Computational Models of Argument
B. Verheij et al. (Eds.)
IOS Press, 2012

doi:10.3233/978-1-61499-111-3-273

Computational Aspects of *cf2* and *stage2* Argumentation Semantics [1]

Wolfgang DVOŘÁK [a] and Sarah Alice GAGGL [b]
[a] *Research Group Theory and Applications of Algorithms, University of Vienna*
[b] *Institute of Information Systems 184, Vienna University of Technology.*

Abstract. We consider two instantiations of the SCC-recursive schema for argumentation semantics, *cf2*, using maximal conflict-free sets as base semantics, and *stage2*, using stage extensions as base semantics. Both of them have been shown to be in general of high complexity. We provide a detailed analysis of possible tractable fragments for these semantics. Moreover we present a labeling based algorithm for computing *cf2* extension, which is complexity-sensitive w.r.t. one of the tractable fragments.

Keywords. Abstract Argumentation, Computational Complexity, Algorithms.

1. Introduction

This work considers two instantiations of the SCC-recursive schema for argumentation semantics [3], *cf2* and *stage2* semantics, which are based on maximal conflict-free sets, so called naive sets. Complementing previous work [10,19,20], we address computational issues with first, a study of possible tractable fragments of the in general intractable reasoning tasks and second, a labeling based algorithm for *cf2* semantics.

Lately, the *cf2* semantics attracted specific attention, as it provides a uniform treatment of odd- and even-length cycles, and it fulfills most evaluation criteria proposed in [2]. One big disadvantage of the *cf2* semantics is that it produces questionable results on AFs with cycles of length ≥ 6 [18,20]. This is due to the fact that the base semantics of *cf2* selects only naive sets. To this end, the *stage2* semantics [10] has been introduced as a combination of the SCC- recursive schema of *cf2* semantics instantiated in the base case with stage semantics [21]. This new semantics includes the advantages of both semantics. The SCC-recursive schema of the *cf2* semantics ensures that the directionality criterion is satisfied, where the stage semantics in the base case repairs the shortcomings arising with *cf2* semantics.

The analysis of computational complexity and in particular identifying tractable cases has always been an important issue in the analysis of argumentation semantics [5,6,8,12,13,15,16] as such an analysis is indispensable for the implementation of efficient algorithms and systems. Especially, the identification of tractable fragments can help to improve the performance for easy instances of in general hard problems. How-

[1] This work has been funded by the Vienna Science and Technology Fund (WWTF) through project ICT08-028.

ever, while properties of $cf2$ and $stage2$ semantics are well-understood, a deeper complexity analysis has somehow been neglected. Only the general hardness of the main reasoning tasks was shown for $cf2$ [20] and $stage2$ [10]. To close this gap, we provide a comprehensive analysis of possible tractable fragments regarding acyclic argumentation frameworks (AFs), even cycle free AFs, bipartite AFs and symmetric AFs. Furthermore, we briefly discuss fixed-parameter tractability for $cf2$ and $stage2$ semantics.

In the second part of the article, we focus on possible implementation methods for the discussed semantics. It turned out that Logic Programming and especially Answer-Set Programming (ASP) allows for rapid prototyping of argumentation systems while providing a good run-time behavior (see [17] for a detailed description of the system ASPARTIX). Furthermore, ASP solvers are developed further continuously which has a positive influence on the performance of those systems.

On the algorithmic side, we present a labeling-based algorithm for $cf2$ semantics. In contrast to the traditional extension-based approach, so called labelings (see e.g. [1]) distinguish two kinds of unaccepted arguments, those which are rejected by the extension and those which are neither rejected nor accepted. This distinction is interesting from a logic perspective but has also proven to be useful for algorithmic issues.

The remainder of the paper is organized as follows. In Section 2 we briefly present the necessary background on abstract argumentation, argumentation semantics and computational complexity. Then, in Section 3 we provide a complexity analysis of the typical tractable fragments for abstract argumentation. Section 4 introduces labelings for $cf2$ and $stage2$ semantics and a labeling-based algorithm for $cf2$ semantics. Finally, we conclude the paper with a discussion of the obtained results.

2. Preliminaries

In this section we introduce the basics of abstract argumentation, the semantics we need for further investigations and necessary notions from complexity theory.

Abstract Argumentation. We start with a definition of abstract argumentation frameworks following [7].

Definition 1 *An* argumentation framework (AF) *is a pair $F = (A, R)$, where A is a finite set of arguments and $R \subseteq A \times A$. The pair $(a, b) \in R$ means that a attacks b. A set $S \subseteq A$* defeats *b (in F) in symbols $S \rightarrowtail b$, if $\exists a \in S$, s.t. $(a, b) \in R$. An $a \in A$ is* defended *by $S \subseteq A$ (in F) iff, $\forall b \in A$, it holds that, if $(b, a) \in R$, then S defeats b (in F). An $a \in A$ is* in conflict with *a $b \in A$, if either $(a, b) \in R$ or $(b, a) \in R$. Moreover, given an AF F, we use A_F to denote the set of it is arguments and resp. R_F to denote its attacks.*

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 AF F. The basic requirement for all semantics is that the sets are conflict-free.

Definition 2 *Let $F = (A, R)$ be an AF. A set $S \subseteq A$ is said to be* conflict-free *(in F), if there are no $a, b \in S$, such that $(a, b) \in R$. We denote the collection of sets which are conflict-free (in F) by $cf(F)$. A set $S \subseteq A$ is maximal conflict-free or* naive*, if $S \in cf(F)$ and for each $T \in cf(F)$, $S \not\subset T$. We denote the collection of all naive sets of F by $naive(F)$. For the empty AF $F_0 = (\emptyset, \emptyset)$, we set $naive(F_0) = \{\emptyset\}$.*

Towards definitions of the semantics we introduce the following formal concepts [7,21].

