On the Complexity of Enumerating the Extensions of Abstract Argumentation Frameworks

Markus Kröll, Reinhard Pichler, and Stefan Woltran
TU Wien, Austria
{kroell, pichler,woltran}@dbai.tuwien.ac.at

Abstract

Several computational problems of abstract argumentation frameworks (AFs) such as skeptical and credulous reasoning, existence of a non-empty extension, verification, etc. have been thoroughly analyzed for various semantics. In contrast, the enumeration problem of AFs (i.e., the problem of computing all extensions according to some semantics) has been left unexplored so far. The goal of this paper is to fill this gap. We thus investigate the enumeration complexity of AFs for a large collection of semantics and, in addition, consider the most common structural restrictions on AFs.

1 Introduction

Within the area of Artificial Intelligence, argumentation has become one of the major fields over the last two decades [Rahwan and Simari, 2009; Bench-Capon and Dunne, 2007]. Its most prominent approach is given by abstract argumentation frameworks (AFs) due to Dung [1995], a simple, yet powerful formalism for modeling and deciding problems that are integral to various advanced argumentation systems, see e.g. [Caminada and Amgoud, 2007]. In AFs one abstracts away from the actual contents of arguments and from the concrete reason of conflicts. Many different semantics have been proposed in the literature to define the set of possible extensions of an AF – each representing a coherent set of arguments that "jointly survive" the conflicts given by the AF. In contrast to other communities, the multitude of semantics is seen as a virtue of formal argumentation rather than a weakness and thus a significant amount of research has been done in order to understand particular features of the available semantics, see e.g. [Baroni *et al.*, 2011a; Dunne *et al.*, 2015].

Also several computational problems of AFs have been thoroughly analyzed such as skeptical and credulous reasoning, existence of a non-empty extension, and the verification problem [Dimopoulos and Torres, 1996; Dunne and Bench-Capon, 2002; Dvořák and Woltran, 2010; Gaggl and Woltran, 2013; Dvořák and Gaggl, 2016; Baroni *et al.*, 2011b]. Since these problems tend to be intractable for most semantics, structural restrictions on the AFs have been investigated and their potential for reducing the complexity of the aforementioned problems has been pinpointed [Coste-Marquis *et al.*, 2005; Dunne, 2007]. For an overview on complexity results, see also [Dvořák, 2012]. The only complexity analysis we are aware of that goes beyond decision problems is about the counting complexity for AFs [Baroni *et al.*, 2010].

In 2015, the first edition of the International Competition on Computational Models of Argumentation (ICCMA) took place and compared the performance of 18 submitted solvers in terms of skeptical and credulous reasoning, finding one extension, and enumerating all extensions [Thimm *et al.*, 2016]. For the next edition of ICCMA[1], in total seven semantics will be considered. This not only shows the increasing interest in solvers that are capable of dealing with multiple semantics, but also witnesses that the enumeration problem is indeed considered to be important by the community. However, in contrast to the detailed picture of the complexity of decision problems, the enumeration complexity has been left unexplored so far.

Formal tools for the complexity analysis of enumeration problems date back to [Johnson *et al.*, 1988] and have been further refined in recent years [Strozecki, 2010; Creignou *et al.*, 2013]. In particular, several notions of tractability have been introduced, since a complexity classification of enumeration problems has to take the size of the output into account.

The interest in enumeration problems spans various fields, with some of the most prominent and natural examples coming from the database area: in query answering, one is typically interested in obtaining *all* answers to a query rather than asking if *some* answer exists. See [Bulatov *et al.*, 2012; Kazana and Segoufin, 2013; Durand *et al.*, 2014; Kröll *et al.*, 2016] for recent works along this line. Enumeration complexity of graph problems has been studied by Goldberg [1991].

As emphasized above, enumeration is a central task for argumentation systems and, in order to understand their adequacy from a complexity theoretic point of view, a systematic study of the enumeration problem of AFs is needed. It is exactly the goal of this paper to provide such an analysis.

Main Contributions. We investigate the enumeration complexity for a total number of 11 multiple-status semantics and, in addition, consider 5 common structural restrictions on AFs, namely bipartite AFs, symmetric AFs (with or without the ad-

[1]http://www.dbai.tuwien.ac.at/iccma17/

ditional restriction of irreflexivity), AFs without even cycles, and implicitly also acyclic AFs. Our main result is a complete picture of the complexity of the enumeration problem of AFs for all these semantics for unrestricted AFs and for AFs with the above listed restrictions. We thus provide an exact separation between tractable and intractable enumeration for all the resulting 66 settings.

Structure. After this introduction, we recall the most relevant basic definitions on abstract argumentation frameworks and on enumeration problems in Section 2. In Section 3, we present an overview of our results. The proof strategy and the main proof ideas will be presented in Section 4. We conclude with Section 5.

2 Background

Abstract argumentation. We first recall some fundamental concepts related to abstract argumentation frameworks (AFs). This will enable us, in Table 1, to give concise definitions of the various semantics of AFs studied in this work.

Formally, an AF is a directed graph (A, R) where A is a finite set of arguments and $R \subseteq A \times A$ is the attack relation.

Given an AF $F = (A, R)$, we define $S^+_F = \{a \mid \exists s \in S : (s, a) \in R\}$, $S^-_F = \{a \mid \exists s \in S : (a, s) \in R\}$ and $S^\oplus_F = S \cup S^+_F$. The characteristic function $\mathcal{F}_F$ of AF F is defined as $\mathcal{F}_F : 2^A \to 2^A$ with $\mathcal{F}_F(S) = \{a \in A \mid \{a\}^-_F \subseteq S^+_F\}$.

A full resolution β of F is a subset $\beta \subseteq R$ such that for any two distinct arguments $a, b \in A$ with $\{(a, b), (b, a)\} \subseteq R$, exactly one of (a, b) and (b, a) is contained in β. We write $\gamma(F)$ to denote the set of all full resolutions of F. For $S \subseteq A$, we write $F|_S$ for the restriction of F to S, i.e., $F|_S = (A \cap S, R \cap (S \times S))$. By $SCCs(F)$ we denote the strongly connected components of the directed graph F. For $a \in A$, let $C_F(a)$ denote the strongly connected component containing a, i.e., $C_F(a) \in SCCs(F)$ unique with $a \in C_F(a)$. An argument $b \in A$ is called component-defeated by S (in F), if there exists $a \in S$ with $(a, b) \in R$ and $a \notin C_F(b)$. We write $D_F(S)$ to denote the set of arguments component defeated by S.

For a set of sets $\mathcal{M}$, we write $\min(\mathcal{M})$ to denote the set of inclusion-minimal sets in $\mathcal{M}$ and $\max(\mathcal{M})$ for the set of inclusion-maximal sets in $\mathcal{M}$.

