Foundations of Implementations for Formal Argumentation

Federico Cerutti
Cardiff University, UK
CeruttiF@cardiff.ac.uk

Sarah A. Gaggl
Technische Universität Dresden, Germany
sarah.gaggl@tu-dresden.de

Matthias Thimm
Universität Koblenz-Landau, Germany
thimm@uni-koblenz.de

Johannes P. Wallner
Technische Universität Wien, Austria
wallner@dbai.tuwien.ac.at

Abstract

We survey the current state of the art of general techniques, as well as specific software systems for solving tasks in abstract argumentation frameworks, structured argumentation frameworks, and approaches for visualizing and analysing argumentation. Furthermore, we discuss challenges and promising techniques such as parallel processing and approximation approaches. Finally, we address the issue of evaluating software systems empirically with links to the International Competition on Computational Models of Argumentation.

1 Introduction

Compared to related areas such as argumentation theory [van Eemeren *et al.*, 2014], research conducted in the formal argumentation community seeks *formal* accounts of argumentation with explicit links to knowledge representation and reasoning, and artificial intelligence [Brachman and Levesque, 2004; Russell and Norvig, 2003]. An

important feature for these accounts is *computability*, i. e., the possibility to provide algorithmic methods to solve problems.

In this paper, we survey general computational techniques and concrete implementations for solving problems related to formal argumentation. We distinguish between: (1) Approaches to abstract argumentation frameworks, (2) Approaches to structured argumentation frameworks (such as ASPIC+ and DeLP), and (3) Other approaches, including semi-formal systems related to visualization of argumentation processes or exchange of arguments on the web.

Between them, the most active research direction within the formal argumentation community[1] is devoted to the first category—algorithms and systems for abstract argumentation frameworks—reviewed in Section 2. The relevant computational problems and their (high) computational complexity have been studied in e. g. [Dunne and Wooldridge, 2009]. Here, we focus on the algorithmic issues and techniques to handle the high computational complexity of some of those problems. The development of implementations has accelerated recently, also due to the foundation of the *International Competition on Computational Models of Argumentation* (ICCMA):[2] besides discussing general techniques we will also survey concrete systems.

We will also look at techniques and systems solving problems for structured approaches to formal argumentation. Due to the multitude of different approaches to structured argumentation, computational techniques and algorithms are usually tailored towards specific approaches. We will discuss them in Section 3.

In order to complement our survey we will also have a brief look at other systems that incorporate some kind of (semi-)formal argumentation such as argument schemes and argumentation technologies (or *debating technologies*) which are popular in many other fields besides the formal argumentation community. In contrast to the perspective of artificial intelligence and knowledge representation usually taken by researchers in the formal argumentation community, the focus of the systems in this third category is on human-computer interaction and supporting critical thinking. We will discuss these systems in Section 4, concluding the survey part of this paper.

In Section 5 we will look beyond the current state of the art of algorithms and systems and current challenges for the development of systems, such as parallelization and approximation algorithms, focusing on abstract and structured argumentation approaches. A recent effort to promote the development of systems for solving argumentation tasks is the ICCMA: the first instance of the competition took place

[1] Approaches in the third category are also addressed by other research communities such as human-computer-interaction and web science.

[2] `http://argumentationcompetition.org` (on 27/04/2017).

in 2015 [Thimm *et al.*, 2016]. We will discuss this competition and general methods for empirically evaluating systems in Section 6.

2 Abstract Argumentation Implementations

In this section we will give an overview of implementations for abstract Argumentation Frameworks (AFs) following the approach from Dung [Dung, 1995] and give an overview of existing systems for Dung's framework as well as for some related formalisms.

One can divide the implementations for abstract AFs into two categories: the *reduction-based approach* and the *direct approach*. The former one reduces the problem at hand into another formalism to exploit existing solvers from the other formalism. We will discuss this method and the dedicated implementations in the following subsection. The other possibility is to design algorithms to directly solve the problem. This implementation method will be presented in Subsection 2.2. For a more detailed discussion on implementation methods for AFs we refer to [Charwat *et al.*, 2015].

Before we go into details on the different approaches we briefly introduce the background on abstract argumentation [Dung, 1995] and the notation we will use in this section. For comprehensive surveys on argumentation semantics the interested reader is referred to [Baroni *et al.*, 2011a].

Definition 2.1. *An* argumentation framework *(AF) is a pair $AF = \langle Ar, att \rangle$, where Ar is a finite set of arguments and $att \subseteq Ar \times Ar$ is the attack relation. The pair $\langle a, b \rangle \in Ar$ means that a attacks b. A set $S \subseteq Ar$ of arguments attacks b (in AF), if there is an $a \in S$, such that $\langle a, b \rangle \in att$. An argument $a \in Ar$ is* defended *by $S \subseteq Ar$ (in AF) iff, for each $b \in Ar$, it holds that, if $\langle b, a \rangle \in att$, then S attacks b (in AF). Given a set $S \subseteq Ar$, $S^+ = \{a \in Ar \mid \langle b, a \rangle \in att, b \in S\}$, and $S^- = \{a \in Ar \mid \langle a, b \rangle \in att, b \in S\}$.*

The inherent conflicts between the arguments are solved by selecting subsets of arguments, where a semantics σ assigns a collection of sets of arguments to an argumentation framework AF. The basic requirement for all semantics is that none of the selected arguments attack each other[3].

[3]We concentrate here on the basic Dung-style argumentation framework, and do not consider approaches like value-based argumentation frameworks (VAFs) [Bench-Capon, 2003] or inconsistency tolerant semantics [Dunne *et al.*, 2009] (where this requirement does not hold), as our main focus is on implementation methods.

Definition 2.2. *Let $AF = \langle Ar, att \rangle$ be an AF. A set $S \subseteq Ar$ is said to be* conflict-free *(in AF), if there are no $a, b \in S$, such that $\langle a, b \rangle \in att$. We denote the collection of sets which are conflict-free (in AF) by $cf(F)$.*

Definition 2.3. *Let $AF = \langle Ar, att \rangle$ be an AF, then $S \in cf(AF)$ is*

- *a* stable *extension, i. e. $S \in \mathcal{E}_{\mathcal{ST}}(AF)$, if each $a \in Ar \setminus S$ is attacked by S in AF;*
- *an* admissible *extension, i. e. $S \in \mathcal{E}_{\mathcal{AD}}(AF)$, if each $a \in S$ is defended by S;*
- *a* preferred *extension, i. e. $S \in \mathcal{E}_{\mathcal{PR}}(AF)$, if $S \in \mathcal{E}_{\mathcal{AD}}(AF)$ and for each $T \in \mathcal{E}_{\mathcal{AD}}(AF)$, $S \not\subset T$;*
- *a* complete *extension, i. e. $S \in \mathcal{E}_{\mathcal{CO}}(AF)$, if $S \in \mathcal{E}_{\mathcal{AD}}(AF)$ and for each $a \in Ar$ defended by S it holds that $a \in S$;*
- *the* grounded *extension (of AF), i. e. the unique set $S = \mathcal{E}_{\mathcal{GR}}(AF)$, if $S \in \mathcal{E}_{\mathcal{CO}}(AF)$ and for each $T \in \mathcal{E}_{\mathcal{CO}}(AF)$, $T \not\subset S$.*

The typical problems of interest in abstract argumentation are the following decision problems for given $AF = \langle Ar, att \rangle$, a semantics σ, $a \in Ar$ and $S \subseteq Ar$:

- Verification Ver_σ: is $S \in \mathcal{E}_\sigma(AF)$?
- Credulous acceptance $Cred_\sigma$: is a contained in at least one σ extension of AF?
- Skeptical acceptance $Skept_\sigma$: is a contained in every σ extension of AF?
- Non-emptiness $Exists_\sigma^{\neg\emptyset}$: is there any $S \in \mathcal{E}_\sigma(AF)$ for which $S \neq \emptyset$?

Computational complexity of decision problems on AFs is well-studied. For an overview see e. g. [Dunne and Wooldridge, 2009].

2.1 Reduction-based Implementations

Reduction-based implementations are a very common approach as one benefits from very sophisticated solvers developed and improved by several communities. The underlying idea is to exploit existing efficient software which has originally been developed for other purposes. To this end, one has to formalize the reasoning problems within other formalisms such as constraint-satisfaction problems (CSP) [Rossi *et al.*, 2006], propositional logic [Biere *et al.*, 2009] or answer-set programming (ASP) [Brewka *et al.*, 2011]. The general methodology of the reduction-based approach is to reduce the problem at hand to the target formalism, run the solver (of the target formalism) and interpret the output as the solutions of the original problem, as depicted in Figure 1.

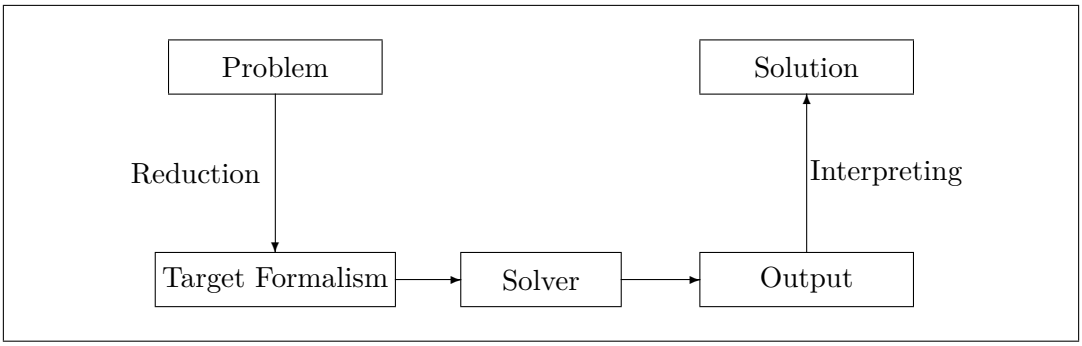


Figure 1: Reduction-based approach.

2.1.1 SAT-based Approach

Reductions to SAT have been first advocated in [Dunne and Bench-Capon, 2002] and [Dunne and Bench-Capon, 2003] and then further developed by Besnard and Doutre [Besnard and Doutre, 2004], and later extended by means of quantified propositional logic [Arieli and Caminada, 2013; Egly and Woltran, 2006]. Several prominent systems use reductions to SAT, such as **Cegartix** [Dvořák *et al.*, 2014] and **{j}ArgSemSAT** [Cerutti *et al.*, 2014c; Cerutti *et al.*, 2016b; Cerutti *et al.*, 2017] that both rely on iterative calls to SAT solvers for argumentation semantics of high complexity (i. e. being located on the second level of the polynomial hierarchy). Further SAT-based systems include **prefMaxSAT** [Vallati *et al.*, 2015; Faber *et al.*, 2016], which uses the MaxSAT approach for the computation of preferred semantics; the **LabSATSolver** [Beierle *et al.*, 2015], which uses propositional formulas based on labellings and, for the subset maximization task, the PrefSat Algorithm [Cerutti *et al.*, 2014a] that then become **{j}ArgSemSAT**. The system **CoQuiAAS** [Lagniez *et al.*, 2015], which also uses SAT encodings for some semantics, will be explained in Subsection 2.1.2, as the maximization task necessary for instance for preferred semantics is performed by means of constraint programming.

Background. Let us consider a set of propositional variables (or atoms) $\mathcal{P}$ and the connectives $\wedge, \vee, \rightarrow$ and $\neg$, denoting respectively the logical conjunction, disjunction, material implication and negation. The constants $\top$ and $\bot$ denote respectively *true* and *false*. In addition, we consider quantified Boolean formulae (QBF) with the universal quantifier $\forall$ and the existential quantifier $\exists$ (both over atoms), that is, given a formula ϕ, then $Qp\phi$ is a QBF, with $Q \in \{\forall, \exists\}$ and $p \in \mathcal{P}$. $Q\{p_1, \ldots, p_n\}\phi$ is a shorthand for $Qp_1 \cdots Qp_n\phi$. A propositional variable p in a QBF ϕ is free if it does not occur within the scope of a quantifier Qp and bound otherwise. If ϕ

contains no free variable, then ϕ is said to be closed and otherwise open. We will write $\phi[p/\psi]$ to denote the result of uniformly substituting each free occurrence of p with ψ in formula ϕ.

An interpretation $I \subseteq \mathcal{P}$ defines for each propositional variable a truth assignment where $p \in I$ indicates that p evaluates to true while $p \notin I$ indicates that p evaluates to false. This generalizes to arbitrary formulae in the standard way: Given a formula ϕ and an interpretation I, then ϕ evaluates to true under I (i. e., I satisfies ϕ) if one of the following holds (with $p \in \mathcal{P}$).

- $\phi = p$ and $p \in I$
- $\phi = \neg p$ and $p \notin I$
- $\phi = \psi_1 \wedge \psi_2$ and both ψ_1 and ψ_2 evaluate to true under I
- $\phi = \psi_1 \vee \psi_2$ and one of ψ_1 and ψ_2 evaluates to true under I
- $\phi = \psi_1 \rightarrow \psi_2$ and ψ_1 evaluates to false or ψ_2 evaluates to true under I
- $\phi = \exists p\psi$ and one of $\psi[p/\top]$ and $\psi[p/\bot]$ evaluates to true under I
- $\phi = \forall p\psi$ and both $\psi[p/\top]$ and $\psi[p/\bot]$ evaluate to true under I.

If an interpretation I satisfies a formula ϕ, denoted by $I \models \phi$, we say that I is a model of ϕ.

Reductions to propositional logic. The first reduction-based approach [Besnard and Doutre, 2004; Egly and Woltran, 2006] we consider here uses propositional logic formulae (without quantifiers) to encode the problem of finding admissible sets. Given an AF $AF = \langle Ar, att \rangle$, for each argument $a \in Ar$ a propositional variable v_a is used. Then, $S \subseteq Ar$ is an extension under semantics σ iff $\{v_a \mid a \in S\} \models \phi$, with ϕ being a propositional formula that evaluates AF AF under semantics σ (below we will present in detail how to translate AFs into formulae). Formally, the correspondence between sets of extensions and models of a propositional formula can be defined as follows.

Definition 2.4. *Let $\mathcal{T} \subseteq 2^{Ar}$ be a collection of sets of arguments and let $\mathcal{I} \subseteq 2^{\mathcal{P}}$ be a collection of interpretations. We say that $\mathcal{T}$ and $\mathcal{I}$ correspond to each other, in symbols $\mathcal{T} \cong \mathcal{I}$, if*

1. *for each $S \in \mathcal{T}$, there exists an $I \in \mathcal{I}$, such that $\{a \mid v_a \in I, a \in Ar\} = S$;*
2. *for each $I \in \mathcal{I}$, there exists an $S \in \mathcal{T}$, such that $\{a \mid v_a \in I, a \in Ar\} = S$; and*

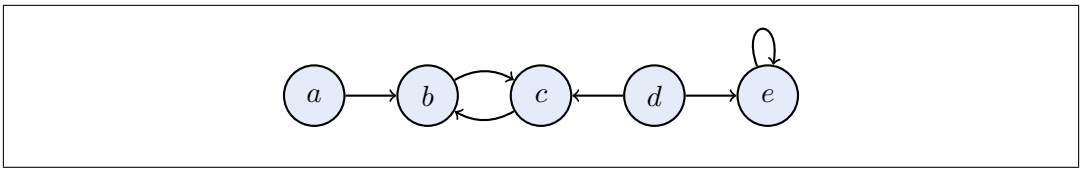


Figure 2: Example argumentation framework.

3. $|\mathcal{T}| = |\mathcal{I}|$.

Given an AF $AF = \langle Ar, att \rangle$, the following formula can be used to solve the enumeration problem of admissible semantics.

$$adm_{Ar,att} := \bigwedge_{a \in Ar} \Big((v_a \rightarrow \bigwedge_{\langle b,a \rangle \in att} \neg v_b) \wedge (v_a \rightarrow \bigwedge_{\langle b,a \rangle \in att} (\bigvee_{\langle c,b \rangle \in att} v_c)) \Big) \tag{1}$$

Note that an empty conjunction is treated as $\top$, whereas the empty disjunction is treated as $\bot$.

The models of $adm_{Ar,att}$ now correspond to the admissible sets of AF, i. e., we have $\mathcal{E}_{\mathcal{AD}}(AF) \cong \{M \mid M \models adm_{Ar,att}\}$. The first conjunction in (1) ensures that the resulting set of arguments is conflict-free, that is, whenever we accept an argument a (i. e., v_a evaluates to true under a model), all its attackers cannot be accepted. The second conjunct expresses the defense of arguments by stating that, if we accept a, then for each attacker b, some defender c must be accepted as well.

Example 2.5. *Let $AF = \langle Ar, att \rangle$ be an AF with $Ar = \{a, b, c, d, e\}$ and $att = \{\langle a, b \rangle, \langle b, c \rangle, \langle c, b \rangle, \langle d, c \rangle, \langle d, e \rangle, \langle e, e \rangle\}$ as depicted in Figure 2. The corresponding propositional formula $adm_{Ar,att}$ is as follows.*

$$\begin{aligned} adm_{Ar,att} \equiv & (v_a \rightarrow \top) \wedge \\ & (v_b \rightarrow (\neg v_a \wedge \neg v_c)) \wedge \\ & (v_c \rightarrow (\neg v_b \wedge \neg v_d)) \wedge \\ & (v_d \rightarrow \top) \wedge \\ & (v_e \rightarrow (\neg v_d \wedge \neg v_e)) \wedge \\ & (v_a \rightarrow \top) \wedge \\ & (v_b \rightarrow (\bot \wedge (v_b \vee v_d))) \wedge \\ & (v_c \rightarrow ((v_a \vee v_c) \wedge \bot)) \wedge \\ & (v_d \rightarrow \top) \wedge \\ & (v_e \rightarrow (\bot \wedge v_d)) \end{aligned}$$

It is easy to see that $\mathcal{I} = \{I_1, I_2, I_3, I_4\}$ *represents the set of models of* $adm_{Ar,att}$*, where*

$$I_1 = \{v_a \mapsto \bot, v_b \mapsto \bot, v_c \mapsto \bot, v_d \mapsto \bot, v_e \mapsto \bot\},$$
$$I_2 = \{v_a \mapsto \top, v_b \mapsto \bot, v_c \mapsto \bot, v_d \mapsto \bot, v_e \mapsto \bot\},$$
$$I_3 = \{v_a \mapsto \bot, v_b \mapsto \bot, v_c \mapsto \bot, v_d \mapsto \top, v_e \mapsto \bot\},$$
$$I_4 = \{v_a \mapsto \top, v_b \mapsto \bot, v_c \mapsto \bot, v_d \mapsto \top, v_e \mapsto \bot\}.$$

As $\mathcal{T} = \{S_1, S_2, S_3, S_4\}$*, with* $S_1 = \{\}$*,* $S_2 = \{a\}$*,* $S_3 = \{d\}$ *and* $S_4 = \{a, d\}$*, is the set of all admissible sets of AF we clearly have the correspondence* $\mathcal{I} \cong \mathcal{T}$ *as desired.*

Reductions to quantified Boolean formulas. For problems beyond NP we require a more expressive formalism than propositional logic. For this purpose we consider QBFs. In the following we will show how to reduce a given AF into a QBF such that the models of the QBF correspond to the preferred extensions of the AF [Egly and Woltran, 2006].

In order to realize the maximality check for preferred semantics we need to be able to compare two sets of atoms w.r.t. set inclusion. Consider the formula

$$Ar < Ar' := \bigwedge_{a \in Ar} (v_a \rightarrow v_{a'}) \wedge \neg \bigwedge_{a' \in Ar'} (v_{a'} \rightarrow v_a),$$

where $Ar' = \{a' \mid a \in Ar\}$. This formula ensures that any model $M \models (Ar < Ar')$ satisfies $\{a \in Ar \mid v_a \in M\} \subset \{a \in Ar \mid v_{a'} \in M\}$. Now we can state the QBF $prf_{Ar,att}$ for preferred extensions. Let the quantified variables be $Ar'_v = \{v_{a'} \mid a' \in Ar'\}$ and $att' = \{\langle a', b' \rangle \mid \langle a, b \rangle \in att\}$. Then

$$prf_{Ar,att} := adm_{Ar,att} \wedge \neg \exists Ar'_v((Ar < Ar') \wedge adm_{Ar',att'}) \,. \tag{2}$$

Thus, for any AF $AF = \langle Ar, att \rangle$ an interpretation I is a model of $prf_{Ar,att}$ iff it satisfies the formula for admissible sets and there exists no "bigger" interpretation I' that also satisfies the the corresponding formula for admissible sets.

Example 2.5 (continued) *There,* I_4 *is the only interpretation which satisfies the QBF* $prf_{Ar,att}$ *and the corresponding set* S_4 *is the only preferred extension of AF.*

Similar approaches have been proposed by Arieli and Caminada in [Arieli and Caminada, 2013] and for Abstract Dialectical Frameworks by Diller *et al.* in **QADF** [Diller *et al.*, 2015].

Iterative application of SAT solvers. The final approach we outline here is based on the idea of iteratively searching for models of propositional formulae and has been instantiated in the systems **{j}ArgSemSAT** [Cerutti *et al.*, 2014a; Cerutti *et al.*, 2014c; Cerutti *et al.*, 2016b] and **Cegartix** [Dvořák *et al.*, 2014]. The idea is to use an algorithm which iteratively constructs formulae and searches for models of these formulae. A new formula is generated based on the model of the previous one (or based on the fact that the previous formula is unsatisfiable). At some point the algorithm reaches a final decision and terminates.

The iterative approach is suitable when the problem to be solved cannot be decided in general—under standard complexity theoretic assumptions—by the satisfiability of a single propositional formula, constructible in polynomial time without quantifiers. This is, for instance, the case with skeptical acceptance under preferred semantics, where the corresponding decision problem is Π_2^P-complete. Instead of reducing the problem to a single QBF formula, the solving task is delegated to the iterative scheme of an algorithm querying a SAT solver multiple times.

The algorithms for preferred semantics work roughly as follows. To compute preferred extensions we traverse the search space of a computationally simpler semantics. For instance, we can iteratively search for admissible sets or complete extensions and iteratively extend them until we reach a maximal set, which is a preferred extension. By generating a new candidate for an admissible set or a complete extension, which is not contained in an already visited preferred extension, we can enumerate all preferred extensions in this manner. This allows answering both credulous and skeptical reasoning problems as well.

For deciding e. g. skeptical acceptance of an argument under preferred semantics one requires, in the worst case, an exponential number of calls to the SAT solver—under standard complexity-theoretic assumptions. However, the actual number of SAT calls in the iterative SAT scheme depends on the number of preferred extensions of the given AF, see [Dvořák *et al.*, 2014].

In the following, we sketch the **Cegartix** approach from [Dvořák *et al.*, 2014] for skeptical acceptance of an argument under preferred semantics. The algorithm returns YES if a is skeptically accepted, NO otherwise. To do so we try to construct a preferred extension which does not contain a. If this is possible we know that a is not skeptically accepted under preferred semantics, otherwise the algorithm returns YES.

1) Check if there is an interpretation I satisfying the formula ϕ (initially $\phi = adm_{Ar,att} \wedge \neg v_a$). If such an interpretation I exists, go to Step 2. Otherwise there is no admissible set which does not contain a, and the algorithm returns YES.

2) Try to add new arguments to I by updating it (as long as possible) with interpretations satisfying the formula

$$adm_{Ar,att} \wedge \neg v_a \wedge (\bigwedge_{a \in Ar, v_a \in I} v_a) \wedge (\bigvee_{a \in Ar, v_a \notin I} v_a).$$

3) For the maximized interpretation I, check if it is possible to add the argument a to it by checking for models of the formula

$$\phi' = adm_{Ar,att} \wedge (\bigwedge_{a \in Ar, v_a \in I} v_a) \wedge (\bigvee_{a \in Ar, v_a \notin I} v_a).$$

If there is an interpretation I' satisfying ϕ', there is a preferred extension which contains a. Otherwise, there is a preferred extension, namely the one represented by the interpretation I, which does not contain the argument a. In this case the algorithm outputs NO and terminates.

4) The algorithm continues with the search for a different preferred extension which does not contain the arguments of I by modifying the formula ϕ as follows:

$$\phi' = \phi \wedge (\bigvee_{a \in Ar, v_a \notin I} v_a).$$

Go to Step 1.

Example 2.5 (continued) *Let us exemplify the algorithm of* ***Cegartix*** *on our AF from Example 2.5, where we want to decide skeptical acceptance of the argument d. We know that there are four interpretations satisfying the formula for admissible sets and only I_1 and I_2 satisfy the formula $\phi = adm_{Ar,att} \wedge \neg v_d$ of Step 1. Let us continue with $I = I_1$ which represents the admissible set $S_1 = \{\}$. In Step 2, we update I by setting v_a to $\top$. Remember, we cannot set v_d to $\top$ as ϕ contains the clause $\neg v_d$. In Step 3 we check if there is an I' satisfying the formula $\phi' = adm_{Ar,att} \wedge v_a \wedge (v_b \vee v_c \vee v_d \vee v_e)$. Indeed $I' = \{v_a \mapsto \top, v_b \mapsto \bot, v_c \mapsto \bot, v_d \mapsto \top, v_e \mapsto \bot\}$ is a model of ϕ', thus we constructed a preferred extensions, namely $S = \{a, d\}$ containing the argument a. In Step 4 we update our formula to $\phi = adm_{Ar,att} \wedge \neg v_d \wedge (v_b \vee v_c \vee v_d \vee v_e)$ and go to Step 1. In the next iteration, we check the new formula ϕ for models, but as ϕ is not satisfiable the algorithm outputs YES and terminates.*

One can use a modified version of the above algorithm to enumerate all preferred extensions. More concretely, one can add the obtained preferred extension from Step 2 to the output-set and then update the formula as in Step 4, while omitting Step 3. Further, the conjunct containing a negated variable for the queried argument

must be removed. The PrefSat approach [Cerutti *et al.*, 2014a] as implemented in the system **{j}ArgSemSAT** [Cerutti *et al.*, 2014c; Cerutti *et al.*, 2016b] uses this method to compute all preferred labellings.

2.1.2 Reductions to Constraint Satisfaction Problems

In the following we introduce reductions to another target formalism, namely Constraint Satisfaction Problems (CSPs) [Rossi *et al.*, 2006], which allow to solve combinatorial search problems. Reductions to CSP have been addressed by Amgoud and Devred [Amgoud and Devred, 2011] and Bistarelli, Pirolandi, and Santini [Bistarelli *et al.*, 2009; Bistarelli and Santini, 2010; Bistarelli and Santini, 2011; Bistarelli and Santini, 2012b; Bistarelli and Santini, 2012a]; the latter works led to the development of the **ConArg** system. Further systems based on CSP are **CoQuiAAS** [Lagniez *et al.*, 2015] and **ASGL** [Sprotte, 2015]. The approach of CSP is inherently related to propositional logic reductions as introduced in Subsection 2.1.1, see also [Walsh, 2000] for a formal analysis of the relation between the two approaches.

A CSP can generally be described by a triple (X, D, C), where $X = \{x_1, \ldots, x_n\}$ is the set of variables, $D = \{D_1, \ldots, D_n\}$ is a set of finite domains for the variables and $C = \{c_1, \ldots, c_m\}$ a set of constraints. Each constraint c_i is a pair (h_i, H_i) where $h_i = (x_{i1}, \ldots, x_{ik})$ is a k-tuple of variables and H_i is a k-ary relation over D. In particular, H_i is a subset of all possible variable values representing the allowed combinations of simultaneous values for the variables in h_i. An assignment v is a mapping that assigns to every variable $x_i \in X$ an element $v(x_i) \in D_i$. An assignment v satisfies a constraint $((x_{i1}, \ldots, x_{ik}), H_i) \in C$ iff $(v(x_{i1}), \ldots, v(x_{ik})) \in H_i$. Finally, a solution is an assignment v to all variables such that all constraints are satisfied, denoted by $(v(x_1), \ldots, v(x_n))$.

Finding a valid assignment of a CSP is in general NP-complete. Nevertheless, several programming libraries support constraint programming, like ECLiPSe,[4] SWI Prolog,[5] Gecode,[6] JaCoP,[7] Choco,[8] Turtle[9] (just to mention some of them) and allow for efficient implementations of CSPs. These constraint programming solvers make use of techniques like backtracking and local search.

