Computing Stable and Preferred Extensions of Dynamic Bipolar Argumentation Frameworks

Gianvincenzo Alfano, Sergio Greco, and Francesco Parisi

Department of Informatics, Modeling, Electronics and System Engineering,
University of Calabria, Italy,
email:{g.alfano, greco, fparisi}@dimes.unical.it

Abstract. Bipolar argumentation frameworks (BAFs) extend Dung's argumentation frameworks to explicitly represent the notion of support between arguments, in addition to that of attack. BAFs can be profitably used to model disputes between two or more agents, with the aim of deciding the sets of arguments that should be accepted to support a point of view in a discussion. However, since new arguments, attacks, and supports are often introduced to take into account new available knowledge, BAFs as well as the set of accepted arguments (under a given semantics) change over the time.
In this paper we tackle the problem of efficiently recomputing sets of accepted arguments of dynamic BAFs (under the preferred and stable semantics). In particular, focusing on a deductive interpretation of support, we introduce an incremental algorithm that, given an initial BAF, an initial extension for it, and an update, computes an extension of the updated BAF.
The experiments show that our technique is faster than computing an extension of the updated BAF from scratch.

1 Introduction

Argumentation has emerged as one of the central fields in Artificial Intelligence [12, 44, 6, 42]. In particular, Dung's abstract argumentation framework [25] is a simple, yet powerful formalism for modelling disputes between two or more agents. The formal meaning of an argumentation framework is given in terms of argumentation semantics, which intuitively tell us the sets of arguments (called *extensions*) that should be accepted to support a point of view in a discussion.

Bipolar argumentation frameworks (BAFs) are an interesting extension of the Dung's frameworks, which allow two kinds of interactions between arguments to be modeled: the *attack* relation (as in Dung's argumentation frameworks) and the *support* relation. Several interpretations of the notion of support have been proposed in the literature [4, 19–21, 14, 45] (see [24] for a comprehensive survey). In this paper, we focus on *deductive* support [14, 45] which is intended to capture the following intuition: if argument a supports argument b then the acceptance of a implies the acceptance of b, and the non-acceptance of b implies the non-acceptance of a.

Although most research in argumentation focused on static frameworks (i.e., frameworks not changing over the time), BAFs are often used to model dynamic systems [8,

31, 41, 7, 23]. In fact, usually a BAF represents a temporary situation, and new arguments, attacks, and supports can be added/retracted to take into account new available knowledge. For instance, for disputes among users of online social networks [40, 2], arguments, attacks, and supports are continuously added/retracted by users to express their point of view in response to the last move made by the adversaries (often disclosing as few arguments/attacks as possible).

However, the definition of evaluation algorithms for dynamic BAFs and the analysis of the computational complexity taking into account such dynamic aspects have been mostly neglected, whereas in these situations incremental computation techniques could greatly improve performance. Recently, focusing on Dung's AFs, in [1] we have proposed a technique for incrementally computing extensions for dynamic AFs. That is, given an AF, an initial extension for it, and an update, we devised an efficient technique for computing an extension of the updated AF.

In this paper, we show that the technique proposed in [1] can be profitably used to compute extensions of dynamic BAFs. Thus, here we address the following problem: given a BAF $\mathcal{B}_0$, an initial extension E_0 for it, and an update u, determine an extension E of the updated BAF $u(\mathcal{B}_0)$, which is obtained from $\mathcal{B}_0$ by applying update u.

We make the following contributions:

1) We identify early-termination conditions for checking whether a given extension for an initial BAF is still an extension for the updated BAF. When these conditions hold, we do not need to recompute an extension of the updated BAF.
2) We build on the meta-argumentation approach proposed in [14] to define a reduction of the problem of determining an extension of an updated BAF to that of determining an extension of a corresponding updated Dung's argumentation framework.
3) We define an incremental algorithm for computing extensions of dynamic BAFs by leveraging on the incremental technique proposed in [1].
4) We performed an experimental analysis showing that our incremental approach outperforms the computation from scratch, where the fastest solvers from the International Competition on Computational Models of Argumentation (ICCMA) [1] taking as input the Dung's argumentation frameworks resulting from the transformation of item 2) are used.

2 Bipolar Argumentation Frameworks

We assume the existence of a set Arg of *arguments*. An *abstract bipolar argumentation framework* (*BAF* for short) [4] is a triple $\langle A, \Sigma, \Pi \rangle$, where (i) $A \subseteq Arg$ is a (finite) set whose elements are referred to as *arguments*, (ii) $\Sigma \subseteq A \times A$ is a binary relation over A whose elements are called *attacks*, (iii) $\Pi \subseteq A \times A$ is a binary relation over A whose elements are called *supports*, and (iv) $\Sigma \cap \Pi = \emptyset$. Thus, a Dung's argumentation framework (AF) [25] is a BAF of the form $\langle A, \Sigma, \emptyset \rangle$.

An argument is an abstract entity whose role is entirely determined by its relationships with other arguments. A BAF can be viewed as a directed graph where each node corresponds to an argument and each edge in the graph corresponds to either an attacks

[1] http://argumentationcompetition.org

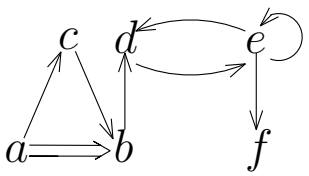

Fig. 1. BAFs $\mathcal{B}_0$ of Example 1

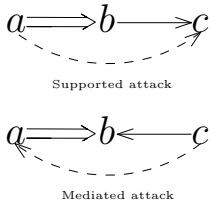

Fig. 2. Supported and mediated attacks

or a support. Given a BAF $\mathcal{B}$, the *bipolar interaction graph* for $\mathcal{B}$ (denoted as $\mathcal{G}_{\mathcal{B}}$) has two kinds of edges: one for the attack relation ($\rightarrow$) and another one for the support relation ($\Rightarrow$), as shown in the following example.

Example 1. Consider the BAF $\mathcal{B}_0 = \langle A_0, \Sigma_0, \Pi_0 \rangle$ where :

- $A_0 = \{a, b, c, d, e, f\}$;
- $\Sigma_0 = \{(a, c),\ (c, b),\ (b, d),\ (d, e),\ (e, d),\ (e, e),\ (e, f)\}$;
- $\Pi_0 = \{(a, b)\}$

The bipolar interaction graph $\mathcal{G}_{\mathcal{B}_0}$ for $\mathcal{B}_0$ is shown in Fig.1. □

Several interpretations of the notion of support have been proposed in the literature [24]. In this paper, we focus on *deductive* support [14] which is intended to capture the following intuition: if argument a supports argument b then the acceptance of a implies the acceptance of b, and thus the non-acceptance of b implies the non-acceptance of a. Given this interpretation of support, the coexistence of the support and attack relations in BAFs entails that new kinds of "implicit" attacks should be considered.