The semantics of an AF is defined in terms of coherent sets of arguments (so-called "extensions") that jointly survive the conflicts given by the AF. In this work, we study the following semantics [Dung, 1995; Verheij, 1996; Caminada *et al.*, 2012; Baroni *et al.*, 2011b; 2005; Dvořák and Gaggl, 2016], which are defined via the extensions of a given AF F: the set of conflict-free sets $cf(F)$, naive sets $naive(F)$, admissible extensions $adm(F)$, preferred extensions $pref(F)$, stable extensions $stable(F)$, complete extensions $comp(F)$, semi-stable extensions $semi(F)$, stage extensions $stage(F)$, resolution-based grounded extensions $resGr(F)$, stage2 extensions $stage2(F)$, and cf2 extensions $cf2(F)$. The formal definitions are given in Table 1. Note that all semantics are multiple-status, i.e. they provide in general more than one extension. We will also occassionally refer to the grounded extension of an AF F, denoted by $grd(F)$; recall that this extension is unique for each AF. Formally, $grd(F)$ is given by the subset-minimal complete extension of F.

$$
\begin{aligned}
cf(F) &= \{S \subseteq A \mid \forall a, b \in S, (a, b) \notin R\}\\
naive(F) &= \max(cf(F))\\
adm(F) &= \{S \in cf(F) \mid S \subseteq \mathcal{F}_F(S)\}\\
pref(F) &= \max(adm(F))\\
stable(F) &= \{S \in adm(F) \mid S^\oplus_F = A\}\\
comp(F) &= \{S \in adm(F) \mid \mathcal{F}_F(S) \subseteq S\}\\
semi(F) &= \{S \in adm(F) \mid \forall T \subseteq A :\\
&\qquad \text{if } S^\oplus_F \subset T^\oplus_F \text{ then } T \notin adm(F)\}\\
stage(F) &= \{S \in cf(F) \mid \forall T \subseteq A :\\
&\qquad \text{if } S^\oplus_F \subset T^\oplus_F \text{ then } T \notin cf(F)\}\\
resGr(F) &= \min\Big(\bigcup_{\beta \in \gamma(F)} \min(comp((A, R \setminus \beta)))\Big)
\end{aligned}
$$

$S \in stage2(F)$ iff:
in case $|SCCs(F)| = 1$: $S \in stage(F)$
else: $\forall C \in SCCs(F)$, $(S \cap C) \in stage2(F|_{C \setminus D_F(S)})$

$S \in cf2(F)$ iff:
in case $|SCCs(F)| = 1$: $S \in naive(F)$
else: $\forall C \in SCCs(F)$, $(S \cap C) \in cf2(F|_{C \setminus D_F(S)})$

Table 1: Semantics of AFs, given $F = (A, R)$.

Several decision problems on AFs have been investigated in the literature. We briefly recall the complexity classification of two of the most prominent decision problems defined as follows: given an AF F, semantics σ, and argument a, decide whether a is contained (i) in at least one σ-extension of F (credulous reasoning); (ii) in all σ-extensions of F (skeptical reasoning). Both problems are tractable for cf and $naive$, and skeptical reasoning is tractable for $comp$ and trivial for adm (since $\emptyset \in adm(F)$ for any AF). Credulous reasoning is NP-complete for adm, $pref$, $stable$, $comp$, $resGr$, and $cf2$, and Σ^P_2-complete for $semi$, $stage$ and $stage2$. Skeptical reasoning is coNP-complete for $stable$, $resGr$, and $cf2$ and Π^P_2-complete for $pref$, $semi$, $stage$ and $stage2$; see e.g. [Dvořák, 2012; Dvořák and Gaggl, 2016] for further details.

Enumeration problems and their complexity. An *enumeration problem* is a pair (L, Sol) such that $L \subseteq \Sigma^*$ (for an alphabet Σ containing at least two symbols) and $\mathsf{Sol} : \Sigma^* \to 2^{\Sigma^*}$ is a function such that for all $x \in \Sigma^*$, we have that $\mathsf{Sol}(x)$ (the set of "solutions") is finite and $\mathsf{Sol}(x) = \emptyset$ iff $x \notin L$.

An *enumeration algorithm* $\mathcal{A}$ for an enumeration problem $\mathcal{P} = (L, \mathsf{Sol})$ is an algorithm which, on input x, outputs exactly the elements from $\mathsf{Sol}(x)$ without duplicates. We denote the output of algorithm $\mathcal{A}$ on x by $\mathcal{A}(x)$.

As far as the computation model of enumeration algorithms is concerned, it is common (cf. [Strozecki, 2010]) to use the RAM model. This is motivated by the fact that a RAM can access parts of exponential-size data in polynomial time. As we will see below, this may be needed to efficiently detect (and avoid the output of) duplicates. We restrict ourselves to polynomially bounded RAM machines, i.e., the size of each register is polynomially bounded in the size of the input.

Let $\mathcal{A}$ be an enumeration algorithm for some problem $\mathcal{P}$. For an input x, let $n = |\mathcal{A}(x)|$. For $0 \le i \le n$, we define $delay(i)$ as follows: $delay(0)$ ("preprocessing") is the time between the start of the algorithm and the first output (or termination of $\mathcal{A}$, if $n = 0$). For $0 < i < n$, $delay(i)$ is the time between outputting solution i and $(i + 1)$. Finally, $delay(n)$ is the time between the last output and the termination of $\mathcal{A}$.

Several notions of *tractability* have been proposed in the literature [Johnson *et al.*, 1988; Creignou *et al.*, 2013; Strozecki, 2010]. We recall the two most relevant ones for our purposes below, namely DelayP ("polynomial delay") and OutputP ("output-polynomial time"). Alternatively, the latter class is sometimes referred to as TotalP ("total polynomial time") [Strozecki, 2010].

The classes DelayP and OutputP are defined as follows. Let $\mathcal{P} = (L, \mathsf{Sol})$ be an enumeration problem.

- $\mathcal{P} \in$ OutputP, if there exists an enumeration algorithm $\mathcal{A}$ for $\mathcal{P}$ and some $m \in \mathbb{N}$, such that on every input x, algorithm $\mathcal{A}$ terminates in time $\mathcal{O}((|x| + |\mathsf{Sol}(x)|)^m)$.
- $\mathcal{P} \in$ DelayP, if there exists an enumeration algorithm $\mathcal{A}$ for $\mathcal{P}$ and some $m \in \mathbb{N}$, such that on every input x and for every $i \in \{0, \dots, |\mathsf{Sol}(x)|\}$, $delay(i)$ is in $\mathcal{O}(|x|^m)$.

