AFs (the one introduced in [48]), and then move to alternative approaches.

4.3.1 Constrained Incomplete AFs

A **constrained incomplete AF** (cIAF) is a pair $\mathsf{cIAF} = (A, \varphi)$ where $A \subseteq \mathcal{U}$ is a set of arguments and φ is a Boolean formula built over the set of propositional variables $\mathsf{AW}_A \cup \mathsf{ATT}_{A\times A}$.[6] The set of **completions** of a given cIAF is

$$\mathsf{completions}(A, \varphi) = \{(A_v, R_v) \mid v \subseteq \mathtt{Prp}_A \text{ and } v \models \varphi\}.$$

Example 3. *Let us consider* $\mathsf{cIAF}_0 = (A, \varphi)$ *with* $A = \{a, b\}$ *and* $\varphi = (\mathtt{aw}_a \wedge \mathtt{aw}_b) \wedge (\mathtt{r}_{a,b} \vee \mathtt{r}_{b,a}) \wedge \neg(\mathtt{r}_{a,b} \wedge \mathtt{r}_{b,a}) \wedge \neg\mathtt{r}_{a,a} \wedge \neg\mathtt{r}_{b,b}$. *The completions of* cIAF_0 *are:*

[6] We have slightly changed the original definition of cIAFs [48], by switching the domain from $\mathcal{U}$ to an arbitrary A, because it allows for naturally plugging-in argumentation dynamics, as we will do in Section 5.2.

Notice that, differently to what happened with previous classes of structures, the set of completions of a cIAF might be empty, since φ can be an inconsistent formula. Moreover, even being consistent, φ might not be satisfied by any valuation representing a non-empty AF, so that we could get the empty AF $(\emptyset, \emptyset)$ as the only completion of a cIAF; e.g., $\mathsf{completions}((\{a\}, \neg \mathtt{aw}_a)) = \{(\emptyset, \emptyset)\}$.

The need of cIAFs. Besides being mathematically interesting, one may wonder why one should use cIAFs. As mentioned, our main motivation is that, while the computational complexity of reasoning tasks associated to the previously introduced formalisms (i.e., (r)IAFs and subclasses) is well-known and relatively low, their modelling power is rather limited. Consider, for instance, a proponent reasoning about the view of her opponent in a very simple debate containing only two arguments $\{a, b\}$. Suppose that a is an argument about public health policies stated by the right-wing presidential candidate. Similarly, b is an argument stated by the left-wing candidate. Imagine that a and b have contradictory conclusions, so they are mutually incompatible. Let us informally understand R as a *defeat* relation here, that is, a relation based on logical incompatibility plus some kind of epistemic-based assessment of the involved arguments (for instance, regarding the reliability of their premisses), as it is usually done in structured argumentation. Now, suppose our proponent knows that her opponent is polarized, in the sense that he (the opponent) is already inclined towards one side of the political spectrum, but she does not know which one; then the possible AFs that the agent attributes to her opponent are exactly the completions of cIAF_0 (see Example 3). As it will be proved later (Proposition 8), there is no rIAF (and therefore no IAF) with the exact set of completions of cIAF_0.

Let us now show how cIAFs can be captured in DL-PA. Let $\mathsf{cIAF} = (A, \varphi)$, and define its **associated valuation** simply as the empty set, that is, $v_{\mathsf{cIAF}} = \emptyset$. (Actually any valuation over $\mathtt{Prp}_A$ will do the job.) The program generating all completions of cIAF is defined as

$$\mathsf{makeComp}^{\mathsf{cIAF}} = \mathsf{vary}(\mathsf{AW}_A); \mathsf{vary}(\mathsf{ATT}_{A\times A}); \varphi?.$$

The behaviour of $\mathsf{makeComp}^{\mathsf{cIAF}_0}$ (see Example 3) is illustrated in Figure 1.

Proposition 6. *Let* $\mathsf{cIAF} = (A, \varphi)$*, then:*

- *If* $(v_{\mathsf{cIAF}}, v) \in ||\mathsf{makeComp}^{\mathsf{cIAF}}||$*, then* $(A_v, R_v) \in \mathsf{completions}(\mathsf{cIAF})$.
- *If* $(A^*, R^*) \in \mathsf{completions}(\mathsf{cIAF})$*, then* $(v_{\mathsf{cIAF}}, v_{(A^*,R^*)}) \in ||\mathsf{makeComp}^{\mathsf{cIAF}}||$.

Reasoning problems for (r)IAFs can be easily adapted to cIAFs: we just have to ensure that the argument about which we formulate the query belongs to all completions. As an example, consider:

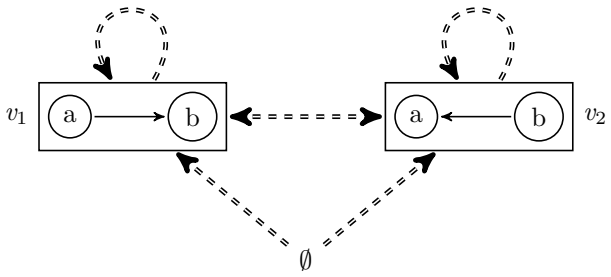


Figure 1: Completions of cIAF_0 seen as valuations over $\mathtt{Prp}_{\{a,b\}}$. Dashed double arrows represent the interpretation of $\mathsf{makeComp}^{\mathsf{cIAF}_0}$; the other valuations over $\mathtt{Prp}_{\{a,b\}}$ are omitted.

σ-Necessary-Credulous-Acceptance (σ-NCA)
Given: A constrained IAF $\mathsf{cIAF} = (A, \varphi)$ and an argument $a \in A$ such that $\models \varphi \rightarrow \mathtt{aw}_a$.
Question: Is it true that for every $(A^*, R^*) \in \mathsf{completions}(\mathsf{cIAF})$ there is an $E \in \sigma(A^*, R^*)$ such that $a \in E$?

Note that requiring $\models \varphi \rightarrow \mathtt{aw}_a$ amounts to requiring $a \in A$ for all $(A, R) \in \mathsf{completions}(A, \varphi)$.

Once again, we can reduce acceptance problems in cIAFs to DL-PA model checking problems.

Proposition 7. *Let* $\mathsf{cIAF} = (A, \varphi)$ *and let* $a \in A$ *such that* $\models \varphi \rightarrow \mathtt{aw}_a$. *Let* $\sigma \in \{st, co, gr, pr, se, id, ea, na, stg\}$. *Then:*

- *The answer to σ-NSA with input* cIAF *and a is yes iff* $v_{\mathsf{cIAF}} \models [\mathsf{makeComp}^{\mathsf{cIAF}}; \mathsf{makeExt}^{\sigma}]\mathtt{in}_a$.
- *The answer to σ-NCA with input* cIAF *and a is yes iff* $v_{\mathsf{cIAF}} \models [\mathsf{makeComp}^{\mathsf{cIAF}}]\langle \mathsf{makeExt}^{\sigma}\rangle \mathtt{in}_a$.
- *The answer to σ-PCA with input* cIAF *and a is yes iff* $v_{\mathsf{cIAF}} \models \langle \mathsf{makeComp}^{\mathsf{cIAF}}; \mathsf{makeExt}^{\sigma}\rangle \mathtt{in}_a$.
- *The answer to σ-PSA with input* cIAF *and a is yes iff* $v_{\mathsf{cIAF}} \models \langle \mathsf{makeComp}^{\mathsf{cIAF}}\rangle [\mathsf{makeExt}^{\sigma}] \mathtt{in}_a$.

We observe that beyond these reasoning problems one may also consider the reasoning task of checking emptiness of the set of completions of a cIAF.

4.3.2 Closely Related Approaches

As mentioned, the idea of shrinking the set of completions of an IAF led to concurrent proposals during the last year. In this subsection, we briefly present the two alternative approaches to our cIAFs of [48].

A more graph-theoretic version of cIAFs. In [54], Jean-Guy Mailly defined his version of cIAFs that we call **cIAFs**^{JM} here to avoid confusion. A cIAF^{JM} is pair of the form (IAF, φ) where $\mathsf{IAF} = (A^F, R^F, A^?, R^?)$ is an IAF and φ is a Boolean formula over $\mathtt{Prp}^{\mathsf{IAF}} = \mathsf{AW}_{A \cup A^?} \cup \mathsf{ATT}_{(A \cup A^?) \times (A \cup A^?)}$. Then the set of **completions of** $\mathsf{cIAF}^{JM} = (\mathsf{IAF}, \varphi)$ is defined as

$$\mathsf{completions}(\mathsf{IAF}) \cap \{(A_v, R_v) \mid v \subseteq \mathtt{Prp}^{\mathsf{IAF}} \text{ and } v \models \varphi\}.$$

IAFs with dependencies. In [41, 42], the team from the University of Calabria formed by Bettina Fazzinga, Sergio Flesca and Filippo Furfaro introduced the notion of IAFs with dependencies.[7] More precisely, their two proposals respectively focus on two restricted classes of IAFs that we have already mentioned: arg-IAFs, and att-IAFs. For the sake of brevity we only present here the notion of arg-IAF with dependencies of [41]. Let A be a set of arguments and let $X, Y \subseteq A$. First, a **dependency over** A is either $X \Rightarrow Y$ or $\mathsf{OP}(X)$ with $\mathsf{OP} \in \{\mathsf{OR}, \mathsf{NAND}, \mathsf{CHOICE}\}$. Second, an **arg-IAF with dependencies** (d-arg-IAF, for short) is a pair $((A, A^?, R), \Delta)$, where $(A, A^?, R)$ is an arg-IAF and Δ is a set of dependencies over $A^?$. Before defining the completions of a d-arg-IAF we need to settle how dependencies are to be interpreted in arg-IAFs. Let (A, R) be an AF and let δ be a dependency over A. We say that (A, R) satisfies δ[8] iff one of the following mutually exclusive clauses holds:

- $\delta = X \Rightarrow Y$ and (if $X \subseteq A$, then $A \cap Y \neq \emptyset$);
- $\delta = \mathsf{OR}(X)$ and $A \cap X \neq \emptyset$,
- $\delta = \mathsf{NAND}(X)$ and $A \cap X \subset X$,
- $\delta = \mathsf{CHOICE}(X)$ and $|A \cap X| = 1$.