Definition 3 *Given an AF $F = (A, R)$ and let $S \subseteq A$. The characteristic function $\mathcal{F}_F : 2^A \to 2^A$ of F is defined as $\mathcal{F}_F(S) = \{x \in A \mid x \text{ is defended by } S\}$. We define the* range *of a set $S \subseteq A$ as $S_R^+ = S \cup \{b \mid \exists a \in S, \text{ s. t. } (a, b) \in R\}$.*

Beside the naive, *cf2* and *stage2* semantics we consider the following semantics.

Definition 4 *Let $F = (A, R)$ be an AF. A set $S \in cf(F)$ is said to be*

- *a* stable *extension (of F), i.e. $S \in stable(F)$, if $S^+ = A$;*
- *an* admissible *extension, i.e. $S \in adm(F)$ if each $a \in S$ is defended by S;*
- *the* grounded *extension (of F), i.e. the unique set $S \in grd(F)$, is the least fixed point of the characteristic function $\mathcal{F}_F$;*
- *a* stage *extension (of F), i.e. $S \in stage(F)$, if $\nexists T \in cf(F)$ with $T_R^+ \supset S_R^+$.*

The *cf2* and *stage2* semantics are based on a decomposition along the strongly connected components (SCCs) of an AF. Hence, we require some further formal machinery. By $SCCs(F)$, we denote the set of *strongly connected components* of an AF $F = (A, R)$, i.e. sets of vertices of the maximal strongly connected[2] sub-graphs of F; Moreover, for an $a \in A$, we denote by $C_F(a)$ the component of F where a occurs in, i.e. the (unique) set $C \in SCCs(F)$, such that $a \in C$. It turns out to be convenient to use two different concepts to obtain sub-frameworks of AFs. Let $F = (A, R)$ be an AF and $S \subseteq A$. Then, $F|_S = ((A \cap S), R \cap (S \times S))$ is the *sub-framework* of F wrt. S, and we also use $F - S = F|_{A \setminus S}$. We note the following relation (which we use implicitly later on), for an AF F and sets S, S': $F|_{S \setminus S'} = F|_S - S' = (F - S')|_S$. We now give the definition of the *cf2* semantics [3].

Definition 5 *Let $F = (A, R)$ be an AF and $S \subseteq A$. A $b \in A$ is* component-defeated *by S (in F), if $\exists a \in S$, s.t. $(a, b) \in R$ and $a \notin C_F(b)$. The set of arguments component-defeated by S in F is denoted by $D_F(S)$.*

Definition 6 *Let $F = (A, R)$ be an AF and $S \subseteq A$. Then, $S \in cf2(F)$, iff*

- *in case $|SCCs(F)| = 1$, then $S \in naive(F)$,*
- *else, $\forall C \in SCCs(F)$, $(S \cap C) \in cf2(F|_C - D_F(S))$.*

In words, the recursive definition $cf2(F)$ is based on a decomposition of the AF F into its $SCCs$ depending on a given set S of arguments.

Recently, a new semantics, namely $stage2$, has been defined [10]. It is a combination of the concepts of stage and *cf2* semantics, where the SCC-recursive schema of *cf2* is instantiated in the base case with stage semantics.

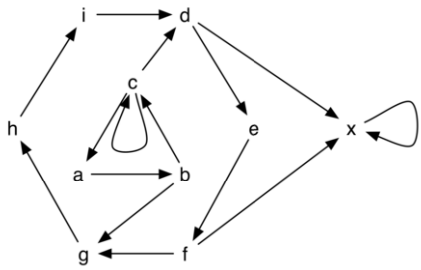
Definition 7 *Let $F = (A, R)$ be an AF and $S \subseteq A$. Then, $S \in stage2(F)$, iff*

- *in case $|SCCs(F)| = 1$, then $S \in stage(F)$,*
- *else, $\forall C \in SCCs(F)$, $(S \cap C) \in stage2(F|_C - D_F(S))$.*

We illustrate the behavior of the introduced semantics in the following example.

[2] A directed graph is called *strongly connected* if there is a directed path from each vertex in the graph to every other vertex of the graph.

Example 1 *Consider the following AF* $F = (A, R)$.



As a, b, c *form an odd cycle, none of the admissible-based semantics accept any arguments, hence* $grd(F) = adm(F) = \{\emptyset\}$*;* $stable(F) = \emptyset$*; whereas* $stage2(F) = \{\{a, d, f, h\}, \{a, e, g, i\}, \{b, d, f, h\}\}$*;* $cf2(F) = stage2(F) \cup \{\{a, f, i\}, \{a, d, g\}, \{a, e, h\}\}$*, and* $stage(F) = \{\{a, d, f, h\}, \{b, d, f, h\}\}$*;* ◊

Towards alternative characterizations of $cf2$ and $stage2$ semantics we require a parametrized notion of reachability [10,20].

Definition 8 *Let* $F = (A, R)$ *be an AF,* $B \subseteq A$*, and* $a, b \in A$*. Then,* b *is* reachable *in* F *from* a modulo B*, in symbols* $a \Rightarrow_F^B b$*, if there exists a path from* a *to* b *in* $F|_B$*, i.e. there exists a sequence* $c_1, \ldots, c_n$ $(n > 1)$ *of arguments such that* $c_1 = a$*,* $c_n = b$*, and* $(c_i, c_{i+1}) \in R \cap (B \times B)$*, for all* i *with* $1 \leq i < n$*.*

Definition 9 *For an AF* $F = (A, R)$*,* $D \subseteq A$*, and a set* S *of arguments, the operator* $\Delta_{F,S}(.)$ *is defined as* $\Delta_{F,S}(D) = \{a \in A \mid \exists b \in S : b \neq a, (b, a) \in R, a \not\Rightarrow_F^{A \setminus D} b\}$*. We denote the least fixed point of* $\Delta_{F,S}(D)$ *as* $\Delta_{F,S}$*.*

Then, $cf2$ and $stage2$ extensions can be characterized as follows.