Clearly, DelayP $\subseteq$ OutputP holds. This inclusion can be seen as follows: suppose that an enumeration algorithm $\mathcal{A}$ for some problem $\mathcal{P}$ works with delay $\mathcal{O}((|x|^m))$ for some $m \ge 1$. Moreover, let $N = |\mathsf{Sol}(x)|$. Then the total time of $\mathcal{A}$ to output all solutions for input x is $\mathcal{O}((N+1) * |x|^m)$. Recall that the factor $(N + 1)$ rather than N is needed for the time between the last output and termination of $\mathcal{A}$. For $N = 0$, we have $\mathcal{O}((N + 1) * |x|^m) = \mathcal{O}(|x|^m) = \mathcal{O}((N + |x|)^m)$. For $N > 0$, the sequence of equalities $\mathcal{O}((N + 1) * |x|^m) = \mathcal{O}(N * |x|^m) = \mathcal{O}((N + |x|)^{m'})$ with $m' = m + 1$ holds, which proves the desired OutputP-membership of $\mathcal{P}$.

Hence, membership in DelayP is the stronger tractability result, which entails membership in OutputP. Conversely, the stronger intractability result is to show that a problem is not in OutputP (under the common assumption that P $\neq$ NP), which entails non-membership in DelayP.

Note that membership in DelayP does not impose a restriction on the space consumption of an enumeration algorithm. More specifically, a DelayP algorithm may require the construction of an index structure for the already found solutions to avoid the output of duplicates. Hence, if there are exponentially many solutions, this index structure and, therefore, the space consumption of the algorithm may become exponential. We will use the notation $\mathsf{DelayP}_\mathsf{P}$ to refer to the class of enumeration problems which can be enumerated with polynomial delay using only polynomial space. In contrast, when we state DelayP membership of a problem, this means that the enumeration may potentially use exponential space.

3 Overview of Results

The overall goal of this work is to study the complexity of the enumeration problem of abstract argumentation under the various semantics recalled in the previous section. We denote by $EnumExt(\mathcal{C}, \sigma)$ the enumeration problem of σ-extensions of argumentation frameworks in a class $\mathcal{C}$ of AFs. An algorithm for the $EnumExt(\mathcal{C}, \sigma)$ problem thus takes $F \in \mathcal{C}$ as input and outputs $\sigma(F)$.

	$\mathcal{C}_0$	$\mathcal{C}_{\text{bip}}$	$\mathcal{C}_{\text{sym}}$	$\mathcal{C}_{(\text{ir,sym})}$	$\mathcal{C}_{\text{noev}}$
cf	**DelayP$_\mathbf{P}$**	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$
naive	**DelayP$_\mathbf{P}$**	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$
adm	**nOP**	**DelayP**	DelayP$_\mathsf{P}$	$\rightarrow$	**DelayP**
pref	$\leftarrow$	nOP	DelayP$_\mathsf{P}$	$\rightarrow$	trivial
stable	$\leftarrow$	**nOP**	**nOP**	DelayP$_\mathsf{P}$	trivial
comp	$\leftarrow$	**nOP**	**DelayP$_\mathbf{P}$**	$\rightarrow$	trivial
semi	$\leftarrow$	nOP	**nOP**	DelayP$_\mathsf{P}$	trivial
stage	$\leftarrow$	nOP	**nOP**	DelayP$_\mathsf{P}$	**nOP**
resGr	**DelayP$_\mathbf{P}$**	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$
stage2	$\leftarrow$	**nOP**	**nOP**	DelayP$_\mathsf{P}$	**nOP**
cf2	**DelayP$_\mathbf{P}$**	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$

Table 2: Summary of tractability/intractability results. Entry "$\leftarrow$" means that hardness carries over from some special case. "$\rightarrow$" means that membership carries over from a more general case.

Recall from Section 2 that, for most of the semantics σ considered here, the principal decision problems such as credulous and skeptical reasoning are intractable. Consequently, several structural restrictions on the AFs have been studied in the literature. In this work, we study three of the most commonly used subclasses of AFs, namely: the restrictions to bipartite graphs, to symmetric graphs and to graphs without even cycles. As far as the restriction to symmetric graphs is concerned, one has often imposed the further restriction of irreflexivity, see e.g. [Coste-Marquis *et al.*, 2005]. In total, we thus consider the following 5 classes of AFs: AFs without any restrictions (which we will denote as $\mathcal{C}_0$), AFs restricted to bipartite graphs (denoted as $\mathcal{C}_{\text{bip}}$), irreflexive, symmetric AFs (denoted as $\mathcal{C}_{(\text{ir,sym})}$), symmetric AFs without irreflexivity restriction (denoted as $\mathcal{C}_{\text{sym}}$), and AFs restricted to graphs with no even cycles (denoted as $\mathcal{C}_{\text{noev}}$).

Our results are summarized in Table 2. Each of the 11 semantics is treated in a separate row. We then have 5 columns for the 5 classes of AFs mentioned above. Table 2 provides a complete complexity classification of the enumeration problem – separating the tractable cases (marked with one of the entries "DelayP", "DelayP$_\mathsf{P}$" or "trivial") from the intractable ones (marked with "nOP", which is short for "not in OutputP" under the common assumption that P $\neq$ NP).

Clearly, if tractability even holds in the unrestricted case (e.g. for *cf* semantics), then tractability propagates to the restricted cases. Likewise, tractability for AFs in $\mathcal{C}_{\text{sym}}$ (e.g., for *pref* semantics) carries over to AFs in $\mathcal{C}_{(\text{ir,sym})}$. We mark this with the entry "$\rightarrow$" in Table 2. Conversely, if at least one of the restricted cases is intractable, then intractability propagates to the unrestricted case. We mark this with the entry "$\leftarrow$". The cases where the structural restriction causes the semantics to have at most one nonempty extension (namely the grounded extension, which can always be computed efficiently) are marked with "trivial".

When filling in the remaining entries in Table 2, we can make use of further dependencies between the different cases.

In particular, under various restrictions, some of the semantics coincide. For instance, in symmetric AFs, every conflict-free extension is admissible. Hence the $\mathsf{DelayP_P}$-membership of enumerating admissible extensions in case of symmetric AFs requires no separate proof. Similarly, for bipartite AFs, every preferred extension is stable. Hence, the intractability of enumerating stable extensions in case of bipartite AFs carries over to preferred extensions. To distinguish the entries in Table 2 which require a separate proof from the ones which follow by the collapse of different semantics in some restricted case, we display the former results in boldface.