The **completions of** $((A, A^?, R), \Delta)$ are defined as those completions of the arg-IAF $(A, A^?, R)$ that satisfy every dependency $\delta \in \Delta$.

The three alternative proposals are already compared in [52]. We will provide some new insights beyond this in the next section. Let us just make a couple of points here. First, note that both versions of cIAFs as well as IAFs with dependencies are clearly inspired by the notion of *constrained AF* [29], which are pairs $((A, R), \varphi)$ where φ is used to shrink the set of *extensions* of (A, R). Second, note that the reasoning tasks associated to both classes of structures are clearly encodable in DL-PA, but we do not work out the details here. Let us just point out that each set of dependencies Δ can be translated into a Boolean formula $t(\Delta)$, and then the program $\mathsf{makeComp}^{((A,A^?,R),\Delta)} = \mathsf{vary}(\mathsf{AW}_{A^?}); t(\Delta)?$ computes all the completions of $((A, A^?, R), \Delta)$ when executed at $v_{(A,R)}$.

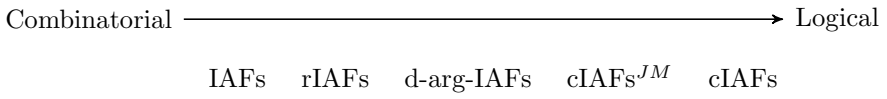
[7] The term *correlations* is used in [41, 42] as the informal counterpart of *dependencies*. We stick to the latter term to avoid confusion.

[8] [41] uses the expression "(A, R) is valid w.r.t. δ", but our expression is more appropriate in a logical analysis.

4.4 Comparison of the Different Approaches

Let us now compare the different approaches to representing qualitative uncertainty about AFs. We start with a couple of general considerations.

Combinatorics vs. logic. The spirit of the seminal works on IAFs was to represent uncertainty by defining completions as directed graphs whose domains and relations fall between given intervals. One may qualify this approach as "combinatorial", since, once the extremes of the interval are given (e.g. A^F and $A^F \cup A^?$), the task of computing completions amounts to finding all possible combinations within the interval. Progressively, other reasoning features that we might qualify as "logical" have been integrated in the definition of completion. For instance, rIAFs introduce a sort of disjunctive reasoning through the addition of $R^{\leftrightarrow}$. We can understand this transition from combinatorics to logic as a sort of spectrum:



Note how at the right-hand extreme (cIAFs), the combinatorial nature of completions has completely vanished.

Graphic representations. One of the appealing features of IAFs is that they admit a very intuitive graphic representation (see Example 1). Interestingly, rIAFs and d-arg-IAFs can also be fully represented in a pictorial manner; see [49] for examples with d-arg-IAFs. As pointed out in [53], cIAFs^{JM} only admit a partial graphic representation. Finally, this pictorial representability is lost by our cIAFs, which completely abstract away from the graph-theoretic definition of IAFs. Hence, in this respect, IAFs with dependencies compare better to cIAFs and cIAFs^{JM}.

Expressivity via sets of completions. Following [53], we can compare the modelling power of each of the previous formalisms for arguing with uncertainty using the sets of completions they can represent. Let $\mathcal{IAF}$ denote the class of all IAFs, and likewise for *att-*$\mathcal{IAF}$, *arg-*$\mathcal{IAF}$, $\mathcal{RIAF}$, *d-arg-*$\mathcal{IAF}$, *d-att-*$\mathcal{IAF}$, *c-*$\mathcal{IAF}$ and *c-*$\mathcal{IAF}^{JM}$. Let $\mathcal{X}$ and $\mathcal{Y}$ be metavariables denoting one of these classes. We say that $\mathcal{X}$ is **at least as expressive as** $\mathcal{Y}$ (in symbols: $\mathcal{X} \succeq \mathcal{Y}$) if, for every $Y \in \mathcal{Y}$ there is a $X \in \mathcal{X}$ such that $\mathsf{completions}(X) = \mathsf{completions}(Y)$. We use $\succ$ to denote the strict part of $\succeq$, we use $\preceq$ to denote the inverse of $\succeq$, and we use $\equiv$ to abbreviate $\succeq \cap \preceq$. For instance, it was proved in [53] that $\mathcal{RIAF} \succ \mathcal{IAF}$.

Proposition 8. *cIAFs are strictly more expressive than IAFs and rIAFs. In other words, for every (r)IAF, there is a cIAF with the same set of completions; but there is a cIAF such that no (r)IAF has the same set of completions.*

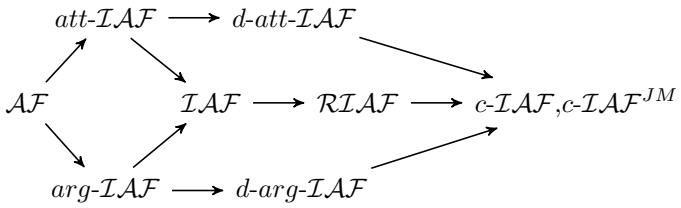


Figure 2: Relative expressivity of formalisms for qualitative uncertainty in formal argumentation. An arrow from $\mathcal{X}$ to $\mathcal{Y}$ means that $\mathcal{X} \preceq \mathcal{Y}$, i.e., $\mathcal{Y}$ is at least as expressive as $\mathcal{X}$. Reflexive and transitive arrows have been omitted.

In the first part of the proof of the previous proposition—see the Appendix—we have used an argument that works for *any* set of directed graphs with domain $\mathcal{U}$ (and not only for the completions of a given rIAF), hence we can state that:

Corollary 1. *For any set* S *of directed graphs with domain* $\mathcal{U}$ *there is a cIAF* cIAF *such that* $\mathsf{S} = \mathsf{completions}(\mathsf{cIAF})$.

In words, cIAFs are a maximally expressive formalism for representing qualitative uncertainty about AFs. Using arguments similar to those employed in the proof of Proposition 8 we can provide the following general result:

Proposition 9. *The relations of Figure 2 hold, where an arrow from* $\mathcal{X}$ *to* $\mathcal{Y}$ *means that* $\mathcal{X} \preceq \mathcal{Y}$ *and where transitive and reflexive arrows are omitted.*

Besides providing a full expressivity map, this proposition highlights the fact that IAFs with dependencies have not been given their most expressive formulation yet. That is, we have arg-IAFs with dependencies [41], and att-IAFs with dependencies [42], but no IAFs with dependencies. This makes that these kinds of structures do not yet permit expressing any set of completions (contrarily to what happens with both cIAFs and cIAFs^{JM}). It seems clear that a mixed version of those formalisms would also be maximally expressive. However, some important design choices are to be made; for instance, whether one permits *mixed dependencies* (those involving uncertain arguments and attacks) or not.

5 Encompassing Dynamics and Uncertainty

As argued in the introduction, there are two fundamental aspects of argumentation that are left out of AFs: the uncertainty about the relevant argumentative information (that is, which arguments and attacks should be taken into account during a debate), and the dynamics of such information. In the previous section we have discussed various ways to represent uncertainty about AFs. As to the dynamics of AFs, it is a well-studied branch of research by now; see e.g. [38, 11] for recent surveys. In this section we sketch how both ideas are to be combined. We start by presenting a well-studied case: control AFs [31], showing that their main reasoning tasks are also encodable in DL-PA. After mentioning some of its

limitations, we proceed to study an extension that combines the kind of dynamics captured by CAFs with the flexibility of cIAFs for representing uncertainty. We close the section by sketching a general theory of dynamics and uncertainty of AFs that provides conceptual tools for conducting future research.

5.1 Control AFs

Control argumentation frameworks were introduced in [31] and applied to argument-based negotiation in [32]. They represent a joint approach to uncertainty and dynamics of AFs. Regarding uncertainty, they are as expressive as rIAFs (Section 4.2). As to dynamics, they capture a parametrised version of what has been called *normal expansion* [9] at the level of each completion.

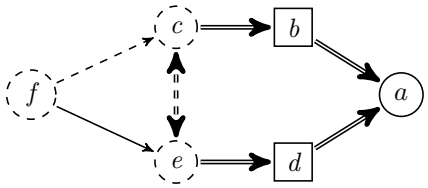
Formally, a **control argumentation framework** is a triple $\mathsf{CAF} = (F, U, C)$ where:

- $F = (A^F, R^F)$ is the *fixed part*, with $R^F \subseteq (A^F \cup A^?) \times (A^F \cup A^?)$, and both A^F and $A^?$ being two finite sets of arguments;
- $U = (A^?, (R^? \cup R^{\leftrightarrow}))$ is the *uncertain part*, where
$$R^?, R^{\leftrightarrow} \subseteq (A^F \cup A^?) \times (A^F \cup A^?)$$
and $R^{\leftrightarrow}$ is symmetric and irreflexive;[9]
- $C = (A^C, R^C)$ is the *control part*, where A^C is yet another finite set of arguments and
$$R^C \subseteq (A^C \times (A^F \cup A^? \cup A^C)) \cup ((A^F \cup A^? \cup A^C) \times A^C);$$
- A^F, $A^?$, and A^C are pairwise disjoint; and
- $R^F, R^?, R^{\leftrightarrow}$, and R^C are pairwise disjoint.

We note $\mathcal{CAF}$ the class of all control AFs.