Given an AF $AF = \langle Ar, att \rangle$, the associated CSP (X, D, C) is specified as $X = Ar$ and for each $a_i \in X$, $D_i = \{0, 1\}$. The constraints are formulated depending on the specific semantics σ. For example, solutions that correspond to conflict-free sets

[4] `http://eclipseclp.org/` (on 27/04/2017).
[5] `http://www.swi-prolog.org/` (on 27/04/2017).
[6] `http://www.gecode.org/` (on 27/04/2017).
[7] `https://github.com/radsz/jacop` (on 27/04/2017).
[8] `http://www.choco-solver.org/` (on 27/04/2017).
[9] `https://github.com/timfel/turtle` (on 27/04/2017).

can be obtained by defining a constraint for each pair of arguments a and b with $\langle a, b\rangle \in att$, where the two variables may not be set to 1 at the same time. Here, the constraint is of the form $((a, b), ((0, 0), (0, 1), (1, 0)))$ which is equivalent to the cases when the propositional formula $(a \rightarrow \neg b)$ evaluates to true.

In the following, we will use the notation from [Amgoud and Devred, 2011], because it reflects the similarities between the CSP approach and the reductions to propositional logic as outlined above.

For admissible semantics we get the following constraints.

$$C_{\mathcal{AD}} = \Big\{(a \rightarrow \bigwedge_{b:\langle b,a\rangle \in att} \neg b) \wedge (a \rightarrow \bigwedge_{b:\langle b,a\rangle \in att} (\bigvee_{c:\langle c,b\rangle \in att} c)) \;\Big|\; a \in Ar\Big\} \tag{3}$$

The first part ensures conflict-free sets and the second part encodes the defense of arguments. Then, for an AF $AF = \langle Ar, att\rangle$ and its associated admissible CSP $(X, D, C_{\mathcal{AD}})$, $(v(x_1), \ldots, v(x_n))$ is a solution of the CSP iff the set $\{x_j, \ldots, x_k\}$ s.t. $v(x_i) = 1$ is an admissible set in AF.

Example 2.5 (continued) *For our AF we obtain the following admissible CSP* $(X, D, C_{\mathcal{AD}})$. *$X = A$, for each $a_i \in X$ we have $D_i = \{0, 1\}$ and*

$$\begin{aligned} C_{\mathcal{AD}} = \{&(a \rightarrow \top) \wedge (a \rightarrow \top), (b \rightarrow \neg a \wedge \neg c) \wedge (b \rightarrow \bot \wedge d),\\ &(c \rightarrow \neg b \wedge \neg d) \wedge (c \rightarrow (a \vee c) \wedge \bot), (d \rightarrow \top) \wedge (d \rightarrow \top),\\ &(e \rightarrow \neg d \wedge \neg e) \wedge (e \rightarrow \bot \vee d)\}. \end{aligned}$$

This CSP has the following solutions: $(0, 0, 0, 0, 0)$, $(1, 0, 0, 0, 0)$, $(0, 0, 0, 1, 0)$, $(1, 0, 0, 1, 0)$ *which correspond to the admissible sets of AF, namely* $\{\}, \{a\}, \{d\}$ *and* $\{a, d\}$.

Most CSP solvers do not support subset maximization. Thus, for preferred semantics, Bistarelli and Santini [2012a] propose an approach that iteratively computes admissible/complete extensions and adds constraints to exclude certain sets, such that one finally obtains the preferred extensions.

Reductions to Weighted Partial Max-SAT. This approach has been implemented in **CoQuiAAS** [Lagniez *et al.*, 2015] and in **prefMaxSAT** [Vallati *et al.*, 2015; Faber *et al.*, 2016] and is particularly tailored to maximization problems as needed to compute preferred semantics. A *Weighted Partial Max-SAT* problem is a problem which maximizes the sum of weights associated to constraints, where the term *partial* means that some constraints have an infinite weight, which means they need to be satisfied. The system **CoQuiAAS** uses a SAT-Solver but the problem of Weighted Partial Max-SAT is more related to Constraint Programming, therefore

we discuss this approach in this section, but of course it is also closely related to the previous section.

The computation of preferred extensions in [Lagniez *et al.*, 2015] is based on complete extensions which are obtained as follows. For an AF $AF = \langle Ar, att \rangle$ and for each $a \in Ar$ we use a boolean variable v_a.

$$comp_{Ar,att} := \bigwedge_{a \in Ar} \Big(v_a \rightarrow (\bigwedge_{b \in Ar : \langle b,a \rangle \in att} \neg v_b) \wedge (v_a \leftrightarrow (\bigwedge_{b \in Ar : \langle b,a \rangle \in att} \bigvee_{c \in Ar : \langle c,b \rangle \in att} v_c)) \Big)$$

The models of $comp_{Ar,att}$ correspond to the complete extensions of AF, i. e., we have $\mathcal{E}_{\mathcal{CO}}(F) \cong \{M | M \models comp_{Ar,att}\}$. Then, the maximal models of $comp_{Ar,att}$ correspond to the preferred extensions of AF. To obtain these one uses the concept of a *maximal satisfiable subset* (MSS). For a set of formulas $\mathcal{F}$ the set of formulas $\mathcal{S} \subseteq \mathcal{F}$ is a MSS iff $\mathcal{S}$ is satisfiable and for each $c \in \mathcal{F} \setminus \mathcal{S}$, $\mathcal{S} \cup \{c\}$ is unsatisfiable.

Now, the computation of preferred extension reduces to the computation of MSSs of the sets of weighted formulas

$$prf_{Ar,att} = \{(comp_{Ar,att}, +\infty), (a_1, 1), \ldots, (a_n, 1)\}$$

where $a_1, \ldots, a_n \in Ar$.

2.1.3 Reductions to Answer Set Programming

The use of logic programming to solve abstract argumentation problems has been initiated by several authors (the survey article by Toni and Sergot [Toni and Sergot, 2011] provides a good overview), including the approach proposed by Nieves *et al.* [Nieves *et al.*, 2008], where the program is re-computed for every input instance; Wakaki and Nitta [Wakaki and Nitta, 2008], who use labelling-based semantics; and the approach by Egly *et al.* [Egly *et al.*, 2010a], which follows extension-based semantics. Here, we focus on the latter—the ASPARTIX approach—[Egly *et al.*, 2010a; Dvořák *et al.*, 2013a; Gaggl *et al.*, 2015], which relies on a query-based implementation where the argumentation framework to be evaluated is provided as an input database. From this point of view, the SAT or CSP methods can be seen as a compiler-like approach to abstract argumentation, while the ASP method acts like an interpreter.

A large collection of such ASP queries is provided by the **ASPARTIX-D** and **ASPARTIX-V** systems. Furthermore, the **DIAMOND** system [Ellmauthaler and Strass, 2014] for *Abstract Dialectical Frameworks* (ADFs), as well as the **GERD** system [Dvořák *et al.*, 2015] for *extended argumentation frameworks* (EAFs) are

based on ASP. In the following, we first give a brief introduction to ASP. We then present how the computation of admissible sets can be encoded in ASP. In order to obtain preferred extensions, it is necessary to check for subset-maximality of admissible sets. We will give pointers to the literature on several approaches for the subset-maximality check and refer to [Charwat *et al.*, 2015] for a detailed discussion.

Background. Let us consider disjunctive logic program under the answer-set semantics [Gelfond and Lifschitz, 1991].[10] We fix a countable set $\mathcal{U}$ of *(domain) elements*, also called *constants*, and suppose a total order $<$ over the domain elements. An *atom* is an expression $p(t_1, \ldots, t_n)$, where p is a *predicate* of arity $n \geq 0$ and each t_i is either a variable or an element from $\mathcal{U}$. An atom is *ground* if it is free of variables. $B_{\mathcal{U}}$ denotes the set of all ground atoms over $\mathcal{U}$.

A *(disjunctive) rule* r with $n \geq 0$, $m \geq k \geq 0$, $n + m > 0$ is of the form

$$a_1 \vee \cdots \vee a_n \leftarrow b_1, \ldots, b_k, \ not\, b_{k+1}, \ldots, \ not\, b_m$$

where $a_1, \ldots, a_n, b_1, \ldots, b_m$ are atoms, and "*not*" stands for *default negation*. An atom a is a positive literal, while *not* a is a default-negated literal. The *head* of r is the set $H(r) = \{a_1, \ldots, a_n\}$ and the *body* of r is $B(r) = B^+(r) \cup B^-(r)$ with $B^+(r) = \{b_1, \ldots, b_k\}$ and $B^-(r) = \{b_{k+1}, \ldots, b_m\}$. A rule r is *normal* if $n \leq 1$ and a *constraint* if $n = 0$. A rule r is *safe* if each variable in $H(r)$ occurs in $B^+(r)$. A rule r is *ground* if no variable occurs in r. A *fact* is a ground rule with a single literal in the head and with an empty body. An *(input) database* is a set of facts. A program is a finite set of safe disjunctive rules. For a program π and an input database D, we often write $\pi(D)$ instead of $D \cup \pi$. If each rule in a program is normal (resp. ground), we call the program normal (resp. ground).

For any program π, let U_π be the set of all constants appearing in π. $Gr(\pi)$ is the set of rules $r\tau$ obtained by applying, to each rule $r \in \pi$, all possible substitutions τ from the variables in r to elements of U_π. An *interpretation* $I \subseteq B_{\mathcal{U}}$ *satisfies* a ground rule r iff $H(r) \cap I \neq \emptyset$ whenever $B^+(r) \subseteq I$ and $B^-(r) \cap I = \emptyset$. I satisfies a ground program π, if each $r \in \pi$ is satisfied by I. A non-ground rule r (resp. a program π) is satisfied by an interpretation I iff I satisfies all groundings of r (resp. $Gr(\pi)$). $I \subseteq B_{\mathcal{U}}$ is an *answer set* of π iff it is a subset-minimal set satisfying the *Gelfond-Lifschitz reduct* $\pi^I = \{H(r) \leftarrow B^+(r) \mid I \cap B^-(r) = \emptyset, r \in Gr(\pi)\}$. For a program π, we denote the set of its answer sets by $\mathcal{AS}(\pi)$.

Reduction to ASP. We now provide fixed queries for admissible sets in such a way that an argumentation framework AF is given as an input database $\widehat{F}$ and the

[10] For further background, see [Eiter *et al.*, 1997; Brewka *et al.*, 2011].

answer sets of the program $\pi_e(\widehat{F})$ are in a certain one-to-one correspondence with the respective extensions, where $e \in \{\mathcal{AD}, \mathcal{PR}\}$. For an AF $AF = \langle Ar, att\rangle$, we define

$$\widehat{F} = \{\ \mathrm{arg}(a) \mid a \in Ar\} \cup \{\mathrm{att}(a,b) \mid \langle a,b\rangle \in att\ \}.$$

We have to guess candidates for the selected type of extensions and then check whether a guessed candidate satisfies the corresponding conditions, where default negation is an appropriate concept to formulate such a guess within a query. In what follows, we use unary predicates $\mathrm{in}(\cdot)$ and $\mathrm{out}(\cdot)$ to perform a guess for a set $S \subseteq Ar$, where $\mathrm{in}(a)$ means $a \in S$.

Similar to Definition 2.4, we define the subsequent notion of correspondence which is relevant for our purposes.

Definition 2.6. *Let $\mathcal{T} \subseteq 2^{\mathcal{U}}$ be a collection of sets of domain elements and let $\mathcal{I} \subseteq 2^{B_{\mathcal{U}}}$ be a collection of sets of ground atoms. We say that $\mathcal{T}$ and $\mathcal{I}$ correspond to each other, in symbols $\mathcal{T} \cong \mathcal{I}$, iff*

1. *for each $S \in \mathcal{T}$, there exists an $I \in \mathcal{I}$, such that $\{a \mid \mathrm{in}(a) \in I\} = S$;*
2. *for each $I \in \mathcal{I}$, there exists an $S \in \mathcal{T}$, such that $\{a \mid \mathrm{in}(a) \in I\} = S$; and*
3. *$|\mathcal{T}| = |\mathcal{I}|$.*

Let $AF = \langle Ar, att\rangle$ be an argumentation framework. The following program fragment guesses, when augmented by $\widehat{F}$, any subset $S \subseteq A$ and then checks whether the guess is conflict-free in AF:

$$\begin{aligned}\pi_{cf} = \{\ & \mathrm{in}(X) \leftarrow \mathit{not}\,\mathrm{out}(X), \mathrm{arg}(X);\\ & \mathrm{out}(X) \leftarrow \mathit{not}\,\mathrm{in}(X), \mathrm{arg}(X);\\ & \leftarrow \mathrm{in}(X), \mathrm{in}(Y), \mathrm{att}(X,Y)\ \}.\end{aligned}$$

The program module $\pi_{\mathcal{AD}}$ for the admissibility test is as follows:

$$\begin{aligned}\pi_{\mathcal{AD}} = \ \pi_{cf} \cup \{\ & \mathrm{defeated}(X) \leftarrow \mathrm{in}(Y), \mathrm{att}(Y,X);\\ & \leftarrow \mathrm{in}(X), \mathrm{att}(Y,X), \mathit{not}\,\mathrm{defeated}(Y)\ \}.\end{aligned}$$

For each conflict-free set one computes the arguments defeated by the set via the predicate defeated/1. The constraint then rules out those sets where an argument in the guessed set is attacked by an argument which is not defeated by the set, thus there is an argument in the conflict-free set which is not defended.

For any AF $AF = \langle Ar, att\rangle$, the admissible sets of AF correspond to the answer sets of $\pi_{\mathcal{AD}}$ augmented by $\widehat{F}$, i. e. $\mathcal{E}_{\mathcal{AD}}(AF) \cong \mathcal{AS}(\pi_{\mathcal{AD}}(\widehat{F}))$.

For semantics beyond NP we need to make use of *disjunction* in the logic program. There are several different ways how to encode these semantics. The first approach was to use the so called *saturation encodings* as pointed out in [Egly *et al.*, 2010a] which are part of **ASPARTIX**. Other encodings also incorporated in **ASPARTIX** are the *metasp encodings* [Dvořák *et al.*, 2013a], and the recently proposed encodings based on *conditional disjunction* which make use of a particular property of preferred semantics as shown in [Gaggl *et al.*, 2015].

2.2 Direct Implementations

A direct implementation refers to a dedicated algorithm for a reasoning problem of a specific semantics. The advantage is that direct implementations directly incorporate some problem-specific shortcuts, which is often not possible—or it leads to limited improvement—in the case of reduction-based implementations.

2.2.1 Labelling-based Algorithms

Many direct implementations are based on an alternative characterization for semantics using certain labelling functions for arguments [Verheij, 1996b; Doutre and Mengin, 2001; Modgil and Caminada, 2009; Nofal *et al.*, 2014b; Nofal *et al.*, 2014a; Verheij, 2007]. A labelling usually assigns each argument one of the following labels $\Lambda = \{\texttt{in}, \texttt{out},$
$\texttt{undec}\}$, which stand for accepted, rejected and undecided arguments. A labelling is a total function $\mathcal{L}ab : Ar \rightarrow \Lambda$. In the following we write $\texttt{x}(\mathcal{L}ab)$ for $\{a \in Ar \mid \mathcal{L}ab(a) = \texttt{x}\}$. For instance, $\texttt{in}(\mathcal{L}ab)$ is the set of all $\texttt{in}$-labeled arguments. Sometimes we will also represent a labelling $\mathcal{L}ab$ as the triple $\langle \texttt{in}(\mathcal{L}ab), \texttt{out}(\mathcal{L}ab), \texttt{undec}(\mathcal{L}ab)\rangle$.

One advantage of labellings is that the label of one argument has an immediate consequence to its neighbours. For example, if an argument a is labeled with $\texttt{in}$, all arguments attacked by a will be labeled with $\texttt{out}$. Such labelling-based algorithms have been materialized in several systems, see Table 1.

Enumeration. Several labelling-based algorithms to enumerate all extensions for various semantics have been proposed. For instance, the algorithm in [Nofal *et al.*, 2014a] makes use of five labels, namely $\Lambda = \{\texttt{in}, \texttt{out}, \texttt{must\_out}, \texttt{blank}, \texttt{undec}\}$, where the additional label $\texttt{blank}$ denotes the not yet labeled arguments and $\texttt{must\_out}$ is assigned to arguments that attack $\texttt{in}$-labeled arguments. Initially all arguments are labeled with $\texttt{blank}$. Then, the algorithm selects an $a \in \texttt{blank}(\mathcal{L}ab)$ which is labeled with $\texttt{in}$ in the left branch and $\texttt{undec}$ in the right branch of the

search tree. Every time an argument a is labeled with `in` all arguments attacked by it are labeled `out` and all remaining arguments which attack a are labeled with `must_out`. These steps are repeated until there are no arguments left to be labeled. The algorithm stores a preferred extension in one branch if each argument has one of the labels `in`, `out` and `undec` and the `in`-labeled arguments are not a subset of a previously stored preferred extension. Then, the algorithm backtracks to try to find all preferred extensions.

For the selection of the next argument to be labeled out from $\texttt{blank}(\mathcal{L}ab)$ the following heuristics are used.

- Don't pick an argument a to label it `in` iff there is a $b \in \{a\}^-$ such that $\mathcal{L}ab(b) \neq \texttt{out}$ and there is no $c \in \{b\}^-$ with $\mathcal{L}ab(c) = \texttt{blank}$.
- Don't pick an argument a to label it `undec` iff each $b \in \{a\}^-$ is either labeled with `out` or `must_out`.
- First select those `blank`-labeled argument to be labeled `in` which are not attacked at all or all its attacker are labeled with `out` or `must_out`.
- Otherwise, select a `blank`-labeled argument to be labeled `in` which attacks the most not `out`-labeled arguments.

Here we have only considered the case of preferred semantics, but for most of the semantics labelling-based algorithms have been proposed in the literature: algorithms for grounded and stable semantics are given in [Modgil and Caminada, 2009]; algorithms for semi-stable and stage semantics can be found in [Caminada, 2007; Caminada, 2010; Modgil and Caminada, 2009]. Recently [Nofal, 2013] studied improved algorithms for enumerating grounded, complete, stable, semi-stable, stage and ideal semantics. Labelling-based Algorithms are implemented in the **ArguLab** [Podlaszewski *et al.*, 2011] system as well as in the **ArgTools** [Nofal *et al.*, 2012].

Decision Procedures. In the following we will exemplify the use of labellings in an algorithm dedicated to credulous reasoning with preferred semantics, following the work of [Verheij, 2007], which is implemented in the **CompArg** system. In credulous reasoning one is only interested if a particular argument is accepted in at least one extension, thus we try to produce a witness (or counter-example) for this argument, instead of computing all extensions.

The algorithm starts with labelling the queried argument with `in` and all the other arguments with `undec`. Then, it iterates the following two steps. Firstly, it checks whether the set of `in`-labeled arguments is conflict-free and if so label all arguments attacking them with `out`. Otherwise terminate the branch of the

algorithm. Secondly, for each argument a which is labeled `out` but not attacked by an argument labelled `in`, it picks an `undec` labeled attacker b of a and label it with `in`. In case there are several such arguments, it starts a new branch of the algorithm for each choice. If no such argument exists it terminates the branch. It stops a branch as soon as no more changes to labellings are made. In that case, it has reached an admissible labelling acting as proof for the credulous acceptance of the queried argument.

Consider the AF of Example 2.5 and the argument c. In the first step we obtain the following intermediate labelling

$$\mathcal{L}ab_1 = \langle\{c\}, \{\}, \{a, b, d, e\}\rangle.$$

As $\texttt{in}(\mathcal{L}ab_1)$ is conflict-free, we label all arguments attacking c with `out`:

$$\mathcal{L}ab_2 = \langle\{c\}, \{b, d\}, \{a, e\}\rangle.$$

Next we need to make arguments b and d *legally out* by labelling at least one of their attacker with `in`. In case of b this is already fulfilled as c is labeled with `in`. However, the argument d has no attacker, so the algorithm stops. We could not construct an admissible labelling for accepting the argument c, thus it is not credulously accepted under preferred semantics.

2.2.2 Dynamic Programming-based Approaches

We briefly mention the dynamic programming-based approach, which is defined on tree decompositions of argumentation frameworks. Many argumentation problems have been shown to be solvable in linear time for AFs of bounded tree-width [Dunne, 2007; Dvořák *et al.*, 2012c; Courcelle, 1989].

First introduced in [Dvořák *et al.*, 2012b], this approach especially aims at the development of efficient algorithms that turn complexity-theoretic results into practice. The algorithms from [Dvořák *et al.*, 2012b] are capable of solving credulous and skeptical reasoning problems under admissible and preferred semantics. Later, this approach was extended to work with stable and complete semantics [Charwat, 2012]. Further fixed-parameter tractability results were obtained for AFs with bounded clique-width [Dvořák *et al.*, 2010] and in the work on backdoor sets for argumentation [Dvořák *et al.*, 2012a]. Negative results for other graph parameters like bounded cycle-rank, directed path-width, and Kelly-width can be found in [Dvořák *et al.*, 2012b].

Systems implemented towards this approach are **dynPARTIX** [Charwat, 2012; Dvořák *et al.*, 2013b] as well as **D-FLAT** [Bliem, 2012; Bliem *et al.*, 2012]. **D-FLAT**

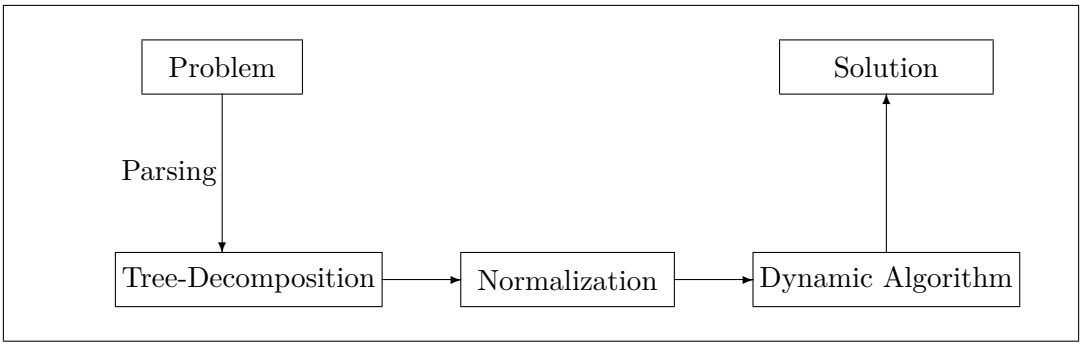


Figure 3: Dynamic-programming approach based on tree-decompositions.

is a general-purpose system that is capable of solving problems from multiple domains. The methodology underlying both of these systems is to build a tree-decomposition of a framework and then run a dynamic programming algorithm on the tree-decomposition to obtain the extensions of the desired semantics, as depicted in Figure 3. For an extensive discussion of the approach we refer the reader to [Charwat *et al.*, 2015].

2.3 Summary

In this section we discussed the two main approaches to implement abstract argumentation frameworks, namely the reduction-based and the direct implementation approach. Systems which implement the reduction-based approach are very popular, as they benefit from highly sophisticated solvers. One can say that they delegate the difficult part of the design of an efficient algorithm to the solvers of the target formalism. This might be the reason why so many solvers make use of this approach (see Table 1). On the other side the direct implementations can incorporate shortcuts if specific properties for certain structures in AFs are known, and in particular when it comes to the reasoning problems of skeptical and credulous acceptance, these algorithms can benefit from them. Many direct implementation algorithms make use of labellings. Table 1 summarizes all systems.

3 Structured Argumentation Implementations

This section gives an overview of algorithmic approaches to structured argumentation [Besnard *et al.*, 2014] and their respective systems. In contrast to abstract argumentation where arguments are interpreted as abstract entities and only logical relationships between arguments are taken into account, structured argumentation

	Direct	Reduction-Based	Type	Reference
{j}ArgSemSAT		Yes	SAT	[Cerutti *et al.*, 2014c; Cerutti *et al.*, 2016b; Cerutti *et al.*, 2017]
ArgTools	Yes		Labellings	[Nofal *et al.*, 2014b]
ArguLab	Yes		Labellings	[Podlaszewski *et al.*, 2011]
ASGL		Yes	CSP	[Sprotte, 2015]
ASPARTIX-D		Yes	ASP, SAT	[Egly *et al.*, 2010a; Gaggl and Manthey, 2015]
ASPARTIX-V		Yes	ASP	[Gaggl *et al.*, 2015]
ASSA	Yes		Matrices	[Hadjisoteriou, 2015]
Carneades	Yes		Labellings	[Gordon *et al.*, 2007]
Cegartix		Yes	SAT	[Dvořák *et al.*, 2014]
CompArg	Yes		Labellings	[Verheij, 2007]
ConArg		Yes	CSP	[Bistarelli *et al.*, 2015]
CoQuiAAS		Yes	SAT	[Lagniez *et al.*, 2015]
DIAMOND		Yes	ASP	[Ellmauthaler and Strass, 2014]
Dungell	Yes		Haskell	[van Gijzel and Nilsson, 2013]
EqArgSolver		Yes	Equations, Labellings	[Rodrigues, 2016]
GERD		Yes	ASP	[Dvořák *et al.*, 2015]
GRIS		Yes	Equations, Labellings	[Gabbay and Rodrigues, 2015]
LabSATSolver		Yes	SAT, Labellings	[Beierle *et al.*, 2015]
LamatzSolver	Yes			[Lamatz, 2015]
prefMaxSAT		Yes	SAT	[Vallati *et al.*, 2015; Faber *et al.*, 2016]
ProGraph	Yes			[Groza and Groza, 2015]
QADF		Yes	QBF, Labellings	[Diller *et al.*, 2015]
ZJU-ARG	Yes		Labellings	[Liao *et al.*, 2013]

Table 1: Summary of abstract argumentation implementations.

considers an argument's internal structure for several aspects including evaluation. Within formal argumentation, formalisms for structured argumentation assume a formalized knowledge base, often in a logical or rule-based form, from which arguments and their relations are constructed. Conceptually, formalisms for structured argumentation often follow the steps of the so-called argumentation process or argumentation pipeline (see e. g. [Dung, 1995, Sections 4 and 5] and [Caminada and Amgoud, 2007, Section 2]):

1. argument construction;
2. determining conflicts among arguments;
3. evaluation of acceptability of arguments; and
4. drawing conclusions.

Argument construction typically refers to the task of building arguments composed of a claim and a derivation of that claim (e. g. a proof tree) from the given knowledge base. Moreover, conflicts need to be recorded, e. g., when claims of two arguments are contradictory, or when the derivation of an argument's claim contradicts with the claim of another argument. Evaluation of acceptability refers to formal means of finding acceptable arguments, and finally conclusions can be drawn from the acceptable arguments.

From a computational point of view, all of the steps of the process taken individually can be quite computationally expensive: for instance even construction of single arguments may be computationally complex (NP-hard in cases); a large number of arguments may be constructed; finding conflicts can be non-trivial; and evaluation of acceptability has in general a high complexity, as in the case of abstract argumentation.

Several algorithmic approaches have been proposed, which result in a quite heterogeneous and evolving field comprising of many different solutions. In the following we highlight properties that distinguish algorithms for structured argumentation from each other.

Reasoning on structural or abstract representation. The first aspect that distinguishes algorithms and systems for structured argumentation is that they may deviate from the conceptual argumentation process. In particular, the approaches can be roughly categorized whether they perform

- (query-based) structural reasoning; or
- reasoning on an abstract representation.