Note that the restriction to *acyclic graphs*, which has also been used in the literature to achieve tractability of decision problems, is implicitly covered by our results: First, since acyclic AFs are a special case of AFs without even cycles, all the tractability results of the latter case carry over to acyclic AFs. Moreover, the only intractable cases for AFs without even cycles are stage and stage2 semantics. However, for acyclic AFs, these two coincide with grounded semantics (cf. [Dvořák and Gaggl, 2016]), which is a trivially tractable case.

4 Proof Strategy and Main Proof Ideas

Before we start discussing the various entries in Table 2, we define the decision problem *ManySol*$(\mathcal{P})$, which is strongly related to an enumeration problem $\mathcal{P}$. For an arbitrary enumeration problem $\mathcal{P} = (L, \mathsf{Sol})$, it is defined as follows:

- *ManySol*$(\mathcal{P})$: Given $x \in L$ and a positive integer m in unary notation, is $|\mathsf{Sol}(x)| \geq m$?

The following property of the *ManySol* problem will make it a useful tool for our intractability proofs.

Proposition 1. *Let $\mathcal{P} = (L, \mathsf{Sol})$ be an enumeration problem. If ManySol$(\mathcal{P}) \notin \mathsf{P}$, then $\mathcal{P} \notin \mathsf{OutputP}$.*

Proof Sketch. Let $\mathcal{P} = (L, \mathsf{Sol})$ be an enumeration problem. We prove the contrapositive, i.e., assume that $\mathcal{P} \in \mathsf{OutputP}$. We have to show that then $ManySol(\mathcal{P}) \in \mathsf{P}$ also holds. By $\mathcal{P} \in \mathsf{OutputP}$, there is an algorithm $\mathcal{A}$ computing $\mathcal{A}(x) = \mathsf{Sol}(x)$ for any $x \in L$ in time $p(|x| + |\mathsf{Sol}(x)|)$ for some polynomial p. Let $x \in L$, $m \geq 0$. Then we can decide whether $|\mathsf{Sol}(x)| \geq m$ as follows: we run algorithm $\mathcal{A}$ on input x for at most $s = p(|x| + m)$ many steps. If the algorithm terminates before s many steps, it outputs $\mathsf{Sol}(x)$ and we can check whether $|\mathsf{Sol}(x)| \geq m$. If $\mathcal{A}$ has not terminated after s many steps, we must have that $|\mathsf{Sol}(x)| > m$ as $\mathcal{A}$ produces an output after $p(|x| + |\mathsf{Sol}(x)|)$ computational steps.

Since m is given in unary, the size of the input to the *ManySol* problem equals $|x| + m$. Hence, we have shown that $ManySol(\mathcal{P})$ can indeed be decided in polynomial time. $\square$

The following decision problem, which is parameterized by a semantics σ, is a special case of *ManySol*$(\mathcal{P})$, provided that σ is guaranteed to have the empty set as an extension.

- $Exists_\sigma^{\neg\emptyset}$: Given an AF $F = (A, R)$, does there exist a set $\emptyset \neq S \subseteq A$ with $S \in \sigma(F)$?

The following property is an immediate consequence of Proposition 1.

Corollary 1. *Let σ be a semantics such that $\emptyset \in \sigma(F)$ for all AFs F and let $\mathcal{C}$ be a class of AFs. If $Exists_\sigma^{\neg\emptyset} \notin \mathsf{P}$ for AFs restricted to $\mathcal{C}$, then $EnumExt(\mathcal{C}, \sigma) \notin \mathsf{OutputP}$.*

We are now ready to present the main proof ideas of the results summarized in Table 2. To this end, we inspect one column of the table after the other.

Unrestricted AFs. First consider the complexity of enumerating conflict-free and naive extensions. For an AF $F = (A, R)$, $S \subseteq A$ is a (maximal) conflict-free set iff it is a (maximal) independent set in the corresponding undirected graph. It was shown in [Johnson *et al.*, 1988] that the *maximal* independent sets can be enumerated in $\mathsf{DelayP_P}$. The same result holds for enumerating *all* independent sets. Hence, we get $EnumExt(\mathcal{C}_0, \sigma) \in \mathsf{DelayP_P}$ with $\sigma \in \{naive, cf\}$.

The tractability result for *cf2* semantics is obtained by applying standard algorithms for recursively enumerating extensions, see e.g. [Cerutti *et al.*, 2014].

Theorem 1. *$EnumExt(\mathcal{C}_0, cf2) \in \mathsf{DelayP_P}$ holds.*

Proof Sketch. Let $F = (A, R)$ with strongly connected components $S_1, \dots, S_m \subseteq A$. W.l.o.g., assume that $S_1, \dots, S_n$ are the bottom components for some $n \leq m$, i.e. strongly connected components without an incoming edge from another component. By the definition of *cf2*, every naive extension on a bottom component is part of some $S \in cf2(F)$. As for each such bottom component the naive extensions are enumerable with polynomial delay using polynomial space, we can fix some naive set $N_i \subseteq S_i$ for $1 \leq i \leq n$. Next we get rid of the arguments of the bottom components $\mathcal{S}_1 = \bigcup S_i$ and the ones which are component defeated by $\mathcal{N}_1 = \bigcup N_i$, resulting in the AF $F^1 = F|_{A \setminus (\mathcal{S}_1 \cup \mathcal{N}_1)}$. As before, we choose a naive extension for every bottom component of F^1, thus again obtaining a union of naive extensions $\mathcal{N}_2$. After repeating this process at most k many times for some $k \leq |A|$, we have that $F^k \neq \emptyset$ and $F^{k+1} = \emptyset$. Then $S = \bigcup_{i=1}^{k} \mathcal{N}_i$ is a *cf2* extension of F. By selecting a single different naive extension of a bottom component of some F^j, the same procedure obtains an $S' \in cf2(F)$ with $S' \neq S$. Through backtracking, we can thus enumerate all extensions $S \in cf2(F)$ with polynomial delay using polynomial space. $\square$

It only remains to establish the tractability of $resGr$ and the intractability of adm semantics, since for all further semantics, the intractability in the unrestricted case carries over from the intractability of some restricted case.

For $resGr$, an enumeration algorithm is given in [Baroni *et al.*, 2011b, Algorithm 2]. This algorithm can be refined to an algorithm with polynomial delay using polynomial space by making use of an algorithm for maximal independent sets and backtracking. For adm, we recall that the $Exists_{adm}^{\neg\emptyset}$ problem is NP-complete, which follows from results in [Dimopoulos and Torres, 1996]. Hence, by Corollary 1, we may conclude that $EnumExt(\mathcal{C}_0, adm) \notin \mathsf{OutputP}$ holds unless $\mathsf{P} = \mathsf{NP}$.