Given a $\mathsf{CAF} = (F, U, C)$, a **control configuration** is a subset of control arguments $CFG \subseteq A^C$. Informally, each control configuration can be seen as a possible argumentative move for the proponent. The **CAF associated to** CFG is $\mathsf{CAF}_{CFG} = (F, C_{CFG}, U)$, where $C_{CFG} = (CFG, R^C \restriction_{A^F \cup A^? \cup CFG})$.

Example 4. *Let us consider the CAF* $\mathsf{CAF}_0 = (F_0, C_0, U_0)$ *where* $A_0^F = \{a\}$, $R_0^F = \{(f, e)\}$, $A_0^? = \{c, e, f\}$, $R_0^? = \{(f, c)\}$, $R_0^{\leftrightarrow} = \{(c, e), (e, c)\}$, $A_0^C = \{b, d\}$, *and* $R_0^C = \{(b, a), (d, a), (c, b), (e, d)\}$. *We represent* CAF_0 *graphically as follows:*



[9] Symmetry and irreflexivity of $R^{\leftrightarrow}$ are not assumed in the original paper [31], but appeared later on in the literature about CAFs [58, 56]. Note that both assumptions do not affect expressivity (in the sense used in Section 4.4) of CAFs.

The notion of **completion** is defined as follows for CAFs:

- $(A^F \cup A^C) \subseteq A^* \subseteq (A^F \cup A^C \cup A^?)$;
- $(R^F \cup R^C) \restriction_{A^*} \subseteq R^* \subseteq (R^F \cup R^C \cup R^? \cup R^{\leftrightarrow}) \restriction_{A^*}$; and
- for every $x, y \in A^*$: $(x, y) \in R^{\leftrightarrow}$ implies $(x, y) \in R^*$ or $(y, x) \in R^*$.

According to this definition, control arguments/attacks behave like fixed arguments/attacks once they have been communicated. Hence, the completions of CAF_0 coincide with those of rIAF_0 (Example 2), i.e., those depicted in Table 2.

Regarding CAFs, defining relevant reasoning tasks gets slightly more complicated because we have to take into account their dynamic dimension. In this context, a natural reasoning task is to find a control configuration (that is, a set of control arguments) such that a certain argument gets accepted by the opponent after the latter learns about them. As before, acceptability is then relative to quantification over completions and extensions. Here is an example:

σ-Necessary-Sceptical-Controllability (σ-NSCon)
Given: A control argumentation framework $\mathsf{CAF} = (F, U, C)$ and an argument $a \in A^F$. **Question:** Is it true that there is a configuration $CFG \subseteq A_C$ such that for every completion (A^*, R^*) of CAF_{CFG} and for every $E \in \sigma(A^*, R^*), a \in E$?

We now move on to explain how to reason about CAFs in DL-PA. Since, uncertainty-wise, control argumentation frameworks are essentially rich incomplete argumentation frameworks, the delicate part in the encoding process comes with their dynamic component, i.e., the control part.

First, given a CAF $\mathsf{CAF} = (F, U, C)$, we define its **associated valuation** as

$$\begin{aligned} v_{\mathsf{CAF}} &= v_{(A^F, R^F \cup R^C)} \\ &= \mathsf{AW}_{A^F} \cup \mathsf{ATT}_{R^F} \cup \mathsf{ATT}_{R^C} \\ &= \{\mathtt{aw}_x \mid x \in A^F\} \cup \{\mathtt{r}_{x,y} \mid (x, y) \in R^F\} \cup \{\mathtt{r}_{x,y} \mid (x, y) \in R^C\}. \end{aligned}$$

Note that v_{CAF} contains all attack variables corresponding to control attacks, but none of them appear in $(A_{v_{\mathsf{CAF}}}, R_{v_{\mathsf{CAF}}})$ since none of the control arguments has been communicated yet. This highlights the fact that in an epistemic interpretation of CAFs, the proponent knows how the opponent will perceive the attack relations regarding all communicable arguments.

To capture the dynamic component of CAF we define the following program:

$$\mathsf{control}^{\mathsf{CAF}} = \mathsf{mkTrueSome}(\mathsf{AW}_{A^C}).$$

Intuitively, $\mathsf{control}^{\mathsf{CAF}}$ nondeterministically chooses some of the possible control configurations of CAF, i.e., some subset of control arguments.

Once we have computed some control configuration, we use the same program as for rIAFs in order to compute completions:

$$\mathsf{makeComp}^{\mathsf{CAF}} = \mathsf{mkTrueSome}(\mathsf{AW}_{A^?}); \mathsf{mkTrueSome}(\mathsf{ATT}_{R^?}); \mathsf{dis}(\mathsf{ATT}_{R^\leftrightarrow}).$$

We again state a correctness result:

Proposition 10. *Let* $\mathsf{CAF} = (F, U, C)$.

- *If* $(v_{\mathsf{CAF}}, v) \in ||\mathsf{control}^{\mathsf{CAF}}; \mathsf{makeComp}^{\mathsf{CAF}}||$ *then there is a control configuration* $CFG \subseteq A^C$ *and a completion* (A^*, R^*) *of* CAF_{CFG} *such that* $(A_v, R_v) = (A^*, R^*)$.
- *For every control configuration* $CFG \subseteq A^C$ *and every* $(A^*, R^*) \in \mathsf{completions}(\mathsf{CAF}_{CFG})$ *there is a valuation* $v \in 2^{\mathtt{Prp}}$ *such that* $(v_{\mathsf{CAF}}, v) \in ||\mathsf{control}^{\mathsf{CAF}}; \mathsf{makeComp}^{\mathsf{CAF}}||$ *and* $(A_v, R_v) = (A^*, R^*)$.

We can then combine the previous programs with $\mathsf{makeExt}$ in order to reduce controllability problems to DL-PA model checking problems.

Proposition 11. *Let* $\sigma \in \{st, co, gr, pr, se, id, ea, na, stg\}$. *Let* $\mathsf{CAF} = (F, U, C)$ *and* $a \in A^F$. *Then:*

- *The answer to* σ*-NSCon with input* CAF *and* a *is yes iff* $v_{\mathsf{CAF}} \models \langle \mathsf{control}^{\mathsf{CAF}} \rangle [\mathsf{makeComp}^{\mathsf{CAF}}; \mathsf{makeExt}^{\sigma}] \mathtt{in}_a$.
- *The answer to* σ*-NCCon with input* CAF *and* a *is yes iff* $v_{\mathsf{CAF}} \models \langle \mathsf{control}^{\mathsf{CAF}} \rangle [\mathsf{makeComp}^{\mathsf{CAF}}] \langle \mathsf{makeExt}^{\sigma} \rangle \mathtt{in}_a$.
- *The answer to* σ*-PCCon with input* CAF *and* a *is yes iff* $v_{\mathsf{CAF}} \models \langle \mathsf{control}^{\mathsf{CAF}}; \mathsf{makeComp}^{\mathsf{CAF}}; \mathsf{makeExt}^{\sigma} \rangle \mathtt{in}_a$.
- *The answer to* σ*-PSCon with input* CAF *and* a *is yes iff* $v_{\mathsf{CAF}} \models \langle \mathsf{control}^{\mathsf{CAF}}; \mathsf{makeComp}^{\mathsf{CAF}} \rangle [\mathsf{makeExt}^{\sigma}] \mathtt{in}_a$.

We close this section by highlighting and making precise two of the main modelling limitations of CAFs that we have already mentioned. Regarding uncertainty, they cannot go further than rIAFs. As to dynamics, the form of communication that they model assumes that uncertainty does not increase. More formally:

Remark 1. $\mathcal{CAF} \equiv \mathcal{RIAF}$.

Remark 2. *Let* $\mathsf{CAF} = (F, U, C)$ *and let* $CFG, CFG' \subseteq A^C$. *Then* $|\mathsf{completions}(\mathsf{CAF}_{CFG})| = |\mathsf{completions}(\mathsf{CAF}_{CFG'})|$.

5.2 Control Constrained Incomplete AFs

One of the advantages of the approaches presented so far is that they can be freely combined. Moreover, the encoding of these formalisms in DL-PA can easily be extrapolated to combined classes of structures. In this subsection, we give evidence of such flexibility by mixing the kind of uncertainty modelled by cIAFs with the kind of dynamics modelled by CAFs. Formally, a **control constrained incomplete AF** (CcIAF) is a tuple $\mathsf{CcIAF} = (A^C, R^C, A^S, \varphi)$ with:

- A^C (control arguments) and A^S (static arguments) are disjoint,
- $R^C \subseteq (A^C \times (A^C \cup A^S)) \cup ((A^C \cup A^S) \times A^C)$,
- φ is a Boolean formula over $\mathsf{AW}_{A^S} \cup \mathsf{ATT}_{A^S \times A^S}$.

Given $\mathsf{CcIAF} = (A^C, R^C, A^S, \varphi)$, the pair (A^S, φ) is its **underlying cIAF**.

The notion of completion is adapted to CcIAFs by combining the intuition behind CAFs and cIAFs, i.e., a completion of (A^C, R^C, A^S, φ) is any AF (A^*, R^*) such that:

- $A^* = A^C \cup A'$,
- $R^* = (R^C \cup R') \restriction_{A^*}$,
- (A', R') is a completion of the underlying cIAF.

The notion of control configuration is also straightforwardly adapted to our new class of structures. More in detail, a **control configuration** of $\mathsf{CcIAF} = (A^C, R^C, A^S, \varphi)$ is any $CFG \subseteq A^C$. The CcIAF associated to CFG is defined as $\mathsf{CcIAF}_{CFG} = (CFG, R^C \restriction_{CFG}, A^S, \varphi)$. We can extrapolate controllability problems to CcIAFs:

σ-Necessary-Sceptical-Controllability (σ-NSCon)
Given: A control constrained incomplete argumentation framework $\mathsf{CcIAF} = (A^C, R^C, A^S, \varphi)$ and an argument $a \in A^S$ s.th. $\models \varphi \rightarrow \mathtt{aw}_a$.
Question: Is it true that there is a configuration $CFG \subseteq A_C$ such that for every completion (A^*, R^*) of CcIAF_{CFG} and for every $E \in \sigma(A^*, R^*), a \in E$?