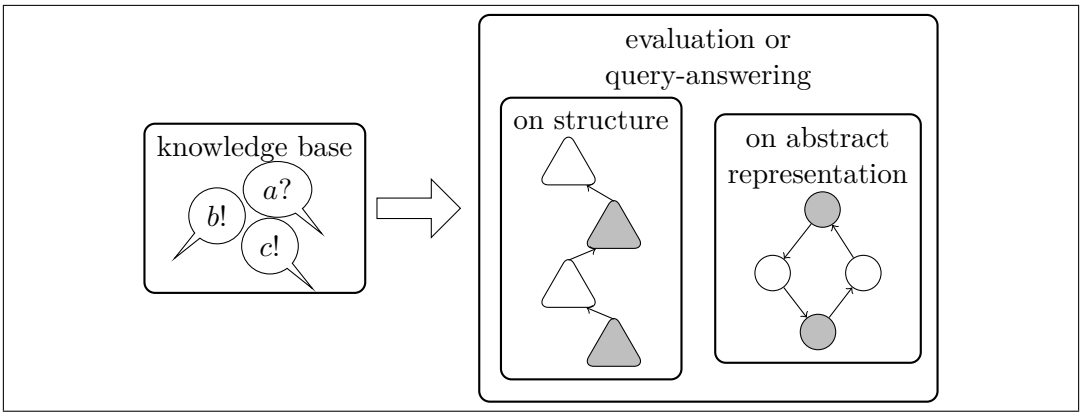


Figure 4: Argumentation process from a computational point of view

The latter classification encompasses algorithms that explicitly construct an abstract representation, e. g. an AF, and perform reasoning solely on that representation. Algorithms following the other approach construct no such representation, but combine argument construction, conflict discovery, and argument evaluation in possibly interleaving steps and take structured information from the input knowledge base into consideration in possibly every step.

Algorithms that perform structural reasoning are typically query-based, i. e., decide acceptability of a certain claim, and construct arguments for and counterarguments against the queried claim from the knowledge base. A structural approach can restrict argument construction more easily than the abstract approach, in particular for query-based reasoning, since structural information can be used to determine which arguments have an effect on the query or the currently processed argument.

On the other hand, the abstract approach first "compiles" the structured knowledge base and subsequently all reasoning can be performed on the abstraction. In some cases "full" knowledge of all arguments occurring in the abstract representation is required to perform reasoning, e. g. for stable semantics. Conceptually, the abstract approach follows more closely the argumentation process. We illustrate structural and abstract approaches to algorithms for structured argumentation in Figure 4. In this figure triangles are arguments with internal structure and round vertices are abstract arguments.

Dedicated and reduction-based approaches. Similarly as for approaches to implement abstract argumentation, we can distinguish between direct or dedicated approaches and reduction-based approaches to implement structured argumentation. An approach is reduction-based if the input is translated to a problem of another

target formalism with available solvers for that problem. Direct algorithms solve the problem at hand with a domain-specific dedicated algorithm. Direct algorithms have the benefit of incorporating domain-specific properties and optimizations more easily. On the other hand, reduction approaches can re-use off-the-shelf solvers. Reduction-based approaches for structured argumentation typically incorporate all involved tasks, i.e., argument construction, conflict evaluation, and deciding acceptability of arguments. When constructing an abstract representation, approaches to structured argumentation can also be hybrid systems, i.e., providing a direct or reduction-based approach for constructing the abstraction, and providing another for abstract reasoning. Usual target systems for reduction-based approaches are Prolog systems, solvers for Boolean satisfiability (SAT) and related formalisms, and solvers for answer-set programming (ASP) [Brewka *et al.*, 2011]. We also call an algorithm or system reduction-based if it incorporates a translation of subproblems to a target language with available solvers.

Considered Approaches. In the following we overview concrete algorithmic approaches to structured argumentation, introducing them with examples and discussing the main computational problems, properties of interest from a computational point of view, and algorithms and systems proposed to solve the problem.[11] We focus on implemented algorithms for abstract rule-based argumentation (in particular concrete instantiations of the general ASPIC+ formalism) [Prakken, 2010; Modgil and Prakken, 2014], assumption-based argumentation (ABA) [Bondarenko *et al.*, 1997; Toni, 2014], argumentation based on logic programming, in particular based on defeasible logic programs (DeLPs) [García and Simari, 2004; García and Simari, 2014], argumentation based on classical logic [Besnard and Hunter, 2008], and Carneades [Gordon *et al.*, 2007]. Complementing information can be found in a review of implementations for defeasible reasoning [Bryant and Krause, 2008], in particular sections 4.2.7, 4.3.1, 4.3.2, 4.3.3, and 4.3.4; in the review for argumentation for the social web [Schneider *et al.*, 2013]; and in the overview on research in argumentation systems given by [Simari, 2011].

3.1 Abstract Rule-Based Argumentation

In this section we focus on systems for abstract rule-based argumentation, in particular concrete instantiations of the ASPIC+ [Prakken, 2010; Modgil and Prakken, 2014] formalism. We begin with a brief introduction to a concrete instantiation

[11] Tools presented and referenced within the following subsections sometimes do not solve the same reasoning tasks proposed for a formalism. We refer the reader to the references for each algorithm and tool for the exact problem definitions that are solved.

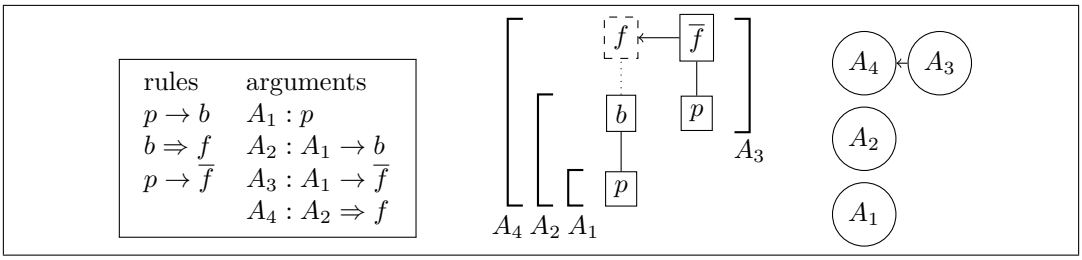


Figure 5: Tweety example knowledge base in ASPIC+ (left) with axiom p, structure of corresponding arguments (middle), and AF (right).

of ASPIC+ following notation of [Modgil and Prakken, 2014]. Input in this formalism is a knowledge base consisting of several components, central among them are (ordinary) premises and axioms, defeasible and strict rules, and preferential information. Semantics are specified via a translation to an abstract argumentation framework. Arguments are constructed by chaining premises or axioms with rules. Conflicts among arguments are defined via so-called undercuts, rebuts, and undermining among arguments, all respecting the preferential information.

We illustrate the concepts in a toy example knowledge base in Figure 5.

Example 3.1. *Figure 5 shows two strict rules (with a simple arrow $\to$) and one defeasible rule (using a double-lined arrow $\Rightarrow$), and assuming p (Tweety is a penguin) to be an axiom, one can infer the four arguments shown in the figure, namely by a strict rule that Tweety is a bird (b), that birds normally fly (via a defeasible rule inferring f), and that penguins do not fly (via a strict rule inferring $\overline{f}$; note that overlining indicates contrariness). The structure of the arguments is visualized in the middle of Figure 5 where we also see the only conflict in this example, namely that argument A_3 attacks A_4 via rebut (contradictory conclusions). On the right of Figure 5 the abstract AF is shown.*

Computational problems for abstract rule-based argumentation include argument construction, conflict discovery, and semantic evaluation. These problems may be tackled in an intertwined way, for instance interleaving construction and evaluation or following more closely the argumentation process step-by-step and thus firstly constructing the abstract argumentation framework and then proceeding by semantical evaluation.

As a rough and general outline for algorithms based on structural reasoning, given a potential conclusion (e. g. Tweety can fly in example Figure 5), arguments can be constructed via backward chaining using rules until premises or axioms are found. For instance, argument A_4 can be constructed from conclusion f and back-

chaining of two rules until axiom p is reached. Counterarguments can be found in a similar manner by back-chaining from conclusions of arguments that would attack the arguments constructed so far. The so constructed arguments, i. e., arguments in favor of the queried claim and the counterarguments, corresponds to a game-theoretic approach to compute acceptability of the given query (and one of its argument in favor) under the specified semantics. For instance, one can conclude that A_3 is contained in an admissible set $\{A_3\}$.

We begin our survey of systems for abstract rule-based argumentation with the **TOAST** system[12] [Snaith and Reed, 2012]. **TOAST** directly follows the steps of the argumentation pipeline by constructing an abstract AF from given input knowledge base and delegates the reasoning tasks to a dedicated AF reasoner, namely the Dung-O-Matic web service [Snaith *et al.*, 2010]. As an example, given the input in Figure 5 (left) the system would return a semantical evaluation of the AF shown on the right of that figure. The **TOAST** and Dung-O-Matic system together provide a system supporting axioms, premises, assumptions, and preferential information (last link and weakest link principles, see also [Modgil and Prakken, 2014]), rules, and a user-specified contrariness relation. The system further supports reasoning on the resulting AF under grounded, preferred, semi-stable, and stable semantics. **TOAST** is available as both a Java-based web service and web form.

Next we overview contributions to systems for abstract rule-based argumentation by Vreeswijk, which influenced subsequent successor systems. These systems follow query-based structural reasoning. Vreeswijk's works for argumentation systems are well summarized in the survey of [Bryant and Krause, 2008, Sections 4.3.1, 4.3.2, 4.3.3, and 4.3.4]. A system that resulted from Vreeswijk's PhD thesis [Vreeswijk, 1993], **IACAS** (InterActive Argumentation System), was written in LISP and is one of the earliest implementations of structured argumentation that is capable of handling input with strict and defeasible rules. This system allows for argument generation for or against a queried claim, and concluding its acceptability taking all the arguments into consideration. Vreeswijk's argumentation system (**AS**) is a Ruby-based implementation that handles strict and defeasible rules and tries to construct an admissible set containing an argument that concludes the queried claim. Two systems based on Vreeswijk's **AS** have been developed, namely the **ASPIC Inference Engine** and **Argue tuProlog** [Bryant *et al.*, 2006].

The **ASPIC Inference Engine** is available from the ASPIC resources at the Cancer Research UK's Advanced Computation Laboratory.[13] It provides both a web-based front-end and a Java-based system that implement query-based structural

[12] `http://www.arg.dundee.ac.uk/toast/` (on 27/04/2017).
[13] `http://aspic.cossac.org` (on 27/04/2017).

reasoning under grounded and (credulous) admissible semantics. The Java-based implementation offers a graphical user-interface.

A reduction approach to the language of Prolog is used in **Argue tuProlog** and the system is presented in [Bryant *et al.*, 2006]. The reduction utilizes a game-theoretic approach for implementing ASPIC, similarly as the previous approaches. In contrast to reduction approaches for other formalisms, **Argue tuProlog** reduces the input to several Prolog queries, i. e., every query for an argument for each player is instantiated as a separate Prolog call and thus the dialogue can be terminated at any time.

We conclude this section with Wietske Visser's Epistemic and Practical Reasoner (**EPR**)[14] [Visser, 2008] which is a direct Java-based implementation that implements query-based reasoning under grounded semantics, (credulous) admissible semantics, and e-p semantics [Prakken, 2006]. The system provides a graphical user-interface, and is documented in detail in Wietske Visser's master's thesis [Visser, 2008].

3.2 Assumption based argumentation

In assumption-based argumentation (ABA) [Bondarenko *et al.*, 1997; Toni, 2014], arguments and conflicts are drawn from three main components: a knowledge base, a set of assumptions, and a contrariness relation. We illustrate these concepts in Figure 6. On the left of Figure 6 we see an ABA framework, with four rules, the set of assumptions A containing a and e, and the contrariness relation relating the two assumptions to be contrary to f and d respectively (denoted via $\overline{a} = f$ and $\overline{e} = d$). Arguments (in squares) and conflicts (with solid arrows) that can be drawn from this framework are shown on the right of the figure. These arguments correspond to proof trees of claims. More concretely, the arguments' structure is based on the rules with the conclusion shown on the top of the squares and attacks take place based on assumptions and their contraries. For instance, the argument with f as the conclusion attacks the argument with conclusion b, since this argument requires the assumption a which is the contrary of f ($\overline{a} = f$). Arguments without assumptions are not attacked, e. g. argument with conclusion c.

Semantics of ABA can be defined via extensions as sets of arguments or, equivalently, as sets of assumptions. For instance, in the example in Figure 6 the set of arguments with claims for c, f, and e (that in this instance uniquely determine the corresponding arguments) is an admissible extension of the ABA framework (no attacks between these arguments are present and all attackers from outside are counterattacked). The corresponding set of assumptions is $\{e\}$.

[14] `http://www.wietskevisser.nl/research/epr/` (on 27/04/2017).

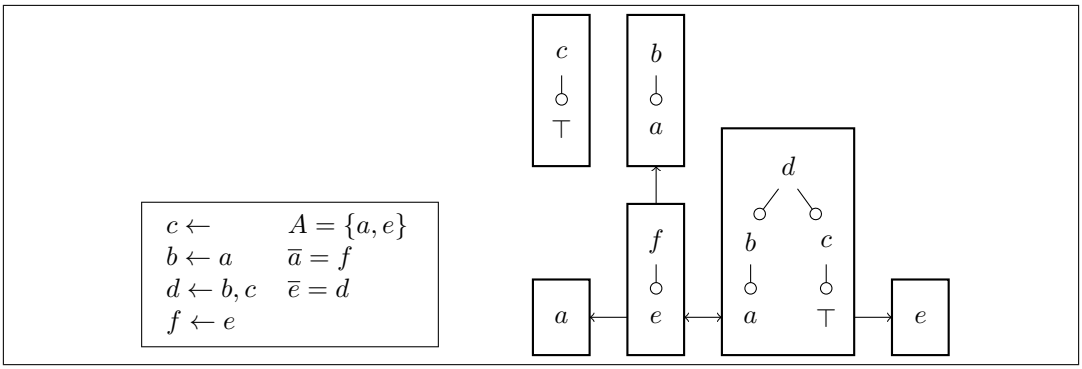


Figure 6: ABA framework (left) and its corresponding arguments and attacks (right)

A typical reasoning task for ABA frameworks is to check whether an argument for a given claim is contained in an extension under a specified semantics. The computational complexity for reasoning with an abstract ABA formalism has been investigated in [Bondarenko *et al.*, 1997]. In [Bondarenko *et al.*, 1997] decision problems for credulous and skeptical acceptance are studied and the complexity ranges from polynomial-time decidable to completeness for Σ_4^P, a class on the fourth level of the polynomial hierarchy.

Common to several algorithms for computing acceptability of a given claim under a specified semantics in a given ABA framework are so-called *dispute derivations* [Craven and Toni, 2016; Dung *et al.*, 2006; Dung *et al.*, 2007; Gaertner and Toni, 2007b; Gaertner and Toni, 2008; Toni, 2013]. Intuitively, dispute derivations can be seen as a game-theoretic constructive proof of acceptability of the given claim by constructing (part of) the argument in favor of the claim as well as constructing (parts of) its counterarguments and their counterarguments. Dispute derivations were proposed for grounded, admissible, and ideal semantics, called respectively GB, AB, and IB[15] dispute derivations [Dung *et al.*, 2007], which are an advancement of the proof trees proposed in [Dung *et al.*, 2006]. In [Gaertner and Toni, 2007b; Gaertner and Toni, 2008] *structured* dispute derivations were proposed that explicitly compute the dialectical structure hidden in dispute derivations, e. g., computing the attack structure explicitly. A parametrized version of dispute derivations was proposed in [Toni, 2013] that have a richer output incorporating both equivalent views of semantics of ABA, namely the view of extensions as sets of arguments and sets of assumptions.

In this paper we illustrate concepts of dispute derivations by showing GB-dispute

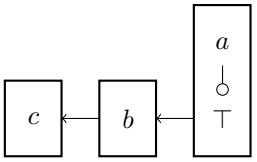
[15]Here, the "B" stands for belief.

derivations [Dung *et al.*, 2007]. In Figure 7 we see on the left a representation of a simple ABA framework with assumptions $A = \{b, c\}$ and a rule that infers a without assumptions. The grounded extension of this ABA framework contains the arguments for c and a, which are uniquely determined in this particular framework. A GB-dispute derivation is a sequence of quadruples (P_i, O_i, A_i, C_i) with integer i denoting the sequence or step. The ingredients for a step are the sentences or nodes for proponent (P_i) and opponent (O_i), the assumptions for defense of the queried claim (A_i) and assumptions for the opponent, so-called culprits (C_i). The component P_i is a set of sentences and both A_i and C_i are sets of assumptions. The second component of the quadruple, O_i, is a set of sets containing sentences. For querying acceptability for a claim α we initialize with $P_0 = \{\alpha\}$, $A_0 = \alpha \cap A$, and empty O_0 and C_0, where A is the set of assumptions in the ABA framework. We next illustrate the basics of GB-dispute derivations by recalling the corresponding sequences from [Dung *et al.*, 2007], where we assume a selection function f that selects at each step either an element in P_i or in O_i and in the latter case an element of the set selected. For a given ABA framework and a selection function f, a GB-dispute derivation of a defense set D for sentence α is a finite sequence of quadruples

$$(P_0, O_0, A_0, C_0), \ldots, (P_i, O_i, A_i, C_i), \ldots, (P_n, O_n, A_n, C_n)$$

with $P_0 = \{\alpha\}$, $A_0 = \alpha \cap A$, and empty O_0 and C_0; $P_n = O_n = \emptyset$ and $A_n = D$; and for every $0 \leq i < n$ and $X = f(P_i, O_i, A_i, C_i)$ the selected element s. t.

1. if $X \in P_i$ then
   (a) if $X \in A$ then
   $$\begin{array}{llllll} P_{i+1} & = & P_i \setminus X, & A_{i+1} & = & A_i, \\ C_{i+1} & = & C_i, & O_{i+1} & = & O_i \cup \{\{\overline{X}\}\} \end{array}$$
   (b) else (there exists a rule $X \leftarrow R$ with body R s.t. $C_i \cap R = \emptyset$)
   $$\begin{array}{llllll} P_{i+1} & = & (P_i \setminus X) \cup R, & A_{i+1} & = & A_i \cup (A \cap R), \\ C_{i+1} & = & C_i, & O_{i+1} & = & O_i \end{array}$$
2. else ($T \in O_i$ is selected with $X \in T$)
   (a) if $X \in A$ then
   $$\begin{array}{llllll} P_{i+1} & = & P_i \cup \{\overline{X}\}, & A_{i+1} & = & A_i \cup (\{\overline{X}\} \cap A), \\ C_{i+1} & = & C_i \cup \{X\}, & O_{i+1} & = & O_i \setminus \{T\} \end{array}$$
   (b) else
   $$\begin{array}{llllll} P_{i+1} & = & P_i, & A_{i+1} & = & A_i, \\ C_{i+1} & = & C_i, & O_{i+1} & = & (O_i \setminus \{T\}) \cup \\ & & & & & \{T \setminus \{X\} \cup R \mid X \leftarrow R \in \mathcal{R}\} \end{array}$$



i	P_i	O_i	A_i	C_i	selected
0	$\{c\}$	$\emptyset$	$\{c\}$	$\emptyset$	–
1	$\emptyset$	$\{\{b\}\}$	$\{c\}$	$\emptyset$	c
2	$\{a\}$	$\emptyset$	$\{c\}$	$\{b\}$	$\{b\}$, b
3	$\{\top\}$	$\emptyset$	$\{c\}$	$\{b\}$	a

Figure 7: ABA with $A = \{b, c\}$, $\overline{b} = a$, $\overline{c} = b$, and rule $a \leftarrow$ (left); GB-dispute derivation for c (right)

with $\mathcal{R}$ the set of rules of the given ABA framework. In Figure 7 we see on the right a sequence of a GB-dispute derivation. Briefly put, in each step in the sequence we select either an element of proponent or opponent, which in turn can either be assumptions or non-assumptions. Depending on the choice, different updates to the step have to be applied. For instance, if we choose an assumption of the proponent, then we remove that assumption from the sentence the proponent holds and add the contrary to the opponent who may construct an argument in favor of the contrary. We can note that each step in the sequence individually is straightforward to compute, however computation relies heavily on the selection function (also on selecting a rule in one case), which is discussed in more detail e. g. in [Gaertner and Toni, 2007b; Craven and Toni, 2016], which also highlights design choices for an algorithm based in dispute derivations.

Several systems have been developed implementing algorithms based on variants of dispute derivations. Current state of the art of dispute-derivation-based algorithms and systems for ABA are query-based and reason on the structural level and generally do not construct the full abstract representation to perform reasoning. Interestingly, most implementations, that build upon dispute derivations, rely on a reduction to Prolog with one exceptions **sxdd** [Craven *et al.*, 2012], which is an implementation in C++.

The system **CaSAPI**,[16] which stands for "Credulous and Sceptical Argumentation: Prolog Implementation", is, as the name suggests, an implementation for ABA in Prolog. In version 2.0 [Gaertner and Toni, 2007a], **CaSAPI** implements GB, AB, and IB dispute derivations to perform query-based structural reasoning. Further, in versions 3.0 [Gaertner and Toni, 2007b] and 4.3 [Gaertner and Toni, 2008; Dung *et al.*, 2007] structured dispute derivations are employed. Nowadays, **CaSAPI** acts as a precursor system for more recent systems.

Several tools with refined dispute derivations and reduction to Prolog have been

[16] `http://www.doc.ic.ac.uk/~ft/CaSAPI/` (on 27/04/2017).

proposed and implemented to perform query-based structural reasoning for ABA.[17] In the tool **proxdd** [Toni, 2013] the parametrized versions of dispute derivations are used. Graph-based versions of dispute derivations have been applied in the systems **grapharg** [Craven *et al.*, 2013] and its follow-up system **abagraph** [Craven and Toni, 2016]. These tools include graphical visualization.

Recently, two systems for ABA were developed which are not based on dispute derivations: **ABAplus**[18] and the system from [Lehtonen *et al.*, 2017], which we call here **ABAToAF**. Both of these systems compute semantics of ABA frameworks via an AF reasoner, ASPARTIX [Egly *et al.*, 2010a], on an abstract representation of the ABA framework.

The system **ABAplus** implements ABA$^+$ [Cyras and Toni, 2016a], an extension of ABA with preferences. More concretely, this system provides computations for flat ABA$^+$ frameworks satisfying the axiom of weak contraposition [Cyras and Toni, 2016b] (this class subsumes flat ABA frameworks). The system **ABAplus** is capable of enumeration of extensions (as sets of assumptions together with their conclusions) under grounded, complete, preferred, stable, and ideal semantics. In contrast to systems described above, **ABAplus** constructs an abstract AF to reason on the ABA, with arguments being sets of assumptions, with the AF being solved via encodings of ASPARTIX. The system **ABAplus** generates arguments, using Python, based on (i) sets of assumptions that deduce contraries of assumptions and (ii) singleton sets of assumptions. Both the ABA$^+$ framework and the enumerated extensions are visualized in a web frontend.

The other system for ABA that relies on an AF reasoner, **ABAToAF**, constructs arguments and attacks, similarly to **ABAplus**, based on sets of assumptions and derived sentences. Argument construction, implemented in Java 8, approximates here the restriction to generate arguments only for those sets of assumptions where at least one sentence can be derived from such a set, but not any proper subset. The system **ABAToAF** solves credulous (under admissible and stable semantics) and skeptical (under stable semantics) acceptance queries via calling an ASP solver on modified ASPARTIX encodings on the constructed AF.

Empirical evaluations of systems for ABA have been carried out for **sxdd** [Craven *et al.*, 2012], **grapharg** [Craven *et al.*, 2013], **abagraph** [Craven and Toni, 2016], and **ABAToAF** [Lehtonen *et al.*, 2017].

The work of [Craven and Toni, 2016], based on preliminary research of [Craven *et al.*, 2013], improves on several computational aspects of dispute derivations by altering the arguments' tree-structure to general graphs and introducing graphical

[17] Available at `http://www.doc.ic.ac.uk/~rac101/proarg/` (on 27/04/2017).

[18] Web front end available at `http://www-abaplus.doc.ic.ac.uk/` (on 27/04/2017) and standalone version at `https://github.com/zb95/2016-ABAPlus/` (on 27/04/2017).

dispute derivations (graph-DDs). In addition to tackle certain circularity questions for computation, in [Craven and Toni, 2016] an improvement for the problems of so-called flabbiness and bloatedness is provided. Briefly put, flabbiness refers to the potential shortcoming that the same sentence or claim is proved in several different ways, and bloatedness talks about deriving a claim in multiple ways in different arguments in an extension. That is, the former talks about computation of claims for individual arguments and the latter talks about computation of extension-based acceptability questions incorporating redundancy. In [Craven and Toni, 2016] graph-DDs are proposed for admissible and grounded semantics.

3.3 Argumentation based on logic programming

In this section we focus on algorithms and systems for argumentation based on logic programming, in particular defeasible logic programming [García and Simari, 2004; García and Simari, 2014].A defeasible logic program (DeLP) consists of strict ($\leftarrow$) and defeasible ($\leftharpoonup$) rules as illustrated in Figure 8. Arguments in a DeLP are composed of a claim (a literal) and a set of defeasible rules. Acceptance of arguments is decided via a dialectical tree, see Figure 8 (right) for an example which includes an argument (A, a) that argues for literal a with set of rules A, arguments $(B_1, \sim b)$ and $(B_2, \sim b)$ that argue for (strongly) negated b, and argument $(E, \sim e)$ that argues for (strongly) negated e. Argument $(B_2, \sim b)$ defeats (A, a) because the former contradicts a subargument of the latter (arguing for b). Such a dialectical tree is then marked conceptually in a bottom-up manner with undefeated U and defeated D, i. e., leaves are undefeated and arguments are defeated if at least one child node is undefeated. Arguments are undefeated if all its children are defeated. Important for determining conflicts are preference relations which can either be given as input or derived via specificity, see [García and Simari, 2004; Stolzenburg *et al.*, 2003] for details. In our example, the argument (A, a) is not warranted, simply because it is defeated by $(B_1, \sim b)$. If the rules used in argument $(B_1, \sim b)$ would be removed from the input DeLP, then argument (A, a) would be warranted.

Complexity of decision problems in DeLP has been studied in [Cecchi *et al.*, 2006], showing complexity results for problems of deciding whether a given structure is an argument in a given DeLP (polynomial-time decidable), existence of arguments (a problem in NP), and further results regarding data complexity.

Algorithms for DeLP, which are based on dialectical trees, inherently solve query-based structural reasoning and check whether the queried claim is acceptable or warranted in a dialectical tree. Regarding enhancements for algorithms for computing acceptance of DeLPs, as stated in the survey of [Bryant and Krause, 2008],

A	B_1	B_2	E
$a \leftarrowtail b$	$\sim b \leftarrowtail d$	$\sim b \leftarrowtail e$	$\sim e \leftarrow g$
$b \leftarrowtail c$	$d \leftarrow$	$e \leftarrowtail f$	$g \leftarrow$
$c \leftarrow$		$f \leftarrow$	

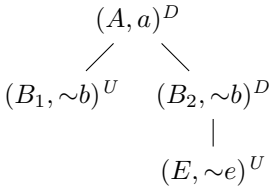


Figure 8: DeLP knowledge base (left) and dialectical tree (right)

three concepts have been proposed to optimize efficiency for deciding acceptance in DeLPs: (i) pruning of dialectical trees [Chesñevar *et al.*, 2000], (ii) using pre-compiled arguments in a dialectical database [Capobianco *et al.*, 2004], and (iii) using parallelism [García and Simari, 2000]. We briefly illustrate these concepts and also refer the reader to the survey [Bryant and Krause, 2008] which includes a section on DeLP (Section 4.2.7).

For pruning of dialectical trees, as can be seen in the example dialectical tree of Figure 8, we do not need to consider all arguments in the tree to determine the dialectical status of the root argument. In particular, since argument $(B_1, \sim b)$ is undefeated, it is immediate that the top argument in this case is defeated. Therefore the right subtree is not relevant for concluding the overall result. Details on general pruning procedures for DeLP can be found in [Chesñevar *et al.*, 2000], in particular how to "choose" the most promising argumentation line (path from root to a leaf in a dialectical tree) that determines an answer to the acceptability question as soon as possible.

In [Capobianco *et al.*, 2004] for speeding up algorithms for ODeLP, a pre-compiled so-called dialectical database is suggested. Briefly put, potential arguments and defeats from the initial knowledge base are pre-compiled. In this way queries can incorporate first look-ups in the pre-compiled dialectical database.

For exploiting parallelism, in [García and Simari, 2000] it is suggested to parallelize computation for (i) finding several arguments for the same conclusion, (ii) discovering several defeaters for an argument, and (iii) finding several argumentation lines.