Bipartite AFs. We first inspect the tractability for adm semantics. It is easy to verify that, given an admissible set S in a bipartite AF $F = (A_1 \cup A_2, R)$, the restrictions $S \cap A_1$

and $S \cap A_2$ are also admissible. The crucial properties of bipartite AFs used here are that, for $i \in \{1,2\}$, all subsets of A_i are conflict-free and all defenders of an argument in A_i must also be in A_i. To compute all admissible sets of F, we have to find the sets $\mathcal{A}_i$ of admissible subsets of A_i for each $i \in \{1,2\}$ separately and then find all conflict-free combinations $S_1 \cup S_2$ with $S_i \in \mathcal{A}_i$. For the computation of $\mathcal{A}_i$, the following lemma, which holds for arbitrary AFs, is crucial.

Lemma 1. *Given an AF (A,R) and a conflict-free set $M \subseteq A$, enumerating all admissible subsets $S \subseteq M$ is in* DelayP.

Proof Sketch. It is shown in [Dunne *et al.*, 2013, Lemma 1] that, given a conflict-free set $M \subseteq A$, there exists a unique maximal admissible subset $S^* \subseteq M$, which can be computed in polynomial time. Then the desired algorithm for enumerating all admissible subsets of M works as follows: we maintain two queues $\mathcal{P}$ and $\mathcal{Q}$ of admissible sets, such that $\mathcal{P}$ contains the already printed and processed ones while $\mathcal{Q}$ contains those which have been found but not printed and processed yet. Initially, $\mathcal{P} = \emptyset$ and $\mathcal{Q} = \{S^*\}$. As long as $\mathcal{Q}$ is not empty, we take an element from $\mathcal{Q}$, output it and search recursively for admissible subsets of $S \setminus \{x\}$ for each $x \in S$. For every admissible set S' thus found, we check if $S' \notin \mathcal{P} \cup \mathcal{Q}$ and, if so, add it to $\mathcal{Q}$.

The correctness of this algorithm is easy to verify. The polynomial delay can be seen as follows: the already found extensions have to be organized in some index structure. In principle, any balanced search tree formalism (e.g., AVL trees) can be used. Yet simpler, we can arrange extensions of an AF with n arguments in a binary tree of depth n where each leaf node (or, equivalently, each path from the root to a leaf) represents an already found extension. A node at depth i is either labelled with "a_i in" or with "a_i out" depending on whether argument a_i is contained in the extension or not. Checking if the current extension is new and extending the tree in case it is, can thus be done in time $\mathcal{O}(n)$. □

Note that Lemma 1 only states DelayP membership rather than $\mathsf{DelayP}_\mathsf{P}$ membership. Indeed, the algorithm in the proof of Lemma 1 crucially depends on an index structure of all the already found solutions to avoid the output of duplicates. Hence, if there are exponentially many solutions, the space requirement of the algorithm becomes exponential as well.

Theorem 2. *$EnumExt(\mathcal{C}_{\text{bip}}, adm) \in$ DelayP holds.*

Proof Sketch. Consider a bipartite AF $F = (A_1 \cup A_2, R)$. Then each of the two sets A_i is conflict-free. As mentioned in the proof of Lemma 1, we can thus compute the unique maximal admissible set $S_i^* \subseteq A_i$ efficiently by applying [Dunne *et al.*, 2013, Lemma 1]. In principle, using our Lemma 1 above, we could compute with polynomial delay the set $\mathcal{A}_i$ of all admissible subsets of A_i and finally check each pair $(S_1, S_2) \in \mathcal{A}_1 \times \mathcal{A}_2$ for conflict-freeness to find and output all admissible sets $S_1 \cup S_2$ with elements from both sides A_1 and A_2. However, the challenge with bipartite AFs is that there may exist exponentially many pairs (S_1, S_2) but only a small number of them yields a conflict-free set $S_1 \cup S_2$.

Hence, we have to be careful to get an overall enumeration algorithm which works in polynomial delay. The solution to overcome this problem is an interleaved output of the admissible subsets S_i of A_i with $i \in \{1,2\}$ and of admissible subsets of the form $S_1 \cup S_2$:

Whenever an admissible subset $S_1 \subseteq A_1$ is processed, we search recursively for the maximal admissible subsets of $S_1 \setminus \{x\}$ for each $x \in S_1$ as in the proof sketch of Lemma 1 and, moreover, we also search for all admissible subsets $S_2 \subseteq A_2$ such that $S_1 \cup S_2$ is conflict-free. For the latter task, we first compute $M \subseteq A_2$ containing all arguments that have no conflict with S_1. Clearly M is conflict-free, since it only has arguments from one side of the bipartite AF. We can then find all admissible subsets $S_2 \subseteq M$ with polynomial delay by Lemma 1. For every S_2 thus found, we output $S_1 \cup S_2$. Note that in this way also all admissible subsets $S_1 \subseteq A_1$ are output (namely when we combine S_1 with $S_2 = \emptyset$) and also admissible subsets $S_2 \subseteq A_2$ are output (namely when we combine S_2 with $S_1 = \emptyset$), since $\emptyset$ is guaranteed to be among the admissible subsets of A_1 and A_2, respectively. □

Before we switch to intractability, we recall that the *stable*, *stage*, *semi*, and *pref* semantics all coincide for bipartite AFs. This follows from the fact that bipartite AFs are odd-cycle free, hence *stable* and *pref* coincide; this also guarantees the existence of a stable extension which implies that stable, semi-stable and stage extensions coincide. Hence, to prove the intractability results of bipartite AFs in Table 2, it suffices to consider the *stable*, *stage2*, and *comp* semantics. To this end, we first recall the following definition. Let $G = (V,E)$ be a directed graph and $K \subseteq V$. Then K is called a *kernel* of G, if K is an independent dominating set, i.e. the nodes in K are pairwise non-adjacent and for every $v \in V \setminus K$ there exists a $k \in K$ with $(k,v) \in E$. As shown by Dimopoulos *et al.* [1997] it is NP-complete to decide whether a given graph has more than two kernels and that this problem remains NP-complete even if the graphs are restricted to bipartite graphs with a single SCC. Building upon this result, we prove the following intractability results for bipartite AFs.

Theorem 3. *Let $\sigma \in \{stable, comp, stage2\}$. Then $EnumExt(\mathcal{C}_{\text{bip}}, \sigma) \notin$ OutputP holds unless P = NP.*

Proof Sketch. "*stable*". Clearly, for an AF $F = (A,R)$, $S \subseteq A$ is a stable extension iff it is a kernel in the corresponding graph. This means that for the class $\mathcal{C}_{\text{bip}}$ of AFs and *stable* semantics, the *ManySol* problem is NP-hard. Hence $EnumExt(\mathcal{C}_{\text{bip}}, stable) \notin$ OutputP by Proposition 1.