Regarding the DL-PA encoding of the reasoning problems that we have just defined, we start by assigning to each CcIAF its **associated valuation**, in a similar way to what we did both with CAFs and with cIAFs:

$$\begin{aligned} v_{\mathsf{CcIAF}} &= \mathsf{ATT}_{R^C} \\ &= \{\mathtt{r}_{x,y} \mid (x,y) \in R^C\}. \end{aligned}$$

The control part of a CcIAF is encoded with the same DL-PA program as in CAFs:

$$\mathsf{control}^{\mathsf{CcIAF}} = \mathsf{mkTrueSome}(\mathsf{AW}_{A^C}).$$

Something analogous happens with the programs for computing completions, where we take over the program we used for cIAFs:

$$\mathsf{makeComp}^{\mathsf{CcIAF}} = \mathsf{vary}(\mathsf{AW}_{A^S}); \mathsf{vary}(\mathsf{ATT}_{A^S \times A^S}); \varphi?.$$

Proposition 12. *Let* $\mathsf{CcIAF} = (A^C, R^C, A^S, \varphi)$.

- *If* $(v_{\mathsf{CcIAF}}, v) \in ||\mathsf{control}^{\mathsf{CcIAF}}; \mathsf{makeComp}^{\mathsf{CcIAF}}||$, *then there is a control configuration* $CFG \subseteq A^C$ *and a completion* (A^*, R^*) *of* CcIAF_{CFG} *such that* $(A_v, R_v) = (A^*, R^*)$.
- *For every control configuration* $CFG \subseteq A^C$ *and every* $(A^*, R^*) \in \mathsf{completions}(\mathsf{CcIAF}_{CFG})$ *there is a valuation* $v \in 2^{\mathtt{Prp}_{A^S \cup A^C}}$ *such that* $(v_{\mathsf{CAF}}, v) \in ||\mathsf{control}^{\mathsf{CcIAF}}; \mathsf{makeComp}^{\mathsf{CcIAF}}||$ *and* $(A_v, R_v) = (A^*, R^*)$.

Once again, we can also reduce reasoning tasks involving CcIAFs to DL-PA model checking problems. Details are left to the reader.

5.3 Towards a General Theory

In [38], a general theory of the dynamics of abstract argumentation systems is developed. The focus of the paper is the dynamics of AFs but, as pointed out by the authors, the theory is *prima facie* applicable to other kinds of argumentation frameworks. In this subsection we apply their categorisation to the formalisms for representing qualitative uncertainty about AFs studied in Section 4. At the same time, we show how DL-PA works as a good logical candidate for formalising many parts of this general theory.

Structural constraints. According to [38], there are different kinds of constraints that one might want to enforce in an argumentation system. The first kind of constraint is concerned with the structure of an AF. [38] distinguishes between *elementary* and *global* structural constraints. The former are defined directly on the components of the framework (adding/removing some arguments/attacks); the latter require some property that the output AF must satisfy, e.g., being odd-loop-free, acyclic, etc. Both kinds of constraints make perfect sense in the kind of structures that we have studied in Section 4. Interestingly, the richer nature of these formalisms allows for further distinctions.

Elementary structural constraints. While in AFs elementary constraints amount to addition/removal of arguments/attacks (or combinations of these, as in the case of AF expansions [9]), we can perform more subtle actions in argumentation frameworks with qualitative uncertainty. Let us illustrate some of these actions for the case of IAFs.

Settling uncertain arguments/attacks. In a debate, an agent may want to promote the epistemic status of an uncertain argument/attack by "settling it". Formally, and restricting our attention to arguments and incomplete AFs,

given $\mathsf{IAF} = (A^F, R^F, A^?, R^?)$ and $a \in A^?$, define the partial function $\mathsf{settle} : (\mathcal{IAF} \times \mathcal{U}) \to \mathcal{IAF}$ by:

$$\mathsf{settle}(\mathsf{IAF}, a) = (A^F \cup \{a\}, A \setminus \{a\}, R^F, R^?).$$

In DL-PA we can compute the completions of the resulting IAF straightforwardly:

$$\mathsf{completions}(\mathsf{settle}(\mathsf{IAF}, a)) = \{(A_v, R_v) \mid (v_{\mathsf{IAF}}, v) \in ||\mathsf{makeComp}^{\mathsf{IAF}}; +\mathtt{aw}_a||\}.$$

Communicating arguments that become uncertain. Another kind of dynamics, formally modelled by moving arguments from $\mathcal{U} \setminus (A^F \cup A^?)$ to $A^?$, can be used to model situations in which argumentation takes place through a communication channel which is not fully trustworthy (say, a messaging app), so that the proponent is not sure whether the opponent received the arguments that were sent. Again, the completions of the resulting IAF can be easily computed within DL-PA, and the same ideas can be applied to communicating attacks instead of arguments.

Communicating arguments with uncertain effects. Yet another kind of action is to communicate arguments whose effects on the opponent's framework are not known. For instance, and within the context of CAFs, one can relax their definition by extending the domain and range of $R^?$ or $R^{\leftrightarrow}$ so as to include A^C.

Belief change methods for logical structures. As for cIAFs (and this applies also to cIAFs^{JM}), elementary changes amount to either augmenting/shrinking the domain A or, more interestingly, changing the epistemic constraint φ. Regarding the latter, methods imported from the belief change literature can be used; for instance, if a new piece of information ψ that is inconsistent with φ is to be added, one could do so by means of an AGM belief revision operator [1]. In that respect, DL-PA has been shown useful to capture belief change operators [47], and these have been applied in turn to AFs [35, 36].

Types of elementary structural changes. Several interesting criteria can be applied to provide a classification of elementary structural changes within frameworks for arguing with qualitative uncertainty. Let us just point out a couple of them. Regarding awareness of arguments, we can distinguish between *internal actions*, *argument-gaining actions*, and *argument-losing actions*. Informally, as the outcome of an internal action, agents neither become aware nor unaware of any new argument.[10] A bit more formally, and restricting our attention to IAFs, we say that the action transforming $\mathsf{IAF}_0 = (A_0^F, A_0^?, R_0^F, R_0^?)$ into $\mathsf{IAF}_1 = (A_1^F, A_1^?, R_1^F, R_1^?)$ is internal whenever $A_0^F \cup A_0^? = A_1^F \cup A_1^?$. For example, the partial function $\mathsf{settle} : (\mathcal{IAF} \times \mathcal{U}) \to \mathcal{IAF}$ defined above is clearly internal.

[10] However, they might change the epistemic status of arguments they are aware of.

Argument-gaining actions formally amount to requiring that $A_0^F \cup A_0^? \subset A_1^F \cup A_1^?$, so that the agent becomes aware of at least one novel argument. The action of communicating uncertain arguments that we have explained above is an example of an argument-gaining action. Finally, in argument-losing actions we have that $A_1^F \cup A_1^? \subset A_0^F \cup A_0^?$, that is, the agent has become unaware of at least one argument.[11] A second criterion for categorizing elementary structural changes would be measuring their impact on the number completions, since, intuitively, the more completions we have, the more uncertainty the formalised agent is dealing with. As examples, the settle function described above always results in a reduction of the number of completions; computing control configurations of CAFs keeps the number of completions constant (see Remark 2 for a precise formulation); and communicating uncertain arguments (also described above) increases the number of completions.

Global structural constraints. In AFs, global structural constraints amount to things like obtaining an acyclic graph, or an odd-loop-free graph, etc. These constraints are motivated by the appealing mathematical properties implied by them. For instance, it is known since [39] that in acyclic AFs, all the four classic semantics collapse. Interestingly, DL-PA can capture many of these constraints. For example, in [36] polynomial formulas characterising the existence of odd- and even-length-loops are constructed. When extrapolated to the more complex formalisms studied here, global constraints can be required either possibly (that is, in at least one completion) or necessarily (in all of them). Furthermore, DL-PA can be used to check if the constraint is satisfied possibly or necessarily. To be more precise, and focusing on IAFs for simplicity: let φ be the formula characterising a targeted global constraint and let IAF be an IAF; then we have that φ is satisfied possibly (resp. necessary) iff $v_{\mathsf{IAF}} \models \langle \mathsf{makeComp}^{\mathsf{IAF}} \rangle \varphi$ (resp. iff $v_{\mathsf{IAF}} \models [\mathsf{makeComp}^{\mathsf{IAF}}]\varphi$). In AFs, global constraints are usually enforced through elementary changes (those described above). Once again, this relation can be studied in DL-PA. For instance, if we want to know if a global constraint φ is possibly enforced in IAF as the result of settling $a \in A^?$, it is enough to model-check whether $v_{\mathsf{IAF}} \models \langle \mathsf{makeComp}^{\mathsf{IAF}}; +\mathtt{aw}_a \rangle \varphi$ holds.