For concrete systems, DeLP reasoning has been implemented in Prolog accessible via the **DeLP client**,[19] and in the general-purpose libraries of **Tweety**[20] [Thimm, 2014]. In **Tweety** both the algorithm outlined in [García and Simari, 2004] for marking a dialectial tree and a translation to an AF have been implemented (the latter does not preserve the dialectical semantics of DeLP and only interprets the ar-

[19] Web interface available at `http://lidia.cs.uns.edu.ar/delp_client/` (on 27/04/2017).
[20] `http://tweetyproject.org` (on 27/04/2017).

guments and counterargument relationship within an abstract framework). **Tweety** also provides a web-interface for DeLP. Also, an abstract machine called JAM (justification abstract machine) [García, 1997] has been designed for DeLP. Furthermore, a reduction to ASP is given in [Thimm and Kern-Isberner, 2008].

Two further notable reduction-based approaches for extensions of DeLP have been proposed and implemented.[21] Possibilistic DeLP (P-DeLP) extends DeLP rules by attaching levels of strength. In [Alsinet *et al.*, 2010] a recursive semantics for P-DeLP has been proposed, the corresponding framework is called RP-DeLP. An ASP-based approach to compute queries for RP-DeLP, i. e., to decide if a literal is warranted in the framework, is presented and experimentally evaluated in [Alsinet *et al.*, 2012], which is based on results and complexity bounds of [Alsinet *et al.*, 2011]. We call the corresponding system **ASP-RP-DeLP**. A reduction-based approach to SAT for multiple outputs of R-DeLP, we call the system **SAT-R-DeLP**, has been presented in [Alsinet *et al.*, 2013] and also experimentally evaluated in that paper. The SAT approach is based on results of [Alsinet *et al.*, 2011].

3.4 Argumentation based on classical logic

In argumentation based on classical logic, or deductive argumentation, arguments and conflicts are generated from a (classical) logic knowledge base [Besnard and Hunter, 2008]. A knowledge base is here a set of formulas and arguments are pairs (S, C) of support S and claim C. The first component is a consistent, minimal (w.r.t. $\subseteq$) subset of the knowledge base that entails the claim, which in turn is a formula. Arguments can be compared w.r.t. conservativeness, i. e., (S, C) is more conservative than (S', C') iff $S \subseteq S'$ and $C' \models C$. Several notions of conflicts among arguments have been studied [Gorogiannis and Hunter, 2011]. We illustrate here the notion of (canonical) undercuts. Argument (S, C) undercuts (S', C') if $C = \neg(\phi_1 \wedge \cdots \wedge \phi_n)$ with $\{\phi_1, \ldots, \phi_n\} \subseteq S'$. Canonical undercuts incorporate notions of maximal conservativeness and canonical enumeration of formulas, i. e., the sequence of formulas ϕ_i in the conjunction C does not matter. In Figure 9 we see on the left (a) a knowledge base and on the right (c) three arguments where the middle one is a canonical undercut of the top one and the bottom one a canonical undercut of the middle one. Note that in contrast to other structured approaches to argumentation, the arrows in formulas in this section denote logical (material) implication, i. e., within formulas $a \to b$ is logically equivalent to $\neg a \leftarrow \neg b$ and $\neg a \vee b$. A further important notion is that of (complete) argument trees. A given argument is the root of an argument tree, for each node its children are its canonical

[21] Available via web-front-end at `http://arinf.udl.cat/rp-delp` (on 27/04/2017).

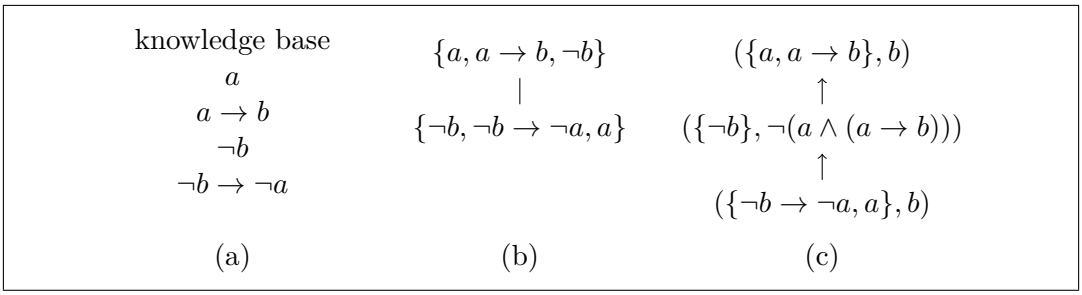


Figure 9: Knowledge base for deductive argumentation (a), inconsistent subsets of that knowledge base (b), and argument tree based on the inconsistent subsets as constructed by compilation-based approach (c)

undercuts, and the support of no node is a subset of the union of supports of all its ancestor nodes.

Computational complexity is in general very high for deductive argumentation [Parsons *et al.*, 2003; Hirsch and Gorogiannis, 2010; Wooldridge *et al.*, 2006; Creignou *et al.*, 2011], as can be intuitively explained from the definitions which incorporate both minimality and entailment properties.[22] Complexity of finding individual arguments has been analyzed in [Parsons *et al.*, 2003], decisions problems concerning instantiation of argument graphs with classical logic in [Wooldridge *et al.*, 2006], and finding argument trees in [Hirsch and Gorogiannis, 2010]. Complexity for problems for deductive argumentation based on propositional logic can reach up to PSPACE.

Proposed algorithms and systems for deductive argumentation are based on minimal unsatisfiable subsets (MUSes) of formulas [Besnard and Hunter, 2006; Besnard *et al.*, 2010], connection graphs [Efstathiou and Hunter, 2011; Efstathiou and Hunter, 2008], reductions to QBF [Besnard *et al.*, 2009] and ASP [Charwat *et al.*, 2012], so-called "contours" [Hunter, 2006b] and approximate arguments [Hunter, 2006a]. Algorithms that utilize contours, approximate arguments, and one MUS-based approach [Besnard and Hunter, 2006] are also discussed in detail in the book [Besnard and Hunter, 2008].

We begin with our algorithmic overview with two MUS-based approaches. The first one [Besnard and Hunter, 2006] falls into the general scheme of knowledge compilation [Darwiche and Marquis, 2002] where a given input is compiled into a

[22] Another explanation for complexity of deductive argumentation is to consider its connection to (propositional) abduction, see [Besnard and Hunter, 2014, Section 7.4]. Complexity of propositional abduction is analyzed in [Eiter and Gottlob, 1995], with problems complete for Σ_2^P a class that is presumably more complex than the class NP.

structure to which one can pose queries that are computationally easier to compute on that structure compared to the original input. For deductive argumentation, the input knowledge base is compiled into a graph consisting of minimal inconsistent subsets of the knowledge base as the vertices and edges between non-disjoint subsets.

In Figure 9 we see in the middle (b) the compiled graph from knowledge base in the left (a). Given an argument, say $(\{a, a \to b\}, b)$ (top right of Figure 9) one can construct an argument tree for this argument using the inconsistent subsets. Note that the support $\{a, a \to b\}$ of this argument is contained in a MUS. The remainder of that MUS ($\neg b$) then is the support for a canonical undercut of the argument, since both parts of the MUS, $\{a, a \to b\}$ and $\{\neg b\}$, each entail a negated conjoined subset of the other, e. g. $\{\neg b\}$ entails $\neg(a \wedge (a \to b))$. Using this line of reasoning recursively, one can construct all counterarguments and in turn the argument tree (shown on the right of Figure 9). For details on the algorithm see [Besnard and Hunter, 2006]. The compilation-based approach has been implemented in the **Tweety** libraries [Thimm, 2014] which can be configured to use different MUS solvers, for instance MARCO [Liffiton *et al.*, 2016] or MIMUS [McAreavey *et al.*, 2014].

Another approach using MUSes [Besnard *et al.*, 2010] directly constructs arguments and counterarguments with a MUS solver, without an "offline" compilation beforehand. The idea underlying argument construction of [Besnard *et al.*, 2010] is that (S, C) is an argument iff $S \cup \{\neg C\}$ is a MUS of the knowledge base together with $\neg C$. Conditions of minimality and entailment for argument (S, C) follow from the fact that if $S \cup \{\neg C\}$ is a MUS, then S is consistent and entails C and S' with $S' \subset S$ does not entail C. The algorithms for argument construction and argument tree generation proposed in [Besnard *et al.*, 2010], **BA** and **BT**, follow this line of reasoning and directly incorporate algorithmic issues like construction of formulas in conjunctive normal form. Algorithm **BA** has been implemented with the MUS solver HYCAM [Grégoire *et al.*, 2009] and experimentally evaluated in [Besnard *et al.*, 2010].

A different approach for generating argument trees for a given claim is proposed in [Efstathiou and Hunter, 2011], building on earlier work in [Efstathiou and Hunter, 2008] which utilizes connection graphs. Connection graphs consist of clauses as vertices and edges between clauses with complementary literals. Briefly put, for a given claim one can reduce the connection graph in such a way that, if non-empty, a support for the claim is contained in the reduced connection graph. In [Efstathiou and Hunter, 2011] this idea is used to construct argument trees. The approach has been implemented in Java in the tool **JArgue** and experimentally evaluated.

Reduction-based approaches are given in [Besnard *et al.*, 2009; Charwat *et al.*, 2012]. The former is a reduction to QBF and the latter to ASP. The latter has

been implemented in the system called **vispartix**[23] within the tool ARVis [Ambroz *et al.*, 2013] for visualizing relations between answer-sets of an ASP encoding. In **vispartix** an AF is generated from a given knowledge base and pre-specified set of claims, and conflicts are constructed as specified in [Gorogiannis and Hunter, 2011], thus partially deviating from other works in this section. The construction process is done via two ASP calls, the first constructing the arguments and the second constructing the attacks. In a final step the AF is visualized. Semantics can be computed via tools developed for AFs.

Algorithms following the concept of contours [Hunter, 2006b] are based on the idea of providing boundaries of what is provable in a knowledge base. Briefly put, an upper (lower) contour stores for a given formula which subsets of the knowledge base entail (do not entail) the formula. Finally, algorithms for approximate arguments [Hunter, 2006a] are based on the idea of relaxing one of the conditions for arguments (consistency, entailment, or minimality).

3.5 Carneades

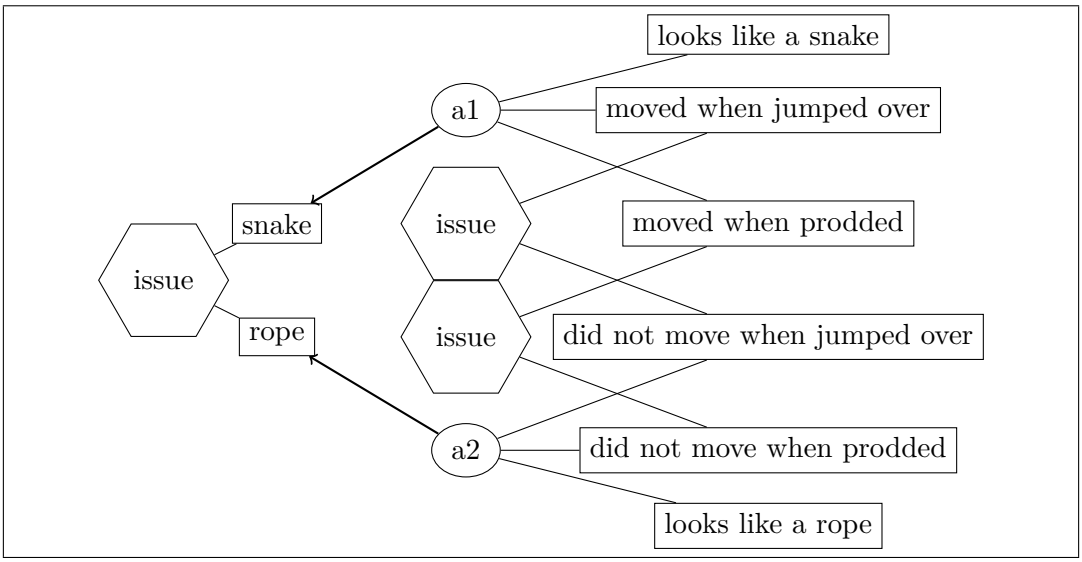


Figure 10: Example Carneades argument graph

Carneades [Gordon and Walton, 2016; Gordon *et al.*, 2007] is both a formal model of argument structure and evaluation, and a system[24] implementing the

[23] `http://www.dbai.tuwien.ac.at/proj/argumentation/vispartix/` (on 27/04/2017).
[24] `https://carneades.github.io/` (on 27/04/2017).

model. Evaluation of acceptance incorporates proof standards [Freeman and Farley, 1996], argument strength, and several ingredients available to a user. We illustrate briefly some of the capabilities of Carneades in a simple example[25] in Figure 10 and refer the reader for more details on the language and acceptability definitions to the literature [Gordon and Walton, 2016]. On the right part of Figure 10 there are six statements, i. e., that an object looks like a snake or a rope, and whether the object moved when jumped over or prodded. *Issue nodes* connect contradictory statements. Two arguments are formed (a1 and a2), which build on their premises (right of the figure) to conclude (left of the figure) that the seen object is indeed a snake or a rope. Let us assume that the object indeed looks like a snake and a rope (e.g. due to poor illumination), but neither did the object move when prodded with a stick nor when jumped over (e. g. by an adventurous person). In this case we conclude that the object is indeed a rope and not a snake (all premises of argument a2 are given but only one for a1).

The system Carneades (currently in version 4.2), features collaborative argument construction, argument visualization, and argument evaluation both for the structured arguments like we have seen in Figure 10 and also for Dung's AFs under grounded, complete, preferred, and stable semantics. Construction of structured arguments relies partially on internal calls to Prolog, and evaluation in the Carneades system can be classified as structural reasoning, since explicit abstract representation in the form of an AF is not utilized. Carneades is also available as a web-service and front-end [Gordon, 2012; Gordon, 2013], and includes a detailed manual.

3.6 Further implementations

Here we give pointers to related algorithms and implementations for structured argumentation that fall outside the previous sections.

In addition to other approaches to structured argumentation, **Tweety** [Thimm, 2014] features an implementation to structured argumentation as proposed in [Thimm and García, 2010]. Further, Wyner et al's [Wyner *et al.*, 2013] approach to instantiate rule-based knowledge bases with strict and defeasible rules as AFs has been encoded in ASP[26] [Strass, 2014].

A translational approach[27] to implement structured argumentation formalisms has been proposed in [van Gijzel and Nilsson, 2014] using Haskell as the programming

[25] Example taken from `http://carneades.github.io/` (on 27/04/2017). Variants of this example are discussed in [Walton *et al.*, 2014].

[26] Main ASP encoding available under `http://sourceforge.net/p/diamond-adf/code/ci/master/tree/lib/theorybase.lp` (on 27/04/2017).

[27] `http://www.cs.nott.ac.uk/~bmv/COMMA/` (on 27/04/2017).

language to capture definitions of these formalisms as directly as possible inside the programming language. For instance, in [van Gijzel and Nilsson, 2014] it is shown how to utilize this approach to translate Carneades to AFs: we call the corresponding system **CarneadesToDung**.

3.7 Summary

In this section we have given an overview of several algorithmic approaches to structured argumentation and their respective systems. Formalisms developed for structured argumentation and their implementations draw a quite heterogeneous picture. In particular, algorithms and systems range from query evaluation on the given structure to reasoning on an abstract representation where structural information is abstracted away. In Table 2 we see a summary of the presented approaches that have implementations and how they can be classified. Systems implementing structural reasoning typically solve queries in the form of deciding acceptance of a given claim and constructing arguments for this claim and counterarguments against the claim in a recursive fashion. Abstract reasoning involves construction of an abstract representation, i. e., an AF, and performing reasoning on this representation resulting typically in sets of extensions. For reduction-based approaches, the column "language" refers to the target formalism of the approach. These systems typically also include parsers or compilers written in an imperative language that translate or reduce the given input to the formalism. In this table, ASP stands for answer-set programming, SAT for satisfiability solvers, and MUS for solvers capable of solving problems related to minimal unsatisfiable subsets of formulas.

The **Tweety** libraries [Thimm, 2014] implement several reasoning tasks from multiple formalisms for structured argumentation. We name the respective approaches in parenthesis for **Tweety**. We note that not all tools mentioned in Table 2 provide reasoning support themselves, i. e., some tools focus on argument construction and delegate evaluation to other systems. The tools **BA** [Besnard *et al.*, 2010] and **vispartix** [Charwat *et al.*, 2012] handle argument construction for deductive argumentation without evaluation, in particular, **BA** generates arguments and **vispartix** an AF. One of **Tweety**'s algorithms translates a given DeLP to an AF and leaves the choice for an AF reasoner to the user. **CarneadesToDung** [van Gijzel and Nilsson, 2014] translates input as specified in the Carneades model to a Dung AF. **TOAST** [Snaith and Reed, 2012] incorporates Dung-O-Matic [Snaith *et al.*, 2010] for evaluation.

	Direct	Reduction	Language	(Query-based) Structural Reasoning	Reasoning on Abstract Representation
ASPIC+					
TOAST	Yes		Java		Yes
ASPIC Inference Engine	Yes		Java	Yes	
EPR	Yes		Java	Yes	
Argue tuProlog		Yes	Prolog	Yes	
ABA					
CaSAPI		Yes	Prolog	Yes	
proxdd		Yes	Prolog	Yes	
abagraph		Yes	Prolog	Yes	
grapharg		Yes	Prolog	Yes	
ABAplus		Yes	ASP		Yes
ABAToAF		Yes	ASP		Yes
DeLP					
DeLP client		Yes	Prolog	Yes	
Tweety (DeLP)	Yes		Java	Yes	
Tweety (DeLP to AF)	Yes		Java		Yes
ASP-RP-DeLP		Yes	ASP	Yes	
SAT-R-DeLP		Yes	SAT	Yes	
Deductive					
JArgue	Yes		Java	Yes	
Tweety (deductive)		Yes	Java/MUS	Yes	
vispartix		Yes	ASP		Yes
BA		Yes	MUS		
Carneades					
Carneades		Yes	Prolog	Yes	
CarneadesToDung			Haskell		Yes

Table 2: Summary table for structured implementations.

4 Other Implementation Approaches

This paper would not be complete without a description of implemented systems that provide a general purpose gateway to formal structures of argumentation. They are, for instance, systems supporting text annotation for producing corpora that can be exploited by argument mining algorithms as well as systems for supporting critical thinking by the means of formal models of argumentation thus reusing elements discussed in previous sections. Our aim here is to summarize the most notable examples with some guidance for the reader interested in using—or reusing—existing implementations.

In particular, we analyse 34 promising implementations chosen among those that are active projects. Since it is beyond the scope of this paper to provide a comprehensive description for each of those, we briefly review them in Section 4.1. Moreover, there are four additional projects that, although they appear to have been discontinued, have been relevant from an academic perspective, and we believe they should be mentioned in order to provide the reader with a complete background. Those are reviewed in Section 4.2, while in Section 4.3 we provide a comparative analysis of the active projects. Finally, the excellent review of Schneider et al. [Schneider *et al.*, 2013] mentions other interesting projects—mostly online platforms—that are briefly discussed in Section 4.4, even if they do not implement any evident formal model of argumentation.

4.1 Active Projects

The following 34 systems are representative among active projects incorporating some argumentation techniques.

AGORA [Hoffmann, 2005; Hoffmann, 2007] is a Computer-Supported Collaborative Argument Visualization (CSCAV) tool. An argument is defined here as a set of statements—claim and one or more reasons—where the reasons jointly provide support for the claim, or are at least meant to support the claim.

AIFdb [Lawrence *et al.*, 2012b] is a database solution for the Argument Web thus implementing the AIF model of arguments [Bex *et al.*, 2013; Rahwan *et al.*, 2011; Chesñevar *et al.*, 2006]. AIFdb offers an array of web service interfaces allowing a wide range of software to interact with the same argument data. Various dataset are available as part of the Argument Corpora [Reed, 2013].

AnalysisWall [Bex *et al.*, 2013] is a collaborative workspace, a touchscreen measuring 11 feet by 7 feet, located at the University of Dundee.

Arg&Dec [Aurisicchio *et al.*, 2015] is a web application for collaborative decision-making, encompassing the quantitative argumentation-based framework QuAD, and its decision matrix model, assisting their comparison through automated transformation.

ArgTeach [Dauphin and Schulz, 2014] is an interactive tutor that facilitates the learning of different labelling semantics in abstract argumentation. It now exists both as a standalone desktop application and as a web application.[28]

ArgTrust [Tang *et al.*, 2012] relates the grounds of an argument to the agent that supplied the information, and can be used as the basis to compute acceptability statuses of arguments that take trust into account.

ArgueApply [Pührer, 2017] is a Java app for mobile phones, with a graphical interface, that lets users put forward arguments, and positive or negative links between arguments, in a fragment of the GRAPPA [Brewka and Woltran, 2014] language.[29]

ArgMed [Hunter and Williams, 2012; Williams *et al.*, 2015] is a project investigating the use of computational argumentation for analysing and aggregating clinical evidence for making recommendations. In addition to the theoretical framework, it also has a public website.[30]

ArguMed [Verheij, 1998] introduces **ARGUE!**, based on the logical system CUMULA that abstractly models defeasible argumentation [Verheij, 1996a]. The development of **ARGUE!** was soon followed by the **ArguMed** family [Verheij, 2003a] based on the DefLog system [Verheij, 2003b], where dialectical arguments consist of statements that can have two types of connections between them: a statement can support another, or a statement can attack another. Dialectical arguments can be evaluated with respect to a set of prima facie justified assumptions.

[28] `http://www-argteach.doc.ic.ac.uk/` (on 27/04/2017).
[29] `http://www.informatik.uni-leipzig.de/~puehrer/ArgueApply/` (on 27/04/2017).
[30] `http://www0.cs.ucl.ac.uk/staff/a.hunter/projects/argmed/` (on 27/04/2017).

Argument Blogging [Bex *et al.*, 2014] allows users to construct debate and discussions across blogs, linking existing and new online resources to form distributed, structured conversations. Arguments and counterarguments can be posed by giving opinions on one's own blog and replying to other bloggers' posts. The resulting argument structure is connected to the Argument Web [Bex *et al.*, 2013], in which argumentative structures are made semantically explicit and machine-processable.

Argunet [Schneider *et al.*, 2007] is a desktop tool coupled with an open source federation system for sharing argument maps.

Arvina [Bex and Reed, 2012; Lawrence *et al.*, 2012a] is a dialogical support system that allows for the structured execution of a reasoning process by implementing dialogue protocols and then allowing users to play the dialogue game against virtual agents and against each other in an instant-messaging environment.

ASPARTIXWeb [Egly *et al.*, 2010b] is a web-based interface to the ASPARTIX system for computing extensions for various semantics of abstract argumentation.[31]

bCisive is a professional argument mapping and critical thinking support system.[32]

CISpaces [Toniolo *et al.*, 2014; Toniolo *et al.*, 2015] is an agent-based tool to help intelligence analysts in acquiring, evaluating, and interpreting information in collaboration. Agents assist analysts in reasoning with different types of evidence to identify what happened and why, what is credible, and how to obtain further evidence. Argument schemes lie at the heart of the tool, and sensemaking agents assist analysts in structuring evidence and identifying plausible hypotheses. A crowdsourcing agent is used to reason about structured information explicitly obtained from groups of contributors, and provenance is used to assess the credibility of hypotheses based on the origin of the supporting information.

Cohere/Compendium [De Liddo and Buckingham Shum, 2010; Shum, 2008] is an open source software for sensemaking using argumentation maps and annotation.

[31] `http://rull.dbai.tuwien.ac.at:8080/ASPARTIX/index.faces` (on 27/04/2017).
[32] `https://www.bcisiveonline.com/` (on 27/04/2017).

ConargWeb is a web-based interface to the Conarg system for computing extensions of Dung's argumentation frameworks.[33]

CoPe_it! [Tzagarakis *et al.*, 2009] is a tool to support synchronous and asynchronous argumentative collaboration in a Web environment. It introduces the notion of incremental formalization of argumentative collaboration. The tool permits a stepwise evolution of the argumentation space, through which formalization is not imposed by the system but is at the user's control. By permitting the users to formalize the discussion as the collaboration proceeds, more advanced services can be made available. Once the collaboration has been formalized to a certain point, CoPe_it! can exhibit an active behavior facilitating the decision making process.

D-BAS [Krauthoff *et al.*, 2016] is a web and dialogue-based system to facilitate online argumentation, with the aim to guide users through statements, their pro-arguments and counterarguments, and adding new arguments as well as conflicts between these arguments.[34]

Debategraph [Macintosh, 2009] is a collaborative debate visualisation tool.

GERD [Dvořák *et al.*, 2015] is a web-based interface of an ASP-based system for enumerating extensions of various semantics of the framework from [Modgil, 2009], which extends Dung's abstract argumentation framework with preferences among arguments.[35]

Gorgias [Kakas and Moraitis, 2003] is a general argumentation framework that combines preference reasoning and abduction. It can form the basis for reasoning about adaptable preference policies in the face of incomplete information from dynamic and evolving environments [Kakas *et al.*, 1994].

Gorgias-B [Spanoudakis *et al.*, 2016] supports the development of applications of argumentation under **Gorgias**. **Gorgias-B** guides the developer to structure their knowledge at several levels. The first level serves for enumerating the possible decisions and arguments that can support these options under some conditions, while each higher level serves for resolving conflicts at the previous level by taking into account default or contextual knowledge.

[33] `http://www.dmi.unipg.it/conarg/` (on 27/04/2017).
[34] `https://dbas.cs.uni-duesseldorf.de/` (on 27/04/2017).
[35] `http://gerd.dbai.tuwien.ac.at/index.php` (on 27/04/2017).

Grafix [Cayrol *et al.*, 2014] is a graphical tool for handling abstract argumentation frameworks and bipolar frameworks. Grafix allows editing and drawing of argumentation graphs (or sets of graphs), and the execution of some "predefined treatments" (called "server treatments") on the current graph(s), such as, e. g., computing various acceptability semantics, or computing the strength of arguments.

GrappaVis is a Java graphical tool to specify GRAPPA [Brewka and Woltran, 2014] and ADF [Brewka *et al.*, 2013] frameworks, evaluate them, and visualize the results of the evaluation. In particular, GRAPPA is a general semantical framework for assigning a precise meaning to graphical models of arguments or labelled argument graphs, which makes them suitable for automatic evaluation. GRAPPA rests on the notion of explicit acceptance conditions, as discussed in ADF [Brewka *et al.*, 2013].[36]

MARFs (Markov Argumentation Random Fields) [Tang *et al.*, 2016] is a system combining elements of formal argumentation theory and probabilistic graphical models. In doing so it provides a principled technique for the merger of probabilistic graphical models and non-monotonic reasoning.

Opinion Space [Faridani *et al.*, 2010] is an online interface incorporating ideas from deliberative polling, dimensionality reduction, and collaborative filtering that allows participants to visualize and navigate through a diversity of comments.

OVA+ [Janier *et al.*, 2014] provides a drag-and-drop interface for analysing textual arguments. It is designed to work with web pages It is available as a web interface and does not require a local installation. It also natively handles AIF structures, and supports real-time collaborative analysis.

Parmenides [Cartwright and Atkinson, 2008; Cartwright *et al.*, 2009; Cartwright and Atikinson, 2009] is primarily a forum by which government bodies can present policy proposals to the public so that users can submit their opinions on the justification presented for a particular policy. Within Parmenides, the justification for action is structured to exploit a specific representation of persuasive argument based on the use of argumentation schemes and critical questions.

[36] `http://www.dbai.tuwien.ac.at/proj/adf/grappavis/` (on 27/04/2017).

PIRIKA (PIlot for the RIght Knowledge and Argument) [Oomidou *et al.*, 2014] is an argument-based communication tool for humans and agents, which supplements current communication systems such as Twitter. It allows for asynchronous argumentation for anyone, anytime, anywhere on any issues, as well as synchronous argumentation and stand-alone argumentation.