Given a BAF $\langle A, \Sigma, \Pi \rangle$, a *supported attack* for an argument $b \in A$ by argument $a_1 \in A$ is a sequence $a_1 \Pi a_2 \Pi \ldots \Pi a_n \Sigma\, b$ with $n \geq 1$. Note that a direct attack $a_1 \Sigma b$ is a supported attack. Thus a supported attack is a (possibly empty) chain of supports followed by an attack. Moreover, we say that there is a *mediated attack* for argument a_1 by argument b if there is an attack $b \Sigma a_n$ and a sequence of supports $a_1 \Pi a_2 \Pi \ldots \Pi a_n$ and with $n \geq 1$. Thus, for a mediated attack the chain of supports ends in a_n which is attacked by b. Supported and mediated attacks are illustrated in Figure 2, where a chain consisting of only one support is considered. The BAF of Example 1 contains a supported attack from argument a to d, and a mediated attack from argument c to a.

Another kind of implicit attack which we do not consider in this paper because of the deductive interpretation of support is the *secondary attack* [21], which occurs when in a BAF there is a sequence $b \Sigma a_1 \Pi a_2 \Pi \ldots \Pi a_n$ with $n \geq 1$. Considering supported and secondary attacks leads to an alternative interpretation of support [21]. However, when considering a deductive interpretation of support, secondary attacks may lead to counterintuitive results [24], though they are useful in contexts where support is interpreted differently.

Given a BAF $\langle A, \Sigma, \Pi \rangle$, we say that a set $S \subseteq A$ *set-attacks* an argument $b \in A$ iff there exists a supported or mediated attack for b by an argument $a \in S$. We use S^+ to denote the set of arguments set-attacked by S. Moreover, we say that a set $S \subseteq A$

defends an argument $a \in A$ iff for each $b \in A$ such that $\{b\}$ set-attacks a, it is the case that S set-attacks b (i.e., $b \in S^+$).

Given a BAF $\langle A, \Sigma, \Pi \rangle$, a set $S \subseteq A$ is *conflict-free* iff there are no two arguments $a, b \in S$ such that $\{a\}$ set-attacks b. Moreover, a conflict-free set $S \subseteq A$ is said to be *admissible* iff it defends all of its arguments.

Example 2. Continuing our example, for the BAF $\mathcal{B}_0$ of Example 1, it is easy to see that $\{a\}$ defends argument b (as $\{a\}$ set-attacks c which set-attacks b). The set $\{a, b\}$ is conflict-free as neither a set-attacks b nor b set-attacks a, while $\{a, d\}$ is not conflict-free as a set-attacks d (by means of the supported attack (a, d)). Moreover, $S = \{c, d, f\}$ is an admissible set as it is conflict-free and S defends all of its arguments: c defends itself from a by the mediated attack from c to a; d is defended by c, and f is defended by d. The set of admissible sets for $\mathcal{B}_0$ is $\{\{\emptyset\}, \{a\}, \{c\}, \{a, b\}, \{c, d\}, \{c, d, f\}\}$. □

Given a BAF $\langle A, \Sigma, \Pi \rangle$, a *preferred extension* (pr) for a BAF is an admissible set which is a maximal (w.r.t $\subseteq$). Furthermore, a conflict-free set $S \subseteq A$ is a *stable extension* (st), if and only if it set-attacks all the arguments in $A \setminus S$. (Note that this implies that S is admissible).

Given a BAF $\mathcal{B}$ and a semantics $\mathcal{S} \in \{\mathtt{pr}, \mathtt{st}\}$, we use $\mathcal{E}_{\mathcal{S}}(\mathcal{B})$ to denote the set of extensions for $\mathcal{B}$ according to $\mathcal{S}$. For the BAF $\mathcal{B}_0$ of Example 1, we have that the set of the stable extensions is $\mathcal{E}_{\mathtt{st}}(\mathcal{B}_0) = \{\{c, d, f\}\}$, while the set of the preferred extensions is $\mathcal{E}_{\mathtt{pr}}(\mathcal{B}_0) = \{\{a, b\}, \{c, d, f\}\}$. □

Labelling. Following the approach of [6], where argumentation semantics have been characterized in terms of *labelling*, we define a labelling function for BAFs. A labelling for a BAF $\mathcal{B} = \langle A, \Sigma, \Pi \rangle$ is a total function $L : A \to \{\text{IN}, \text{OUT}, \text{UN}\}$ assigning to each argument a label: $L(a) = \text{IN}$ means that argument a is accepted, $L(a) = \text{OUT}$ means that a is rejected, while $L(a) = \text{UN}$ means that a is undecided.

Let $in(L) = \{a \mid a \in A \wedge L(a) = \text{IN}\}$, $out(L) = \{a \mid a \in A \wedge L(a) = \text{OUT}\}$, and $un(L) = \{a \mid a \in A \wedge L(a) = \text{UN}\}$. In the following, we also use the triple $\langle in(L), out(L), un(L) \rangle$ to represent the labelling L.

Given a BAF $\mathcal{B} = \langle A, \Sigma, \Pi \rangle$, a labelling L for $\mathcal{B}$ is said to be *admissible (or legal)* if $\forall a \in in(L) \cup out(L)$ it holds that (*i*) $L(a) = \text{OUT}$ iff $\exists\, b \in A$ such that $a \in \{b\}^+$ and $L(b) = \text{IN}$; and (*ii*) $L(a) = \text{IN}$ iff $L(b) = \text{OUT}$ for all $b \in A$ such that $a \in \{b\}^+$. Moreover, L is a *complete* labelling iff conditions (*i*) and (*ii*) hold for all $a \in A$.

Between extensions and complete labellings there is a bijective mapping defined as follows: for each extension E there is a unique labelling $L = \langle E, E^+, A \setminus (E \cup E^+) \rangle$ and for each labelling L there is a unique extension $in(L)$. We say that L is the labelling *corresponding* to E. For instance, considering the BAF $\mathcal{B}_0$ of Example 1, the labelling corresponding to the preferred extension $E = \{a, b\}$ is $L = \langle \{a, b\}, \{c, d\}, \{e, f\} \rangle$.

In the following, we say that the *status of an argument* a w.r.t. a labelling L (or its corresponding extension $in(L)$) is IN (resp., OUT, UN) iff $L(a) = \text{IN}$ (resp., $L(a) = \text{OUT}$, $L(a) = \text{UN}$). We will avoid to mention explicitly the labelling (or the extension) whenever it is understood.

2.1 Updates

An *update* u for a BAF $\mathcal{B}_0$ allows us to change $\mathcal{B}_0$ into a BAF $\mathcal{B}$ by adding or removing an argument, an attack, or a support. The addition (resp., deletion) of an argument a will be denoted as $+a$ (resp. $-a$), whereas the addition (resp., deletion) of an attack from a to b will be denoted as $+(a \rightarrow b)$ (resp., $-(a \rightarrow b)$). Moreover, the addition (resp., deletion) of a support from a to b will be denoted as $+(a \Rightarrow b)$ (resp., $-(a \Rightarrow b)$).