Acceptability constraints. The second kind of constraint distinguished by [38] is concerned with the output of the argument evaluation process in an argumentation system. The distinction elementary/global applies here, too. When restricted to AFs, one might want to enforce a set of arguments to be part of (or equal to) at least one (or every) extension; this is an elementary acceptability constraint. This kind of enforcement is probably the most studied throughout the literature on abstract argumentation, since the work of [9], as it has

[11] The last type of action connects with a recent thoughtful study of the notion of *forgetting an argument* in the context of AFs [12]. Interestingly, we can capture within IAFs the distinction, made in [33], between *forgetting-as-becoming-unaware* (moving an argument from $A^F \cup A^?$ to $\mathcal{U} \setminus (A^F \cup A^?)$), and *forgetting-as-becoming-ignorant* (moving an argument from A^F to $A^?$).

a clear informal counterpart in real-life argumentation: persuading an opponent basically amounts to enforcing some targeted arguments. Furthermore, and from a more technical perspective, one might also want to enforce some kind of global acceptability constraint: controlling the cardinality of the set of extensions, its structure, etc. Again, qualitative uncertainty introduces a new layer of quantification: acceptability enforcement can be pursued possibly (i.e., in at least one completion) or necessarily (in all of them). Just as it happens with AFs, acceptability constraints are usually enforced through a (combination of) structural changes such as the ones we have described above. As an example, the reasoning tasks of both CAFs and CcIAFs are a way of enforcing a possible/necessary acceptability constraint through the performance of a combination of elementary structural changes that do not increase uncertainty (activating control arguments).

Semantic constraints. Finally, the third kind of constraint distinguished by [38] affects the semantics that has been chosen to evaluate arguments. Informally, enforcing a semantic constraint amounts to a change in the standards applied within the argument evaluation process. To this respect, not only the parameter σ can be switched to σ', but one could also move from credulous to sceptical acceptability, and *vice versa*. Just as before, the formalisms studied in Section 4 introduce an additional layer of quantification to be taken into account when formulating semantic constraints: we can move from a 'possible' semantics (arguments should be accepted in at least one completion) to a 'necessary' semantics (they should be accepted in all), and backwards. As we have shown throughout the paper (e.g., in Proposition 3), the distinction between possible and necessary acceptability can be transparently captured in DL-PA.

6 Discussion, Related Work and Future Directions

We have taken the logical encoding of AFs and their extensions a step further by moving from encodings in propositional logic and *quantified Boolean formulas* (QBF) to encodings in a simple version of dynamic logic DL-PA. Approaches to argumentation reasoning problems based on SAT-solvers typically use Besnard and Doutre's encoding of AFs and their semantics in propositional logic [19], as well as its extension to QBF for semantics requiring maximality checking; see e.g. [57] for a recent such approach, and [26] for a review of approaches to abstract argumentation reasoners. Based on our work, one could use DL-PA model checkers instead of SAT-solvers in order to automatically decide the reasoning problems that we have investigated here. This would however have to await such model checkers, which for the time being do not exist yet. Alternatively, one could resort to translations from DL-PA to QBF and use solvers for the latter. This is currently pursued in the LILaC group at IRIT.

On the whole, all we have done in DL-PA can as well be done in equally

expressive logical frameworks like propositional logic or QBF. The advantage over the former is that (1) some semantics can be expressed more compactly in DL-PA, such as the preferred semantics: it is one level higher in the polynomial hierarchy than the other semantics and can therefore not be captured by a polynomial propositional logic formula, while a polynomial DL-PA formula is given in [36],[12] and (2) the reasoning problems can be expressed directly as DL-PA programs. The advantage over QBFs is that the DL-PA encoding of reasoning problems by means of programs is more natural than the rather complex QBF encodings that one can find in the literature. Actually, most of the works on arguing with qualitative uncertainty use QBF encodings and algorithms for determining the complexity of associated reasoning tasks (see e.g. [13] or [58]). All advantages already pointed out by [36] of using DL-PA instead of QBF for encoding argumentative semantics are preserved by our encodings. In particular, "extension construction programs such as $\mathsf{makeExt}^{\sigma}$ capture things in a more general, flexible and natural way than a QBF encoding".

Getting closer to a theorem proving approach. Our encoding of formalisms for arguing with qualitative uncertainty can be qualified as *hybrid*, since it combines some previous semantic reasoning with reasoning inside DL-PA. For instance, in order to compute the completions of an IAF, one first needs to find its associated valuation (which is reasoning outside the logic, using semantic objects), then has to write down the makeComp program, and finally one reasons in DL-PA to find the makeComp-successors of the associated valuation. We followed this hybrid method because we found intuitive the identification of directed graphs with propositional valuations over $\mathtt{Prp}$. However, we can adopt results from [35, 37, 36] to get a more homogeneous method here. For instance, given $\mathsf{IAF} = (F, U)$, instead of computing its associated valuation we can write down a propositional formula that characterizes its fixed elements (similar to what is done in [35] for standard AFs and in our proof of Proposition 8 in the Appendix):

$$\mathsf{Th}(\mathsf{IAF}) = \bigwedge_{x \in A^F} \mathtt{aw}_x \wedge \bigwedge_{x \in \mathcal{U} \setminus A^F} \neg \mathtt{aw}_x \wedge \bigwedge_{(x,y) \in R^F} \mathtt{r}_{x,y} \bigwedge_{(x,y) \in \mathcal{U} \times \mathcal{U} \setminus R^F} \neg \mathtt{r}_{x,y}.$$

If we combine this formula with the makeComp program and the converse operator we obtain a formula whose models completely characterize the set of completions of IAF:

$$\mathsf{completions}(\mathsf{IAF}) = \{(A_v, R_v) \mid v \in ||\langle \big(\mathsf{Th}(\mathsf{IAF})?; \mathsf{makeComp}^{\mathsf{IAF}}\big)^{\smile} \rangle \top ||\}.$$

Novel contents w.r.t. our conference paper [48]. This work is based on our previous conference paper [48], which we have improved and extended in three main different directions. First, in Section 3, (i) we capture argumentation

[12] Remember that our adaptation of the formula Preferred of [36] captures preferred semantics in the more general setting of a set of background arguments $\mathcal{U}$ and is also polynomial in the size of $\mathcal{U}$.

semantics in DL-PA that had not been captured before (naive, semi-stable, stage, ideal and eager semantics), and (ii) we also adapt previous encodings to our more general setting (in particular, we adapt the encodings of complete, preferred and grounded semantics [36] to the assumption of the existence of a background universe of arguments $\mathcal{U}$, which is in turn useful for modelling both dynamics and uncertainty about AFs). Second, we discussed some closely related works that appeared since (sections 4.3.2 and 4.4). Third, we provided new results regarding the combination of dynamics and uncertainty in abstract argumentation: sections 5.2 and 5.3 are entirely new. Finally, there are also several small improvements w.r.t. the conference version, some of which are signalled throughout the paper.

Epistemic aspects of argumentation. In recent years, a few papers dealing with the combination of epistemic logic and formal argumentation have appeared. Broadly speaking, these works can be divided into two main branches: (i) those trying to provide a formalisation of the notion of justified belief based on argumentative tools such as [44, 62, 63, 61, 21, 22]; and (ii) those using epistemic models for reasoning about uncertain AFs such as [60, 59, 49, 50]. Clearly, the second one is strongly connected—both conceptually and technically—to some of the ideas presented here. The main first difference is that the formalisms used in this paper lack a tool for capturing higher-order epistemic attitudes, that is, a tool capable of representing not only what an agent thinks of her opponent's argumentative situation (her AF), but also about what the agent thinks that her opponent thinks about the agent's argumentative situation, and so on. This is an important point, since this kind of mental attitude has been successfully employed under the name of *recursive opponent models* within the sub-field of strategic argumentation (see, e.g., [64]). However, the incorporation of this type of multiple agency together with a full dynamic toolkit would mean to replace DL-PA by the strong modelling power of dynamic epistemic logic [34]. This comes at the price of a blow-up in the computational complexity of the associated reasoning tasks. One might however follow [27] and employ lightweight epistemic logics where disjunctions in the scope of epistemic operators are forbidden. That would represent a compromise between modelling multiple agency/dynamics, on the one side, and modelling uncertainty, on the other side, since any form of uncertainty that goes beyond IAFs (see Figure 2) would have to be excluded from this approach. A second important difference is that, unlike epistemic logic, none of the formalisms studied in this paper allow for modelling *the actual world*, i.e., what is true independently of what the formalised agent thinks. This notion is in turn needed for distinguishing between *knowledge* (which is usually required to be true) and *belief* (which is often merely required to be consistent). However, this limitation seems easier to be overcome: It suffices to augment IAFs (and their extensions) with a *distinguished completion*, informally accounting for what the actual AF is.

Further semantics. Yet another direction for future work is extending our DL-PA encoding to semantics that have not been considered in Section 3. A specially interesting case is the recently introduced family of weak admissibility-based semantics [10], since most of the associated reasoning tasks have been shown to be PSPACE-complete, matching the complexity of the DL-PA model checking problem [5].

An alternative notion of expressivity. In a very recent paper [2], Alfano et al. invented a rewriting technique in order to reduce general IAFs to their strict subclasses arg-IAFs and att-IAF, and yet to a proper subclass of arg-IAFs.[13] More concretely, they show [2, Theorem 7] that the completions of the rewritten incomplete AF can be mapped (through another transformation) to the completions of the original one. They moreover claim that "This result entails that arg-IAFs (resp. farg-IAF, att-IAF) have the same expressivity of general IAFs, though arg-IAFs (resp. farg-IAF, att-IAF) have a simpler structure". This clearly conflicts with the expressivity map that we provided in Proposition 9 and Figure 2, which is based on the notion of expressivity first introduced in [53] and later used in [52, 48]. Although a detailed comparison of both notions of expressivity is out of the scope of this discussion, we would just like to mention that the one employed here seems more useful for intuitive modelling purposes (i.e., to find out what kind of situations the formalised agent is able to represent in her mind), while Alfano et. al's seems more interesting from a technical perspective (actually, it is used to extend complexity results regarding IAFs to their proper subclasses). Be as it may, the work done in [2] opens an interesting research question: can the rewriting technique be extended to more expressive formalisms (in our sense), such as rIAFs or cIAFs?