Quaestio-it [Evripidou and Toni, 2014] is based on a framework for modelling and analysing social discussions. It offers debating infrastructure for opinion exchanges between users and providing support for extracting intelligent answers to user-posed questions.

Rationale is a professional argument mapping and critical thinking support system.[37]

Reason [Introne, 2009] is a platform for supporting group decisions by leveraging the argumentative structure of deliberative conversation to drive a decision support algorithm. The platform uses argument visualization to mediate the collaborators' conversation.

Truthmapping is a professional, collaborative argument mapping tool.[38]

4.2 Discontinued Projects

In addition to the 34 systems discussed in Section 4.1, we briefly mention the following four as well. Although discontinued at the time of writing, those works have significantly impacted the research field and are still inspirational.

Avicenna [Rahwan *et al.*, 2011] is an OWL-based argumentation system that consists of three main tiers: the data tier, the middle tier, and the client tier. The argumentation ontology is stored in the form of RDF statements (triples) in the back-end database, which constitutes the data tier. The middle tier is responsible for reasoning based on description logics and the interface to the web, through which applications in the client tier connect.

[37] `http://rationale.austhink.com/` (on 27/04/2017).
[38] `https://www.truthmapping.com/` (on 27/04/2017).

Dispute Finder [Ennals *et al.*, 2010] is a browser extension that alerts a user when information they read online is disputed by a source that they might trust. Dispute Finder examines the text on the page that the user is browsing and highlights any phrases that resemble known disputed claims. If a user clicks on a highlighted phrase then Dispute Finder shows her a list of articles that support other points of view.

SEAS [Lowrance *et al.*, 2008] is a collaborative, semi-automatic approach to evidential reasoning that uses template-based structured argumentation. Graphical depictions of arguments readily convey lines of reasoning, from evidence through to conclusions, making it easy to compare and contrast alternative lines of reasoning.

Trellis [Chklovski *et al.*, 2003] allows users to add their observations, viewpoints, and conclusions as they analyze information by making semantic annotations to documents and other on-line resources. Users can associate specific claims with particular locations in documents used as "sources" for analysis, and then structure these statements into an argument detailing pros and cons on a certain issue.

4.3 Comparative Analysis

To provide a concise overview over the *active* systems discussed in Section 4.1, we identified seven features that characterize the commonalities and differences among those systems, namely whether a system

(F1) is able to handle some form of structured argumentation;

(F2) gives the ability to manipulate arguments;

(F3) is collaborative;

(F4) enables a dialogue between different parties involved in its usage; and, in particular, if it

(F5) enables a dialogue based on speech acts;

(F6) includes a reasoner based on Dung's theory of abstract argumentation; or if it

(F7) includes a reasoner not based on Dung's theory of abstract argumentation.

It is evident that F5 is a specific case of F4: if a system offers speech acts, by definition it also offers a dialogue system. Moreover, F6 and F7 only apparently are mutually exclusive: indeed, a system can offer multiple choices of reasoners—the

case of **CISpaces**—or it can encompass Dung's theory of abstract argumentation as a special case—e. g. **MARFs**.

Table 3 provides a comparative overview of the 34 active projects from Section 4.1 with respect to the seven features identified. This list of features is clearly far from being complete or unquestionable. However, it is sufficient for describing a large variety of possible usages of the systems.

Indeed, if a system supports F1 and F6, it is evident that it can be used in the *conventional* meaning of structured argumentation and perhaps it implements a specific approach for structured argumentation [Besnard *et al.*, 2014]. This is, for instance, the case of **OVA+**, which allows to represent and reason about ASPIC+ knowledge bases. Moreover, since **OVA+** also possesses the feature F2, it is evident that it can be used interactively; and since it possesses F3 as well, it can used in a distributed fashion.

It is worth noticing that there is only one system exhibiting all the seven features, CISpaces, which is unfortunately not (yet) available as an open-source implementation. Differently from OVA, CISpaces implements a subset of ASPIC, notably the ability to express only defeasible rules, and it follows a customised methodology for handling preferences, similar to ASPIC+ but using AFRA [Baroni *et al.*, 2011b] as the meta-representation system. However, it also encompasses both the ability to use an evolution of **ArgTrust** as a web-service, as well as models of probabilistic reasoning based on [Li *et al.*, 2012].

To conclude this analysis, it is worth showing the chronological evolution of all 38 systems reviewed in this survey, depicted in Figure 11. It is evident that 2014 has been the most prolific year, as also testified by the significant number (19) of demo submissions to COMMA 2014.

4.4 Projects for Informal Argumentation

Following the review of Schneider et al. [2013], there are further systems worth mentioning that make use of "informal" argumentation techniques. Indeed, they tend to be closer to user experience and they generally have a low entry barrier. At the same time, they do not offer much support for structuring arguments in a formal fashion, nor automated reasoning capabilities.

There is a large number of social networking debating systems such as Arguehow,[39] Climate CoLab [Gürkan *et al.*, 2010], ConsiderIt [Kriplean *et al.*, 2011],

[39] `http://arguehow.com/` (on 27/04/2017).

	F1	F2	F3	F4	F5	F6	F7
AGORA	Yes	Yes	Yes				
AIFdb	Yes		Yes			Yes	
AnalysisWall	Yes	Yes	Yes			Yes	
Arg&Dec		Yes	Yes	Yes			Yes
ArgTeach						Yes	
ArgTrust	Yes	Yes					Yes
ArgueApply		Yes	Yes	Yes		Yes	Yes
ArgMed	Yes	Yes				Yes	
ArguMed	Yes	Yes					Yes
Argument Blogging	Yes	Yes	Yes				
Argunet	Yes	Yes	Yes				
Arvina	Yes			Yes	Yes	Yes	
ASPARTIXWeb						Yes	
bCisive	Yes	Yes					
CISpaces	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Cohere/Compendium	Yes	Yes	Yes				
ConargWeb		Yes				Yes	
CoPe_it!	Yes	Yes	Yes				
D-BAS	Yes		Yes	Yes			
Debategraph	Yes	Yes	Yes	Yes			
GERD						Yes	Yes
Grafix						Yes	Yes
GrappaVis	Yes					Yes	Yes
Gorgias	Yes	Yes					Yes
Gorgias-B	Yes	Yes					
MARFs	Yes					Yes	Yes
Opinion Space			Yes				
OVA+	Yes	Yes	Yes	Yes		Yes	
Parmenides	Yes			Yes			
PIRIKA		Yes	Yes			Yes	
Quaestio-it	Yes	Yes	Yes	Yes			Yes
Rationale	Yes	Yes					
Reason	Yes	Yes	Yes				
Truthmapping	Yes	Yes	Yes	Yes			

Table 3: Comparative overview of systems (discontinued systems are omitted) using some form of formal argumentation. F1: structured argumentation; F2: argument manipulation; F3: collaborative; F4: enables dialogues, F5: based on speech acts; F6: Dung's reasoner, or F7: non-Dung's reasoner.

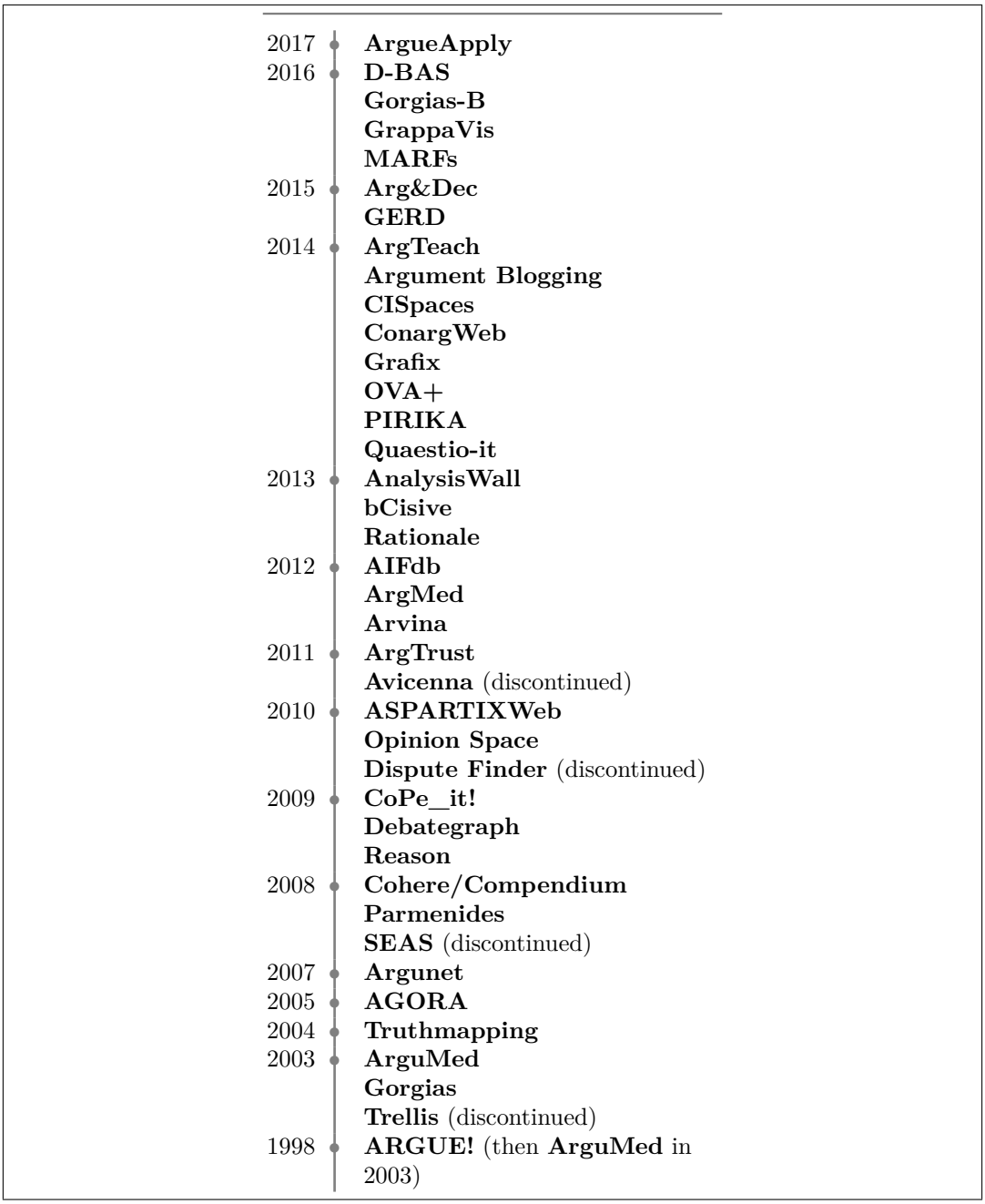


Figure 11: History of systems from Section 4, both active and discontinued. The year refers to the first tracked publication or to the first time the system appears online.

ConvinceMe,[40], CreateDebate,[41] Debate.org,[42] Debatepidia,[43] Debatewise,[44] Hypernews,[45] and LivingVote.[46] Further systems worth mentioning are, e. g., Belvedere,[47] an open-source critical thinking support system; the Cabanac's annotation system[48] for investigating social validation of arguments in comments; and DiscourseDB,[49] that is used to collaboratively collect policy-related commentary.

5 Challenges

In this section we discuss current challenges in devising and implementing algorithms for solving problems related to formal argumentation. In particular, for abstract argumentation problems we discuss parallel algorithms (Section 5.1), approximation algorithms (Section 5.2), and dynamic selection of algorithms depending on graph features (Section 5.3). We also have a brief look at advanced techniques and the related challenges for some structured argumentation approaches (Section 5.4).

5.1 Parallelization

Reasoning tasks related to computational models of argumentation in general, and abstract argumentation in particular, are usually hard from the perspective of computational complexity, cf. e. g. [Dunne and Wooldridge, 2009]. In order to make systems applicable to real-world scenarios, specific measures have to be taken in order to overcome the NP-complexity barrier—or even higher. One such measure is to use *parallelization*. Modern computing systems usually provide many CPU cores that allow for multiple threads to be executed in parallel. Moreover, grid- or cluster-based systems collect the computational capacity of many single machines and provide an abstraction with access to many computing cores. In order to exploit the computational power of such parallel systems, algorithms have to be devised that allow for the decomposition of complex problems, independent solving of the individual sub-problems, and an effective aggregation of the partial results into a global solution. While not every computational problem allows for such a parallelization—or

[40] `http://hamschank.com/convinceme/index.html` (on 27/04/2017).
[41] `http://www.createdebate.com/` (on 27/04/2017).
[42] `http://debate.org` (on 27/04/2017).
[43] `http://www.debatepedia.com/` (on 27/04/2017).
[44] `http://debatewise.org/` (on 27/04/2017).
[45] `http://sourceforge.net/projects/hypernews/` (on 27/04/2017).
[46] `http://www.livingvote.org/` (on 27/04/2017).
[47] `http://belvedere.sourceforge.net/` (on 27/04/2017).
[48] `http://www.irit.fr/~Guillaume.Cabanac/expe/` (on 27/04/2017).
[49] `http://www.discoursedb.org/` (on 27/04/2017).

at least does not allow for parallelization with a significant gain in performance—parallelization has been applied to many NP-complete (or harder) problems in the past with some success, most notably to the problem SAT [Hölldobler *et al.*, 2011] allowing for considerable speed-ups on certain subclasses of instances.

For abstract argumentation, a natural feature to exploit for devising parallel algorithms is SCC-recursiveness [Baroni *et al.*, 2005]. A semantics is SCC-recursive if the problem of enumerating the extensions for the graph as a whole can be be decomposed in computing the extensions of its *strongly connected components*[50] (SCC). Once SCCs have been identified, extensions can be computed on each SCC separately and the resulting sub-extensions can be joined in order to obtain the extensions of the whole graph paying attention to the inter-dependencies among different SCCs.[51] This basic approach is followed by the algorithm presented in [Cerutti *et al.*, 2015], which itself is an enhancement to the previously published algorithm from [Cerutti *et al.*, 2014e].

The approach for parallelizing the computation of extensions in abstract argumentation outlined in [Cerutti *et al.*, 2015] is effective as long as the number of SCCs is "relatively" large in comparison to the size of the argumentation framework. Computing the SCCs of a graph can be done in polynomial time (see e. g. Tarjan's algorithm [Tarjan, 1972]) and, thus, the computational overhead of decomposing the problem is negligible in comparison to the computational effort of computing extensions, which is, as discussed before, often NP-hard or harder, depending on the chosen semantics. The computational effort required for the aggregation step is highly dependent on the actual instance of the problem and may be exponential in the worst case, as a sub-graph may possess an exponential number of extensions [Baumann and Strass, 2014] that need to be aggregated. However, for "reasonable" instances, this step is also negligible in comparison to the effort of computing extensions. As the empirical evaluation in [Cerutti *et al.*, 2015] suggests, exploiting SCC-recursiveness for parallelization may yield a speedup (up to 280%) when increasing the number of cores from 1 to 4.

Another approach to parallelization is not based on decomposing a problem into sub-problems, but on parallel execution of different algorithms for the whole problem. For many computationally hard problems there is usually a limited number of algorithms that can solve "most" of the instances in reasonable time, and the core problem is to determine *which* algorithm should be selected to solve a particular instance. This problem is called the *Algorithm selection* problem and will be dis-

[50] A subgraph of a directed graph is a strongly connected component, if there is a directed path from every vertex to each vertex and the subgraph is maximal.

[51] Other decomposition methods might take advantages of I/O-multipoles [Baroni *et al.*, 2014], but no approaches have been yet proposed.

cussed in more detail in Section 5.3. It is worth noticing that [Vallati *et al.*, 2017] proposes a first parallel algorithm selection approach. A straightforward solution to this problem is to devise a meta-algorithm that runs several algorithms on the original problem in parallel. As soon as the first algorithm terminates, the meta-algorithm terminates as well and the result of the meta-algorithm is the result of this algorithm. This approach, also called *variant-based parallel computation*, has been implemented in [Craven *et al.*, 2012] for the problem of deciding acceptance of arguments in *assumption-based argumentation* (ABA)[52] and has been applied in the medical domain. More specifically, the approach of [Craven *et al.*, 2012] is based on discussion games and different algorithms for solving acceptance use different expansion strategies in advancing the game.

The two approaches from above are complementary in the way how parallelization is realized. While the first approach uses a single algorithm and decomposes the problem instance into a parallel execution, the second approach uses multiple algorithms on the whole problem. Of course, combinations of the paradigms are imaginable.

5.2 Approximation Techniques

Parallelization offers an approach to overcome complexity barriers while maintaining soundness and completeness. A different and also often applied approach is to give up soundness and/or completeness and devise *approximation algorithms*, see e. g. [Vazirani, 2002; Cormen *et al.*, 2009]. Roughly, an approximation algorithm is not expected to solve the problem correctly but only within a certain margin of error. On the other hand, an approximation algorithm is expected to be more efficient than a correct algorithm.

In general, an algorithm A is said to be an ϵ-approximation algorithm for an optimization problem P (with $\epsilon > 0$), if for every instance the output of A is in the interval $[(1-\epsilon)C, (1+\epsilon)C]$, where C is the optimal solution, and ϵ thus represents the relative error in the approximation. Usually, one is interested in polynomial-time ϵ-approximation algorithms with ϵ being as small as possible. In case the algorithm returns more refined solutions—i. e. it decreases the ϵ-approximation further—if provided with additional runtime, it belongs to the class of *anytime algorithms*.

Approximation techniques for problems of abstract argumentation have not been investigated in-depth yet, with only very few exceptions. For example, the equational approach to abstract argumentation (see also [Gabbay, 2012; Gabbay and Rodriguez, 2014]) views an argumentation framework as a generator of equations for value

[52] While ABA is actually an approach to structured argumentation, we discuss it here as it is the only known parallel approach to structured argumentation.

assignments V such that $V(X) = 1$ indicates that X is in; $V(X) = 0$ indicates that X is out; and $V(X) \in (0, 1)$ that X is undecided. In [Gabbay and Rodriguez, 2014] the authors introduce an iteration schema for computing complete extensions, starting from an arbitrary assignment V_0 and then, by use of a specific update rule, generating a sequence of assignments $V_0, V_1, \ldots$. In [Gabbay and Rodriguez, 2014] it is shown that this sequence will eventually converge and form a complete extension. This algorithm can therefore be interpreted as an anytime algorithm for computing complete extensions, but a thorough analysis of this algorithm in terms of approximation quality has not been done yet.

In the area of *probabilistic abstract argumentation* [Li *et al.*, 2012; Thimm, 2012; Hunter, 2014], which is concerned with combining abstract argumentation frameworks with probabilistic reasoning, approximation techniques from probabilistic reasoning have been applied to overcome the additional complexity necessary to deal with quantitative uncertainty [Hadoux *et al.*, 2015; Li *et al.*, 2012]. As probabilistic abstract argumentation is a topic that will be covered in later volumes of this handbook, we omit discussing these techniques here.

In summary, approximation techniques for computational models of arguments are still underdeveloped, but may gain attention in the near future.

5.3 Algorithm Selection

In Section 5.1 we already discussed the variant-based parallel computation approach of [Craven *et al.*, 2012] which is a specific solution for solving the *Algorithm Selection* problem by running different algorithms for the same problem in parallel. If parallelization is not possible for devising an algorithm, another solution is given by the *algorithm portfolio* approach [Rice, 1976; Leyton-Brown *et al.*, 2003; Xu *et al.*, 2008]. A portfolio is a meta-algorithm that has access to several specific algorithms for solving the same problem. When presented with a problem instance, the meta-algorithm selects one of those specific algorithms. In the case of *dynamic portfolios*, the meta-algorithm first extracts some *features* of the problem instance and then selects an algorithm that has, in a preprocessing step, proven to be the best algorithm for instances with the given features. This approach has been proven quite successful in solving many hard problems, such as SAT [Xu *et al.*, 2008].

The crucial step in developing a dynamic portfolio algorithm is to define which features are relevant both to assess the quality of the algorithms in the preprocessing step and to select the appropriate algorithm during runtime. Furthermore, it is important that the overhead introduced for computing features of the problem instance during runtime is "reasonably" small. In [Vallati *et al.*, 2014b; Cerutti *et al.*, 2014b] the authors presented 50 features of abstract argumenta-

tion frameworks and derived empirical performance models (EPMs) to determine the "best" implementation for enumerating preferred extensions, given CPU-time as evaluation criterion and a limited set of solvers. The features considered there were basic graph theory-based measures such as size of the graph, average degree of arguments, flow hierarchy, and so on. The two EPMs presented in [Cerutti *et al.*, 2014b] show an overall accuracy of 80% (classification) and, depending on the implementation, the ability to predict quite accurately the runtime required by a solver to enumerate the preferred extensions (regression). Unsurprisingly, the set of most informative features—according to a greedy forward search-based on the Correlation-based Feature Selection attribute evaluator [Hall, 1998] and with respect to the experimental setting used by the authors—includes the density of the argumentation graph, as well as number of SCCs and the size of the maximum SCC. When the computed EPMs have been applied to the problem of algorithm selection, both of them perform significantly well: in 78% of cases (resp. 75%) the classification-based EPM (the regression-based EPM) selects the best implementation. In most of the cases, 83%, both EPMS select the same algorithm, which is the correct one in 82% of cases.

Complete static and dynamic portfolios have been proposed in [Cerutti *et al.*, 2016d], and parallel portfolios are proposed and discussed in [Vallati *et al.*, 2017]. However, it is still unclear whether there may be better features to use for the selection problem or whether a combination of different techniques discussed in this section may yield improved performance. In [Brochenin *et al.*, 2015], *abstract solvers* [Nieuwenhuis *et al.*, 2006] are used as a formal machinery to formally specify different algorithms addressing extension-enumeration problems. By using these formalizations, algorithms could be combined and extended to more effective algorithms. Hence, using this machinery to also include the concepts discussed in this section may be a fruitful endeavor.

5.4 Advanced Techniques for Structured Argumentation

In structured argumentation, further computational problems than argument evaluation may occur. Many approaches to structured argumentation consider a knowledge base formalized in some logical formalism, and then derive arguments and conflicts between them on top of that, cf. Figure 4. Therefore, additional computational effort is required to construct arguments and to discover the conflict relationship between them. In general, computational approaches to structured argumentation can be categorized in two classes: those that use abstract argumentation frameworks as the underlying argument evaluation mechanism and those that provide proprietary evaluation mechanisms.

For the class of approaches providing proprietary evaluation mechanisms—such as Defeasible Logic Programming and earlier versions of Deductive Argumentation—the processes of argument construction, defeat discovery, and argument evaluation are usually intertwined, but each step still imposes some challenges.[53]

For argument construction, an important issue is relevance of arguments. In particular, for approaches building on classical logics—such as Deductive Argumentation—the number of arguments that can be derived from knowledge base may be potentially infinite. Given a specific query to the knowledge base, usually only those arguments are constructed that are relevant to the query and possess a certain normal form (in Deductive Argumentation these are the *maximally conservative undercuts*). In [Besnard and Hunter, 2006] an effective method for constructing both arguments and the defeat relation for a certain query is presented. This method relies on a preprocessing step that generates a so-called *compilation* from a knowledge base, which is an undirected graph with vertices being the minimal inconsistent subsets of the knowledge base and two vertices are connected if they have a non-empty intersection. Given a specific query, a traversal algorithm allows the complete construction of an argument tree from this compilation. Considering only *approximate arguments* [Hunter, 2006a]—e. g. arguments which are not necessarily minimal—also allows to gain efficiency by trading-off completeness or soundness (to some extent).

Another advanced technique for structured argumentation is *pruning of dialectical trees* in, e. g., Defeasible Logic Programming [Chesñevar *et al.*, 2000; Chesñevar and Simari, 2007; Rotstein *et al.*, 2011]. This technique also offers a solution to refrain from considering all arguments for evaluating a query. This is realized by only expanding the dialectical tree so far until the evaluation status of the query is decided. For example, if an argument possesses multiple attackers, and it can already be decided that the first attacker is ultimately accepted and defeats the argument, then there is no need to evaluate the acceptance status of the remaining attackers as it can already be decided that the argument under consideration is not acceptable. Yet another approach to address the very same issue is to evaluate different argumentation lines in a dialectical tree in parallel [García and Simari, 2000].

[53] For those approaches relying on abstract argumentation for argument evaluation, similar sophisticated techniques as outlined in this and the previous sections apply, but will not be discussed separately.

6 Evaluation of Implementations

While theoretical approaches to computational models of argumentation are usually analytically evaluated using rationality postulates or comparison of behavior on toy examples—see e. g. [Gorogiannis and Hunter, 2011; Caminada and Amgoud, 2005; Amgoud, 2014]—the evaluation of algorithms and implementations focuses on the three aspects of *correctness*, *performance*, and *usability*. The correctness of algorithms and implementations is usually shown in an analytical way and involves showing that the algorithmic representation corresponds to the formal definition, e. g. that the result of performing an algorithm indeed returns the grounded extension of a given abstract argumentation framework. In order to evaluate an *algorithm* with respect to performance, one usually conducts an analytical runtime or complexity analysis. For the performance evaluation of *implementations* an empirical evaluation on either artificial or real-world benchmarks and runtime measurement on the corresponding computational problems is essential for obtaining a comparative analysis of different approaches. Finally, in order to evaluate the usability of implementations, user studies have to be performed.

For the remainder of this section, we will focus on the problem of empirical performance evaluation of implementations of computational models of argumentation. In particular, we will focus on evaluations of implementations that solve problems for abstract argumentation frameworks, cf. Section 2. Those problems are an important aspect of any evaluation of implementations as well, as they provide clear formalizations of what are the expected outcomes of computational tasks. Another important aspect of such evaluations is the identification of suitable benchmarks, i. e. abstract argumentation graphs, that can be used to compare the performance of different implementations, which we discuss in Section 6.1. We discuss existing comparative analyses, in particular the *International Competition on Computational Models of Argumentation* (ICCMA),[54] in Section 6.2.

6.1 Benchmark Examples

A crucial issue in setting up an evaluation of an implementation of abstract argumentation problems is the identification of argument graphs that are used as benchmark examples. Ideally, real-world applications would provide these kind of benchmark graphs in order to test implementations on actually existing problems. Unfortunately, the availability of real-world benchmarks for argumentation problems is quite limited, some few exceptions are [Cabrio *et al.*, 2013; Cabrio and Villata, 2014b;

[54] `http://argumentationcompetition.org` (on 27/04/2017).

Cabrio and Villata, 2014a] and AIFdb.[55] Moreover, these benchmarks are tailored towards problems of argument mining [Wells, 2014] and their representation as abstract argumentation frameworks usually lead to topologically simple graphs, such as cycle-free graphs, which are unsuitable for comparing abstract argumentation solvers: all classical semantics coincide with grounded semantics on cycle-free graphs [Dung, 1995]. In order to compare solvers for—among others—preferred and stable semantics, artificially-generated argumentation graphs have been used so far.

Generating graphs for testing computational approaches or hypotheses on physical or social phenomena has already some tradition in network theory [Erdös and Rényi, 1959; Albert and Barabási, 2002; Pfeiffer *et al.*, 2012; Tabourier *et al.*, 2011; Barabasi and Albert, 1999]. However, it is questionable whether these graph models are suitable to model argumentation problems. For instance, the Barabási-Albert model [Barabasi and Albert, 1999] generates networks based on *preferential attachment*. The concept *preferential attachment* refers to the tendency of nodes that have already many connections to other nodes, to receive even more connections in the evolution of the network: an example of this phenomenon is the saying "the rich get richer, while the poor get poorer." To the best of our knowledge, there is no evidence that real-world argumentation adheres to this concept. Another concept from network theory often (indirectly) implemented in graph models is that of *triangle closure*, i. e., the tendency of nodes directly connecting to the neighbors of its neighbors (as in the saying "the friend of my friend is also my friend"). This concept is hardly applicable to argumentation graphs as this would imply that *defense* (an argument attacking the attacker of another argument) tends also to be a *direct attack* (the first argument attacking the argument it also defends).

Graph models from network theory also usually generate undirected graphs. Adapting a model to generate directed edges is of course trivial, but it is questionable whether the resulting graphs have any interpretation with respect to the original intention of the model.