Proposition 1 ([10,20]) *For any AF* F*,* $cf2(F) = \{S \mid S \in naive(F) \cap naive([[F - \Delta_{F,S}]])\}$ *and* $stage2(F) = \{S \mid S \in naive(F) \cap stage([[F - \Delta_{F,S}]])\}$*.*

In [10] it has been shown that for any AF F, $stable(F) \subseteq stage2(F) \subseteq cf2(F)$.

Computational Complexity. We now turn to complexity issues. We assume the reader has knowledge about standard complexity classes, i.e. P, NP, coNP and logarithmic space L. Nevertheless, we briefly recapitulate the concept of oracle machines and some related complexity classes. Let $\mathcal{C}$ notate some complexity class. By a $\mathcal{C}$-oracle machine we mean a (polynomial time) Turing machine which can access an oracle that decides a given (sub)-problem in $\mathcal{C}$ within one step. We denote the class of decision problems, that can be solved by such machines, as $\mathrm{P}^{\mathcal{C}}$ if the underlying Turing machine is deterministic and $\mathrm{NP}^{\mathcal{C}}$ if the underlying Turing machine is non-deterministic. The class $\Sigma_2^P = \mathrm{NP}^{\mathrm{NP}}$, denotes the problems which can be decided by a non-deterministic polynomial time algorithm that has access to an NP-oracle. The class $\Pi_2^P = \mathrm{coNP}^{\mathrm{NP}}$ is defined as the complementary class of Σ_2^P, i.e. $\Pi_2^P = \mathrm{co}\Sigma_2^P$. The relations between the complexity classes used in this work are $\mathrm{P} \subseteq \mathrm{NP}$ (coNP) $\subseteq \Sigma_2^P$ (Π_2^P).
We are interested in the following decision problems (for a semantics σ).

Table 1. Computational Complexity ($\mathcal{C}$-c denotes completeness for class $\mathcal{C}$).

	naive	*stable*	*stage*	*cf2*	*stage2*
$Cred_\sigma$	in L	NP-c	Σ_2^P-c	NP-c	Σ_2^P-c
$Skept_\sigma$	in L	coNP-c	Π_2^P-c	coNP-c	Π_2^P-c
Ver_σ	in L	in L	coNP-c	in P	coNP-c

- $Cred_\sigma$: Given AF $F = (A, R)$ and $a \in A$. Is a contained in *some* $S \in \sigma(F)$?
- $Skept_\sigma$: Given AF $F = (A, R)$ and $a \in A$. Is a contained in *each* $S \in \sigma(F)$?
- Ver_σ: Given AF $F = (A, R)$ and $S \subseteq A$. Is $S \in \sigma(F)$?

The complexity landscape for semantics based on maximal conflict-free sets is given in Table 1 (see [5,6,10,16,20]). The general complexity of *cf2* has been studied in [20] while the complexity of *stage2* has been studied in [10].

3. Complexity Analysis

As already mentioned, both *cf2* and *stage2* semantics are computationally intractable, i.e. the former is on the NP-layer while the latter is even on the second level of the polynomial hierarchy, naturally the issue of identifying tractable instances arises. Towards our analysis of tractable fragments we first identify a relation between credulous and skeptical acceptance. By the following result, whenever credulous acceptance is tractable we immediately get tractability for skeptical acceptance.

Proposition 2 *Given an AF $F = (A, R)$ and $a \in A$ such that $(a, a) \notin R$. Then, a is skeptically accepted with cf2 (resp. stage2) iff no $\{b \mid (b, a) \in R$ or $(a, b) \in R\}$ is credulously accepted with cf2 (resp. stage2).*

Proof. For the proof we abstract from the concrete semantics *cf2*, *stage2* and consider an arbitrary semantics σ with $\sigma(F) \subseteq naive(F)$.
$\Rightarrow$: Consider $E \in \sigma(F)$ with $a \in E$. As $E \in cf(F)$, clearly $\{b \mid (b, a) \in R$ or $(a, b) \in R\} \cap E = \emptyset$.
$\Leftarrow$: Consider $E \in \sigma(F)$ with $\{b \mid (b, a) \in R$ or $(a, b) \in R\} \cap E = \emptyset$. As $E \in naive(F)$ and $(a, a) \notin R$ we have $a \in E$. □

In the following we consider different graph classes which where proposed as tractable fragments for abstract argumentation in the literature and study the complexity of *stage2* and *cf2* semantics on these graph classes.

Acyclic Argumentation Frameworks. One tractable fragment for argumentation is the class of acyclic AFs. Tractability is due to the fact that on acyclic AFs most semantics coincide with the grounded semantics [7]. This result extends to *cf2* and *stage2*.

Theorem 1 *For acyclic AFs and $\sigma \in \{cf2, stage2\}$ the problems $Cred_\sigma$ and $Skept_\sigma$ are in P.*

Proof. We first show that, on acyclic AFs, grounded, *cf2* and *stage2* semantics coincide. Having a look at the SCC-recursive schema applied to acyclic AFs, then the base seman-

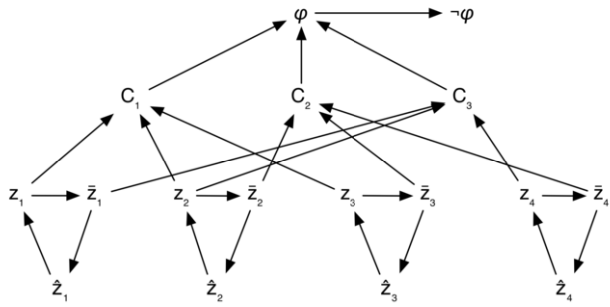


Figure 1. AF F_φ for the 3-CNF φ.

tics is only applied to AFs consisting of a single argument and no attack. Thus semantics coincide if they coincide on these AFs. We have $grd(\{a\},\emptyset) = naive(\{a\},\emptyset) = stage(\{a\},\emptyset) = \{\{a\}\}$ and thus the assertion follows. Now the complexity results are immediate by the fact that these problems are in P for grounded semantics. □

Even Cycle Free Argumentation Frameworks. By a result in [9], reasoning with admissible-based semantics in AFs without even-length cycles is tractable. Unsurprisingly this result does not extend to *cf2* and *stage2* semantics.