Funding

The research activity of both authors is partially supported by the EU ICT-48 2020 project TAILOR (No. 952215). Part of this research was carried on when Antonio Yuste was employed by the University of Málaga through a Post-doctoral contract type A.3.1. of *Plan Propio de Investigación, Transferencia y Divulgación Científica*.

Acknowledgements

We thank Sylvie Doutre and Jean-Guy Mailly for previous discussions on the topic of this paper, specially for triggering the idea of constrained incomplete argumentation frameworks.

[13] The so-called *fact-uncertain AFs* (farg-IAFs), which are argument-incomplete AFs where all uncertain arguments are not attacked.

References

[1] Carlos Alchourrón, Peter Gärdenfors, and David Makinson. On the logic of theory change: Partial meet contraction and revision functions. *J. of Symbolic Logic*, 50:510–530, 1985.

[2] Gianvincenzo Alfano, Sergio Greco, Francesco Parisi, and Irina Trubitsyna. Incomplete argumentation frameworks: Properties and complexity. In *The Thirty-Sixth AAAI Conference on Artificial Intelligence*, pages 5451–5460. AAAI Press, 2022.

[3] Leila Amgoud and Srdjan Vesic. A new approach for preference-based argumentation frameworks. *Annals of Mathematics and Artificial Intelligence*, 63(2):149–183, 2011.

[4] Katie Atkinson, Pietro Baroni, Massimiliano Giacomin, Anthony Hunter, Henry Prakken, Chris Reed, Guillermo Simari, Matthias Thimm, and Serena Villata. Towards artificial argumentation. *AI Magazine*, 38(3):25–36, 2017.

[5] Philippe Balbiani, Andreas Herzig, François Schwarzentruber, and Nicolas Troquard. DL-PA and DCL-PC: model checking and satisfiability problem are indeed in PSPACE. *CoRR*, abs/1411.7825, 2014.

[6] Philippe Balbiani, Andreas Herzig, and Nicolas Troquard. Dynamic logic of propositional assignments: a well-behaved variant of PDL. In *28th Annual ACM/IEEE Symposium on Logic in Computer Science*, pages 143–152. IEEE, 2013.

[7] Pietro Baroni, Martin Caminada, and Massimiliano Giacomin. Abstract argumentation frameworks and their semantics. In Pietro Baroni, Dov M Gabbay, Massimiliano Giacomin, and Leendert van der Torre, editors, *Handbook of Formal Argumentation*, pages 159–236. College Publications, 2018.

[8] Pietro Baroni, Federico Cerutti, Massimiliano Giacomin, and Giovanni Guida. Encompassing attacks to attacks in abstract argumentation frameworks. In Claudio Sossai and Gaetano Chemello, editors, *European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty*, pages 83–94. Springer, 2009.

[9] Ringo Baumann and Gerhard Brewka. Expanding argumentation frameworks: Enforcing and monotonicity results. In Pietro Baroni, Federico Cerutti, Massimiliano Giacomin, and Guillermo Ricardo Simari, editors, *Proceedings of the COMMA 2010*, volume 216 of *Frontiers in AI and Applications*, pages 75–86. IOS Press, 2010.

[10] Ringo Baumann, Gerhard Brewka, and Markus Ulbricht. Revisiting the foundations of abstract argumentation - semantics based on weak admissibility and weak defense. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020*, pages 2742–2749. AAAI Press, 2020.

[11] Ringo Baumann, Sylvie Doutre, Jean-Guy Mailly, and Johannes P Wallner. Enforcement in formal argumentation. *Journal of Applied Logics*, 2631(6):1623, 2021.

[12] Ringo Baumann, Dov M. Gabbay, and Odinaldo Rodrigues. Forgetting an argument. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020*, pages 2750–2757. AAAI Press, 2020.

[13] Dorothea Baumeister, Matti Järvisalo, Daniel Neugebauer, Andreas Niskanen, and Jörg Rothe. Acceptance in incomplete argumentation frameworks. *Artificial Intelligence*, 295:103470, 2021.

[14] Dorothea Baumeister, Daniel Neugebauer, and Jörg Rothe. Credulous and skeptical acceptance in incomplete argumentation frameworks. In *Proceedings of the COMMA 2018*, volume 305 of *Frontiers in AI and Applications*, pages 181–192. IOS Press, 2018.

[15] Dorothea Baumeister, Daniel Neugebauer, Jörg Rothe, and Hilmar Schadrack. Complexity of verification in incomplete argumentation frameworks. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18)*, pages 1753–1760. AAAI Press, 2018.

[16] Dorothea Baumeister, Daniel Neugebauer, Jörg Rothe, and Hilmar Schadrack. Verification in incomplete argumentation frameworks. *Artificial Intelligence*, 264:1–26, 2018.

[17] Trevor JM Bench-Capon and Paul E Dunne. Argumentation in artificial intelligence. *Artificial Intelligence*, 171(10-15):619–641, 2007.

[18] Philippe Besnard, Claudette Cayrol, and Marie-Christine Lagasquie-Schiex. Logical theories and abstract argumentation: A survey of existing works. *Argument & Computation*, 11(1-2):41–102, 2020.

[19] Philippe Besnard and Sylvie Doutre. Checking the acceptability of a set of arguments. In J.P. Delgrande and T. Schaub, editors, *10th International Workshop on Non-Monotonic Reasoning*, pages 59–64, 2004.

[20] Stefano Bistarelli and Francesco Santini. ConArg: A constraint-based computational framework for argumentation systems. In *IEEE 23rd International Conference on Tools with Artificial Intelligence, ICTAI 2011*, pages 605–612. IEEE Computer Society, 2011.

[21] Alfredo Burrieza and Antonio Yuste-Ginel. Basic beliefs and argument-based beliefs in awareness epistemic logic with structured arguments. In Henry Prakken, Stefano Bistarelli, Francesco Santini, and Carlo Taticchi, editors, *Proceedings of the COMMA 2020*, volume 326 of *Frontiers in AI and Applications*, pages 123–134. IOS Press, 2020.

[22] Alfredo Burrieza and Antonio Yuste-Ginel. An awareness epistemic framework for belief, argumentation and their dynamics. In Joseph Y. Halpern and Andrés Perea, editors, *Proceedings Eighteenth Conference on Theoretical Aspects of Rationality and Knowledge*, volume 335 of *EPTCS*, pages 69–83, 2021.

[23] Martin WA Caminada, Walter A Carnielli, and Paul E Dunne. Semi-stable semantics. *Journal of Logic and Computation*, 22(5):1207–1254, 2012.

[24] Claudette Cayrol, Caroline Devred, and Marie-Christine Lagasquie-Schiex. Handling ignorance in argumentation: Semantics of partial argumentation frameworks. In Khaled Mellouli, editor, *European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty 2007*, volume 4724 of *LNCS*, pages 259–270. Springer, 2007.

[25] Claudette Cayrol and Marie-Christine Lagasquie-Schiex. On the acceptability of arguments in bipolar argumentation frameworks. In Lluís Godo, editor, *European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, volume 3571 of *LNCS*, pages 378–389. Springer, 2005.

[26] Federico Cerutti, Sarah A Gaggl, Matthias Thimm, and Johannes Wallner. Foundations of implementations for formal argumentation. *IfCoLog Journal of Logics and their Applications*, 4(8):2623–2705, 2017.

[27] Martin C. Cooper, Andreas Herzig, Faustine Maffre, Frédéric Maris, Elise Perrotin, and Pierre Régnier. A lightweight epistemic logic and its application to planning. *Artificial Intelligence*, 298:103437, 2021.

[28] Sylvie Coste-Marquis, Caroline Devred, Sébastien Konieczny, Marie-Christine Lagasquie-Schiex, and Pierre Marquis. On the merging of Dung's argumentation systems. *Artificial Intelligence*, 171(10-15):730–753, 2007.

[29] Sylvie Coste-Marquis, Caroline Devred, and Pierre Marquis. Constrained argumentation frameworks. In P. Doherty, J. Mylopoulos, and C. Welty, editors, *Proceedings of the Tenth International Conference on Principles of Knowledge Representation and Reasoning*, page 112–122. AAAI Press, 2006.

[30] Marcos Cramer and Leon van der Torre. SCF2-an argumentation semantics for rational human judgments on argument acceptability. In C. Beierle, M. Ragni, F. Stolzenburg, and M. Thimm, editors, *Proceedings of the 8th Workshop on Dynamics of Knowledge and Belief*, pages 24–35. CEUR-WS.org, 2019.

[31] Yannis Dimopoulos, Jean-Guy Mailly, and Pavlos Moraitis. Control argumentation frameworks. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 4678–4685. AAAI Press, 2018.

[32] Yannis Dimopoulos, Jean-Guy Mailly, and Pavlos Moraitis. Argumentation-based negotiation with incomplete opponent profiles. In N. Agmon, M. E. Taylor, E. Elkind, and M. Veloso, editors, *18th International Conference on Autonomous Agents and MultiAgent Systems AAMAS 2019*, page 1252–1260, 2019.

[33] Hans P. van Ditmarsch, Andreas Herzig, Jérôme Lang, and Pierre Marquis. Introspective forgetting. *Synthese*, 169(2):405–423, 2009.

[34] Hans P. van Ditmarsch, Wiebe van der Hoek, and Barteld Kooi. *Dynamic Epistemic Logic*. Springer, 2007.

[35] Sylvie Doutre, Andreas Herzig, and Laurent Perrussel. A dynamic logic framework for abstract argumentation. In C. Baral, G. De Giacomo, and T. Eiter, editors, *Fourteenth International Conference on the Principles of Knowledge Representation and Reasoning*. AAAI Press, 2014.

[36] Sylvie Doutre, Andreas Herzig, and Laurent Perrussel. Abstract argumentation in dynamic logic: Representation, reasoning and change. In Beishui Liao, Thomas Ågotnes, and Yi N. Wang, editors, *Dynamics, Uncertainty and Reasoning*, pages 153–185. Springer, 2019.