Finally, from the perspective of challenging benchmarks for abstract argumentation, the graphs generated by such models are usually also not adequate. Initial experiments for ICCMA'15 [Thimm *et al.*, 2016] (see also below and the next section) suggest that those generated graphs usually contain an empty or a very small grounded extensions, usually no stable extensions (also due to the triangle closure property), and very few and small complete and preferred extensions. The latter observation is due to the fact that these graph models aim at modeling the "small world" property of many real-world graphs.[56] This leads to many arguments di-

[55] `http://corpora.aifdb.org` (on 27/04/2017).

[56] This property basically states that there are always "relatively short" paths from any node to every other node [Watts and Strogatz, 1998], provided that the network is connected and not too

rectly or indirectly being in conflict with each other. However, these models have been used for benchmark generation in earlier evaluations of implementations of abstract argumentation solvers [Bistarelli *et al.*, 2013; Bistarelli *et al.*, 2014].

In order to provide challenging benchmarks, ICCMA'15 used proprietary graph generators, each addressing different aspects of computationally hard graphs for specific semantics. For example, the `StableGenerator` aims at generating graphs with many stable extensions, and thus also many complete and preferred extensions. Graphs generated by this generator pose substantial combinatorial challenges for solvers addressing the computational tasks of determining (skeptical or credulous) acceptance of arguments and of enumerating extensions. For a given number of arguments, this generator first identifies a subset of these arguments to form an acyclic subgraph which will contain the grounded extension. Afterwards, another subset of arguments is randomly selected and attacks are randomly added from some arguments within this set to all arguments outside the set (except to the arguments identified in the first step). This process is repeated until a number of desired stable extensions is reached. The source code for this and other generators can be found in the source code repository[57] of *probo* [Cerutti *et al.*, 2014f], the benchmark suite used to run the competition. Another general tool for generating argumentation frameworks from a set given graph features is given by AFBenchGen[58] [Cerutti *et al.*, 2014d; Cerutti *et al.*, 2016a].

6.2 Comparative Analysis

The first systematic evaluations of implementations of abstract argumentation solvers have been conducted in [Bistarelli *et al.*, 2013; Bistarelli *et al.*, 2014]. In these evaluations a small number of implementations have been evaluated with respect to runtime on graphs generated by different graph models from social networking theory such as the Barabási-Albert model (see above). A similar performance evaluation is provided in [Vallati *et al.*, 2014a; Cerutti *et al.*, 2016d]. In addition, in [Cerutti *et al.*, 2016c] the authors discuss the effect of solver and instances configuration on performance.

A large-scale and systematic comparison of different implementations of computational models of argumentation is offered by the *International Competition on Computational Models of Argumentation* (ICCMA)[59], which has already been men-

complete. For example the theory of "six degrees of separation" suggests that in the social network of the known world the longest shortest path between any two persons is six.

[57] `http://sourceforge.net/p/probo/code/HEAD/tree/trunk/src/net/sf/probo/generators/` (on 27/04/2017).

[58] `https://sourceforge.net/projects/afbenchgen/` (on 27/04/2017).

[59] `http://argumentationcompetition.org` (on 27/04/2017).

tioned before and is an international event established in 2014. The first instance of the competition took place in 2015 and focused on comparing implementations for various decision and enumeration problems in abstract argumentation.

The competition in 2015 received 18 solvers from research groups in Austria, China, Cyprus, Finland, France, Germany, Italy, Romania, and UK. It was conducted using the benchmark framework *probo* [Cerutti *et al.*, 2014f], which provides the possibility to run the instances on the individual solvers, verify the results, measure the runtime, and log the results accordingly. The software *probo* is written in Java and requires the implementation of a simple command line interface from the participating solvers.[60] All benchmark graphs—generated using proprietary generation algorithms, see previous section—were made available in two file formats. The *trivial graph format*[61] (TGF) is a simple representation of a directed graph which simply lists all appearing vertices and edges. The *Aspartix format* (APX) [Egly *et al.*, 2008] is an abstract argumentation-specific format which represents an argumentation framework as facts in a logic programming-like way. In order to verify the answers of solvers, the solutions for all instances were computed in advance using the *Tweety libraries for logical aspects of artificial intelligence and knowledge representation*[62] [Thimm, 2014]. Tweety contains naïve algorithms for all considered semantics that implement the formal definitions of all semantics in a straightforward manner and thus provides verified reference implementations for all considered problems. Besides serving as the benchmark framework for executing the competition, *probo* also contains several abstract classes and interfaces for solver specification that can be used by participants in order to easily comply with the solver interface specification.

The competition in 2015 evaluated the runtime performance of the solvers for four different semantics and four different computational tasks, yielding a total of 16 tracks. Among the best solvers throughout all tracks were CoQuiAAS, ArgSemSAT, and LabSATSolver (see also Section 2). For detailed performance comparisons and current competitions see the webpage of ICCMA.[63]

7 Discussion

In this paper we discussed (1) approaches for addressing reasoning problems in abstract argumentation frameworks; (2) approaches for handling structured argu-

[60] See `http://argumentationcompetition.org/2015/iccma15notes_v3.pdf` (on 27/04/2017) for the formal interface description.

[61] `http://en.wikipedia.org/wiki/Trivial_Graph_Format` (on 27/04/2017).

[62] `http://tweetyproject.org` (on 27/04/2017).

[63] `http://argumentationcompetition.org` (on 27/04/2017).

mentation frameworks; and (3) other approaches that might be relevant to the argumentation community although they do not belong to the previous two classes.

As per approaches for abstract argumentation frameworks, it is beyond doubt that currently the majority of proposals adopt a reduction-based approach (Section 2.1), thus relying on SAT-solvers, or CSP-solvers, or ASP-solvers. However, we have covered the few direct implementations as discussed in Section 2.2.

Coming to approaches for structured argumentation frameworks, we considered the four large families developed in some 20 years of studies, viz. (in alphabetical order) ABA, ASPIC+, Deductive argumentation, and DeLP. We also considered the case of Carneades, which is both a formal model of argument structure and evaluation, and a system implementing the model.

Then, we reviewed 34 implemented systems that provide a general purpose gateway to formal structures of argumentation. They can be systems for producing corpora that can be exploited by argument mining algorithms as well as system for supporting critical thinking by the means of formal models of argumentation.

This touches one of the main topic of discussion still open in the community, namely applying machine learning techniques for automatic argument elicitation from natural language text, or *argument mining*, see [Budzynska *et al.*, 2014; Wells, 2014]. This is a fast growing research field, but at the same time, it encompasses a large variety of topics, from mining legal arguments, to mining tweets, and it is unlikely to have a *one-size-fits-all* approach. At the same time, this is an extremely young research field and best practices did not yet emerge in the community.

While we did not devote space to argument mining techniques, we instead discussed what are the main challenges we envisage for implementation of formal argumentation, as well as what are sensible ways for comparing different implementations. In particular, we reviewed (Section 5) the few approaches for making systems applicable to real-world scenarios, and thus overcoming the NP-complexity barrier, namely parallelization and approximation techniques. Moreover, machine learning techniques might also play an important role in selecting the right solver for a specific problem. There are, indeed, some embryonic approaches for automatic algorithm selection on the basis of abstract argumentation frameworks features. However, most—if not all—of the reviewed approaches consider abstract argumentation frameworks only.

This leads us to the last element of discussion we touched in this paper (Section 6), namely how to compare different systems by the means of benchmarks and competitions. Although the community already made a move in the context of abstract argumentation, with the first edition of the International Competition of Computational Models of Argumentation, we still have a long way ahead for addressing questions related to structured argumentation. Comparative studies on different

formalisms, i. e. [Schulz and Caminada, 2015] and [Heyninck and Straßer, 2016], might shed some light on common grounds, thus allowing for a fair comparison.

Acknowledgements

This work has been funded by the Austrian Science Fund (FWF): P25521, I2854 and P30168, and by Academy of Finland under grants 251170 COIN and 284591.

References

[Albert and Barabási, 2002] Réka Albert and Albert-Laszló Barabási. Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74(1):47–97, 2002.

[Alsinet *et al.*, 2010] Teresa Alsinet, Ramón Béjar, and Lluis Godo. A characterization of collective conflict for defeasible argumentation. In Pietro Baroni, Federico Cerutti, Massimiliano Giacomin, and Guillermo R. Simari, editors, *Proceedings of the 3rd International Conference on Computational Models of Argument (COMMA 2010)*, volume 216 of *Frontiers in Artificial Intelligence and Applications*, pages 27–38. IOS Press, 2010.

[Alsinet *et al.*, 2011] Teresa Alsinet, Ramón Béjar, Lluis Godo, and Francesc Guitart. Maximal ideal recursive semantics for defeasible argumentation. In Salem Benferhat and John Grant, editors, *Proceedings of the 5th International Conference on Scalable Uncertainty Management (SUM 2011)*, volume 6929 of *Lecture Notes in Computer Science*, pages 96–109. Springer, 2011.

[Alsinet *et al.*, 2012] Teresa Alsinet, Ramón Béjar, Lluis Godo, and Francesc Guitart. Using answer set programming for an scalable implementation of defeasible argumentation. In *Proceedings of the IEEE 24th International Conference on Tools with Artificial Intelligence (ICTAI 2012)*, pages 1016–1021. IEEE, 2012.

[Alsinet *et al.*, 2013] Teresa Alsinet, Ramón Béjar, Lluis Godo, and Francesc Guitart. On the implementation of a multiple output algorithm for defeasible argumentation. In Weiru Liu, V. S. Subrahmanian, and Jef Wijsen, editors, *Proceedings of the 7th International Conference on Scalable Uncertainty Management (SUM 2013)*, volume 8078 of *Lecture Notes in Computer Science*, pages 71–77. Springer, 2013.

[Ambroz *et al.*, 2013] Thomas Ambroz, Günther Charwat, Andreas Jusits, Johannes P. Wallner, and Stefan Woltran. ARVis: Visualizing relations between answer sets. In Pedro Cabalar and Tran Cao Son, editors, *Proceedings of the 12th International Conference on Logic Programming and Nonmonotonic Reasoning, (LPNMR 2013)*, volume 8148 of *Lecture Notes in Artificial Intelligence*, pages 73–78. Springer, 2013.

[Amgoud and Devred, 2011] Leila Amgoud and Caroline Devred. Argumentation frameworks as constraint satisfaction problems. In Salem Benferhat and John Grant, editors, *Proceedings of the 5th International Conference on Scalable Uncertainty Management (SUM 2011)*, volume 6929 of *Lecture Notes in Computer Science*, pages 110–122. Springer, 2011.

[Amgoud, 2014] Leila Amgoud. Postulates for logic-based argumentation systems. *International Journal of Approximate Reasoning*, 55(9):2028–2048, 2014.

[Arieli and Caminada, 2013] Ofer Arieli and Martin W.A. Caminada. A QBF-based formalization of abstract argumentation semantics. *Journal of Applied Logic*, 11(2):229–252, 2013.

[Aurisicchio *et al.*, 2015] Marco Aurisicchio, Pietro Baroni, Dario Pellegrini, and Francesca Toni. Comparing and integrating argumentation-based with matrix-based decision support in Arg&Dec. In Elizabeth Black, Sanjay Modgil, and Nir Oren, editors, *Proceedings of the 3rd International Workshop on Theory and Applications of Formal Argumentation (TAFA 2015), Revised Selected Papers*, volume 9524 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2015.

[Barabasi and Albert, 1999] Albert-Laszlo Barabasi and Reka Albert. Emergence of scaling in random networks. *Science*, 286(5439):11, 1999.

[Baroni *et al.*, 2005] Pietro Baroni, Massimiliano Giacomin, and Giovanni Guida. SCC-recursiveness: a general schema for argumentation semantics. *Artificial Intelligence*, 168(1-2):165–210, 2005.

[Baroni *et al.*, 2011a] Pietro Baroni, Martin Caminada, and Massimiliano Giacomin. An introduction to argumentation semantics. *The Knowledge Engineering Review*, 26(4):365–410, 2011.

[Baroni *et al.*, 2011b] Pietro Baroni, Federico Cerutti, Massimiliano Giacomin, and Giovanni Guida. AFRA: argumentation framework with recursive attacks. *International Journal of Approximate Reasoning*, 52(1):19–37, 2011.

[Baroni *et al.*, 2014] Pietro Baroni, Guido Boella, Federico Cerutti, Massimiliano Giacomin, Leendert van der Torre, and Serena Villata. On the Input/Output behavior of argumentation frameworks. *Artificial Intelligence*, 217(0):144–197, 2014.

[Baumann and Strass, 2014] Ringo Baumann and Hannes Strass. On the Maximal and Average Numbers of Stable Extensions. In Elizabeth Black, Sanjay Modgil, and Nir Oren, editors, *Proceedings of the 2nd International Workshop on Theory and Applications of Formal Argumentation (TAFA 2013)*, volume 8306 of *Lecture Notes in Computer Science*, pages 111–126. Springer, 2014.

[Beierle *et al.*, 2015] Christoph Beierle, Florian Brons, and Nico Potyka. A software system using a SAT solver for reasoning under complete, stable, preferred, and grounded argumentation semantics. In Steffen Hölldobler, Markus Krötzsch, Rafael Peñaloza, and Sebastian Rudolph, editors, *Proceedings of the 38th Annual German Conference on AI (KI 2015)*, volume 9324 of *Lecture Notes in Computer Science*, pages 241–248. Springer, 2015.

[Bench-Capon, 2003] T J M Bench-Capon. Persuasion in Practical Argument Using Value Based Argumentation Frameworks. *Journal of Logic and Computation*, 13(3):429–448, 2003.

[Besnard and Doutre, 2004] Philippe Besnard and Sylvie Doutre. Checking the acceptability of a set of arguments. In James P. Delgrande and Torsten Schaub, editors, *Proceedings of the 10th International Workshop on Non-Monotonic Reasoning (NMR 2004)*, pages 59–

64, 2004.

[Besnard and Hunter, 2006] Philippe Besnard and Anthony Hunter. Knowledgebase compilation for efficient logical argumentation. In Patrick Doherty, John Mylopoulos, and Christopher A. Welty, editors, *Proceedings of the 10th International Conference on Principles of Knowledge Representation and Reasoning (KR 2006)*, pages 123–133. AAAI Press, 2006.

[Besnard and Hunter, 2008] Philippe Besnard and Anthony Hunter. *Elements of Argumentation*. MIT Press, 2008.

[Besnard and Hunter, 2014] Philippe Besnard and Anthony Hunter. Constructing argument graphs with deductive arguments: a tutorial. *Argument & Computation*, 5(1):5–30, 2014.

[Besnard *et al.*, 2009] Philippe Besnard, Anthony Hunter, and Stefan Woltran. Encoding deductive argumentation in quantified Boolean formulae. *Artificial Intelligence*, 173(15):1406–1423, 2009.

[Besnard *et al.*, 2010] Philippe Besnard, Éric Grégoire, Cédric Piette, and Badran Raddaoui. MUS-based generation of arguments and counter-arguments. In Reda Alhajj, James B. D. Joshi, and Mei-Ling Shyu, editors, *Proceedings of the IEEE International Conference on Information Reuse and Integration, (IRI 2010)*, pages 239–244. IEEE Systems, Man, and Cybernetics Society, 2010.

[Besnard *et al.*, 2014] Philippe Besnard, Alejandro Javier García, Anthony Hunter, Sanjay Modgil, Henry Prakken, Guillermo R. Simari, and Francesca Toni. Introduction to structured argumentation. *Argument & Computation*, 5(1):1–4, 2014.

[Bex and Reed, 2012] Floris Bex and Chris Reed. Dialogue Templates for Automatic Argument Processing. In Bart Verheij, Stefan Szeider, and Stefan Woltran, editors, *Proceedings of the 4th International Conference on Computational Models of Argument (COMMA 2012)*, volume 245 of *Frontiers in Artificial Intelligence and Applications*, pages 366–377. IOS Press, 2012.

[Bex *et al.*, 2013] Floris Bex, John Lawrence, Mark Snaith, and Chris Reed. Implementing the argument web. *Communications of the ACM*, 56(10):66, 2013.

[Bex *et al.*, 2014] Floris Bex, Mark Snaith, John Lawrence, and Chris Reed. Argublogging: An application for the argument web. *Journal of Web Semantics*, 25:9–15, 2014.

[Biere *et al.*, 2009] Armin Biere, Marijn Heule, Hans van Maaren, and Toby Walsh, editors. *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*. IOS Press, 2009.

[Bistarelli and Santini, 2010] Stefano Bistarelli and Francesco Santini. A common computational framework for semiring-based argumentation systems. In Helder Coelho, Rudi Studer, and Michael Wooldridge, editors, *Proceedings of the 19th European Conference on Artificial Intelligence (ECAI 2010)*, volume 215 of *Frontiers in Artificial Intelligence and Applications*, pages 131–136. IOS Press, 2010.

[Bistarelli and Santini, 2011] Stefano Bistarelli and Francesco Santini. ConArg: A constraint-based computational framework for argumentation systems. In Taghi M. Khoshgoftaar and Xingquan (Hill) Zhu, editors, *Proceedings of the 23rd IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2011)*, pages 605–612.

IEEE Computer Society Press, 2011.

[Bistarelli and Santini, 2012a] Stefano Bistarelli and Francesco Santini. ConArg: a tool to solve (weighted) abstract argumentation frameworks with (soft) constraints. *CoRR*, abs/1212.2857, 2012.

[Bistarelli and Santini, 2012b] Stefano Bistarelli and Francesco Santini. Modeling and solving AFs with a constraint-based tool: ConArg. In Sanjay Modgil, Nir Oren, and Francesca Toni, editors, *Proceedings of the 1st International Workshop on Theory and Applications of Formal Argumentation (TAFA 2011)*, volume 7132 of *Lecture Notes in Computer Science*, pages 99–116. Springer, 2012.

[Bistarelli *et al.*, 2009] Stefano Bistarelli, Daniele Pirolandi, and Francesco Santini. Solving weighted argumentation frameworks with soft constraints. In Javier Larrosa and Barry O'Sullivan, editors, *Proceedings of the 14th Annual ERCIM International Workshop on Constraint Solving and Constraint Logic Programming (CSCLP 2009), Revised Selected Papers*, volume 6384 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 2009.

[Bistarelli *et al.*, 2013] Stefano Bistarelli, Fabio Rossi, and Francesco Santini. A first comparison of abstract argumentation systems: A computational perspective. In Domenico Cantone and Marianna Nicolosi Asmundo, editors, *Proceedings of the 28th Italian Conference on Computational Logic*, volume 1068 of *CEUR Workshop Proceedings*, pages 241–245. CEUR-WS.org, 2013.

[Bistarelli *et al.*, 2014] Stefano Bistarelli, Fabio Rossi, and Francesco Santini. Benchmarking hard problems in random abstract AFs: The stable semantics. In Simon Parsons, Nir Oren, Chris Reed, and Federico Cerutti, editors, *Proceedings of the 5th International Conference on Computational Models of Argument (COMMA 2014)*, volume 266 of *Frontiers in Artificial Intelligence and Applications*. IOS Press, 2014.

[Bistarelli *et al.*, 2015] Stefano Bistarelli, Fabio Rossi, and Francesco Santini. A comparative test on the enumeration of extensions in abstract argumentation. *Fundamenta Informaticae*, 140(3-4):263–278, 2015.

[Bliem *et al.*, 2012] Bernhard Bliem, Michael Morak, and Stefan Woltran. D-FLAT: Declarative problem solving using tree decompositions and answer-set programming. *TPLP*, 12(4-5):445–464, 2012.

[Bliem, 2012] Bernhard Bliem. Decompose, Guess & Check - Declarative problem solving on tree decompositions. Master's thesis, Vienna University of Technology, 2012. `http://permalink.obvsg.at/AC07814499`.

[Bondarenko *et al.*, 1997] Andrei Bondarenko, Phan Minh Dung, Robert A. Kowalski, and Francesca Toni. An abstract, argumentation-theoretic approach to default reasoning. *Artificial Intelligence*, 93:63–101, 1997.

[Brachman and Levesque, 2004] Ronald J. Brachman and Hector J. Levesque. *Knowledge Representation and Reasoning*. The Morgan Kaufmann Series in Artificial Intelligence. Morgan Kaufmann Publishers, 2004.

[Brewka and Woltran, 2014] Gerhard Brewka and Stefan Woltran. GRAPPA: A semantical framework for graph-based argument processing. In Torsten Schaub, Gerhard Friedrich, and Barry O'Sullivan, editors, *Proceedings of the 21st European Conference on Artifi-*

cial Intelligence (ECAI 2014) - Including Prestigious Applications of Intelligent Systems (PAIS 2014), volume 263 of *Frontiers in Artificial Intelligence and Applications*, pages 153–158. IOS Press, 2014.

[Brewka *et al.*, 2011] Gerhard Brewka, Thomas Eiter, and Mirosław Truszczyński. Answer set programming at a glance. *Commun. ACM*, 54(12):92–103, 2011.

[Brewka *et al.*, 2013] Gerhard Brewka, Stefan Ellmauthaler, Hannes Strass, Johannes Peter Wallner, and Stefan Woltran. Abstract dialectical frameworks revisited. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence IJCAI 2013*, pages 803–809. AAAI Press, 2013.

[Brochenin *et al.*, 2015] Remi Brochenin, Thomas Linsbichler, Marco Maratea, Johannes Peter Wallner, and Stefan Woltran. Abstract solvers for dung's argumentation frameworks. In Elizabeth Black, Sanjay Modgil, and Nir Oren, editors, *Proceedings of the 3rd International Workshop on Theory and Applications of Formal Argument (TAFA 2015)*, volume 9524 of *Lecture Notes in Computer Science*. Springer, 2015.

[Bryant and Krause, 2008] Daniel Bryant and Paul J. Krause. A review of current defeasible reasoning implementations. *The Knowledge Engineering Review*, 23(3):227–260, 2008.

[Bryant *et al.*, 2006] Daniel Bryant, Paul J. Krause, and Gerard Vreeswijk. Argue tuProlog: A lightweight argumentation engine for agent applications. In Paul E. Dunne and Trevor J. M. Bench-Capon, editors, *Proceedings of the 1st International Conference on Computational Models of Argument (COMMA 2006)*, volume 144 of *Frontiers in Artificial Intelligence and Applications*, pages 27–32. IOS Press, 2006.

[Budzynska *et al.*, 2014] Katarzyna Budzynska, Mathilde Janier, Juyeon Kang, Chris Reed, Patrick Saint-Dizier, Manfred Stede, and Olena Yaskorska. Towards Argument Mining from Dialogue. In Simon Parsons, Nir Oren, Chris Reed, and Federico Cerutti, editors, *Proceedings of the 5th International Conference on Computational Models of Argument (COMMA 2014)*, volume 266 of *Frontiers in Artificial Intelligence and Applications*, pages 185–196. IOS Press, 2014.

[Cabrio and Villata, 2014a] Elena Cabrio and Serena Villata. Node: A benchmark of natural language arguments. In Simon Parsons, Nir Oren, Chris Reed, and Federico Cerutti, editors, *Proceedings of the 5th International Conference on Computational Models of Argument (COMMA 2014)*, volume 266 of *Frontiers in Artificial Intelligence and Applications*, pages 449–450. IOS Press, 2014.

[Cabrio and Villata, 2014b] Elena Cabrio and Serena Villata. Towards a benchmark of natural language arguments. In Sébastien Konieczny and Hans Tompits, editors, *Proceedings of the 15th International Workshop on Non-Monotonic Reasoning (NMR 2014)*, 2014.

[Cabrio *et al.*, 2013] Elena Cabrio, Serena Villata, and Fabien Gandon. A support framework for argumentative discussions management in the web. In Philipp Cimiano, Óscar Corcho, Valentina Presutti, Laura Hollink, and Sebastian Rudolph, editors, *Proceedings of the 10th Extended Semantic Web Conference (ESWC 2013)*, volume 7882 of *Lecture Notes in Computer Science*. Springer, 2013.

[Caminada and Amgoud, 2005] Martin Caminada and Leila Amgoud. An Axiomatic Account of Formal Argumentation. In Manuela M. Veloso and Subbarao Kambhampati,

editors, *Proceedings of the 20th National Conference on Artificial Intelligence and the 17th Innovative Applications of Artificial Intelligence Conference (AAAI 2005)*, pages 608–613. AAAI Press / The MIT Press, 2005.

[Caminada and Amgoud, 2007] Martin Caminada and Leila Amgoud. On the evaluation of argumentation formalisms. *Artificial Intelligence*, 171(5-6):286–310, 2007.

[Caminada, 2007] Martin Caminada. An algorithm for computing semi-stable semantics. In Khaled Mellouli, editor, *Proceedings of the 9th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU 2007)*, volume 4724 of *Lecture Notes in Computer Science*, pages 222–234. Springer, 2007.

[Caminada, 2010] Martin Caminada. An algorithm for stage semantics. In Pietro Baroni, Federico Cerutti, Massimiliano Giacomin, and Guillermo R. Simari, editors, *Proceedings of the 3rd International Conference on Computational Models of Argument (COMMA 2010)*, volume 216 of *Frontiers in Artificial Intelligence and Applications*, pages 147–158. IOS Press, 2010.

[Capobianco *et al.*, 2004] Marcela Capobianco, Carlos Iván Chesñevar, and Guillermo R. Simari. An argument-based framework to model an agent's beliefs in a dynamic environment. In Iyad Rahwan, Pavlos Moraitis, and Chris Reed, editors, *Proceedings of the 1st International Workshop on Argumentation in Multi-Agent Systems (ArgMAS 2004), Revised Selected and Invited Papers*, volume 3366 of *Lecture Notes in Computer Science*, pages 95–110. Springer, 2004.

[Cartwright and Atikinson, 2009] Dan Cartwright and Katie Atikinson. Using Computational Argumentation to Support E-participation. *IEEE Intelligent Systems*, pages 42–52, 2009.

[Cartwright and Atkinson, 2008] Dan Cartwright and Katie Atkinson. Political Engagement Through Tools for Argumentation. In Philippe Besnard, Sylvie Doutre, and Anthony Hunter, editors, *Proceedings of the 2nd International Conference on Computational Models of Argument (COMMA 2008)*, volume 172 of *Frontiers in Artificial Intelligence and Applications*, pages 116–127. IOS Press, 2008.

[Cartwright *et al.*, 2009] Dan Cartwright, Katie Atkinson, and Trevor Bench-Capon. Supporting argument in e-democracy. In Alexander Prosser and Peter Parycek, editors, *Proceedings of the 3rd Conference on Electronic Democracy (EDEM 2009)*, pages 151–160, 2009.

[Cayrol *et al.*, 2014] Claudette Cayrol, Sylvie Doutre, and Marie-Christine Lagasquie-Schiex. GRAFIX: a Tool for Abstract Argumentation. In Simon Parsons, Nir Oren, Chris Reed, and Federico Cerutti, editors, *Proceedings of the 5th International Conference on Computational Models of Argument (COMMA 2014)*, volume 266 of *Frontiers in Artificial Intelligence and Applications*, pages 453–454. IOS Press, 2014.

[Cecchi *et al.*, 2006] Laura A. Cecchi, Pablo R. Fillottrani, and Guillermo R. Simari. On the complexity of DeLP through game semantics. In Jürgen Dix and Anthony Hunter, editors, *Proceedings of the 11th International Workshop on Nonmonotonic Reasoning (NMR 2006)*, pages 390–398, 2006.

[Cerutti *et al.*, 2014a] Federico Cerutti, Paul E. Dunne, Massimiliano Giacomin, and Mauro

Vallati. Computing preferred extensions in abstract argumentation: a SAT-based approach. In Elizabeth Black, Sanjay Modgil, and Nir Oren, editors, *Proceedings of the 2nd International Workshop on Theory and Applications of Formal Argumentation, Revised Selected papers (TAFA 2013)*, volume 8306 of *Lecture Notes in Computer Science*, pages 176–193. Springer, 2014.

[Cerutti *et al.*, 2014b] Federico Cerutti, Massimiliano Giacomin, and Mauro Vallati. Algorithm Selection for Preferred Extensions Enumeration. In Simon Parsons, Nir Oren, Chris Reed, and Federico Cerutti, editors, *Proceedings of the 5th International Conference on Computational Models of Argument (COMMA 2014)*, volume 266 of *Frontiers in Artificial Intelligence and Applications*, pages 221–232. IOS Press, 2014.