"*stage2*". Since *stable* and *stage* semantics coincide for bipartite AFs, the enumeration of *stage* extensions shares the hardness of *stable* ones. Since the underlying NP-hardness results for kernels holds even for bipartite graphs with a single SCC, we also have $EnumExt(\mathcal{C}, stage2) \notin$ OutputP.

"*comp*". We can use the reduction of the NP-hardness proof in [Dimopoulos *et al.*, 1997] to show that for the class $\mathcal{C}_{\text{bip}}$ of AFs and *comp* semantics, the *ManySol* problem is NP-hard. Instead of kernels, satisfying truth assignments correspond to non-trivial complete extensions, which are subsets of the non-trivial kernels of the bipartite graph. □

Symmetric AFs. In symmetric AFs, every argument defends itself against any attacks. Hence, *cf* and *adm* semantics coincide and so do *naive* and *pref* semantics. The $\mathsf{DelayP}_\mathsf{P}$ membership of cf and $naive$ semantics thus carries over to adm and $pref$ semantics. For $cf2$, and $resGr$, $\mathsf{DelayP_P}$ membership carries over from the unrestricted case. For the remaining semantics, no collapse with an already shown tractable case occurs in case of symmetric AFs. Indeed, for the following semantics, we can show intractability:

Theorem 4. *Let* $\sigma \in \{stable, stage, semi, stage2\}$. *Then* $EnumExt(\mathcal{C}_{\text{sym}}, \sigma) \notin \mathsf{OutputP}$ *holds unless* $\mathsf{P} = \mathsf{NP}$.

Proof Sketch. In [Dvořák, 2012] it was shown that for an arbitrary AF F one can construct in polynomial time a symmetric AF F' with self-loops such that $stage(F) = stage(F') = semi(F')$ and $stable(F) = stable(F')$. In other words, the problem of enumerating the stage extensions of an arbitrary AF F can be reduced to the problem of enumerating the stage (resp. semi-stable) extensions of a symmetric AF $F' \in \mathcal{C}_{\text{sym}}$. Likewise, the problem of enumerating the stable extensions of an arbitrary AF F can be reduced to the problem of enumerating the stable extensions of a symmetric AF $F' \in \mathcal{C}_{\text{sym}}$. This means that $EnumExt(\mathcal{C}_{\text{sym}}, \sigma) \notin \mathsf{OutputP}$ holds for $\sigma \in \{stage, stable, semi\}$ unless $\mathsf{P} = \mathsf{NP}$.

For $stage2$, we observe that the aforementioned reduction from F to F' preserves strong connectivity. Using the AF F to be constructed in the proof of Theorem 7, we have that $stage(F) = stage(F') = stage2(F')$, thus proving intractability also for $EnumExt(\mathcal{C}_{\text{sym}}, stage2)$. □

It only remains to consider $comp$ semantics. To study this case, we define the following decision problem:

- *ExtSolComplete* Given an AF $F = (A, R)$ and sets of arguments $I, O \subseteq A$, is there some $S \in comp(F)$ with $I \subseteq S$ and $O \cap S = \emptyset$?

While this problem is intractable for arbitrary AFs, it can be shown tractable for $comp$ semantics over symmetric AFs.

Lemma 2. *ExtSolComplete is in* P *for symmetric AFs.*

Proof Sketch. Intuitively, I (resp. O) contains the arguments that we want to be *in* (resp. *out*). A polynomial time algorithm $\mathcal{A}$ for solving this decision problem works as follows: On input $F = (A, R)$ and $I, O \subseteq A$, we first check whether $I \in cf(F)$. If this is not the case then $\mathcal{A}$ ends in a rejecting state; otherwise, we compute the minimal fixpoint S of $\mathcal{F}_F$ such that $I \subseteq S$ holds. Since, for symmetric AFs, conflict-free sets are admissible, S is the minimal complete set containing I. So $\mathcal{A}$ accepts the input iff $S \cap O = \emptyset$. □

Similarly to algorithms in [Creignou and Hébrard, 1997] and [Kröll *et al.*, 2016], we can use algorithm $\mathcal{A}$ for *ExtSolComplete* from Lemma 2 to design an efficient enumeration algorithm for complete extensions on symmetric AFs.

Theorem 5. $EnumExt(\mathcal{C}_{\text{sym}}, comp) \in \mathsf{DelayP_P}$ *holds.*

Proof Sketch. Let $F = (A, R)$ with $A = \{a_1, \dots, a_n\}$. The idea of the enumeration algorithm is the following: We extend single arguments to complete extensions successively, by repeatedly adding arguments to such partial extensions as well as fixing a set of arguments that cannot be added to such partial extensions. Starting with the argument a_1, the enumeration algorithm first obtains the answers of algorithm $\mathcal{A}$ from Lemma 2 on input $(F, \{a_1\}, \{\})$ and $(F, \{\}, \{a_1\})$. For every input $\Omega = (F, I_\Omega, O_\Omega)$ accepted by $\mathcal{A}$ we add a_2 to either I_Ω or O_Ω resulting in two new inputs $\Omega_1 = (F, I_\Omega \cup \{a_2\}, O_\Omega)$ and $\Omega_1 = (F, I_\Omega, O_\Omega \cup \{a_2\})$. Repeating this n times, we obtain all complete extensions I_Ω for inputs Ω of $\mathcal{A}$ such that $\mathcal{A}$ accepts and $I_\Omega \cup O_\Omega = \{a_1, \dots, a_n\}$. By a depth-first computation and backtracking, this enumeration can be done with polynomial delay using polynomial space. □

Irreflexive, symmetric AFs. We now consider AFs which are symmetric *and* irreflexive. Clearly, all tractability results carry over from symmetric AFs. It remains to consider those cases, for which the enumeration problem of symmetric AFs with self-loops is intractable. The restriction to irreflexive, symmetric AFs implies that stable and preferred semantics coincide [Coste-Marquis *et al.*, 2005]. Hence, $\mathsf{DelayP_P}$-membership of $pref$ semantics carries over to $stable$, and consequently to $semi$ and $stage$ semantics (since we are guaranteed that a stable extension exists). Finally, $stage2$ and $stage$ extensions coincide, since in symmetric AFs different components are not connected at all. Hence, we also have $\mathsf{DelayP_P}$-membership for $stage2$ semantics.

No-even AFs. Let $F = (A, R)$ be an AF without even cycles. As shown in [Dunne and Bench-Capon, 2001], the grounded extension $S \subseteq A$ is the only complete (and also preferred and semi-stable) extension and is computable in polynomial time. Moreover, S is the only candidate for a stable extension and we can test whether $S \in stable(F)$ efficiently. Hence, for any of the semantics $stable$, $pref$, $comp$, and $semi$, the enumeration problem becomes trivial. It remains to consider the adm, $stage$, and $stage2$ semantics. Below, we show that adm leads to tractability while the latter two lead to intractability.