We will use $u(\mathcal{B}_0)$ to denote the BAF resulting from the application of an update u to the initial BAF $\mathcal{B}_0$. For instance, for BAF $\mathcal{B}_0 = \langle A_0, \Sigma_0, \Pi_0 \rangle$, if $u = +(f \Rightarrow b)$, we have that $u(\mathcal{B}_0) = +(f \Rightarrow b)(\mathcal{B}_0) = \langle A_0, \Sigma_0, \Pi_0 \cup \{(f, b)\} \rangle$, while if $u = -(b \rightarrow d)$, we have that $u(\mathcal{B}_0) = \langle A_0, \Sigma_0 \setminus \{(b, d)\}, \Pi_0 \rangle$.

Applying an update u to a BAF implies that its semantics (set of extensions or labellings) may change. Continuing our running example, for the BAF $\mathcal{B}_0$ of Example 1 and the update $u = +(f \Rightarrow b)$, we have that the set of the stable extensions for the updated BAF $\mathcal{B} = +(f \Rightarrow b)(\mathcal{B}_0)$ is $\mathcal{E}_{\mathtt{st}}(\mathcal{B}) = \{\{c, d\}\}$, while the set of the preferred extensions is $\mathcal{E}_{\mathtt{pr}}(\mathcal{B}) = \{\{a, b\}, \{c, d\}\}$. In fact, the addition of the support between f and b entails that additional implicit attacks must be considered: a supported attack between f and d, and a mediated one between c and f.

In the following, for the sake of the presentation, we consider only *feasible* updates which are defined as follows. Adding an argument as well as removing an attack/support are feasible updates. The deletion of an argument is feasible only if a is *isolated*, that is there is no argument b attacking/supporting a or being attacked/supported by a, where a is not necessarily distinct from b. The addition of an attack (resp., support) between a and b is feasible only if a and b are arguments of the initial BAF $\mathcal{B}_0$ and there is no already a support (resp. attack) between a and b in $\mathcal{B}_0$. Clearly, the general updates can be simulated by a sequence of feasible updates. For instance, an isolated argument a can be deleted after deleting all attacks and supports involving a (by performing feasible updates). Analogously, adding an attack or a support between an argument a in $\mathcal{B}_0$ and a new argument b can be simulated as a sequence of updates of the form $+b, +(a \rightarrow b)$ or $+b, +(a \Rightarrow b)$.

It is worth noting that if $\mathcal{B}$ is obtained from $\mathcal{B}_0$ through the addition (resp. deletion) of a set S of isolated argument, then, let E_0 be an extension for $\mathcal{B}_0$, it is the case that $E = E_0 \cup S$ (resp. $E = E_0 \setminus S$) is an extension for $\mathcal{B}$ that can be trivially computed. Thus, in the following we do not discuss further updates of the form $+a$ or $-a$. We will focus on updates of the forms $\pm(c \rightarrow d)$ and $\pm(e \Rightarrow f)$, where $\pm$ means either $+$ or $-$.

3 Incremental Computation

Given a BAF $\mathcal{B}_0$, a preferred/stable extension E_0 for $\mathcal{B}_0$, and an update u, our approach to recompute a preferred/stable extension E of an updated BAF $u(\mathcal{B}_0)$ consists of the following three main steps.

1) We identify conditions ensuring that a given extension E_0 for the initial BAF $\mathcal{B}_0$ (under the preferred or stable semantic) is still an extension for the updated BAF $u(\mathcal{B}_0)$. In such case, the update u does not invalidate the initial extension E_0, and it will be immediately returned by our algorithm.

2) Given $\mathcal{B}_0$, E_0 and u, we transform the initial BAF $\mathcal{B}_0$ into an equivalent Dung's AF $\mathcal{M}_0$, a corresponding extension E_0^m, and an update u^m for $\mathcal{M}_0$. Our transformation is based on the meta-argumentation approach proposed in [14], though in our case we need to take into account the initial extension E_0 and the update u.
3) Given the AF $\mathcal{M}_0$, its extension E_0^m, and the update u^m for $\mathcal{M}_0$, we use the incremental technique proposed in [1] for computing an extension E^m of the updated AF $u^m(\mathcal{M}_0)$ w.r.t. E_0^m. The technique identifies a reduced AF sufficient to compute an extension of the whole AF and use state-of-the-art algorithms to recompute an extension of the reduced AF only. Then from an extension E^m of the updated AF $u^m(\mathcal{M}_0)$, we derive an extension E of an updated BAF $u(\mathcal{B}_0)$.

In the next sections we describe in detail the three steps above.

3.1 Early-Termination Conditions

Given a BAF $\mathcal{B}_0$, an initial extension E_0 whose corresponding labelling is L_0, and an update u, for each pair of initial statuses $L_0(a)$ and $L_0(b)$ of the arguments involved in the update, Tables 1 – 4 tell us if E_0 is still an extension after performing the update under the preferred or stable semantics.

Proposition 1 (Extension preservation for addition/deletion of an attack/support). *Let $\mathcal{B}_0$ be a BAF, $\mathcal{S} \in \{\mathtt{pr}, \mathtt{st}\}$ a semantics, $E_0 \in \mathcal{E}_{\mathcal{S}}(\mathcal{B}_0)$ an extension of $\mathcal{B}_0$ under semantics $\mathcal{S}$, L_0 the labelling corresponding to E_0, and u an update. If $\mathcal{S}$ is in the cell $\langle L_0(a), L_0(b)\rangle$ of Table 1 and $u = +(a \to b)$ [resp., Table 2 and $u = +(a \Rightarrow b)$; Table 3 and $u = -(a \to b)$; Table 4 and $u = -(a \Rightarrow b)$)], then $E_0 \in \mathcal{E}_{\mathcal{S}}(u(\mathcal{B}_0))$.*

If some of the conditions of Proposition 1 holds, then the given initial extension of the initial BAF is still an extension of the updated one, and thus the above-mentioned steps 2) and 3) can be skipped — the algorithm just returns the initial extension which is also an extension for the updated BAF. For instance, considering the BAF $\mathcal{B}_0$ of Example 1, the update $u = -(b \to d)$, and preferred extension $E_0 = \{c, d, f\}$, since $L_0(b) = \text{OUT}$ and $L_0(d) = \text{IN}$, Table 3 says that $E_0 = \{c, d, f\}$ is still an extension of the BAF $u(\mathcal{B}_0)$. Similarly, considering $u = +(c \Rightarrow f)$, and again the preferred extension $E_0 = \{c, d, f\}$, since $L_0(c) = L_0(f) = \text{IN}$, Table 2 tell us that E_0 is still an extension of the updated BAF.

Conditions similar to those of Proposition 1 were identified in [1] for updates of Dung's AFs, where support is not considered. However, those conditions could be used only at step 3) when applying the technique of [1] to the meta-argumentation framework $\mathcal{M}_0$, that is, after performing the transformation of step 2). Therefore, to avoid to uselessly perform step 2), we provided Proposition 1, which can be used to check for cases for which the initial extension is preserved directly on the input BAF.