[Cerutti *et al.*, 2014c] Federico Cerutti, Massimiliano Giacomin, and Mauro Vallati. ArgSemSAT: solving argumentation problems using SAT. In Simon Parsons, Nir Oren, Chris Reed, and Federico Cerutti, editors, *Proceedings of the 5th International Conference on Computational Models of Argument (COMMA 2014)*, volume 266 of *Frontiers in Artificial Intelligence and Applications*, pages 455–456. IOS Press, 2014.

[Cerutti *et al.*, 2014d] Federico Cerutti, Massimiliano Giacomin, and Mauro Vallati. Generating challenging benchmark AFs. In Simon Parsons, Nir Oren, Chris Reed, and Federico Cerutti, editors, *Proceedings of the 5th International Conference on Computational Models of Argument (COMMA 2014)*, volume 266 of *Frontiers in Artificial Intelligence and Applications*. IOS Press, 2014.

[Cerutti *et al.*, 2014e] Federico Cerutti, Massimiliano Giacomin, Mauro Vallati, and Marina Zanella. A SCC Recursive Meta-Algorithm for Computing Preferred Labellings in Abstract Argumentation. In Chitta Baral and Giuseppe De Giacomo, editors, *Proceedings of the 14th International Conference on Principles of Knowledge Representation and Reasoning (KR 2014)*, pages 42–51, 2014.

[Cerutti *et al.*, 2014f] Federico Cerutti, Nir Oren, Hannes Strass, Matthias Thimm, and Mauro Vallati. A benchmark framework for a computational argumentation competition (demo paper). In Simon Parsons, Nir Oren, Chris Reed, and Federico Cerutti, editors, *Proceedings of the 5th International Conference on Computational Models of Argumentation (COMMA 2014)*, volume 266 of *Frontiers in Artificial Intelligence and Applications*. IOS Press, 2014.

[Cerutti *et al.*, 2015] Federico Cerutti, Ilias Tachmazidis, Sotirios Batsakis, Mauro Vallati, Massimiliano Giacomin, and Grigoris Antoniou. Exploiting Parallelism for Hard Problems in Abstract Argumentation. In Blai Bonet and Sven Koenig, editors, *Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI 2015)*, pages 1475–1481. AAAI Press, 2015.

[Cerutti *et al.*, 2016a] Federico Cerutti, Mauro Vallati, and Massimiliano Giacomin. Generating Structured Argumentation Frameworks: AFBenchGen2. In Pietro Baroni, Thomas F. Gordon, Tatjana Scheffler, and Manfred Stede, editors, *Proceedings of the 6th International Conference on Computational Models of Argument (COMMA 2016)*, pages 467–468, 2016.

[Cerutti *et al.*, 2016b] Federico Cerutti, Mauro Vallati, and Massimiliano Giacomin.

jArgSemSAT: an efficient off-the-shelf solver for abstract argumentation frameworks. In James P. Delgrande and Frank Wolter, editors, *15th International Conference on Principles of Knowledge Representation and Reasoning (KR2016)*, pages 541–544, 2016.

[Cerutti *et al.*, 2016c] Federico Cerutti, Mauro Vallati, and Massimiliano Giacomin. On the Effectiveness of Automated Configuration in Abstract Argumentation Reasoning. In Pietro Baroni, Thomas F. Gordon, Tatjana Scheffler, and Manfred Stede, editors, *Proceedings of the 6th International Conference on Computational Models of Argument (COMMA 2016)*, pages 199–206. IOS Press, 2016.

[Cerutti *et al.*, 2016d] Federico Cerutti, Mauro Vallati, and Massimiliano Giacomin. Where Are We Now? State of the Art and Future Trends of Solvers for Hard Argumentation Problems. In Pietro Baroni, Thomas F. Gordon, Tatjana Scheffler, and Manfred Stede, editors, *Proceedings of the 6th International Conference on Computational Models of Argument (COMMA 2016)*, pages 207–218. IOS Press, 2016.

[Cerutti *et al.*, 2017] Federico Cerutti, Mauro Vallati, and Massimiliano Giacomin. An Efficient Java-Based Solver for Abstract Argumentation Frameworks: jArgSemSAT. *International Journal on Artificial Intelligence Tools*, 26(02):1750002–1–26, 2017.

[Charwat *et al.*, 2012] Günther Charwat, Johannes P. Wallner, and Stefan Woltran. Utilizing ASP for Generating and Visualizing Argumentation Frameworks. In Michael Fink and Yuliya Lierler, editors, *Proceedings of the 5th Workshop on Answer Set Programming and Other Computing Paradigms (ASPOCP 2012)*, pages 51–65, 2012.

[Charwat *et al.*, 2015] Günther Charwat, Wolfgang Dvořák, Sarah Alice Gaggl, Johannes Peter Wallner, and Stefan Woltran. Methods for solving reasoning problems in abstract argumentation - A survey. *Artificial Intelligence*, 220:28–63, 2015.

[Charwat, 2012] Günther Charwat. Tree-decomposition based algorithms for abstract argumentation frameworks. Master's thesis, Vienna University of Technology, 2012. `http://permalink.obvsg.at/AC07812654`.

[Chesñevar and Simari, 2007] Carlos Iván Chesñevar and Guillermo R. Simari. A lattice-based approach to computing warranted beliefs in skeptical argumentation frameworks. In Manuela M. Veloso, editor, *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007)*, pages 280–285, 2007.

[Chesñevar *et al.*, 2000] Carlos Iván Chesñevar, Guillermo R. Simari, and Alejandro Javier García. Pruning search space in defeasible argumentation. In *Proceedings of the Workshop on Advances and Trends in AI. International Conferences of the Chilean Computer Science Society*, pages 46–55, 2000.

[Chesñevar *et al.*, 2006] Carlos Iván Chesñevar, Jarred McGinnis, Sanjay Modgil, Iyad Rahwan, Chris Reed, Guillermo R. Simari, Matthew South, Gerard A. W. Vreeswijk, and Steven Willmot. Towards an argument interchange format. *The Knowledge Engineering Review*, 21(04):293, 2006.

[Chklovski *et al.*, 2003] Timothy Chklovski, Yolanda Gil, Varun Ratnakar, and John Lee. Trellis: Supporting decision making via argumentation in the semantic web. In Katia Sycara and John Mylopoulos, editors, *Proceedings of the 2nd International Semantic Web Conference (ISWC 2003)*. Citeseer, 2003.

[Cormen *et al.*, 2009] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. MIT Press, 2009.

[Courcelle, 1989] B Courcelle. The monadic second-order logic of graphs, II: infinite graphs of bounded width. *Math. Systems Theory*, 21:187–221, 1989.

[Craven and Toni, 2016] Robert Craven and Francesca Toni. Argument graphs and assumption-based argumentation. *Artificial Intelligence*, 233:1–59, 2016.

[Craven *et al.*, 2012] Robert Craven, Francesca Toni, Cristian Cadar, Adrian Hadad, and Matthew Williams. Efficient argumentation for medical decision-making. In Gerhard Brewka, Thomas Eiter, and Sheila A. McIlraith, editors, *Proceedings of the 13th International Conference on Principles of Knowledge Representation and Reasoning (KR 2012)*. AAAI Press, 2012.

[Craven *et al.*, 2013] Robert Craven, Francesca Toni, and Matthew Williams. Graph-based dispute derivations in assumption-based argumentation. In Elizabeth Black, Sanjay Modgil, and Nir Oren, editors, *Proceedings of the 2nd International Workshop on Theory and Applications of Formal Argumentation (TAFA 2013), Revised Selected papers*, volume 8306 of *Lecture Notes in Computer Science*, pages 46–62. Springer, 2013.

[Creignou *et al.*, 2011] Nadia Creignou, Johannes Schmidt, Michael Thomas, and Stefan Woltran. Complexity of logic-based argumentation in post's framework. *Argument & Computation*, 2(2-3):107–129, 2011.

[Cyras and Toni, 2016a] Kristijonas Cyras and Francesca Toni. ABA+: assumption-based argumentation with preferences. In James P. Delgrande and Frank Wolter, editors, *15th International Conference on Principles of Knowledge Representation and Reasoning (KR2016)*, pages 553–556. AAAI Press, 2016.

[Cyras and Toni, 2016b] Kristijonas Cyras and Francesca Toni. ABA+: assumption-based argumentation with preferences. *CoRR*, abs/1610.03024, 2016.

[Darwiche and Marquis, 2002] Adnan Darwiche and Pierre Marquis. A knowledge compilation map. *Journal of Artificial Intelligence Research*, 17:229–264, 2002.

[Dauphin and Schulz, 2014] Jeremie Dauphin and Claudia Schulz. Arg Teach - A Learning Tool for Argumentation Theory. In *Proceedings of the 26th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2014)*, pages 776–783. IEEE Computer Society, 2014.

[De Liddo and Buckingham Shum, 2010] Anna De Liddo and Simon Buckingham Shum. Cohere: A prototype for contested collective intelligence. In Anna De Liddo, Simon Buckingham Shum, Gregorio Convertino, Ágnes Sándor, and Mark Klein, editors, *ACM Computer Supported Cooperative Work (CSCW 2010) - Workshop: Collective Intelligence In Organizations - Toward a Research Agenda*, 2010.

[Diller *et al.*, 2015] Martin Diller, Johannes Peter Wallner, and Stefan Woltran. Reasoning in abstract dialectical frameworks using quantified boolean formulas. *Argument & Computation*, 6(2):149–177, 2015.

[Doutre and Mengin, 2001] Sylvie Doutre and Jérôme Mengin. Preferred extensions of argumentation frameworks: Query answering and computation. In Rajeev Goré, Alexander Leitsch, and Tobias Nipkow, editors, *Proceedings of the 1st International Joint Confer-*

ence on Automated Reasoning (IJCAR 2001), volume 2083 of *Lecture Notes in Computer Science*, pages 272–288. Springer, 2001.

[Dung *et al.*, 2006] Phan Minh Dung, Robert A. Kowalski, and Francesca Toni. Dialectic proof procedures for assumption-based, admissible argumentation. *Artificial Intelligence*, 170(2):114–159, 2006.

[Dung *et al.*, 2007] Phan Minh Dung, Paolo Mancarella, and Francesca Toni. Computing ideal sceptical argumentation. *Artificial Intelligence*, 171(10-15):642–674, 2007.

[Dung, 1995] 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):321–358, 1995.

[Dunne and Bench-Capon, 2002] P E Dunne and T J M Bench-Capon. Coherence in finite argument systems. *Artificial Intelligence*, 141:187–203, 2002.

[Dunne and Bench-Capon, 2003] Paul E Dunne and T J M Bench-Capon. Two party immediate response disputes: properties and efficiency. *Artificial Intelligence*, 149(2):221–250, 2003.

[Dunne and Wooldridge, 2009] Paul E. Dunne and Michael Wooldridge. Complexity of abstract argumentation. In Iyad Rahwan and Guillermo R. Simari, editors, *Argumentation in AI*, chapter 5, pages 85–104. Springer-Verlag, 2009.

[Dunne *et al.*, 2009] P E Dunne, A Hunter, P McBurney, S Parsons, and M Wooldridge. Inconsistency tolerance in weighted argument systems. In *AAMAS '09: Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems*, pages 851–858, Richland, SC, 2009. International Foundation for Autonomous Agents and Multiagent Systems.

[Dunne, 2007] Paul E. Dunne. Computational properties of argument systems satisfying graph-theoretic constraints. *Artificial Intelligence*, 171(10-15):701–729, 2007.

[Dvořák *et al.*, 2010] Wolfgang Dvořák, Stefan Szeider, and Stefan Woltran. Reasoning in argumentation frameworks of bounded clique-width. In Pietro Baroni, Federico Cerutti, Massimiliano Giacomin, and Guillermo R. Simari, editors, *Proceedings of the 3rd Conference on Computational Models of Argument (COMMA 2010)*, Frontiers in Artificial Intelligence and Applications, pages 219–230. IOS Press, 2010.

[Dvořák *et al.*, 2012a] Wolfgang Dvořák, Sebastian Ordyniak, and Stefan Szeider. Augmenting tractable fragments of abstract argumentation. *Artificial Intelligence*, 186:157–173, 2012.

[Dvořák *et al.*, 2012b] Wolfgang Dvořák, Reinhard Pichler, and Stefan Woltran. Towards fixed-parameter tractable algorithms for abstract argumentation. *Artificial Intelligence*, 186:1–37, 2012.

[Dvořák *et al.*, 2012c] Wolfgang Dvořák, Stefan Szeider, and Stefan Woltran. Abstract argumentation via monadic second order logic. In Eyke Hüllermeier, Sebastian Link, Thomas Fober, and Bernhard Seeger, editors, *Proceedings of the 6th International Conference on Scalable Uncertainty Management (SUM 2012)*, volume 7520 of *Lecture Notes in Computer Science*, pages 85–98. Springer, 2012.

[Dvořák *et al.*, 2013a] Wolfgang Dvořák, Sarah A. Gaggl, Johannes P. Wallner, and Stefan Woltran. Making use of advances in answer-set programming for abstract argumentation systems. In Hans Tompits, Salvador Abreu, Johannes Oetsch, Jörg Pührer, Dietmar Seipel, Masanobu Umeda, and Armin Wolf, editors, *Proceedings of the 19th International Conference on Applications of Declarative Programming and Knowledge Management (INAP 2011), Revised Selected Papers*, volume 7773 of *Lecture Notes in Artificial Intelligence*, pages 114–133. Springer, 2013.

[Dvořák *et al.*, 2013b] Wolfgang Dvořák, Michael Morak, Clemens Nopp, and Stefan Woltran. dynPARTIX - A dynamic programming reasoner for abstract argumentation. In Hans Tompits, Salvador Abreu, Johannes Oetsch, Jörg Pührer, Dietmar Seipel, Masanobu Umeda, and Armin Wolf, editors, *Proceedings of the 19th International Conference on Applications of Declarative Programming and Knowledge Management (INAP 2011), Revised Selected Papers*, volume 7773 of *Lecture Notes in Artificial Intelligence*, pages 259–268. Springer, 2013.

[Dvořák *et al.*, 2014] Wolfgang Dvořák, Matti Järvisalo, Johannes Peter Wallner, and Stefan Woltran. Complexity-sensitive decision procedures for abstract argumentation. *Artificial Intelligence*, 206:53–78, 2014.

[Dvořák *et al.*, 2015] Wolfgang Dvořák, Sarah Alice Gaggl, Thomas Linsbichler, and Johannes Peter Wallner. Reduction-Based Approaches to Implement Modgil's Extended Argumentation Frameworks. In Thomas Eiter, Hannes Strass, Miroslaw Truszczynski, and Stefan Woltran, editors, *Advances in Knowledge Representation, Logic Programming, and Abstract Argumentation - Essays Dedicated to Gerhard Brewka on the Occasion of His 60th Birthday*, volume 9060 of *Lecture Notes in Computer Science*, pages 249–264. Springer, 2015.

[Efstathiou and Hunter, 2008] Vasiliki Efstathiou and Anthony Hunter. Algorithms for effective argumentation in classical propositional logic: A connection graph approach. In Sven Hartmann and Gabriele Kern-Isberner, editors, *Proceedings of the 5th International Symposium on Foundations of Information and Knowledge Systems (FoIKS 2008)*, volume 4932 of *Lecture Notes in Computer Science*, pages 272–290. Springer, 2008.

[Efstathiou and Hunter, 2011] Vasiliki Efstathiou and Anthony Hunter. Algorithms for generating arguments and counterarguments in propositional logic. *International Journal of Approximate Reasoning*, 52(6):672–704, 2011.

[Egly and Woltran, 2006] Uwe Egly and Stefan Woltran. Reasoning in argumentation frameworks using quantified boolean formulas. In Paul E. Dunne and Trevor J. M. Bench-Capon, editors, *Proceedings of the 1st International Conference on Computational Models of Argument (COMMA 2006)*, volume 144 of *Frontiers in Artificial Intelligence and Applications*, pages 133–144. IOS Press, 2006.

[Egly *et al.*, 2008] Uwe Egly, Sarah A. Gaggl, and Stefan Woltran. ASPARTIX: Implementing Argumentation Frameworks Using Answer-Set Programming. In Maria G. de la Banda and Enrico Pontelli, editors, *Proceedings of the 24th International Conference on Logic Programming (ICLP 2008)*, volume 5366 of *Lecture Notes in Computer Science*, pages 734–738. Springer, 2008.

[Egly *et al.*, 2010a] Uwe Egly, Sarah A. Gaggl, and Stefan Woltran. Answer-set programming encodings for argumentation frameworks. *Argument & Computation*, 1(2):147–177, 2010.

[Egly *et al.*, 2010b] Uwe Egly, Sarah Alice Gaggl, Paul Wandl, and Stefan Woltran. ASPARTIX Conquers the Web. Software demonstration at the 3rd International Conference on Computational Models of Argument (COMMA 2010), `http://comma2010.unibs.it/demos/Egly_etal.pdf`, 2010.

[Eiter and Gottlob, 1995] Thomas Eiter and Georg Gottlob. The Complexity of Logic-Based Abduction. *Journal of the ACM*, 42(1):3–42, 1995.

[Eiter *et al.*, 1997] Thomas Eiter, Georg Gottlob, and Heikki Mannila. Disjunctive datalog. *ACM Transactions on Database Systems*, 22(3):364–418, 1997.

[Ellmauthaler and Strass, 2014] Stefan Ellmauthaler and Hannes Strass. The DIAMOND system for computing with abstract dialectical frameworks. In Simon Parsons, Nir Oren, Chris Reed, and Federico Cerutti, editors, *Proceedings of the 5th International Conference on Computational Models of Argument (COMMA 2014)*, volume 266 of *Frontiers in Artificial Intelligence and Applications*, pages 233–240. IOS Press, 2014.

[Ennals *et al.*, 2010] Rob Ennals, Beth Trushkowsky, and John Mark Agosta. Highlighting disputed claims on the web. In Michael Rappa, Paul Jones, Juliana Freire, and Soumen Chakrabarti, editors, *Proceedings of the 19th International Conference on World Wide Web (WWW 2010)*, pages 341–350, New York, New York, USA, 2010. ACM Press.

[Erdös and Rényi, 1959] Paul Erdös and Alfréd Rényi. On random graphs I. *Publ. Math. Debrecen*, 6:290–297, 1959.

[Evripidou and Toni, 2014] Valentinos Evripidou and Francesca Toni. Quaestio-it.com: a social intelligent debating platform. *Journal of Decision Systems*, 23(3):333–349, 2014.

[Faber *et al.*, 2016] Wolfgang Faber, Mauro Vallati, Federico Cerutti, and Massimiliano Giacomin. Solving set optimization problems by cardinality optimization with an application to argumentation. In Gal A. Kaminka, Maria Fox, Paolo Bouquet, Eyke Hüllermeier, Virginia Dignum, Frank Dignum, and Frank van Harmelen, editors, *22nd European Conference on Artificial Intelligence (ECAI 2016) – Including Prestigious Applications of Artificial Intelligence (PAIS 2016)*, pages 966–973, 2016.

[Faridani *et al.*, 2010] Siamak Faridani, Ephrat Bitton, Kimiko Ryokai, and Ken Goldberg. Opinion space. In Elizabeth D. Mynatt, Don Schoner, Geraldine Fitzpatrick, Scott E. Hudson, W. Keith Edwards, and Tom Rodden, editors, *Proceedings of the 28th International Conference on Human Factors in Computing Systems (CHI 2010)*, pages 1175–1184, New York, New York, USA, 2010. ACM Press.

[Freeman and Farley, 1996] Kathleen Freeman and Arthur M. Farley. A model of argumentation and its application to legal reasoning. *Artificial Intelligence and Law*, 4(3-4):163–197, 1996.

[Gabbay and Rodrigues, 2015] Dov M. Gabbay and Odinaldo Rodrigues. Equilibrium states in numerical argumentation networks. *Logica Universalis*, 9(4):411–473, 2015.

[Gabbay and Rodriguez, 2014] Dov M. Gabbay and Odinaldo Rodriguez. A self-correcting iteration schema for argumentation networks. In Simon Parsons, Nir Oren, Chris Reed,

and Federico Cerutti, editors, *Proceedings of the 5th International Conference on Computational Models of Argument (COMMA 2014)*, volume 266 of *Frontiers in Artificial Intelligence and Applications*, pages 377–384. IOS Press, 2014.

[Gabbay, 2012] Dov M. Gabbay. Equational approach to argumentation networks. *Argument & Computation*, 3(2-3):87–142, 2012.

[Gaertner and Toni, 2007a] Dorian Gaertner and Francesca Toni. CaSAPI: a system for credulous and sceptical argumentation. In Guillermo R. Simari and Paolo Torroni, editors, *Workshop on Argumentation for Non-Monotonic Reasoning (NMR 2007)*, pages 80–95, 2007.

[Gaertner and Toni, 2007b] Dorian Gaertner and Francesca Toni. Computing arguments and attacks in assumption-based argumentation. *IEEE Intelligent Systems*, 22(6):24–33, 2007.

[Gaertner and Toni, 2008] Dorian Gaertner and Francesca Toni. Hybrid argumentation and its properties. In Philippe Besnard, Sylvie Doutre, and Anthony Hunter, editors, *Proceedings of the 2nd International Conference on Computational Models of Argument (COMMA 2008)*, volume 172 of *Frontiers in Artificial Intelligence and Applications*, pages 183–195. IOS Press, 2008.

[Gaggl and Manthey, 2015] Sarah Alice Gaggl and Norbert Manthey. ASPARTIX-D: ASP Argumentation Reasoning Tool – Dresden. In Matthias Thimm and Serena Villata, editors, *System Descriptions of the First International Competition on Computational Models of Argumentation (ICCMA'15)*, pages 29–32, 2015.

[Gaggl *et al.*, 2015] Sarah Alice Gaggl, Norbert Manthey, Alessandro Ronca, Johannes Peter Wallner, and Stefan Woltran. Improved answer-set programming encodings for abstract argumentation. *TPLP*, 15(4-5):434–448, 2015.

[García and Simari, 2000] Alejandro Javier García and Guillermo R. Simari. Parallel defeasible argumentation. *Journal of Computer Science & Technology*, 1(2):37–51, 2000.

[García and Simari, 2004] Alejandro Javier García and Guillermo R. Simari. Defeasible logic programming: An argumentative approach. *TPLP*, 4(1-2):95–138, 2004.

[García and Simari, 2014] Alejandro Javier García and Guillermo R. Simari. Defeasible logic programming: DeLP-servers, contextual queries, and explanations for answers. *Argument & Computation*, 5(1):63–88, 2014.

[García, 1997] Alejandro Javier García. Defeasible logic programming: Definition and implementation. Master's thesis, Universidad Nacional del Sur, Computer Science Department, Bahía Blanca, Argentina, 1997.

[Gelfond and Lifschitz, 1991] Michael Gelfond and Vladimir Lifschitz. Classical negation in logic programs and disjunctive databases. *New Generation Computing*, 9(3-4):365–385, 1991.

[Gordon and Walton, 2016] Thomas F. Gordon and Douglas Walton. Formalizing balancing arguments. In Pietro Baroni, Thomas F. Gordon, Tatjana Scheffler, and Manfred Stede, editors, *Proceedings of the 6th International Conference on Computational Models of Argument (COMMA 2016)*, volume 287 of *Frontiers in Artificial Intelligence and Applications*, pages 327–338. IOS Press, 2016. Revised version under `http:`

//www.tfgordon.de/publications/.

[Gordon *et al.*, 2007] Thomas F. Gordon, Henry Prakken, and Douglas N. Walton. The Carneades model of argument and burden of proof. *Artificial Intelligence*, 171(10-15):875–896, 2007.

[Gordon, 2012] Thomas F. Gordon. The Carneades web service. In Bart Verheij, Stefan Szeider, and Stefan Woltran, editors, *Proceedings of the 4th International Conference on Computational Models of Argument (COMMA 2012)*, volume 245 of *Frontiers in Artificial Intelligence and Applications*, pages 517–518. IOS Press, 2012.

[Gordon, 2013] Thomas F. Gordon. Introducing the Carneades web application. In Enrico Francesconi and Bart Verheij, editors, *Proceedings of the International Conference on Artificial Intelligence and Law (ICAIL 2013)*, pages 243–244. ACM, 2013.

[Gorogiannis and Hunter, 2011] Nikos Gorogiannis and Anthony Hunter. Instantiating abstract argumentation with classical logic arguments: Postulates and properties. *Artificial Intelligence*, 175:1479–1497, 2011.

[Grégoire *et al.*, 2009] Éric Grégoire, Bertrand Mazure, and Cédric Piette. Using local search to find msses and muses. *European Journal of Operational Research*, 199(3):640–646, 2009.

[Groza and Groza, 2015] Serban Groza and Adrian Groza. ProGraph: towards enacting bipartite graphs for abstract argumentation frameworks. In Matthias Thimm and Serena Villata, editors, *System Descriptions of the First International Competition on Computational Models of Argumentation (ICCMA'15)*, pages 29–32, 2015.

[Gürkan *et al.*, 2010] Ali Gürkan, Luca Iandoli, Mark Klein, and Giuseppe Zollo. Mediating debate through on-line large-scale argumentation: Evidence from the field. *Information Sciences*, 180(19):3686–3702, 2010.

[Hadjisoteriou, 2015] Evgenios Hadjisoteriou. Computing Argumentation with Matrices. In Claudia Schulz and Daniel Liew, editors, *2015 Imperial College Computing Student Workshop (ICCSW 2015)*, volume 49 of *OpenAccess Series in Informatics (OASIcs)*, pages 29–36, Dagstuhl, Germany, 2015. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[Hadoux *et al.*, 2015] Emmanuel Hadoux, Aurélie Beynier, Nicolas Maudet, Paul Weng, and Anthony Hunter. Optimization of probabilistic argumentation with markov decision models. In Qiang Yang and Michael Wooldridge, editors, *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI 2015)*, pages 2004–2010. AAAI Press, 2015.

[Hall, 1998] Mark A. Hall. *Correlation-based Feature Subset Selection for Machine Learning*. PhD thesis, University of Waikato, Hamilton, New Zealand, 1998.

[Heyninck and Straßer, 2016] Jesse Heyninck and Christian Straßer. Relations between assumption-based approaches in nonmonotonic logic and formal argumentation. In Gabriele Kern-Isberner and Renata Wassermann, editors, *Proceedings of the 16th International Workshop on Non-Monotonic Reasoning (NMR 2016)*, 2016.

[Hirsch and Gorogiannis, 2010] Robin Hirsch and Nikos Gorogiannis. The complexity of the warranted formula problem in propositional argumentation. *Journal of Logic and Computation*, 20(2):481–499, 2010.

[Hoffmann, 2005] Michael H. G. Hoffmann. Logical argument mapping: A method for overcoming cognitive problems of conflict management. *International Journal of Conflict Management*, 16(4):304–334, 2005.

[Hoffmann, 2007] Michael H. G. Hoffmann. Logical argument mapping. In Simon Buckingham Shum, Mikael Lind, and Hans Weigand, editors, *Proceedings of the 2nd International Conference on Pragmatic Web (ICPW 2007)*, volume 280 of *ACM International Conference Proceeding Series*, pages 41–47, New York, New York, USA, 2007. ACM.

[Hölldobler *et al.*, 2011] Steffen Hölldobler, Norbert Manthey, Van Hau Nguyen, Julian Stecklina, and Peter Steinke. A short overview on modern parallel SAT-solvers. In *Proceedings of the International Conference on Advanced Computer Science and Information Systems*, pages 201–206, 2011.

[Hunter and Williams, 2012] Anthony Hunter and Matthew Williams. Aggregating evidence about the positive and negative effects of treatments. *Artificial intelligence in medicine*, 56(3):173–90, 2012.

[Hunter, 2006a] Anthony Hunter. Approximate arguments for efficiency in logical argumentation. In Jürgen Dix and Anthony Hunter, editors, *Proceedings of the 11th International Workshop on Non-Monotonic Reasoning (NMR 2006)*, pages 383–388, 2006.