Theorem 2 *For AFs without even-length cycles: $Cred_{cf2}$ is NP-complete, $Skept_{cf2}$ is coNP-complete, $Cred_{stage2}$ is NP-hard, and $Skept_{stage2}$ is coNP-hard.*

Proof. The membership part for *cf2* follows immediately from the complexity results for arbitrary AFs. For the hardness part we reduce the NP-hard SAT (resp. coNP hard UNSAT) problem to *Cred* (resp. *Skept*).

Given a 3-CNF formula $\varphi = \bigwedge_{j=1}^{m} C_j$ over atoms Z with $C_j = l_{j1} \vee l_{j2} \vee l_{j3}$ $(1 \leq j \leq m)$, the corresponding AF $F_\varphi = (A_\varphi, R_\varphi)$ is built as follows:

$$A_\varphi = Z \cup \bar{Z} \cup \hat{Z} \cup \{C_1, \ldots, C_m\} \cup \{\varphi, \neg\varphi\}$$
$$R_\varphi = \{(z,\bar{z}),(\bar{z},\hat{z}),(\hat{z},z) \mid z \in Z\} \cup \{(C_j,\varphi) \mid 1 \leq j \leq m\} \cup \{(\varphi,\neg\varphi)\} \cup$$
$$\{(z,C_j) \mid j \in \{1,\ldots,m\}, z \in \{l_{j1},l_{j2},l_{j3}\}\} \cup$$
$$\{(\bar{z},C_j) \mid j \in \{1,\ldots,m\}, \neg z \in \{l_{j1},l_{j2},l_{j3}\}\}$$

Figure 1 illustrates the AF F_φ of the formula $\varphi = (z_1 \vee z_2 \vee z_3) \wedge (\neg z_2 \vee \neg z_3 \vee \neg z_4) \wedge (\neg z_1 \vee z_2 \vee z_4)$.

An SCC of F_φ either consists of a single argument or is a cycle of length three which is not attacked by another SCC. As stage and naive semantics coincide on both we have $cf2(F_\varphi) = stage2(F_\varphi)$. Thus, in the remainder of the proof we only consider *cf2* semantics. We now claim that (1) φ is satisfiable iff (2) φ is credulously accepted in F_φ iff (3) $\neg\varphi$ is not skeptically accepted in F_φ.

(1) $\Rightarrow$ (2): φ is satisfiable and thus it has a model $M \subseteq Z$. Consider now the set $E = M \cup \{\bar{z} \mid z \in Z \setminus M\} \cup \{\varphi\}$. We next show that E is a *cf2* extension of F_φ. It is easy to check that $E \in naive(F_\varphi)$. So let us consider $\Delta_{F_\varphi,E}$. As M model of φ each C_i

is either attacked by a $z_i \in E$ or $\bar{z}_i \in E$, and as there are no attacks from C_i to $Z \cup \bar{Z}$ we obtain $C_i \in \Delta_{F_\varphi,E}, 1 \leq i \leq m$. Similarly, $\neg\varphi$ is attacked by φ and as $\neg\varphi$ has no outgoing attacks also $\neg\varphi \in \Delta_{F_\varphi,E}$. Now consider $Z \cup \bar{Z} \cup \hat{Z}$. Those arguments are not attacked from outside their SCCs, hence none of the arguments is contained in $\Delta_{F_\varphi,E}$. Now consider $F' = [[F_\varphi - \Delta_{F_\varphi,E}]] = (Z \cup \bar{Z} \cup \hat{Z} \cup \{\varphi\}, \{(z,\bar{z}), (\bar{z},\hat{z}), (\hat{z},z) \mid z \in Z\})$. It is easy to see that $E \in naive(F')$ and thus we finally obtain that $E \in cf2(F_\varphi)$. Hence, φ is credulously accepted.

$(1) \Leftarrow (2)$: Let $E \in cf2(F_\varphi)$ such that $\varphi \in E$. As E is conflict-free and $\varphi \in E$ we have $C_i \notin E$ for $1 \leq i \leq m$. Moreover $C_i \in \Delta_{F_\varphi,E}$. Assume the contrary, then there exists a $C_i \in [[F_\varphi - \Delta_{F_\varphi,E}]]$ and as C_i is not strongly connected to any argument, it is an isolated argument in the separation and thus in any naive set of $[[F_\varphi - \Delta_{F_\varphi,E}]]$, a contradiction. Now as $C_i \in \Delta_{F_\varphi,E}$, for each C_i there exists $l \in Z \cup \bar{Z}$ and $l \in E$ such that l attacks C_i (which is equivalent to $l \in C_i$). Notice, as E is conflict-free it can not happen that $\{z, \bar{z}\} \subseteq E$. Finally, we obtain that $M = E \cap Z$ is a model of φ.

$(2) \Leftrightarrow (3)$: This is by the fact that in F_φ the argument $\neg\varphi$ is only connected to φ and thus each naive (resp. $cf2$) extension of F_φ either contains φ or $\neg\varphi$. □

While even cycle free AFs are tractable for admissible-based semantics, in particular for stable semantics, they are still hard for $cf2$, $stage2$ and also for stage semantics [13].

Bipartite Argumentation Frameworks. Bipartite AFs have been shown to be tractable for admissible based semantics [8]. In the following we show that they are also tractable for $cf2$ and $stage2$ semantics.