3.2 The Meta-Argumentation Framework

Given the initial BAF $\mathcal{B}_0$ and an updated u for it, we define the corresponding meta-argumentation framework as follows.

update $+(a \to b)$		$L_0(b)$		
		IN	UN	OUT
$L_0(a)$	IN			pr, st
	UN			pr
	OUT	pr,st		pr,st

Table 1. Cases for which $E_0 \in \mathcal{E}_{\mathcal{S}}(u(\mathcal{B}_0))$ *for* $u = +(a \to b)$.

update $+(a \Rightarrow b)$		$L_0(b)$		
		IN	UN	OUT
$L_0(a)$	IN	pr,st		
	UN			
	OUT	pr,st	pr	pr,st

Table 2. Cases for which $E_0 \in \mathcal{E}_{\mathcal{S}}(u(\mathcal{B}_0))$ *for* $u = +(a \Rightarrow b)$.

update $-(a \to b)$		$L_0(b)$		
		IN	UN	OUT
$L_0(a)$	IN	NA	NA	
	UN	NA		pr
	OUT	pr,st	pr	pr,st

Table 3. Cases for which $E_0 \in \mathcal{E}_{\mathcal{S}}(u(\mathcal{B}_0))$ *for* $u = -(a \to b)$.

update $-(a \Rightarrow b)$		$L_0(b)$		
		IN	UN	OUT
$L_0(a)$	IN	pr,st	NA	NA
	UN	pr		NA
	OUT	pr,st	pr	

Table 4. Cases for which $E_0 \in \mathcal{E}_{\mathcal{S}}(u(\mathcal{B}_0))$ *for* $u = -(a \Rightarrow b)$.

Definition 1 (Meta AF). *Let $\mathcal{B} = \langle A, \Sigma, \Pi \rangle$ be a BAF, and u an update for $\mathcal{B}$ of the form $u = \pm(c \to d)$ or $u = \pm(e \Rightarrow f)$. Then, the meta-AF for $\mathcal{B}$ w.r.t. u is $\mathcal{M} = \langle A^m, \Sigma^m \rangle$ where:*

$$\begin{aligned}
i)\ A^m \;=\; & \{\, a \mid\ a \in A\} \cup \{X_{a,b}, Y_{a,b} \mid\ (a,b) \in \Sigma\} \cup \\
& \{X_{c,d}, Y_{c,d} \mid\ u = +(c \to d)\} \cup \\
& \{Z_{a,b} \mid\ (a,b) \in \Pi\} \cup \\
& \{Z_{e,f} \mid\ u = +(e \Rightarrow f)\} \\
ii)\ \Sigma^m \;=\; & \{(a, X_{a,b}), (X_{a,b}, Y_{a,b}), (Y_{a,b}, b) \mid\ (a,b) \in \Sigma\} \cup \\
& \{(c, X_{c,d}), (X_{c,d}, Y_{c,d}) \mid\ u = +(c \to d)\} \cup \\
& \{(b, Z_{a,b}), (Z_{a,b}, a) \mid\ (a,b) \in \Pi\} \cup \\
& \{(f, Z_{e,f}) \mid\ u = +(e \Rightarrow f)\}
\end{aligned}$$

As an example, the (interaction graph of the) meta AF $\mathcal{M}_0$ for the for the BAF $\mathcal{B}_0$ of Example 1 w.r.t the update $u = +(f \Rightarrow b)$ is shown in Figure 3.

Our definition of meta-argumentation framework builds on that proposed in [14] and consider additional (meta)arguments (e.g., $Z_{f,b}$ in Figure 3) and attacks (e.g., $(b, Z_{f,b})$ in Figure 3) that will allow us to simulate (positive) updates to be performed on BAF $\mathcal{B}_0$ by means of updates performed on the corresponding the meta-AF $\mathcal{M}_0$. (We do not need to add additional arguments/attack to simulate negative updates, as it is sufficient to remove attacks already present in the original construction of [14]). The meta AF of Definition 1 collapses to the construction of [14] if we have no updates.

Thus, updates for a given BAFs can be modeled by means of updates for the corresponding meta-AF, as detailed in what follows.

Definition 2 (Updates for the Meta AF). *Let $\mathcal{B} = \langle A, \Sigma, \Pi \rangle$ be a BAF, and u an update for $\mathcal{B}$ of the form $u = \pm(c \to d)$ or $u = \pm(e \Rightarrow f)$. The corresponding update*

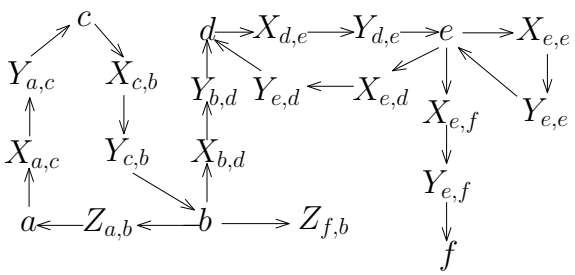

Fig. 3. Meta AF $\mathcal{M}_0$ for the BAF $\mathcal{B}_0$ of Figure 1 w.r.t. the update $u = +(f \Rightarrow b)$.

u^m for the meta-AF $\mathcal{M}$ for $\mathcal{B}$ w.r.t. u is as follows:

$$u^m = \begin{cases} +(Z_{e,f}, e)\ if\ u = +(e \Rightarrow f) \\ -(Z_{e,f}, e))\ if\ u = -(e \Rightarrow f)) \\ +(Y_{c,d}, d)\ if\ u = +(c \rightarrow d) \\ -(Y_{c,d}, d))\ if\ u = -(c \rightarrow d)) \end{cases}$$

Continuing our running example, the update for the meta AF $\mathcal{M}_0$ corresponding to the BAF $\mathcal{B}_0$ of Example 1 and the update $u = +(f \Rightarrow b)$ is $u^m = +(Z_{f,b}, f)$.

The last ingredient we need before being ready to apply the incremental technique of [1] is the initial extension E_0^m for the AF $\mathcal{M}_0$. It is obtained from that of initial BAF $\mathcal{B}_0$ by essentially propagating the labels of the arguments in $\mathcal{B}_0$ as follows.

Definition 3 (Initial Extension/Labelling for the Meta AF). *Given the BAF $\mathcal{B}_0 = \langle A, \Sigma, \Pi \rangle$ and its initial labelling L_0, the corresponding initial labelling L_0^m for the meta-AF $\mathcal{M}_0 = \langle A^m, \Sigma^m \rangle$ is as follows:*

- $L_0^m(a) = L_0(a)$ *for all* $a \in A \cap A^m$
- $L_0^m(X_{a,b}) = \text{IN}\ if L_0(a) = \text{OUT}$, $L_0^m(X_{a,b}) = \text{OUT}\ if L_0(a) = \text{IN}$, *and* $L_0^m(X_{a,b}) = \text{UN}\ if L_0(a) = \text{UN}$, *for all* $X_{a,b} \in A^m$
- $L_0^m(Y_{a,b}) = L_0(a)$ *for all* $Y_{a,b} \in A^m$
- $L_0^m(Z_{a,b}) = \text{IN}\ if L_0(b) = \text{OUT}$, $L_0^m(Z_{a,b}) = \text{OUT}\ if L_0(b) = \text{IN}$, *and* $L_0^m(Z_{a,b}) = \text{UN}\ if L_0(b) = \text{UN}$, *for all* $Z_{a,b} \in A^m$

For instance, with reference to Figure 3, and the preferred extension $E_0 = \{c, d, f\}$ for the BAF $\mathcal{B}_0$ of Example 1, we have that the labelling corresponding to E_0 is $L_0 = \langle \{c, d, f\}, \{a, b, e\}, \emptyset \rangle$, and thus $L_0^m(Z_{f,b}) = \text{IN}$ since $L_0^m(b) = L_0(b) = \text{OUT}$. Moreover, $L_0^m(Y_{e,f}) = L_0(e) = \text{OUT}$.