[Hunter, 2006b] Anthony Hunter. Contouring of knowledge for intelligent searching for arguments. In Gerhard Brewka, Silvia Coradeschi, Anna Perini, and Paolo Traverso, editors, *Proceedings of the 17th European Conference on Artificial Intelligence (ECAI 2006), Including Prestigious Applications of Intelligent Systems (PAIS 2006)*, volume 141 of *Frontiers in Artificial Intelligence and Applications*, pages 250–254. IOS Press, 2006.

[Hunter, 2014] Anthony Hunter. Probabilistic qualification of attack in abstract argumentation. *International Journal of Approximate Reasoning*, 55(2):607–638, 2014.

[Introne, 2009] Joshua E. Introne. Supporting group decisions by mediating deliberation to improve information pooling. In Stephanie Teasley, Erling Havn, Wolfgang Prinz, and Wayne Lutters, editors, *Proceedings of the ACM 2009 International Conference on Supporting Group Work*, pages 189–198. ACM, 2009.

[Janier *et al.*, 2014] Mathilde Janier, John Lawrence, and Chris Reed. OVA+: an Argument Analysis Interface. In Simon Parsons, Nir Oren, Chris Reed, and Federico Cerutti, editors, *Proceedings of the 5th International Conference on Computational Models of Argument (COMMA 2014)*, volume 266 of *Frontiers in Artificial Intelligence and Applications*, pages 463–464. IOS Press, 2014.

[Kakas and Moraitis, 2003] Antonis Kakas and P Moraitis. Argumentation based decision making for autonomous agents. In Jeffrey S. Rosenschein, Tuomas Sandholm, Michael Wooldridge, and Makoto Yokoo, editors, *Proceedings of the second international joint conference on Autonomous Agents and Multiagent Systems*, pages 883–890, 2003.

[Kakas *et al.*, 1994] Antonis C. Kakas, Paolo Mancarella, and Phan Minh Dung. The Acceptability Semantics for Logic Programs. In Pascal Van Hentenryck, editor, *Proceedings of the Eleventh International Conference on Logic Programming*, pages 504–519, Cambridge, MA, USA, 1994. MIT Press.

[Krauthoff *et al.*, 2016] Tobias Krauthoff, Michael Baurmann, Gregor Betz, and Martin

Mauve. Dialog-based online argumentation. In Pietro Baroni, Thomas F. Gordon, Tatjana Scheffler, and Manfred Stede, editors, *Proceedings of the 6th International Conference on Computational Models of Argument (COMMA 2016)*, volume 287 of *Frontiers in Artificial Intelligence and Applications*, pages 33–40. IOS Press, 2016.

[Kriplean *et al.*, 2011] Travis Kriplean, Jonathan T. Morgan, Deen Freelon, Alan Borning, and Lance Bennett. ConsiderIt: Improving structured public deliberation. In Desney S. Tan, Saleema Amershi, Bo Begole, Wendy A. Kellogg, and Manas Tungare, editors, *Proceedings of the International Conference on Human Factors in Computing Systems (CHI 2011), Extended Abstracts Volume*, pages 1831–1836. ACM, 2011.

[Lagniez *et al.*, 2015] Jean-Marie Lagniez, Emmanuel Lonca, and Jean-Guy Mailly. Coquiaas: A constraint-based quick abstract argumentation solver. In *Proceedings of the 27th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2015)*, pages 928–935. IEEE Computer Society, 2015.

[Lamatz, 2015] Nico Lamatz. LamatzSolver-v0.1: A grounded extension finder based on the Java-Collection-Framework. In Matthias Thimm and Serena Villata, editors, *System Descriptions of the First International Competition on Computational Models of Argumentation (ICCMA'15)*, pages 29–32, 2015.

[Lawrence *et al.*, 2012a] John Lawrence, Floris Bex, and Chris Reed. Dialogues on the Argument Web: Mixed Initiative Argumentation with Arvina. In Bart Verheij, Stefan Szeider, and Stefan Woltran, editors, *Proceedings of the 4th International Conference on Computational Models of Argument (COMMA 2012)*, volume 245 of *Frontiers in Artificial Intelligence and Applications*, pages 513–514. IOS Press, 2012.

[Lawrence *et al.*, 2012b] John Lawrence, Floris Bex, Chris Reed, and Mark Snaith. Aifdb: Infrastructure for the argument web. In Bart Verheij, Stefan Szeider, and Stefan Woltran, editors, *Proceedings of the 4th International Conference on Computational Models of Argument (COMMA 2012)*, volume 245 of *Frontiers in Artificial Intelligence and Applications*, pages 515–516. IOS Press, 2012.

[Lehtonen *et al.*, 2017] Tuomo Lehtonen, Johannes P. Wallner, and Matti Järvisalo. From structured to abstract argumentation: Assumption-based acceptance via AF reasoning. In Alessandro Antonucci, Laurence Cholvy, and Odile Papini, editors, *14th European Conference, ECSQARU 2017*, 2017. To appear.

[Leyton-Brown *et al.*, 2003] Kevin Leyton-Brown, Eugene Nudelman, Galen Andrew, Jim McFadden, and Yoav Shoham. A portfolio approach to algorithm selection. In Georg Gottlob and Toby Walsh, editors, *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI 2003)*, pages 1542–1542. Morgan Kaufmann, 2003.

[Li *et al.*, 2012] Hengfei Li, Nir Oren, and Timothy J. Norman. Probabilistic Argumentation Frameworks. In Sanjay Modgil, Nir Oren, and Francesca Toni, editors, *Proceedings of the 1st International Workshop on Theorie and Applications of Formal Argumentation (TAFA 2011), Revised Selected Papers*, volume 7132 of *Lecture Notes in Computer Science*. Springer, 2012.

[Liao *et al.*, 2013] Beishui Liao, Liyun Lei, and Jianhua Dai. Computing preferred labellings by exploiting sccs and most sceptically rejected arguments. In Elizabeth Black, Sanjay

Modgil, and Nir Oren, editors, *Proceedings of the 2nd International Workshop on Theory and Applications of Formal Argumentation (TAFA 2013), Revised Selected papers*, volume 8306 of *LNCS*, pages 194–208. Springer, 2013.

[Liffiton *et al.*, 2016] Mark H. Liffiton, Alessandro Previti, Ammar Malik, and Joao Marques-Silva. Fast, flexible MUS enumeration. *Constraints*, 21(2):223–250, 2016.

[Lowrance *et al.*, 2008] John Lowrance, Ian Harrison, Andres Rodriguez, Eric Yeh, Tom Boyce, Janet Murdock, Jerome Thomere, and Ken Murray. Template-based structured argumentation. In *Knowledge Cartography*, pages 307–333. Springer, 2008.

[Macintosh, 2009] Ann Macintosh. Moving Toward Intelligent Policy Development? *IEEE Intelligent Systems*, 24(5):79–82, 2009.

[McAreavey *et al.*, 2014] Kevin McAreavey, Weiru Liu, and Paul C. Miller. Computational approaches to finding and measuring inconsistency in arbitrary knowledge bases. *International Journal of Approximate Reasoning*, 55(8):1659–1693, 2014.

[Modgil and Caminada, 2009] Sanjay Modgil and Martin Caminada. Proof theories and algorithms for abstract argumentation frameworks. In Iyad Rahwan and Guillermo R. Simari, editors, *Argumentation in Artificial Intelligence*, pages 105–132. Springer, 2009.

[Modgil and Prakken, 2014] Sanjay Modgil and Henry Prakken. The $ASPIC^+$ framework for structured argumentation: a tutorial. *Argument & Computation*, 5(1):31–62, 2014.

[Modgil, 2009] Sanjay Modgil. Reasoning about preferences in argumentation frameworks. *Artificial Intelligence*, 173(9-10):901–934, 2009.

[Nieuwenhuis *et al.*, 2006] R. Nieuwenhuis, A. Oliveras, and C. Tinelli. Solving SAT and SAT modulo theories: From an abstract Davis-Putnam-Logemann-Loveland procedure to DPLL(T). *Journal of the ACM*, 53(6):937–977, 2006.

[Nieves *et al.*, 2008] Juan C. Nieves, Mauricio Osorio, and Ulises Cortés. Preferred extensions as stable models. *TPLP*, 8(4):527–543, 2008.

[Nofal *et al.*, 2012] Samer Nofal, Paul E. Dunne, and Katie Atkinson. On preferred extension enumeration in abstract argumentation. In Bart Verheij, Stefan Szeider, and Stefan Woltran, editors, *Proceedings of the 4th Conference on Computational Models of Argument (COMMA 2012)*, volume 245 of *Frontiers in Artificial Intelligence and Applications*, pages 205–216. IOS Press, 2012.

[Nofal *et al.*, 2014a] Samer Nofal, Katie Atkinson, and Paul E. Dunne. Algorithms for argumentation semantics: Labeling attacks as a generalization of labeling arguments. *Journal of Artificial Intelligence Research*, 49:635–668, 2014.

[Nofal *et al.*, 2014b] Samer Nofal, Katie Atkinson, and Paul E. Dunne. Algorithms for decision problems in argument systems under preferred semantics. *Artificial Intelligence*, 207:23–51, 2014.

[Nofal, 2013] Samer Nofal. *Algorithms for Argument Systems.* PhD thesis, University of Liverpool, 2013. `http://research-archive.liv.ac.uk/12173/`.

[Oomidou *et al.*, 2014] Yutaka Oomidou, Yuki Katsura, Hajime Sawamura, Jacques Riche, and Takeshi Hagiwara. Asynchronous Argumentation System PIRIKA for Anyone, Anytime, Anywhere, with the Balanced Semantics. In Simon Parsons, Nir Oren, Chris Reed,

and Federico Cerutti, editors, *Proceedings of the 5th International Conference on Computational Models of Argument (COMMA 2014)*, volume 266 of *Frontiers in Artificial Intelligence and Applications*, pages 471–472. IOS Press, 2014.

[Parsons *et al.*, 2003] Simon Parsons, Michael Wooldridge, and Leila Amgoud. Properties and complexity of some formal inter-agent dialogues. *Journal of Logic and Computation*, 13(3):347–376, 2003.

[Pfeiffer *et al.*, 2012] Joseph J. Pfeiffer, III, Timothy La Fond, Sebastián Moreno, and Jennifer Neville. Fast generation of large scale social networks with clustering. *CoRR*, abs/1202.4805, 2012.

[Podlaszewski *et al.*, 2011] Mikolaj Podlaszewski, Martin Caminada, and Gabriella Pigozzi. An implementation of basic argumentation components. In Liz Sonenberg, Peter Stone, Kagan Tumer, and Pinar Yolum, editors, *Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2011)*, pages 1307–1308. IFAAMAS, 2011.

[Prakken, 2006] Henry Prakken. Combining sceptical epistemic reasoning with credulous practical reasoning. In Paul E. Dunne and Trevor J. M. Bench-Capon, editors, *Proceedings of the 1st International Conference on Computational Models of Argument (COMMA 2006)*, volume 144 of *Frontiers in Artificial Intelligence and Applications*, pages 311–322. IOS Press, 2006.

[Prakken, 2010] Henry Prakken. An abstract framework for argumentation with structured arguments. *Argument & Computation*, 1(2):93–124, 2010.

[Pührer, 2017] Jörg Pührer. ArgueApply: A mobile app for argumentation. In Marcello Balduccini and Tomi Janhunen, editors, *Proceedings of the 14th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR 2017)*. Springer, 2017. To appear.

[Rahwan *et al.*, 2011] Iyad Rahwan, Bita Banihashemi, Chris Reed, Douglas Walton, and Sherief Abdallah. Representing and classifying arguments on the Semantic Web. *The Knowledge Engineering Review*, 26(04):487–511, 2011.

[Reed, 2013] Chris Reed. Argument Corpora. Technical report, University of Dundee, 2013.

[Rice, 1976] John R. Rice. The algorithm selection problem. *Advances in Computers*, 15:65–118, 1976.

[Rodrigues, 2016] Odinaldo Rodrigues. Introducing eqargsolver: An argumentation solver using equational semantics. In Matthias Thimm, Federico Cerutti, Hannes Strass, and Mauro Vallati, editors, *Proceedings of the First International Workshop on Systems and Algorithms for Formal Argumentation (SAFA) co-located with the 6th International Conference on Computational Models of Argument (COMMA 2016)*, volume 1672 of *CEUR Workshop Proceedings*, pages 22–33. CEUR-WS.org, 2016.

[Rossi *et al.*, 2006] Francesca Rossi, Peter van Beek, and Toby Walsh. *Handbook of Constraint Programming (Foundations of Artificial Intelligence)*. Elsevier Science Inc., New York, NY, USA, 2006.

[Rotstein *et al.*, 2011] Nicolás D. Rotstein, Sebastian Gottifredi, Alejandro Javier García, and Guillermo R. Simari. A heuristics-based pruning technique for argumentation trees.

In Salem Benferhat and John Grant, editors, *Proceedings of the 5th International Conference on Scalable Uncertainty Management (SUM 2011)*, volume 6929 of *Lecture Notes in Computer Science*, pages 177–190. Springer, 2011.

[Russell and Norvig, 2003] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Pearson Education, second edition, 2003.

[Schneider *et al.*, 2007] David C Schneider, Christian Voigt, and Gregor Betz. Argunet-a software tool for collaborative argumentation analysis and research. In *7th Workshop on Computational Models of Natural Argument (CMNA VII)*, 2007.

[Schneider *et al.*, 2013] Jodi Schneider, Tudor Groza, and Alexandre Passant. A review of argumentation for the social semantic web. *Semantic Web*, 4(2):159–218, 2013.

[Schulz and Caminada, 2015] Claudia Schulz and Martin W. A. Caminada. On the equivalence between assumption-based argumentation and logic programming. In Sarah Gaggl, Juan Carlos Nieves, and Hannes Strass, editors, *1st International Workshop on Argumentation and Logic Programming (ArgLP 2015)*, 2015.

[Shum, 2008] Simon Buckingham Shum. Cohere: Towards web 2.0 argumentation. In Philippe Besnard, Sylvie Doutre, and Anthony Hunter, editors, *Proceedings of the 2nd International Conference on Computational Models of Argument (COMMA 2008)*, volume 172 of *Frontiers in Artificial Intelligence and Applications*, pages 97–108. IOS Press, 2008.

[Simari, 2011] Guillermo R. Simari. A brief overview of research in argumentation systems. In Salem Benferhat and John Grant, editors, *Proceedings of the 5th International Conference on Scalable Uncertainty Management (SUM 2011)*, volume 6929 of *Lecture Notes in Computer Science*, pages 81–95. Springer, 2011.

[Snaith and Reed, 2012] Mark Snaith and Chris Reed. TOAST: online aspic$^+$ implementation. In Bart Verheij, Stefan Szeider, and Stefan Woltran, editors, *Proceedings of the 4th International Conference on Computational Models of Argument (COMMA 2012)*, volume 245 of *Frontiers in Artificial Intelligence and Applications*, pages 509–510. IOS Press, 2012.

[Snaith *et al.*, 2010] Mark Snaith, Joseph Devereux, John Lawrence, and Chris Reed. Pipelining argumentation technologies. In Pietro Baroni, Federico Cerutti, Massimiliano Giacomin, and Guillermo R. Simari, editors, *Proceedings of the 3rd International Conference on Computational Models of Argument (COMMA 2010)*, volume 216 of *Frontiers in Artificial Intelligence*. IOS Press, 2010.

[Spanoudakis *et al.*, 2016] Nikolaos I. Spanoudakis, Antonis C. Kakas, and Pavlos Moraitis. Gorgias-B: Argumentation in Practice. In Pietro Baroni, Thomas F. Gordon, Tatjana Scheffler, and Manfred Stede, editors, *Proceedings of the 6th International Conference on Computational Models of Argument (COMMA 2016)*, pages 477–478, 2016.

[Sprotte, 2015] Kilian Sprotte. ASGL: Argumentation Semantics in Gecode and Lisp. In Matthias Thimm and Serena Villata, editors, *System Descriptions of the First International Competition on Computational Models of Argumentation (ICCMA'15)*, pages 41–44, 2015.

[Stolzenburg *et al.*, 2003] Frieder Stolzenburg, Alejandro Javier García, Carlos Iván Chesñe-

var, and Guillermo R. Simari. Computing generalized specificity. *Journal of Applied Non-Classical Logics*, 13(1):87–113, 2003.

[Strass, 2014] Hannes Strass. Implementing instantiation of knowledge bases in argumentation frameworks. In Simon Parsons, Nir Oren, Chris Reed, and Federico Cerutti, editors, *Proceedings of the 5th International Conference on Computational Models of Argument (COMMA 2014)*, volume 266 of *Frontiers in Artificial Intelligence and Applications*, pages 475–476. IOS Press, 2014.

[Tabourier *et al.*, 2011] Lionel Tabourier, Camille Roth, and Jean-Philippe Cointet. Generating constrained random graphs using multiple edge switches. *Journal of Experimental Algorithmics*, 16:1–15, 2011.

[Tang *et al.*, 2012] Yuqing Tang, Kai Cai, Peter McBurney, Elizabeth Sklar, and Simon Parsons. Using argumentation to reason about trust and belief. *Journal of Logic and Computation*, 22(5):979–1018, 2012.

[Tang *et al.*, 2016] Yuqing Tang, Nir Oren, and Katia P. Sycara. Markov Argumentation Random Fields. In Dale Schuurmans and Michael P. Wellman, editors, *Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI 2016)*. AAAI Press, 2016.

[Tarjan, 1972] Robert E. Tarjan. Depth-first search and linear graph algorithms. *SIAM Journal on Computing*, 1(2):146–160, 1972.

[Thimm and García, 2010] Matthias Thimm and Alejandro Javier García. Classification and strategical issues of argumentation games on structured argumentation frameworks. In Wiebe van der Hoek, Gal A. Kaminka, Yves Lespérance, Michael Luck, and Sandip Sen, editors, *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2010)*, pages 1247–1254. IFAAMAS, 2010.

[Thimm and Kern-Isberner, 2008] Matthias Thimm and Gabriele Kern-Isberner. On the relationship of defeasible argumentation and answer set programming. In Philippe Besnard, Sylvie Doutre, and Anthony Hunter, editors, *Proceedings of the 2nd International Conference on Computational Models of Argument (COMMA 2008)*, volume 172 of *Frontiers in Artificial Intelligence and Applications*, pages 393–404. IOS Press, 2008.

[Thimm *et al.*, 2016] Matthias Thimm, Serena Villata, Federico Cerutti, Nir Oren, Hannes Strass, and Mauro Vallati. Summary report of the first international competition on computational models of argumentation. *AI Magazine*, 37(1):102, 2016.

[Thimm, 2012] Matthias Thimm. A Probabilistic Semantics for Abstract Argumentation. In Luc De Raedt, Christian Bessière, Didier Dubois, Patrick Doherty, Paolo Frasconi, Fredrik Heintz, and Peter J. F. Lucas, editors, *Proceedings of the 20th European Conference on Artificial Intelligence (ECAI 2012)*, volume 242 of *Frontiers in Artificial Intelligence and Applications*. IOS Press, 2012.

[Thimm, 2014] Matthias Thimm. Tweety: A comprehensive collection of java libraries for logical aspects of artificial intelligence and knowledge representation. In Chitta Baral, Giuseppe De Giacomo, and Thomas Eiter, editors, *Proceedings of the Fourteenth International Conference on Principles of Knowledge Representation and Reasoning (KR 2014)*. AAAI Press, 2014.

[Toni and Sergot, 2011] Francesca Toni and Marek Sergot. Argumentation and answer set

programming. In Marcello Balduccini and Tran C. Son, editors, *Logic Programming, Knowledge Representation, and Nonmonotonic Reasoning: Essays in Honor of Michael Gelfond*, volume 6565 of *Lecture Notes in Computer Science*, pages 164–180. Springer, 2011.

[Toni, 2013] Francesca Toni. A generalised framework for dispute derivations in assumption-based argumentation. *Artificial Intelligence*, 195:1–43, 2013.

[Toni, 2014] Francesca Toni. A tutorial on assumption-based argumentation. *Argument & Computation*, 5(1):89–117, 2014.

[Toniolo *et al.*, 2014] Alice Toniolo, Timothy Dropps, Robin W. Ouyang, John A. Allen, Timothy J. Norman, Nir Oren, Mani B. Srivastava, and Paul Sullivan. Argumentation-based collaborative intelligence analysis in CISpaces. In Simon Parsons, Nir Oren, Chris Reed, and Federico Cerutti, editors, *Proceedings of the 5th International Conference on Computational Models of Argument (COMMA 2014)*, volume 266 of *Frontiers in Artificial Intelligence and Applications*, pages 481–482. IOS Press, 2014.

[Toniolo *et al.*, 2015] Alice Toniolo, Timothy J. Norman, Anthony Etuk, Federico Cerutti, Robin Wentao Ouyang, Mani Srivastava, Nir Oren, Timothy Dropps, John A. Allen, and Paul Sullivan. Agent Support to Reasoning with Different Types of Evidence in Intelligence Analysis. In Rafael H. Bordini, Edith Elkind, Gerhard Weiss, and Pinar Yolum, editors, *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2015)*, pages 781–789, 2015.

[Tzagarakis *et al.*, 2009] Manolis Tzagarakis, Nikos Karousos, Nikos Karacapilidis, and Dora Nousia. Unleashing argumentation support systems on the Web: The case of CoPe_it! In *Proceedings of the Web Science (WebSci'09)*, 2009.

[Vallati *et al.*, 2014a] Mauro Vallati, Federico Cerutti, and Massimiliano Giacomin. Argumentation extensions enumeration as a constraint satisfaction problem: a performance overview. In Richard Booth, Giovanni Casini, Szymon Klarman, Gilles Richard, and Ivan José Varzinczak, editors, *Proceedings of the International Workshop on Defeasible and Ampliative Reasoning (DARe@ECAI 2014)*, volume 1212 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2014.

[Vallati *et al.*, 2014b] Mauro Vallati, Federico Cerutti, and Massimiliano Giacomin. Argumentation frameworks features: an initial study. In Torsten Schaub, Gerhard Friedrich, and Barry O'Sullivan, editors, *Proceedings of the 21st European Conference on Artificial Intelligence (ECAI 2014)*, volume 263 of *Frontiers in Artificial Intelligence and Applications*, pages 1117–1118. IOS Press, 2014.

[Vallati *et al.*, 2015] Mauro Vallati, Federico Cerutti, Wolfgang Faber, and Massimiliano Giacomin. prefMaxSAT: Exploiting MaxSAT for Enumerating Preferred Extensions. In Matthias Thimm and Serena Villata, editors, *System Descriptions of the First International Competition on Computational Models of Argumentation (ICCMA'15)*, pages 58–61, 2015.

[Vallati *et al.*, 2017] Mauro Vallati, Federico Cerutti, and Massimiliano Giacomin. On the combination of argumentation solvers into parallel portfolios. In *Proceedings of the 30th Australasian Joint Conference on Artificial Intelligence*, 2017. To appear.

[van Eemeren *et al.*, 2014] Frans H. van Eemeren, Bart Garssen, Erik C. W. Krabbe, Francisca A. Snoeck Henkemans, Bart Verheij, and Jean H. M. Wagemans. *Handbook of Argumentation Theory*. Springer Netherlands, 2014.

[van Gijzel and Nilsson, 2013] Bas van Gijzel and Henrik Nilsson. Towards a framework for the implementation and verification of translations between argumentation models. In Rinus Plasmeijer, editor, *Proceedings of the 25th Symposium on Implementation and Application of Functional Languages (IFL 2013)*, page 93. ACM, 2013.

[van Gijzel and Nilsson, 2014] Bas van Gijzel and Henrik Nilsson. A principled approach to the implementation of argumentation models. In Simon Parsons, Nir Oren, Chris Reed, and Federico Cerutti, editors, *Proceedings of the 5th International Conference on Computational Models of Argument (COMMA 2014)*, volume 266 of *Frontiers in Artificial Intelligence and Applications*, pages 293–300. IOS Press, 2014.

[Vazirani, 2002] Vijay V. Vazirani. *Approximation Algorithms*. Springer, 2002.

[Verheij, 1996a] Bart Verheij. *Formal studies of argumentation and defeat*. PhD thesis, Universiteit Maastricht, 1996.

[Verheij, 1996b] Bart Verheij. Two approaches to dialectical argumentation: admissible sets and argumentation stages. In John-Jules Ch. Meyer and Linda C. van der Gaag, editors, *Proceedings of the Eighth Dutch Conference on Artificial Intelligence (NAIC'96)*, pages 357–368, Utrecht, NL, 1996.

[Verheij, 1998] Bart Verheij. Argue! - an implemented system for computer-mediated defeasible argumentation. In Han La Poutre and Jaap van den Herik, editors, *NAIC '98. Proceedings of the Tenth Netherlands/Belgium Conference on Artificial Intelligence*, pages 57–66, 1998.

[Verheij, 2003a] Bart Verheij. Artificial argument assistants for defeasible argumentation. *Artificial Intelligence and Law*, 150(1-2):291–324, 2003.

[Verheij, 2003b] Bart Verheij. DefLog: on the Logical Interpretation of Prima Facie Justified Assumptions. *Journal of Computational Logic*, 13(3):319–346, 2003.

[Verheij, 2007] Bart Verheij. A labeling approach to the computation of credulous acceptance in argumentation. In Manuela M. Veloso, editor, *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007)*, pages 623–628, 2007.

[Visser, 2008] Wietske Visser. Implementation of argument-based practical reasoning. Master's thesis, Utrecht University, 2008.

[Vreeswijk, 1993] Gerard A. Vreeswijk. *Studies in Defeasible Argumentation*. PhD thesis, Department of Computer Science, Free University Amsterdam, 1993.

[Wakaki and Nitta, 2008] Toshiko Wakaki and Katsumi Nitta. Computing argumentation semantics in answer set programming. In Hiromitsu Hattori, Takahiro Kawamura, and Tsuyoshi Ide, editors, *New Frontiers in Artificial Intelligence, (JSAI 2008) Conference and Workshops, Revised Selected Papers*, volume 5447 of *Lecture Notes in Computer Science*, pages 254–269, 2008.

[Walsh, 2000] Toby Walsh. SAT v CSP. In Rina Dechter, editor, *Proceedings of the 6th International Conference on Principles and Practice of Constraint Programming (CP*

2000), volume 1894 of *Lecture Notes in Computer Science*, pages 441–456. Springer, 2000.

[Walton *et al.*, 2014] Douglas Walton, Christopher W. Tindale, and Thomas F. Gordon. Applying recent argumentation methods to some ancient examples of plausible reasoning. *Argumentation*, 28(1):85–119, 2014.

[Watts and Strogatz, 1998] Duncan J. Watts and Steven H. Strogatz. Collective dynamics of small-world networks. *Nature*, 393(6684):440–442, 1998.

[Wells, 2014] Simon Wells. Argument mining: Was ist das? In *Proceedings of the 14th International Workshop on Computational Models of Natural Argument (CMNA 2014)*, 2014.

[Williams *et al.*, 2015] Matt Williams, Zi Wei Liu, Anthony Hunter, and Fergus Macbeth. An updated systematic review of lung chemo-radiotherapy using a new evidence aggregation method. *Lung cancer (Amsterdam, Netherlands)*, 87(3):290–5, 2015.

[Wooldridge *et al.*, 2006] Michael Wooldridge, Paul E. Dunne, and Simon Parsons. On the complexity of linking deductive and abstract argument systems. In Anthony Cohn, editor, *Proceedings of the 21st National Conference on Artificial Intelligence (AAAI 2006*, pages 299–304. AAAI Press, 2006.

[Wyner *et al.*, 2013] Adam Wyner, Trevor J. M. Bench-Capon, and Paul E. Dunne. On the instantiation of knowledge bases in abstract argumentation frameworks. In João Leite, Tran Cao Son, Paolo Torroni, Leon van der Torre, and Stefan Woltran, editors, *Proceedings of the 14th International Workshop on Computational Logic in Multi-Agent Systems (CLIMA XIV)*, volume 8143 of *Lecture Notes in Computer Science*, pages 34–50. Springer, 2013.

[Xu *et al.*, 2008] Lin Xu, Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown. Satzilla: Portfolio-based algorithm selection for sat. *Journal of Artificial Intelligence Research*, 32:565–606, 2008.

Received