[37] Sylvie Doutre, Faustine Maffre, and Peter McBurney. A dynamic logic framework for abstract argumentation: adding and removing arguments. In Salem Benferhat, Karim Tabia, and Moonis Ali, editors, *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems.*, volume 10351 of *LNCS*, pages 295–305. Springer, 2017.

[38] Sylvie Doutre and Jean-Guy Mailly. Constraints and changes: a survey of abstract argumentation dynamics. *Argument & Computation*, 9:223–248, 2018.

[39] Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, 77(2):321–357, 1995.

[40] Bettina Fazzinga, Sergio Flesca, and Filippo Furfaro. Revisiting the notion of extension over incomplete abstract argumentation frameworks. In Christian Bessiere, editor, *International Joint Conference in Artificial Intelligence*, pages 1712–1718. IJCAI Organization, 7 2020.

[41] Bettina Fazzinga, Sergio Flesca, and Filippo Furfaro. Reasoning over argument-incomplete AAFs in the presence of correlations. In Zhi-Hua Zhou, editor, *International Joint Conference in Artificial Intelligence*, pages 189–195. IJCAI Organization, 2021.

[42] Bettina Fazzinga, Sergio Flesca, and Filippo Furfaro. Reasoning over Attack-incomplete AAFs in the Presence of Correlations. In M. Bienvenu, G. Lakemeyer, and E. Erdem, editors, *Proceedings of the 18th International Conference on Principles of Knowledge Representation and Reasoning*, pages 301–311, 11 2021.

[43] Davide Grossi. On the logic of argumentation theory. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems*, pages 409–416. IFAMA, 2010.

[44] Davide Grossi and Wiebe van der Hoek. Justified beliefs by justified arguments. In Chitta Baral, Giuseppe De Giacomo, and Thomas Eiter, editors, *International Conference on Principles of Knowledge Representation and Reasoning*. AAAI Press, 2014.

[45] Mathieu Guillaume, Marcos Cramer, Leendert van der Torre, and Christine Schiltz. Reasoning on conflicting information: An empirical study of formal argumentation. *PloS one*, 17(8):e0273225, 2022.

[46] David Harel, Dexter Kozen, and Jerzy Tiuryn. *Dynamic Logic*. MIT Press, 2000.

[47] Andreas Herzig. Belief change operations: a short history of nearly everything, told in dynamic logic of propositional assignments. In Chitta Baral and Giuseppe De Giacomo, editors, *International Conference on Principles of Knowledge Representation and Reasoning*. AAAI Press, 2014.

[48] Andreas Herzig and Antonio Yuste-Ginel. Abstract argumentation with qualitative uncertainty: An analysis in dynamic logic. In Pietro Baroni, Christoph Benzmüller, and Yi N. Wáng, editors, *Logic and Argumentation*, volume 13040 of *LNCS*, pages 190–208. Springer, 2021.

[49] Andreas Herzig and Antonio Yuste-Ginel. Multi-agent abstract argumentation frameworks with incomplete knowledge of attacks. In Zhi-Hua Zhou, editor, *International Joint Conference in Artificial Intelligence*, pages 1922–1928. IJCAI Organization, 2021.

[50] Andreas Herzig and Antonio Yuste-Ginel. On the Epistemic Logic of Incomplete Argumentation Frameworks. In M. Bienvenu, G. Lakemeyer, and E. Erdem, editors, *Proceedings of the 18th International Conference on Principles of Knowledge Representation and Reasoning*, pages 681–685, 11 2021.

[51] Anthony Hunter, Sylwia Polberg, Nico Potyka, Tjitze Rienstra, and Matthias Thimm. Probabilistic argumentation: A survey. In Dov Gabbay, Massimiliano Giacomin, Guillermo R. Simari, and Matthias Thimm, editors, *Handbook of Formal Argumentation*, volume 2, pages 159–236. College Publications, 2021.

[52] Jean-Guy Mailly. Yes, no, maybe, I don't know: Complexity and application of abstract argumentation with incomplete knowledge. *Argument & Computation*, 13(3):291–324.

[53] Jean-Guy Mailly. A note on rich incomplete argumentation frameworks. *arXiv preprint arXiv:2009.04869*, 2020.

[54] Jean-Guy Mailly. Constrained incomplete argumentation frameworks. In Jiřina Vejnarová and Nic Wilson, editors, *Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, volume 12897 of *LNCS*, pages 103–116. Springer, 2021.

[55] Sanjay Modgil, Francesca Toni, Floris Bex, Ivan Bratko, Carlos I. Chesñevar, Wolfgang Dvořák, Marcelo A. Falappa, Xiuyi Fan, Sarah Alice Gaggl, Alejandro J. García, María P. González, Thomas F. Gordon, João Leite, Martin Možina, Chris Reed, Guillermo R. Simari, Stefan Szeider, Paolo Torroni, and Stefan Woltran. The added value of argumentation. In *Agreement Technologies*, pages 357–403. Springer, 2013.

[56] Andreas Niskanen. *Computational Approaches to Dynamics and Uncertainty in Abstract Argumentation*. PhD thesis, Helsingin yliopisto, 2020.

[57] Andreas Niskanen and Matti Järvisalo. μ-toksia: An efficient abstract argumentation reasoner. In Diego Calvanese, Esra Erdem, and Michael Thielscher, editors, *Proceedings of the 17th International Conference on Principles of Knowledge Representation and Reasoning*, pages 800–804, 2020.

[58] Andreas Niskanen, Daniel Neugebauer, Matti Järvisalo, et al. Controllability of control argumentation frameworks. In Christian Bessiere, editor, *International Joint Conference in Artificial Intelligence*. IJCAI Organization, 2020.

[59] Carlo Proietti and Antonio Yuste-Ginel. Dynamic epistemic logics for abstract argumentation. *Synthese*, 199(3):8641–8700, 2021.

[60] François Schwarzentruber, Srdjan Vesic, and Tjitze Rienstra. Building an epistemic logic for argumentation. In Luis Fariñas del Cerro, Andreas Herzig, and Jérôme Mengin, editors, *Logics in Artificial Intelligence*, volume 7519 of *LNCS*, pages 359–371. Springer, 2012.

[61] Chenwei Shi, Sonja Smets, and Fernando R Velázquez-Quesada. Logic of justified beliefs based on argumentation. *Erkenntnis*, pages 1–37, 2021.

[62] Chenwei Shi, Sonja Smets, and Fernando R. Velázquez-Quesada. Argument-based belief in topological structures. In Jérôme Lang, editor, *Proceedings Sixteenth Conference on Theoretical Aspects of Rationality and Knowledge*, EPTCS. Open Publishing Association, 2017.

[63] Chenwei Shi, Sonja Smets, and Fernando R. Velázquez-Quesada. Beliefs supported by binary arguments. *Journal of Applied Non-Classical Logics*, 28(2-3):165–188, 2018.

[64] Matthias Thimm. Strategic argumentation in multi-agent systems. *KI-Künstliche Intelligenz*, 28(3):159–168, 2014.

Appendix

In this appendix, we provide selected proofs and proof sketches for the results found throughout the paper.

[**Theorem 1**]

Proof. We shall just prove the first bullet for $\sigma = se$. The other cases are simpler and follow similar arguments, while the second bullet follows easily from the first one and the meaning of $\mathsf{makeExt}$. Let us first state without proof a couple of needed lemmas

Lemma 1. $(v, v_1) \in ||\mathsf{copy}(\mathsf{IN}_{\mathcal{U}}); \mathsf{makeExt}^{\sigma}||$ *implies* $\mathsf{E}_v = \{x \mid \mathtt{in}'_x \in v_1\}$ *and* $(A_v, R_v) = (A_{v_1}, R_{v_1})$.

Lemma 2. *Let* $v \subseteq \mathtt{Prp}$ *s.th.* $v \models \mathsf{Well}$ *and* $\{x \mid \mathtt{in}'_x \in v\} \subseteq \{x \mid \mathtt{aw}_x \in v\}$, *then:*

- $v \models \mathsf{IncludesCp}$ *iff* $\{x \mid \mathtt{in}'_x \in v\}^{\oplus} \subseteq \mathsf{E}_v^{\oplus}$.
- $v \models \mathsf{IncludedInCp}$ *iff* $\mathsf{E}_v^{\oplus} \subseteq \{x \mid \mathtt{in}'_x \in v\}^{\oplus}$.

($\Rightarrow$) Suppose that $v \models \mathsf{Semistable}$, which amounts to
$v \models \mathsf{Complete}$ and $v \models [\mathsf{copy}(\mathsf{IN}_{\mathcal{U}}); \mathsf{makeExt}^{co}]\,(\mathsf{IncludesCp} \rightarrow \mathsf{IncludedInCp})$. The first conjunct is equivalent, by the same item we are proving but for $\sigma = co$, to $\mathsf{E}_v \in \mathsf{co}(A_v, R_v)$. So, we just need to show that E_v has a maximal range among complete extensions. Suppose $E' \in \mathsf{co}(A_v, R_v)$. By by the same item that we are proving but for $\sigma = co$ and Lemma 1, we have that $E' = \mathsf{E}_{v_1}$ and $\mathsf{E}_v = \{x \mid \mathtt{in}'_x \in v_1\}$ for some $(v, v_1) \in ||\mathsf{copy}(\mathsf{IN}_{\mathcal{U}}); \mathsf{makeExt}^{co}||$. Suppose that $\mathsf{E}_v^{\oplus} \subseteq \mathsf{E}_{v_1}^{\oplus}$. Note that v_1 satisfies the antecedent of Lemma 2 (this is deducible from $v \models \mathsf{Complete}$ and Lemma 1). Hence, we have that $\mathsf{E}_v^{\oplus} \subseteq \mathsf{E}_{v_1}^{\oplus}$ is equivalent to $v_1 \models \mathsf{IncludesCp}$. Since we know that $v_1 \models \mathsf{IncludesCp} \rightarrow \mathsf{IncludedInCp}$, we can deduce $v_1 \models \mathsf{IncludedInCp}$, which by Lemma 2 again, amounts to $\mathsf{E}_{v_1}^{\oplus} \subseteq \mathsf{E}_v^{\oplus}$. Since E_{v_1} was an arbitrary complete extension of (A_v, R_v), we can conclude that the range of E_v is maximal among the ranges of complete extensions.