Theorem 6. $EnumExt(\mathcal{C}_{\text{noev}}, adm) \in \mathsf{DelayP}$ *holds.*

Proof Sketch. For AF $F \in \mathcal{C}_{\text{noev}}$, $comp$ and grd semantics coincide. Hence, the grounded extension of F (which can be computed efficiently) is the unique maximal admissible set of F. In particular, it is conflict-free and contains all admissible sets of F as subsets. By Lemma 1, we can enumerate all admissible sets $S \subseteq M$ with polynomial delay. □

Theorem 7. *Let* $\sigma \in \{stage, stage2\}$. *Then* $EnumExt(\mathcal{C}_{\text{noev}}, \sigma) \notin \mathsf{OutputP}$ *holds unless* $\mathsf{P} = \mathsf{NP}$.

Proof Sketch. By Proposition 1, it suffices to show that the *ManySol* problem of $stage$ and $stage2$ extensions is NP-hard on AFs without even cycles. The proof is by reduction from SAT. For $stage$ semantics, we can take over almost literally the construction from [Dvořák, 2012, Theorem 30] (which, in turn, uses ideas similar to [Dvořák *et al.*, 2014, Proposition 13]). The only modification needed is to delete an auxiliary argument q from the AF constructed there.

Apart from a self-cycle, the resulting AF is acyclic, i.e., every argument builds its own SCC. The main task to prove NP-hardness also for $stage2$ semantics is a general one: we are given an acyclic AF and we have to extend it in such a way that the resulting AF consists of a single SCC. The challenge here is that we must not introduce an even cycle.

Our construction proceeds in two steps: first, we assume an appropriate topological sort of the arguments in the directed acyclic graph (ignoring self-cycles) from [Dvořák, 2012, Theorem 30] and we introduce "backward paths" of length 2 between any two successive vertices. That is, let A be sorted as $A = \{a_1, \dots, a_N\}$. Then, for every $i \in \{1, \dots, N-1\}$, we add vertices u_i and arcs $(a_{i+1}, u_i), (u_i, a_i), (u_i, u_i)$. The self-cycles prevent the u_i's from being selected in a stage extension. The idea of these paths of length 2 is that every simple cycle in the resulting graph consists of a "forward" arc (i.e., from some a_i to some a_{i+k}) and a sequence of $k-1$ "backward paths" of length 2. Hence, these cycles have odd length. Moreover, since a_1 can be chosen such that every other vertex in A is reachable from it, the additional paths of length 2 thus make sure that the graph has a single SCC.

The second step then adds further auxiliary vertices v_i, v_i' together with a cycle of length 3 for every $i \in \{1, \dots, N-1\}$: $(v_i, u_i), (u_i, v_i'), (v_i', v_i), (v_i', v_i')$; the AF thus remains to consist of a single component. Moreover, every stage extension contains the "auxiliary" vertices v_i and, thus, has all vertices u_i in its range. This is needed to guarantee that the maximality condition of stage extensions only depends on the original vertices in A and not on the question as to which of the additional vertices u_i are contained in the range.

Analogously to [Dvořák, 2012, Theorem 30], one can show that the resulting AF is guaranteed to have $2m$ stage extensions, where m denotes the number of clauses in the instance φ of the SAT problem. Further stage extensions exist if and only if φ is satisfiable. Hence, the *ManySol* problem is NP-hard and intractability of $\mathit{EnumExt}(\mathcal{C}_{\mathrm{noev}}, \sigma)$ for the semantics $\sigma \in \{\mathit{stage}, \mathit{stage2}\}$ follows from Proposition 1. $\square$

5 Conclusion

In this work, we have started a systematic study of the enumeration problem in the context of abstract argumentation. As our main result, we have identified the border between tractable and intractable enumeration for AFs under 11 of the most common semantics of AFs. A great variety of settings has been explored by considering unrestricted AFs as well as AFs with common structural restrictions (namely, bipartite graphs, symmetric graphs with or without self-loops, graphs without even cycles, and implicitly also acyclic graphs).

These tractability and intractability results can now be used as guidance for developers of argumentation tools which compute the set of extensions under a given semantics. Developers have to make sure that, in the tractable cases identified here, the tools indeed work efficiently. In particular, for reduction-based argumentation tools (working by reduction to SAT or ASP), it is not guaranteed a priori that favorable structural properties of an input AF are preserved under the reduction and exploited by the target solvers.

Note that further structural restrictions of AFs have been studied in the literature, such as AFs with no odd cycles. Since they provide a generalization of the class $\mathcal{C}_{\mathrm{bip}}$ of AFs restricted to bipartite graphs, our intractability results for the semantics pref, stable, comp, semi, stage, and $\mathit{stage2}$ carry over to AFs with no odd cycles. Likewise, the $\mathsf{DelayP_P}$ membership for the semantics cf, naive, resGr, and $\mathit{cf2}$ carries over from the unrestricted case $\mathcal{C}_0$. In contrast, the adm semantics remains as an interesting open question. Further structural restrictions to be considered can be found in [Dunne, 2007]. A particularly interesting class is given by AFs which, when interpreted as undirected graphs, have bounded treewidth. For this class, MSO encodings as given in [Dunne, 2007; Dvořák *et al.*, 2012] readily pave the way for showing tractable enumeration via meta-theorems as provided in [Flum *et al.*, 2002; Bagan, 2006; Courcelle, 2009].

Our study has revealed that the boundary between easy and hard cases of enumeration may well differ from related decision problems. For instance, as recalled in Section 2, both credulous and skeptical reasoning with *cf2* semantics is intractable. Nevertheless, the enumeration of all *cf2* extensions is feasible with polynomial delay. Hence, for future work, the search for further tractable fragments of computational problems in the area of argumentation should take the enumeration problem as an additional focus into account.

Acknowledgments

This research was supported by the Austrian Science Fund (FWF): P25207-N23, P25518-N23, I2854-N35, W1255-N23. We thank Wolfgang Dvořák for many fruitful discussions and his help.

References

[Bagan, 2006] Guillaume Bagan. MSO queries on tree decomposable structures are computable with linear delay. In *Proc. CSL'06*, volume 4207 of *LNCS*, pages 167–181. Springer, 2006.

[Baroni *et al.*, 2005] Pietro Baroni, Massimiliano Giacomin, and Giovanni Guida. SCC-recursiveness: a general schema for argumentation semantics. *Artificial Intelligence*, 168(1-2):162–210, 2005.