The following proposition characterizes the relationship between extensions of updated BAFs and extension of updated for meta AFs.

Proposition 2. *Let $\mathcal{B}_0 = \langle A, \Sigma, \Pi \rangle$ be a BAF, $\mathcal{S} \in \{\texttt{pr}, \texttt{st}\}$ a semantics, and $E_0 \in \mathcal{E}_\mathcal{S}(\mathcal{B}_0)$ an extension of $\mathcal{B}_0$ under $\mathcal{S}$.*
Let $\mathcal{M}_0$ be meta AF for $\mathcal{B}_0$ w.r.t. u, E_0^m the initial $\mathcal{S}$-extension for $\mathcal{M}_0$ corresponding to E_0, and u^m the update for $\mathcal{M}_0$ corresponding to u.
Then, there is $E \in \mathcal{E}_\mathcal{S}(u(\mathcal{B}_0))$ iff there is $E^m \in \mathcal{E}_\mathcal{S}(u^m(\mathcal{M}_0))$ such that $E = E^m \cap A$.

Algorithm 1 Incr-BAF($\mathcal{B}_0, u, E_0, \mathcal{S}$, Solver$_\mathcal{S}$)

Input: BAF $\mathcal{B}_0 = \langle A_0, \Sigma_0\, \Pi_0 \rangle$,
update u of the form $u = \pm(a \Rightarrow b)$ or $u = \pm(a \rightarrow b)$,
an initial $\mathcal{S}$-extension E_0,
semantics $\mathcal{S} \in \{\mathtt{pr}, \mathtt{st}\}$,
function Solver$_\mathcal{S}(\mathcal{A})$ returning an $\mathcal{S}$-extension for AF $\mathcal{A}$ if it exists, $\perp$ otherwise;
Output: An $\mathcal{S}$-extension E for $u(\mathcal{B}_0)$ if it exists, $\perp$ otherwise;;

1: **if** $checkProp(\mathcal{B}_0, u, E_0, \mathcal{S})$ **then**
2: **return** E_0**;**
3: Let $\mathcal{M}_0 = \langle A^m, \Sigma^m \rangle$ be the the meta-AF for $\mathcal{B}_0$ w.r.t. u (cf. Definition 1);
4: Let u^m be the update for $\mathcal{M}_0$ corresponding to u (cf. Definition 2)
5: Let E_0^m be the initial $\mathcal{S}$-extension for $\mathcal{M}_0$ corresponding to E_0; (cf. Definition 3)
6: Let $E^m =$ Incr-Alg($\mathcal{M}_0, u^m, \mathcal{S}, E_0^m$, Solver$_\mathcal{S}$) [1]
7: **if** $(E^m \neq \perp)$ **then**
8: **return** $E = (E^m \cap A_0)$;
9: **else**
10: **return** $\perp$;

3.3 Incremental Algorithm

Given a BAF $\mathcal{B}_0$, a semantics $\mathcal{S} \in \{\mathtt{pr}, \mathtt{st}\}$, an extension $E_0 \in \mathcal{E}_\mathcal{S}(\mathcal{B}_0)$, and an update u of the form $u = \pm(a \Rightarrow b)$ or $u = \pm(a \rightarrow b)$, we define an incremental algorithm (Algorithm 1) for computing an extension E of the updated BAF $u(\mathcal{B}_0)$, if it exists (note that for the stable semantics, the set of extensions $\mathcal{E}_{\mathtt{st}}(u(\mathcal{B}_0))$ of the updated AF may be empty; in this case, the algorithm returns $\perp$).

Algorithm 1 works as follows. It first checks if the initial extension E_0 is still an extension of the updated BAF at Line 1, where $checkProp(\mathcal{B}_0, u, E_0, \mathcal{S})$ is a function returning *true* iff some of the conditions of Proposition 1 holds. If this is the case, it immediately returns the initial extension. Otherwise, it computes the (meta) AF $\mathcal{M}_0$ (Line 3), the update u^m for $\mathcal{M}_0$ (Line 4), and the initial $\mathcal{S}$-extension E_0^m for $\mathcal{M}_0$ (Line 5). Next it invokes the incremental algorithm Incr-Alg proposed in [1] with input parameters $\mathcal{M}_0, u^m, \mathcal{S}, E_0^m$, and Solver$_\mathcal{S}$, where Solver$_\mathcal{S}$ is any external solver that can compute an $\mathcal{S}$-extension for the input AF.

Roughly speaking, the technique in [1] uses state-of-the-art algorithms to compute an extension on a subset of the input AF. More in detail, it consists of three steps:
(*i*) First a sub-AF, called *reduced* AF, is identified on the basis of the the set of arguments *influenced* by an update [37–39] and additional information provided by the initial extension. In our running example, given the meta AF $\mathcal{M}_0$ of Figure 3 the update $u^m = +(Z_{f,b}, f)$, and the extension E_0^m corresponding to the preferred extension $E_0 = \{c, d, f\}$, the influenced set consists of argument f only. The reduced AF is built by adding to the subgraph induced by the influenced set, additional arguments and attacks containing needed information on the "external context", i.e. information about the status of arguments which are attacking some argument in the influenced set. Continuing our example, the reduced AF consists of the two arguments $Z_{f,b}$ and f and the attack $(Z_{f,b}, f)$ between them.

(*ii*) Second, a non-incremental algorithm (e.g., *Cegartix* [28] for $\mathcal{S} = \texttt{pr}$, *ASPARTIX-D* [36] for $\mathcal{S} = \texttt{st}$) is used to compute an extension of the reduced AF — this is done by calling function $\mathsf{Solver}_{\mathcal{S}}$ in Algorithm 1. In our example the external solver returns the preferred extension $E^r = \{Z_{f,b}\}$ for the reduced AF.
(*iii*) Finally, the final extension E_m of the whole (meta) AF is obtained by merging a portion of its initial extension with that computed for the reduced AF (i.e., $E^r = \{Z_{f,b}\}$ in our example) by the external solver $\mathsf{Solver}_{\mathcal{S}}$. The result of the merging operation in our example will be $E_m = \{c, d, Y_{c,b}, Z_{a,b}, Z_{f,b}, X_{a,c}, Y_{c,b}, Y_{d,e}, X_{e,d}, X_{e,e}, X_{e,f}\}$.

After calling the incremental algorithm Incr-Alg of [1] over the (meta-) AF $\mathcal{M}_0$, the extension of the updated BAF (if any) is obtained by projecting out the extension E^m returned by Incr-Alg over the set of arguments A_0 of the initial BAF (Line 8). In our example, we obtain the extension $E = \{c, d\} = E^m \cap A_0$ for the updated BAF.