Theorem 3 *For bipartite AFs the problems $Cred_{cf2}$, $Skept_{cf2}$, Ver_{cf2} are in P.*

Proof. Given a bipartite AF (A_1, A_2, R) with $A = A_1 \cup A_2$. Start with $E_1 = A_1$ and $E_2 = \emptyset$, iterating (1) $E_2 := E_2 \cup \{b \in A_2 \mid E_1 \not\rightarrowtail b\}$ and (2) $E_1 := E_1 \setminus \{a \in E_1 \mid E_2 \rightarrowtail a\}$ until E_1, E_2 reach a fixed point. By results in [8] the above algorithm works in polynomial time and results the stable extension $E_1 \cup E_2$, with E_1 being the set of credulously accepted arguments (w.r.t. stable semantics) in A_1. We next show that this algorithm also applies to $cf2$. To this end let C_1 be the set of credulously accepted arguments in A_1 and S_2 the set of skeptically accepted arguments in A_2. We claim that after each iteration step it holds that (i) $E_1 \supseteq C_1$, (ii) $E_2 \subseteq S_2$ and (iii) $A_1 \setminus E_1 \subseteq \Delta_{F,S_2}$.

As an induction base observe that $E_1 = A_1$ and $E_2 = \emptyset$ trivially satisfies (i)-(iii). Now for the induction step assume (i)-(iii) holds before applying the iteration step, we have to show that it also holds afterwards.

First consider (ii): E_2 is only changed if there is a $b \in A_2$ and $E_1 \not\rightarrowtail b$. But by (iii) this means that for all $E \in cf2(F)$ all attackers of b are contained in $\Delta_{F,E}$. Hence, for each $E \in cf2(F)$, the argument b is isolated in the AF $[[F - \Delta_{F,E}]]$ and thus clearly $b \in E$. Hence, $b \in S_2$ and (ii) is satisfied. Now consider (i): By (ii) an argument a is only removed from E_1 if it is attacked by an skeptically accepted argument. But then a can not be credulously accepted, i.e. $a \notin C_1$, and thus still $E_1 \supseteq C_1$. Finally consider (iii): If an argument a is removed from E_1 it is attacked by an argument b such that for $E \in cf2(F)$ all attackers of b are contained in $\Delta_{F,E}$. But then clearly $a \not\rightarrowtail_F^{A \setminus \Delta_{F,E}} b$ and thus $a \in \Delta_{F,E}$. Now using that $E_1 \cup E_2$ is a stable extension, the fixed point of the above algorithm is also a $cf2$ extension. Thus, $E_1 = C_1$ and $E_2 = S_2$. By symmetry we finally obtain that in bipartite AFs, the credulously (resp. skeptically) accepted arguments w.r.t.

$cf2$ coincide with the credulously (resp. skeptically) accepted arguments w.r.t. $stable$[3]. Hence, the P results for stable semantics in [8] carry over to $cf2$ semantics. □

Even though credulous and skeptical acceptance of $cf2$ and stable semantics coincide on bipartite AFs, they propose different extensions. For instance consider a cycle of length 6 with $(\{0,1,2,3,4,5\}, \{(i, i+1 \mod 6) \mid 0 \leq i \leq 5\})$. This is certainly a bipartite AF and proposes the $cf2$ extension $\{0,3\}$ which is not stable. However, for $stage2$ and stable semantics, also the extensions coincide.

Theorem 4 *For bipartite AFs* $Cred_{stage2}$, $Skept_{stage2}$, Ver_{stage2} *are in P.*

Proof. Bipartite AFs are odd cycle free and therefore coherent [7]. Hence stable and stage semantics coincide. We show that also $stable(F) = stage2(F)$. The relation $stable(F) \subseteq stage2(F)$ holds in general [10]. Now let us consider $S \in stage2(F)$. As $S \in naive(F)$, to show $S \in stable(F)$ it suffices to show that $S^+ = A$. Clearly $\Delta_{F,S} \subseteq S^+$. [4] Now let us consider the AF $[[F - \Delta_{F,S}]]$ which is also odd cycle free (the class of bipartite AFs is closed under the deletion of arguments). Hence, the condition $S \in stage([[F - \Delta_{F,S}]])$ is equivalent to $S \in stable([[F - \Delta_{F,S}]])$ and thus $A \setminus \Delta_{F,S} \subseteq S^+$. Finally we obtain $A \subseteq S^+$ and hence $S \in stable(F)$. Now the theorem follows from $stable(F) = stage2(F)$ and the results for stable semantics in [8]. □

Symmetric AFs. Finally we consider symmetric AFs, which where studied in [5]. In symmetric AFs all SCCs are isolated in the sense that there is no attack from one SCC to another (otherwise by symmetry, there would be an attack back and thus, those SCCs would merge to just one). Hence, $cf2$ coincides with naive semantics while $stage2$ coincides with stage semantics. We immediately obtain the complexity result for $cf2$ and $stage2$ by the corresponding results for naive and stage. In the first case this clearly leads to tractability. In the latter one we have to be more careful. If we follow [5] and assume that symmetric AFs are also irreflexive then, we have tractability by the fact that such AFs are coherent and stable semantics are tractable. However, without the assumption of irreflexiveness, the tractability results for stable and stage semantics do not hold. Thus, they do not hold for $stage2$ as well.

Further Considerations & Related Work. An other interesting approach towards tractability comes from parametrized complexity theory. For so called fixed-parameter tractability (fpt), one identifies problem parameters, for instance parameters measuring the graph structure, such that computational costs heavily depend on the parameter but are only polynomial in the size of the instance. Now, if only considering problem instances with bounded parameter, one obtains a polynomial time algorithm.

First investigation for fixed-parameter tractability regarding abstract argumentation where undertaken for the graph parameters tree-width [8,12] and clique-width [15]. The work in [14] shows that also reasoning with $cf2$ semantics is fpt w.r.t. tree-width and clique-width. Moreover, using the building blocks provided there, one can easily construct a monadic second order logic encoding for $stage2$ semantics, and by the results presented in [14] this implies fpt w.r.t. tree-width and clique-width.