[Baroni *et al.*, 2010] Pietro Baroni, Paul E. Dunne, and Massimiliano Giacomin. On extension counting problems in argumentation frameworks. In *Proc. COMMA 2010*, volume 216 of *FAIA*, pages 63–74. IOS Press, 2010.

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

[Baroni *et al.*, 2011b] Pietro Baroni, Paul E. Dunne, and Massimiliano Giacomin. On the resolution-based family of abstract argumentation semantics and its grounded instance. *Artificial Intelligence*, 175(3-4):791–813, 2011.

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

[Bulatov *et al.*, 2012] Andrei A. Bulatov, Víctor Dalmau, Martin Grohe, and Dániel Marx. Enumerating homomorphisms. *J. Comput. Syst. Sci.*, 78(2):638–650, 2012.

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

[Caminada *et al.*, 2012] Martin Caminada, Walter A. Carnielli, and Paul E. Dunne. Semi-stable semantics. *J. Log. Comput.*, 22(5):1207–1254, 2012.

[Cerutti *et al.*, 2014] Federico Cerutti, Massimiliano Giacomin, Mauro Vallati, and Marina Zanella. An SCC recursive meta-algorithm for computing preferred labellings in abstract argumentation. In *Proc. KR 2014*, pages 42–51, 2014.

[Coste-Marquis *et al.*, 2005] Sylvie Coste-Marquis, Caroline Devred, and Pierre Marquis. Symmetric argumentation frameworks. In *Proc. ECSQARU 2005*, volume 3571 of *LNCS*, pages 317–328. Springer, 2005.

[Courcelle, 2009] Bruno Courcelle. Linear delay enumeration and monadic second-order logic. *Discrete Applied Mathematics*, 157(12):2675–2700, 2009.

[Creignou and Hébrard, 1997] Nadia Creignou and J-J Hébrard. On generating all solutions of generalized satisfiability problems. *Informatique théorique et applications*, 31(6):499–511, 1997.

[Creignou *et al.*, 2013] Nadia Creignou, Arne Meier, Julian-Steffen Müller, Johannes Schmidt, and Heribert Vollmer. Paradigms for parameterized enumeration. In *Proc. MFCS 2013*, volume 8087 of *LNCS*, pages 290–301. Springer, 2013.

[Dimopoulos and Torres, 1996] Yannis Dimopoulos and Alberto Torres. Graph theoretical structures in logic programs and default theories. *Theor. Comput. Sci.*, 170(1-2):209–244, 1996.

[Dimopoulos *et al.*, 1997] Yannis Dimopoulos, Vangelis Magirou, and Christos H Papadimitriou. On kernels, defaults and even graphs. *Annals of Mathematics and Artificial Intelligence*, 20(1-4):1–12, 1997.

[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):321–357, 1995.

[Dunne and Bench-Capon, 2001] Paul E. Dunne and Trevor J. M. Bench-Capon. Complexity and combinatorial properties of argument systems. *Univ. Liverpool, Tech. report*, 2001.

[Dunne and Bench-Capon, 2002] Paul E. Dunne and Trevor J. M. Bench-Capon. Coherence in finite argument systems. *Artificial Intelligence*, 141(1/2):187–203, 2002.

[Dunne *et al.*, 2013] Paul E. Dunne, Wolfgang Dvořák, and Stefan Woltran. Parametric properties of ideal semantics. *Artificial Intelligence*, 202:1–28, 2013.

[Dunne *et al.*, 2015] Paul E. Dunne, Wolfgang Dvořák, Thomas Linsbichler, and Stefan Woltran. Characteristics of multiple viewpoints in abstract argumentation. *Artificial Intelligence*, 228:153–178, 2015.

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

[Durand *et al.*, 2014] Arnaud Durand, Nicole Schweikardt, and Luc Segoufin. Enumerating answers to first-order queries over databases of low degree. In *Proc. PODS 2014*, pages 121–131. ACM, 2014.

[Dvořák and Gaggl, 2016] Wolfgang Dvořák and Sarah Alice Gaggl. Stage semantics and the SCC-recursive schema for argumentation semantics. *J. Log. Comput.*, 26(4):1149–1202, 2016.

[Dvořák and Woltran, 2010] Wolfgang Dvořák and Stefan Woltran. Complexity of semi-stable and stage semantics in argumentation frameworks. *Inf. Process. Lett.*, 110(11):425–430, 2010.

[Dvořák *et al.*, 2012] Wolfgang Dvořák, Stefan Szeider, and Stefan Woltran. Abstract argumentation via monadic second order logic. In *Proc. SUM 2012*, volume 7520 of *LNCS*, pages 85–98. Springer, 2012.

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

[Dvořák, 2012] Wolfgang Dvořák. *Computational Aspects of Abstract Argumentation*. PhD thesis, Technische Universität Wien, 2012.

[Flum *et al.*, 2002] Jörg Flum, Markus Frick, and Martin Grohe. Query evaluation via tree-decompositions. *J. ACM*, 49(6):716–752, 2002.

[Gaggl and Woltran, 2013] Sarah Alice Gaggl and Stefan Woltran. The cf2 argumentation semantics revisited. *J. Log. Comput.*, 23(5):925–949, 2013.

[Goldberg, 1991] Leslie Ann Goldberg. *Efficient algorithms for listing combinatorial structures*. PhD thesis, University of Edinburgh, UK, 1991.

[Johnson *et al.*, 1988] David S. Johnson, Christos H. Papadimitriou, and Mihalis Yannakakis. On generating all maximal independent sets. *Inf. Process. Lett.*, 27(3):119–123, 1988.

[Kazana and Segoufin, 2013] Wojciech Kazana and Luc Segoufin. Enumeration of first-order queries on classes of structures with bounded expansion. In *Proc. PODS 2013*, pages 297–308. ACM, 2013.

[Kröll *et al.*, 2016] Markus Kröll, Reinhard Pichler, and Sebastian Skritek. On the complexity of enumerating the answers to well-designed pattern trees. In *Proc. ICDT 2016*, volume 48 of *LIPIcs*, pages 22:1–22:18, 2016.

[Rahwan and Simari, 2009] Iyad Rahwan and Guillermo R. Simari, editors. *Argumentation in Artificial Intelligence*. Springer, 2009.

[Strozecki, 2010] Yann Strozecki. *Enumeration complexity and matroid decomposition*. PhD thesis, Université Paris Diderot – Paris 7, 2010.

[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):102–104, April 2016.

[Verheij, 1996] Bart Verheij. Two approaches to dialectical argumentation: admissible sets and argumentation stages. In *Proc. NAIC*, pages 357–368, 1996.