4 Experimental Results

We implemented a C++ prototype and, for each semantics $\mathcal{S} \in \{\texttt{pr}, \texttt{st}\}$, we compared the performance of Algorithm 1 with that of the best ICCMA'15 solver for the computational task $\mathcal{S}$-SE, that is the task of determining some $\mathcal{S}$-extension. In particular, given a BAF $\mathcal{B}_0$, a semantics $\mathcal{S} \in \{\texttt{pr}, \texttt{st}\}$, an extension $E_0 \in \mathcal{E}_{\mathcal{S}}(\mathcal{B}_0)$, and an update u of the form $u = \pm(a \Rightarrow b)$ or $u = \pm(a \rightarrow b)$, we compare the following two strategies for computing an extension $E \in \mathcal{E}_{\mathcal{S}}(u(\mathcal{B}_0))$ of the updated BAF:

- **Incremental computation**, that is, Algorithm 1 with input $\mathcal{B}_0, u, E_0, \mathcal{S}$, and $\mathsf{Solver}_{\mathcal{S}}$;
- **Computation from scratch**, where an extension E of the updated BAF $u(\mathcal{B}_0)$ is computed by running $\mathsf{Solver}_{\mathcal{S}}$ over the (meta-)AF for $u^m(\mathcal{M}_0)$.

where $\mathsf{Solver}_{\mathcal{S}}$ is *Cegartix* [28] for $\mathcal{S} = \texttt{pr}$ and *ASPARTIX-D* [36] for $\mathcal{S} = \texttt{st}$ (these solvers are the winners of the ICCMA'15 competition for the computational task $\mathcal{S}$-SE).

Dataset. We generated a set of BAFs by starting from AFs used as benchmarks at ICCMA'15, available at *http://argumentationcompetition.org/2015/results.html*. Given a percentage $p \in \{10\%, 20\%\}$ of support, for each AF $\mathcal{A}_d = \langle A_d, \Sigma_d \rangle$ in the ICCMA dataset, we generate two BAFs $\mathcal{B}_0 = \langle A_d, \Sigma^p, \Pi^p \rangle$ as follows. We selected $p \times |\Sigma_d|$ attacks in Σ_d in a random way, and converted them into supports. That is, let $\Sigma^r \subseteq \Sigma_d$ be the set of the chosen $p \times |\Sigma_d|$ attacks in Σ_d. For each $(a, b) \in Att^r$, we added a support randomly chosen in $\{(a, b), (b, a)\}$ to Π^p. Finally, we set $\Sigma^p = \Sigma_d \setminus \Sigma^r$.

Methodology. For each semantics $\mathcal{S} \in \{\texttt{pr}, \texttt{st}\}$, for each BAF $\mathcal{B}_0 = \langle A_0, \Sigma_0, \Pi_0 \rangle$ in the dataset, we considered every $\mathcal{S}$-extension E_0 of $\mathcal{B}_0$ as an initial extension. Then, we randomly selected an update u of the form $+(a \rightarrow b)$ (with $a, b \in A_0$ and $(a, b) \notin \Sigma_0$), or $+(a \Rightarrow b)$ (with $a, b \in A_0$ and $(a, b) \notin \Pi_0$), or $-(a \rightarrow b)$ (with $(a, b) \in \Sigma_0$), or $-(a \Rightarrow b)$ (with $(a, b) \in \Pi_0$). Next, we computed an $\mathcal{S}$-extension E for the updated BAF $u(\mathcal{B}_0)$ by calling Algorithm 1. Finally, the average run time of our incremental algorithm to compute an $\mathcal{S}$-extension was compared with the average run time of *Cegartix* for $\mathcal{S} = \texttt{pr}$ and *ASPARTIX-D* for $\mathcal{S} = \texttt{st}$ to compute an $\mathcal{S}$-extension for $u^m(\mathcal{M}_0)$ from scratch.

Results. Figure 4 reports the average run times (log scale) of the incremental computation (*Incr-BAF*) and the computation from scratch. Each data point reported in the figure is the average time over 30 runs. The figure shows how the running time vary w.r.t. the semantics (preferred or stable), the percentage (10% or 20%) of edges of the type support in the initial BAF, and the number of arguments in the meta AF $\mathcal{M}_0$ for the given BAF and update. It is worth noting that the number of arguments and attacks in the meta AF $\mathcal{M}_0$ is much greater then the number of arguments and attacks/supports in the initial BAF $\mathcal{B}_0 = \langle A_0, \Sigma_0, \Pi_0 \rangle$, whose size (in terms of nodes and edges in the interaction graph) is that of the original AF $\mathcal{A}_d = \langle A_d, \Sigma_d \rangle$ in the ICCMA dataset used to build $\mathcal{B}_0$. Specifically, from Definition 1, it is easy to see that the number of arguments of $\mathcal{M}_0$ turns out to be $|A_0| + |\Pi_0| + 2 \times |\Sigma_0|$, while the number of attacks is $2 \times |\Pi_0| + 3 \times |\Sigma_0|$.

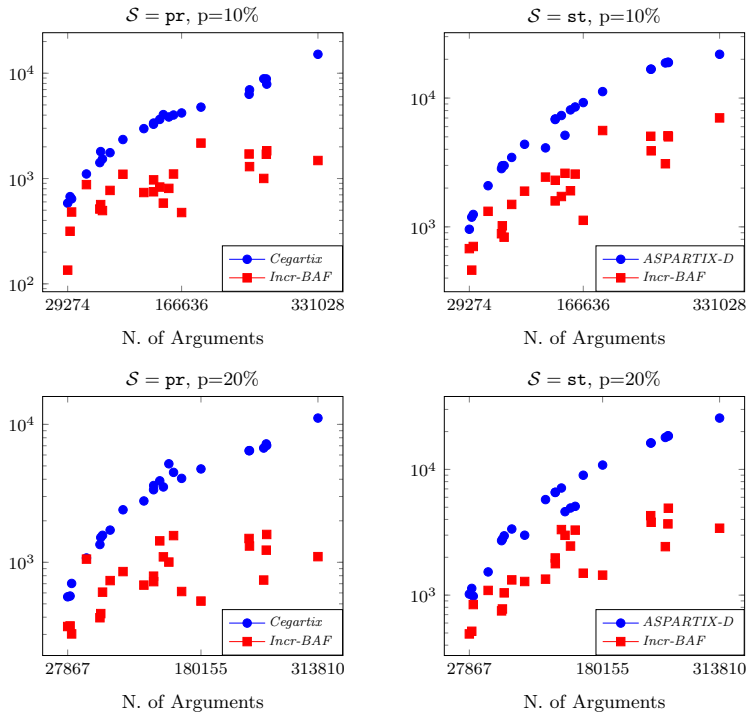

Fig. 4. Run times (ms) of ICCMA solvers and *Incr-BAF*, for semantics $\mathcal{S}$ and percentage p of supports showed on the top of each graph, versus the number of arguments in the meta AF.