[3] By $stable(F) \subseteq stage2(F) \subseteq cf2(F)$ and Proposition 2 this also extends to $stage2$ semantics. However this does not cover the complexity of the Ver_{stage2} problem.

[4] In general, $S \in stable(F)$ iff $S \in naive(F) \cap stable([[F - \Delta_{F,S}]])$.

Another approach towards fpt is the so called backdoor approach, using the distance to a tractable fragment as parameter [13]. In particular it was shown that the backdoor approach does not help in the case of stage semantics and as the counter examples for stage semantics immediately carry over to $stage2$ semantics[5] there is no benefit in applying the backdoor approach to $stage2$ semantics. However, in the case of $cf2$ semantics and the tractable fragments of acyclic AFs and symmetric AFs, the backdoor approach looks promising.

4. Computing Extensions & Labelings

In this section we focus on the computation of extensions and labelings for $cf2$ and $stage2$ semantics. The advantage of using ASP for the computation of the extensions is that one can first guess all possible assignments/labelings and then check if the guesses fulfill all requirements for the specific semantics.

The ASP encodings for the $cf2$ semantics, according to the alternative characterization of Proposition 1, have been published in [19]. Due to the lack of space we only sketch how we adapted the $cf2$ encodings for $stage2$, where the modularity of ASP encodings makes this modification quite simple[6]. To be more precise, the checking module is modified from checking whether $S \in naive([[F - \Delta_{F,S}]])$, to $S \in stage([[F - \Delta_{F,S}]])$, where the ASP encodings for stage semantics can be found in [11].

Labelings for cf2 and stage2. Now, we turn to the general labeling-based approach. For an overview about labelings w.r.t. most argumentation semantics we refer to [1], where also a labeling for $cf2$ semantics is included. However, we give a slightly different definition of a $cf2$ labeling which reflects more of the intuition of $cf2$ semantics.

Definition 10 *Let $F = (A, R)$ be an AF. A* labeling *is a total function $\mathcal{L} : A \to \{in, out, undec\}$.*

Then, a labeling can be denoted as a triple $\mathcal{L} = (\mathcal{L}_{in}, \mathcal{L}_{out}, \mathcal{L}_{undec})$, where $\mathcal{L}_l = \{a \in A \mid \mathcal{L}(a) = l\}$. The following definition of a naive labeling slightly differs from the traditional definition, as there are no arguments labeled out. We need this special form of the naive labeling for the definition of the $cf2$ labeling.

Definition 11 *Let $F = (A, R)$ be an AF. Then, $\mathcal{L} \in naive_{\mathcal{L}}(F)$, iff*

- *for all $a \in \mathcal{L}_{in}$ there is no $b \in \mathcal{L}_{in}$ such that $(a, b) \in R$,*
- *$\mathcal{L}_{undec} = \{a \in A \setminus \mathcal{L}_{in}\}$ and $\mathcal{L}_{out} = \emptyset$,*
- *for all $a \in \mathcal{L}_{undec}$ there is an argument $b \in \mathcal{L}_{in}$, such that a is in conflict with b.*

Next, we define $cf2$ labelings, where an argument is labeled out iff it is attacked by an argument labeled in which does not belong to the same SCC.

Definition 12 *Let $F = (A, R)$ be an AF. Then, $\mathcal{L} \in cf2_{\mathcal{L}}(F)$, iff*

[5] Adding an argument that attacks itself and has a symmetric conflict with the original arguments does not change stage semantics, but ensures that stage semantics coincides with $stage2$ semantics. Indeed such an operation just increases the distance to a tractable fragment by one.

[6] All ASP encodings have been incorporated in the system ASPARTIX and are online available at `http://rull.dbai.tuwien.ac.at:8080/ASPARTIX/`.

- *in case* $|SCCs(F) = 1|$, *then* $\mathcal{L} \in naive_{\mathcal{L}}(F)$.
- *otherwise,* $\forall C \in SCCs(F), \mathcal{L}|_{C \setminus D_F(\mathcal{L}_{in})} \in cf2_{\mathcal{L}}(F|_C - D_F(\mathcal{L}_{in}))$, *and* $\forall a \in D_F(\mathcal{L}_{in}) \Leftrightarrow \mathcal{L}(a) = out$.

It is easy to see that there is a one-to-one mapping between $cf2$ extensions and labelings, s.t. each extension S corresponds to a labeling $\mathcal{L}$ with $\mathcal{L}_{in} = S$ and $\mathcal{L}_{out} = \Delta_{F,S}$.

To define the $stage2$ labeling, we start with the conflict-free and stage labeling according to [1].

Definition 13 *Let* $F = (A, R)$ *be an AF. Then,* $\mathcal{L}$ *is a* conflict-free labeling *of* F*, i. e.* $\mathcal{L} \in cf_{\mathcal{L}}(F)$*, iff*

- *for all* $a \in \mathcal{L}_{in}$ *there is no* $b \in \mathcal{L}_{in}$ *such that* $(a, b) \in R$,
- *for all* $a \in \mathcal{L}_{out}$ *there exists a* $b \in \mathcal{L}_{in}$ *such that* $(b, a) \in R$

Then, $\mathcal{L}$ *is a* stage labeling *of* F*, i. e.* $\mathcal{L} \in stage_{\mathcal{L}}(F)$*, iff* $\mathcal{L} \in cf_{\mathcal{L}}(F)$ *and there is no* $\mathcal{L}' \in cf_{\mathcal{L}}(F)$ *with* $\mathcal{L}'_{undec} \subset \mathcal{L}_{undec}$*. Then,* $\mathcal{L}$ *is a* $stage2$ labeling *of* F*, i. e.* $\mathcal{L} \in stage2_{\mathcal{L}}(F)$*, iff* $\mathcal{L} \in cf_{\mathcal{L}}(F) \cap stage_{\mathcal{L}}([[F - \Delta_{F,\mathcal{L}_{in}}]])$*, where* $\Delta_{F,\mathcal{L}_{in}} \subseteq \mathcal{L}_{out}$.