($\Leftarrow$) Suppose that $\mathsf{E}_v \in \mathsf{se}((A_v, R_v))$, which amounts to
(i) $\mathsf{E}_v \in \mathsf{co}(A_v, R_v)$ and (ii) the range of E_v is maximal among those of the complete extensions of (A_v, R_v). From (i) and the same item we are proving but for $\sigma = co$, we obtain $v \models \mathsf{Complete}$. Hence we just need to show that the second conjunct of $\mathsf{Semistable}$ is true at v. For doing so, suppose that $(v, v_1) \in ||\mathsf{copy}(\mathsf{IN}_{\mathcal{U}}); \mathsf{makeExt}^{co}||$ and $v_1 \models \mathsf{IncludesCp}$. From both lemmas and the previous assertion, we can arrive to $v_1 \models \mathsf{IncludedInCp}$.

□

[**Proposition 2**]

Proof. For the first item, suppose $(v_{\mathsf{IAF}}, v) \in ||\mathsf{makeComp}^{\mathsf{IAF}}||$. We recall from Proposition 1 that $||\mathsf{mkTrueSome}(\mathsf{P})|| = \{(v', v'') \mid v'' = v' \cup S, S \subseteq \mathsf{P}\}$ for any set of atoms P. By the semantics of the sequential composition operator ";", $(v_{\mathsf{IAF}}, v) \in ||\mathsf{makeComp}^{\mathsf{IAF}}||$ amounts to saying that $v = v_{\mathsf{IAF}} \cup \mathsf{P}$ for some $\mathsf{P} \subseteq \mathsf{AW}_{A^?} \cup \mathsf{ATT}_{R^?}$. From this statement, and applying the definition of (A_v, R_v) and the one of completion, we obtain that $(A_v, R_v) \in \mathsf{completions}(\mathsf{IAF})$.

For the second item, suppose that $(A^*, R^*) \in \mathsf{completions}(\mathsf{IAF})$, which amounts to $A^F \subseteq A^* \subseteq A^F \cup A^?$ and $R^F \restriction_{A^*} \subseteq R^* \subseteq (R^F \cup R^?) \restriction_{A^*}$. Now, remember that $v_{(A^*,R^*)} = \mathsf{AW}_{A^*} \cup \mathsf{ATT}_{R^*}$. From the two previous statements and the definition of v_{IAF}, we can deduce that the set of variables whose truth values differ from v to $v_{(A^*,R^*)}$ must be a subset of $\mathsf{AW}_{A^?} \cup \mathsf{ATT}_{R^?}$, which, as argued before, amounts to saying that $(v_{\mathsf{IAF}}, v_{(A^*,R^*)}) \in ||\mathsf{makeComp}^{\mathsf{IAF}}||$. □

[**Proposition 3**]

Sketch of proof. The result follows from the definition of the reasoning task, the correctness of each $\mathsf{makeExt}^{\sigma}$ (Theorem 1), Proposition 2, and the interpretation of [.] and ⟨.⟩ in DL-PA. □

[**Proposition 4**]

Sketch of proof. The proof is analogous to that of Proposition 2, but takes into account the observation that, when applied to the *symmetric* relation $R^{\leftrightarrow} = \{(x_1, y_1), (y_1, x_1), ..., (x_n, y_n), (y_n, x_n)\}$, every execution of $\mathsf{dis}(\mathsf{ATT}_{R^{\leftrightarrow}})$ makes true either $\mathtt{r}_{x_i,y_i}$, or $\mathtt{r}_{y_i,x_i}$ or both, for every $1 \leq i \leq n$. This ensures that the last clause of the definition of completion for rIAFs is captured in the DL-PA program $\mathsf{makeComp}^{\mathsf{rIAF}}$. □

[**Proposition 5**]

Sketch of proof. The result follows from the definition of the reasoning problem, the correctness of $\mathsf{makeExt}^{\sigma}$ (Theorem 1), the correctness of $\mathsf{makeComp}^{\mathsf{rIAF}}$ (Proposition 4), and the semantics of DL-PA. □

[**Proposition 6**]

Sketch of proof. The interpretation of $\mathsf{vary}(\mathsf{AW}_A); \mathsf{vary}(\mathsf{ATT}_{A\times A})$, when restricted to $2^{\mathtt{Prp}_A \setminus \mathsf{IN}_A}$, is actually the total relation $2^{\mathtt{Prp}_A \setminus \mathsf{IN}_A} \times 2^{\mathtt{Prp}_A \setminus \mathsf{IN}_A}$. Hence from $v_{\mathsf{cIAF}} = \emptyset$ we have an execution of $\mathsf{vary}(\mathsf{AW}_A); \mathsf{vary}(\mathsf{ATT}_{A\times A})$ that goes to *any* valuation in $2^{\mathtt{Prp}_A \setminus \mathsf{IN}_A}$. Then the execution of $\varphi?$ filters those valuations of $2^{\mathtt{Prp}_A \setminus \mathsf{IN}_A}$ that satisfy the constraint of cIAF, i.e., the set of valuations of $2^{\mathtt{Prp}_A \setminus \mathsf{IN}_A}$ representing the set of completions of cIAF.

□

[**Proposition 7**]

Sketch of proof. The result follows from from the definition of the reasoning task, the correctness of $\mathsf{makeExt}^{\sigma}$ (Theorem 1), Proposition 6, and the semantics of DL-PA. □

[**Proposition 8**]

Proof. We only have to prove $c\text{-}\mathcal{IAF} \succ \mathcal{RIAF}$ because $c\text{-}\mathcal{IAF} \succ \mathcal{IAF}$ follows from $\mathcal{RIAF} \succ \mathcal{IAF}$ [53] and the transitivity of $\succ$.

To prove $c\text{-}\mathcal{IAF} \succeq \mathcal{RIAF}$, suppose rIAF is a rIAF with $\mathsf{completions}(\mathsf{rIAF}) = \{(A_1^*, R_1^*), ..., (A_n^*, R_n^*)\}$. For every AF (A, R) defined over $\mathcal{U}$ we can write its *theory* (see e.g. [37]), that is, the propositional formula

$$\mathsf{Th}(A,R) = \bigwedge_{x \in A} \mathtt{aw}_x \wedge \bigwedge_{x \in \mathcal{U} \setminus A} \neg \mathtt{aw}_x \wedge \bigwedge_{(x,y) \in R} \mathtt{r}_{x,y} \wedge \bigwedge_{(x,y) \in \mathcal{U} \times \mathcal{U} \setminus R} \neg \mathtt{r}_{x,y}.$$

It is then easy to show that for any valuation $v \subseteq \mathtt{Prp}$, we have that $v \models \mathsf{Th}(A, R)$ iff $(A_v, R_v) = (A, R)$. Now, letting $\rho = \bigvee_{1 \leq i \leq n} \mathsf{Th}(A_i^*, R_i^*)$, we have that

$$\mathsf{completions}(\mathcal{U}, \rho) = \mathsf{completions}(\mathsf{rIAF}).$$

In order to prove that $\mathcal{RIAF} \not\succeq c\text{-}\mathcal{IAF}$ it suffices to show that the cIAF of Example 3 (called cIAF_0) cannot be expressed as a rIAF. Reasoning towards a contradiction, suppose that there is a rIAF $\mathsf{rIAF} = (A^F, R^F, A^?, R^?, R^{\leftrightarrow})$ with the same set of completions as cIAF_0. Then we would have $(a, b) \in R^F \cup R^? \cup R^{\leftrightarrow}$ (since (a, b) appears in a completion of rIAF). We show that the last statement is absurd. If $(a, b) \in R^F$ then (a, b) should appear in all completions of rIAF where a and b are present, but this is not true. If $(a, b) \in R^?$ then we reason by cases on $(b, a) \in R^F \cup R^? \cup R^{\leftrightarrow}$: the first one is impossible, since (b, a) would be in every completion where a and b appear, and that is not the case; the second one is absurd because we would have an extension with neither (a, b) nor (b, a), and this is not the case; the third one is impossible because it would imply $(a, b) \in R^{\leftrightarrow}$, but we have assumed that $(a, b) \in R^?$, and we know that $R^? \cap R^{\leftrightarrow} = \emptyset$ by definition. Finally, suppose that $(a, b) \in R^{\leftrightarrow}$, which implies $(b, a) \in R^{\leftrightarrow}$ (by symmetry of $R^{\leftrightarrow}$), which is impossible because we would have a completion containing both (a, b) and (b, a), but this is not the case. □

[**Proposition 10**]

Sketch of proof. The proof is analogous to those of propositions 2 and 4. The essential difference lies in the fact that the previous execution of $\mathsf{control}^{\mathsf{CAF}}$ is needed to nondeterministically choose a control configuration of CAF. Also, note that $\mathsf{ATT}_{R^C} \subseteq v_{\mathsf{CAF}}$ is essential for obtaining the needed control attacks in the corresponding completion. □

[**Proposition 11**]

Sketch of proof. The result follows from the definition of the reasoning task, the correctness of $\mathsf{makeExt}^{\sigma}$ (Theorem 1), Proposition 10, and the semantics of DL-PA. □