From the results reported in Figure 4, we can draw the following conclusions:

- Our algorithm outperforms the competitors that compute the extensions from scratch. In particular, the time saved by the incremental computation increases exponentially with respect to the size of the input BAF.
- The improvements obtained for the two semantics (preferred and stable) are similar. That is, our incremental approach is quite insensitive w.r.t. the semantics adopted.

– The improvements obtained increase when increasing the percentage of support from 10% to 20%. In fact, for a given fixed number $n = |\Sigma_0| + |\Pi_0|$ of the edges in the interaction graph for BAF $\mathcal{B}_0$, it is the case that increasing the percentage of edges in Π_0 (and thus decreasing $|\Sigma_0|$) yields to smaller meta AFs.

5 Related Work

A comprehensive introduction to (static) abstract argumentation frameworks (AFs) can be found in [6], while [24] provides a survey of bipolar argumentation frameworks (BAFs). Although the idea underlying AFs is simple and intuitive, most of the semantics proposed so far suffer from a high computational complexity [27, 26, 29, 30, 32–35]. Complexity bounds and evaluation algorithms for AFs have been deeply studied in the literature, but most of this research focused on static frameworks, whereas, in practice, AFs (as well as BAFs) are not static systems [8, 31, 41, 7, 23].

There have been significant efforts coping with dynamics aspects of Dung's abstract argumentation frameworks. as discussed in what follows. [15, 16] have investigated the principles according to which a grounded extension of a Dung's abstract argumentation frameworks does not change when the set of arguments/attacks are changed. [17, 18] have addressed the problem of revising the set of extensions of an argumentation framework, and studied how the extensions can evolve when a new argument is considered. They focus on adding one argument interacting with one initial argument (i.e. an argument which is not attacked by any other argument). [13] have studied the evolution of the set of extensions after performing a change operation (addition/removal of arguments/interaction). Dynamic argumentation has been applied to decision-making of an autonomous agent by [5], where it is studied how the acceptability of arguments evolves when a new argument is added to the decision system. The division-based method, proposed by [41] and then refined by [7], divides the updated framework into two parts: affected and unaffected, where only the status of affected arguments is recomputed after updates. Recently, [46] introduced a matrix representation of argumentation frameworks and proposed a matrix reduction that, when applied to dynamic argumentation frameworks, resembles the division-based method in [41].

Other relevant works on dynamic aspects of Dung's argumentation frameworks include the following. [8] have proposed an approach exploiting the concept of splitting of logic programs to deal with dynamic argumentation. The technique considers weak expansions of the initial AF, where added arguments never attack previous ones. [11] have investigated whether and how it is possible to modify a given AF so that a desired set of arguments becomes an extension, whereas [43] have studied equivalence between two AFs when further information (another AF) is added to both AFs. [9] have focused on expansions where new arguments and attacks may be added but the attacks among the old arguments remain unchanged, while [10] have characterized update and deletion equivalence, where adding/deleting arguments/attacks is allowed (deletions were not considered by [43, 9]).

Bipolarity in argumentation is discussed in [4], where a survey of the use of bipolarity is given, as as well as a formal definition of BAF that extends the Dung's concept of argumentation framework by including supports is provided. The notion of support has

been found useful in many application domains, including decision making [3]. However, as discussed in [24], different interpretations of the concept of support have been proposed. The acceptability of arguments in the presence of the support relation was first investigated in [19]. Then, to handle bipolarity in argumentation, [20, 21] proposed an approach based on using the concept of *coalition* of arguments, where arguments are grouped together, and defeats occur between coalitions. However, when considering a deductive interpretation of support [14, 45], as we did in this paper, coalitions may lead to counterintuitive results [24], though they are useful in contexts where support is interpreted differently. Changes in bipolar argumentation frameworks have been studied in [22], where it is shown how the addition of one argument together with one support involving it (and without any attack) impacts the extensions of the updated BAF.

To the best of our knowledge, this is the first paper addressing the problem of efficiently and incrementally computing extensions of dynamic BAFs.

6 Conclusion and Future Work

We introduced a technique for the incremental computation of extensions of dynamic BAFs. Following the meta-argumentation approach [24], according to which BAFs are translated into semantically equivalent AFs, we introduced a translation where updates and initial extensions of BAFs are taken into account. Then, we exploited the incremental algorithm recently proposed in [1] and computed extensions of the meta AFs, from which the updated extensions of BAFs are obtained. Our experiments showed that the incremental technique outperforms the computation from scratch.

Although in this paper we focused on updates consisting of adding/removing one attack/support, our technique can be extended to deal with sets of updates performed simultaneously. Indeed, the construction described in [38] for reducing the application of a set of updates to the application of a single attack update can be easily extended to deal with multiple updates for BAFs.

Moreover, our technique can be extended to consider *second-order attacks* [14] for BAFs, that is, (i) attacks from an argument or an attack to another attack and (ii) attacks from an argument to a support. This allows the representation of a kind of *defeasible support*, according to which the support itself can be attacked. In fact, analogously to what done in Section 3.2, we can build on the definition of meta-AF introduced in [14] for encoding second-order attacks, and then we can extend it to deal with updates for such kind of BAFs. The incremental algorithm of [1] could be used again by taking as input the meta AF resulting from such transformation.

Finally, we also plan to extend our technique to deal with other interpretations of support, particularly the approach in [20, 21] where meta AFs are also adopted to cope with bipolarity in argumentation.

References

1. Gianvincenzo Alfano, Sergio Greco, and Francesco Parisi. Efficient computation of extensions for dynamic abstract argumentation frameworks: An incremental approach. In *IJCAI*, pages 49–55, 2017.