Again there is a one-to-one mapping between $stage2$ extensions and labelings, and each extension S corresponds to a labeling $\mathcal{L}$ with $\mathcal{L}_{in} = S$ and $\mathcal{L}_{out} = S^+ \setminus S$.

A Labeling Algorithm for $cf2$. In the following we present a labeling-based algorithm computing $cf2$-labelings/extensions. This algorithm is complexity-sensitive in the following sense. From Theorem 1 we know that on acyclic AFs, $cf2$ coincides with the grounded semantics and thus can be computed in polynomial time. To this end, the following algorithm is designed in the way that on acyclic AFs, there is no need for recursive calls. Notice that the other tractable fragments, i.e. symmetric and bipartite AFs, may propose an exponential number of extensions (the tractability for reasoning tasks was via some shortcut preventing us from computing all extensions) and thus not allow for an efficient computation of all extensions.

The following proposition identifies two rules to propagate already computed labels.

Proposition 3 *For AF* $F = (A, R)$ *and labeling* $\mathcal{L} = (\mathcal{L}_{in}, \mathcal{L}_{out}, \mathcal{L}_{undec}) \in cf2(F)$. *Let* $a \in A$*, then* $att(a) = \{b \in A \mid (b, a) \in R\}$ *denotes all attackers of* a.

1. *For every* $a \in A$*: if* $att(a) \subseteq \mathcal{L}_{out} \wedge (a, a) \notin R$ *then* $a \in \mathcal{L}_{in}$.
2. *For every* $a \in A$*: if* $\exists b \in \mathcal{L}_{in}, O \subseteq \mathcal{L}_{out} : (b, a) \in R \wedge a \not\Rightarrow_F^{A \setminus O} b$ *then* $a \in \mathcal{L}_{out}$.

Proof. (1) As mentioned above $a \in \mathcal{L}_{out}$ iff $a \in \Delta_{F,\mathcal{L}_{in}}$. If all attackers of a are in $\Delta_{F,\mathcal{L}_{in}}$ we get that $\{a\}$ is an isolated argument in $[[F - \Delta_{F,S}]]$. Now, as $\mathcal{L} \in naive([[F - \Delta_{F,S}]])$ and $(a, a) \notin R$ we finally get $a \in \mathcal{L}_{in}$. (2) Using $\exists b \in \mathcal{L}_{in}, O \subseteq \mathcal{L}_{out} : (b, a) \in R \wedge a \not\Rightarrow_F^{A \setminus O} b$ and $O \subseteq \mathcal{L}_{out} = \Delta_{F,\mathcal{L}_{in}}$, we obtain that $\exists b \in \mathcal{L}_{in} : (b, a) \in R \wedge a \not\Rightarrow_F^{A \setminus \mathcal{L}_{out}} b$. As $\Delta_{F,\mathcal{L}_{in}}$ is a fixed point we obtain that $a \in \Delta_{F,\mathcal{L}_{in}}$ and thus also $a \in \mathcal{L}_{out}$. □

Description of Algorithm 1. The $cf2$ labeling algorithm requires as input an AF $F = (A, R)$ and a labeling $\mathcal{L} = (\mathcal{L}_{in}, \mathcal{L}_{out}, \mathcal{L}_{undec})$. If $cf2_{\mathcal{L}}(F, \mathcal{L})$ is started with the initial labeling $\mathcal{L} = (\emptyset, \emptyset, A)$, it returns all $cf2$ labelings of F. At the beginning, the two sets X and Y are computed. Where X identifies those arguments in $\mathcal{L}_{undec}$ which can directly be labeled with in, and Y identifies those arguments in $\mathcal{L}_{undec}$ which can di-

Algorithm 1 $cf2_{\mathcal{L}}(F, \mathcal{L})$

Require: AF $F = (A, R)$, labeling $\mathcal{L} = (\mathcal{L}_{in}, \mathcal{L}_{out}, \mathcal{L}_{undec})$;
Ensure: Return all *cf2* labelings of F.
1: $X = \{a \in \mathcal{L}_{undec} \mid att(a) \subseteq \mathcal{L}_{out}\}$;
2: $Y = \{a \in \mathcal{L}_{undec} \mid \exists b \in \mathcal{L}_{in}, (b, a) \in R, a \not\Rightarrow_F^{A \setminus \mathcal{L}_{out}} b\}$;
3: **while** $(X \cup Y) \neq \emptyset$ **do**
4: $\mathcal{L}_{in} = \mathcal{L}_{in} \cup X, \mathcal{L}_{out} = \mathcal{L}_{out} \cup Y, \mathcal{L}_{undec} = \mathcal{L}_{undec} \setminus (X \cup Y)$;
5: update X and Y;
6: **end while**
7: $B = \{a \in \mathcal{L}_{undec} \mid \mathcal{L}_{in} \cup \{a\} \in cf(F)\}$;
8: **if** $B \neq \emptyset$ **then**
9: $C = \{a \in B \mid \nexists b \in B : b \Rightarrow_F^{A \setminus \mathcal{L}_{out}} a, a \not\Rightarrow_F^{A \setminus \mathcal{L}_{out}} b\}$;
10: $\mathcal{E} = \emptyset$;
11: **for all** $\mathcal{L}' \in naive_{\mathcal{L}}(F|_C)$ **do**
12: update $\mathcal{L}$ with $\mathcal{L}'$;
13: $\mathcal{E} = \mathcal{E} \cup cf2_{\mathcal{L}}(F, \mathcal{L})$;
14: **end for**
15: **return** $\mathcal{E}$;
16: **else**
17: **return** $\{(\mathcal{L}_{in}, \mathcal{L}_{out}, \mathcal{L}_{undec})\}$;
18: **end if**

rectly be labeled with *out* according to Proposition 3. These new labeling modifications are performed in the "while-loop" till a fixed point is reached. Next, the set B identifies all arguments which are labeled *undec* and are not in conflict with the arguments in $\mathcal{L}_{in}$. Then, if $B \neq \emptyset$, the set C identifies the next SCCs to be labeled. Note here, C does not contain all arguments of an SCC, but all arguments which can be labeled *in*. To be more precise, self-attacking arguments are omitted in C. Next, in Line 11 a separated procedure identifies all naive labelings of the sub-framework $F|_C$. For each naive labeling $\mathcal{L}'$ we update the actual labeling $\mathcal{L}$ with $\mathcal{L}'$ and call $cf2_{\mathcal{L}}(F, \mathcal{L})$ recursively. Note, this step is a branch between different *cf2*-extensions. Finally, the algorithm returns all *cf2* labelings of F.