2. Teresa Alsinet, Josep Argelich, Ramn Bjar, Csar Fernndez, Carles Mateu, and Jordi Planes. An argumentative approach for discovering relevant opinions in twitter with probabilistic valued relationships. *Pattern Recognition Letters*, In press, 2017.
3. Leila Amgoud, Jean-François Bonnefon, and Henri Prade. An argumentation-based approach to multiple criteria decision. In *ECSQARU*, pages 269–280, 2005.
4. Leila Amgoud, Claudette Cayrol, and Marie-Christine Lagasquie-Schiex. On the bipolarity in argumentation frameworks. In *NMR*, pages 1–9, 2004.
5. Leila Amgoud and Srdjan Vesic. Revising option status in argument-based decision systems. *J. Log. Comput.*, 22(5):1019–1058, 2012.
6. Pietro Baroni, Martin Caminada, and Massimiliano Giacomin. An introduction to argumentation semantics. *The Knowledge Engineering Review*, 26(4):365–410, 2011.
7. Pietro Baroni, Massimiliano Giacomin, and Beishui Liao. On topology-related properties of abstract argumentation semantics. A correction and extension to dynamics of argumentation systems: A division-based method. *AI*, 212:104–115, 2014.
8. Ringo Baumann. Splitting an argumentation framework. In *LPNMR*, pages 40–53, 2011.
9. Ringo Baumann. Normal and strong expansion equivalence for argumentation frameworks. *AI*, 193:18–44, 2012.
10. Ringo Baumann. Context-free and context-sensitive kernels: Update and deletion equivalence in abstract argumentation. In *ECAI*, pages 63–68, 2014.
11. Ringo Baumann and Gerhard Brewka. Expanding argumentation frameworks: Enforcing and monotonicity results. In *COMMA*, pages 75–86, 2010.
12. Trevor J. M. Bench-Capon and Paul E. Dunne. Argumentation in artificial intelligence. *AI*, 171(1015):619 – 641, 2007.
13. Pierre Bisquert, Claudette Cayrol, Florence Dupin de Saint-Cyr, and Marie-Christine Lagasquie-Schiex. Characterizing change in abstract argumentation systems. In *Trends in Belief Revision and Argumentation Dynamics*, volume 48, pages 75–102. 2013.
14. Guido Boella, Dov M. Gabbay, Leendert W. N. van der Torre, and Serena Villata. Support in abstract argumentation. In *COMMA*, pages 111–122, 2010.
15. Guido Boella, Souhila Kaci, and Leendert W. N. van der Torre. Dynamics in argumentation with single extensions: Abstraction principles and the grounded extension. In *ECSQARU*, pages 107–118, 2009.
16. Guido Boella, Souhila Kaci, and Leendert W. N. van der Torre. Dynamics in argumentation with single extensions: Attack refinement and the grounded extension. In *ArgMAS Workshop*, pages 150–159, 2009.
17. Claudette Cayrol, Florence Dupin de Saint-Cyr, and Marie-Christine Lagasquie-Schiex. Revision of an argumentation system. In *KR*, pages 124–134, 2008.
18. Claudette Cayrol, Florence Dupin de Saint-Cyr, and Marie-Christine Lagasquie-Schiex. Change in abstract argumentation frameworks: Adding an argument. *JAIR*, 38:49–84, 2010.
19. Claudette Cayrol and Marie-Christine Lagasquie-Schiex. On the acceptability of arguments in bipolar argumentation frameworks. In *ECSQARU*, pages 378–389, 2005.
20. Claudette Cayrol and Marie-Christine Lagasquie-Schiex. Bipolar abstract argumentation systems. In *Argumentation in Artificial Intelligence*, pages 65–84. 2009.
21. Claudette Cayrol and Marie-Christine Lagasquie-Schiex. Coalitions of arguments: A tool for handling bipolar argumentation frameworks. *Int. J. Intell. Syst.*, 25(1):83–109, 2010.
22. Claudette Cayrol and Marie-Christine Lagasquie-Schiex. Change in abstract bipolar argumentation systems. In *SUM*, pages 314–329, 2015.
23. Günther Charwat, Wolfgang Dvorák, Sarah Alice Gaggl, Johannes Peter Wallner, and Stefan Woltran. Methods for solving reasoning problems in abstract argumentation - A survey. *AI*, 220:28–63, 2015.

24. Andrea Cohen, Sebastian Gottifredi, Alejandro Javier Garc´ıNa, and Guillermo Ricardo Simari. A survey of different approaches to support in argumentation systems. *Knowledge Eng. Review*, 29(5):513–550, 2014.
25. Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *AI*, 77(2):321–358, 1995.
26. Paul E. Dunne. The computational complexity of ideal semantics. *AI*, 173(18):1559–1591, 2009.
27. Paul E. Dunne and Michael Wooldridge. Complexity of abstract argumentation. In *Argumentation in Artificial Intelligence*, pages 85–104. 2009.
28. Wolfgang Dvorák, Matti Järvisalo, Johannes Peter Wallner, and Stefan Woltran. Complexity-sensitive decision procedures for abstract argumentation. *AI*, 206:53–78, 2014.
29. Wolfgang Dvorák, Reinhard Pichler, and Stefan Woltran. Towards fixed-parameter tractable algorithms for argumentation. In *KR*, 2010.
30. Wolfgang Dvorák and Stefan Woltran. Complexity of semi-stable and stage semantics in argumentation frameworks. *Information Processing Letters*, 110(11):425–430, 2010.
31. Marcelo A. Falappa, Alejandro Javier Garcia, Gabriele Kern-Isberner, and Guillermo Ricardo Simari. On the evolving relation between belief revision and argumentation. *The Knowledge Engineering Review*, 26(1):35–43, 2011.
32. Bettina Fazzinga, Sergio Flesca, and Francesco Parisi. Efficiently estimating the probability of extensions in abstract argumentation. In *SUM*, pages 106–119, 2013.
33. Bettina Fazzinga, Sergio Flesca, and Francesco Parisi. On the complexity of probabilistic abstract argumentation frameworks. *ACM Trans. Comput. Log.*, 16(3):22, 2015.
34. Bettina Fazzinga, Sergio Flesca, and Francesco Parisi. On efficiently estimating the probability of extensions in abstract argumentation frameworks. *Int. J. Approx. Reasoning*, 69:106–132, 2016.
35. Bettina Fazzinga, Sergio Flesca, Francesco Parisi, and Adriana Pietramala. PARTY: A mobile system for efficiently assessing the probability of extensions in a debate. In *DEXA*, pages 220–235, 2015.
36. Sarah Alice Gaggl and Norbert Manthey. ASPARTIX-D ready for the competition, 2015.
37. Sergio Greco and Francesco Parisi. Efficient computation of deterministic extensions for dynamic abstract argumentation frameworks. In *ECAI*, pages 1668–1669, 2016.
38. Sergio Greco and Francesco Parisi. Incremental computation of deterministic extensions for dynamic argumentation frameworks. In *JELIA*, pages 288–304, 2016.
39. Sergio Greco and Francesco Parisi. Incremental computation of grounded semantics for dynamic abstract argumentation frameworks. In *COREDEMA*, pages 66–81, 2016.
40. Nadin Kökciyan, Nefise Yaglikci, and Pinar Yolum. Argumentation for resolving privacy disputes in online social networks: (extended abstract). In *AAMAS*, pages 1361–1362, 2016.
41. Bei Shui Liao, Li Jin, and Robert C. Koons. Dynamics of argumentation systems: A division-based method. *AI*, 175(11):1790–1814, 2011.
42. Sanjay Modgil and Henry Prakken. Revisiting preferences and argumentation. In *IJCAI*, pages 1021–1026, 2011.
43. Emilia Oikarinen and Stefan Woltran. Characterizing strong equivalence for argumentation frameworks. *AI*, 175(14-15):1985–2009, 2011.
44. Iyad Rahwan and Guillermo R. Simari. *Argumentation in Artificial Intelligence*. Springer Publishing Company, Incorporated, 1st edition, 2009.
45. Serena Villata, Guido Boella, Dov M. Gabbay, and Leendert W. N. van der Torre. Modelling defeasible and prioritized support in bipolar argumentation. *Ann. Math. Artif. Intell.*, 66(1-4):163–197, 2012.
46. Yuming Xu and Claudette Cayrol. The matrix approach for abstract argumentation frameworks. In *Proc. of International Workshop on Theory and Applications of Formal Argumentation (TAFA)*, pages 243–259, 2015.