Due to space limitations we only discuss briefly the necessary modifications of Algorithm 1 for computing *stage2* labelings. As each *stage2* extension is also a *cf2* extension we can apply Proposition 3 to *stage2* as well, but we have to take into account the different definition of $\mathcal{L}_{out}$. To be more precise, we have to modify the definition of the sets X, Y such that $\{a\} \cup \mathcal{L}_{in} \in cf(F)$. Moreover, in Line 11 we have to replace $naive_{\mathcal{L}}(F|_C)$ by $stage_{\mathcal{L}}(F|_C)$.

Finally, we highlight that although the worst case run-time of the algorithm is exponential in the size of the AF, it is polynomial if one considers both the number of extensions and the size of the AF.

5. Conclusion

We discussed two computational aspects for the *cf2* and *stage2* semantics, namely computational complexity and implementation methods. We studied the typical tractable

fragments for argumentation semantics for $cf2$ and $stage2$ semantics, where it turned out that the acyclic, bipartite and irreflexive symmetric AFs are tractable while even cycle free AFs remain hard. Furthermore, we proposed labelings for $cf2$ and $stage2$ and to complete the picture, we provided a labeling algorithm for $cf2$ semantics.

Considering other characterizations of semantics like the equational approach [18] may allow for further tractable fragments and provide different algorithms. Hence a careful comparison of the different approaches would be an interesting topic for future research. Finally let us mention that as both $cf2$ and $stage2$ semantics satisfy the directionality property [3,10] they are amenable for the splitting techniques presented in [4].

References

[1] P. Baroni, M. Caminada, and M. Giacomin. An introduction to argumentation semantics. *Knowledge Eng. Review*, 26(4):365–410, 2011.

[2] P. Baroni and M. Giacomin. On principle-based evaluation of extension-based argumentation semantics. *Artif. Intell.*, 171(10-15):675–700, 2007.

[3] P. Baroni, M. Giacomin, and G. Guida. Scc-recursiveness: A general schema for argumentation semantics. *Artif. Intell.*, 168(1-2):162–210, 2005.

[4] R. Baumann. Splitting an argumentation framework. In J. P. Delgrande and W. Faber, ed., *Proc. LPNMR 2011*, volume 6645 of *LNCS*, 40–53. Springer, 2011.

[5] S. Coste-Marquis, C. Devred, and P. Marquis. Symmetric argumentation frameworks. In L. Godo, ed., *Proc. ECSQARU 2005*, volume 3571 of *LNCS*, 317–328. Springer, 2005.

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

[7] P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artif. Intell.*, 77(2):321–358, 1995.

[8] P. E. Dunne. Computational properties of argument systems satisfying graph-theoretic constraints. *Artif. Intell.*, 171(10-15):701–729, 2007.

[9] P. E. Dunne and T.J.M. Bench-Capon. Complexity and combinatorial properties of argument systems. Technical report, Dept. of Computer Science, University of Liverpool, 2001.

[10] W. Dvořák and S. A. Gaggl. Incorporating stage semantics in the scc-recursive schema for argumentation semantics. In . *Proc. NMR 2012*, 2012.

[11] W. Dvořák, S. A. Gaggl, J. P. Wallner, and S. Woltran. Making use of advances in answer-set programming for abstract argumentation systems. *CoRR*, abs/1108.4942, 2011.

[12] W. Dvořák, R. Pichler, and S. Woltran. Towards fixed-parameter tractable algorithms for abstract argumentation. *Artificial Intelligence*, 186(0):1 – 37, 2012.

[13] W. Dvořák, S. Ordyniak and S. Szeider. Augmenting tractable fragments of abstract argumentation. *Artificial Intelligence*, 186(0):157–173, 2012.

[14] W. Dvořák, S. Szeider, and S. Woltran. Abstract Argumentation via Monadic Second Order Logic. Accepted for SUM 2012 (available as Technical Report DBAI-TR-2012-79, TU Wien)

[15] W. Dvořák, S. Szeider, and S. Woltran. Reasoning in argumentation frameworks of bounded clique-width. In P. Baroni, F. Cerutti, M. Giacomin, and G. R. Simari, ed., *Proc. COMMA 2010*, FAIA, 219–230, IOS Press, 2010.

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

[17] U. Egly, S. A. Gaggl, and S. Woltran. Answer-set programming encodings for argumentation frameworks. *In Argument and Computation*, 1(2):147–177, 2010.

[18] Dov M. Gabbay. The equational approach to cf2 semantics. *CoRR*, abs/1203.0220, 2012.

[19] S. A. Gaggl and S. Woltran. cf2 semantics revisited. In P. Baroni, F. Cerutti, M. Giacomin, and G. R. Simari, ed., *Proc. COMMA 2010*, FAIA, 243–254. IOS Press, 2010.

[20] S. A. Gaggl and S. Woltran. The cf2 argumentation semantics revisited. *Journal of Logic and Computation*, 2012, doi: 10.1093/logcom/exs011.

[21] B. Verheij. Two approaches to dialectical argumentation: admissible sets and argumentation stages. In J. Meyer and L. van der Gaag, ed., *Proc. NAIC'96*, 357–368, 1996.