dbai

TECHNICAL REPORT

INSTITUT FÜR INFORMATIONSSYSTEME

ABTEILUNG DATENBANKEN UND ARTIFICIAL INTELLIGENCE

Towards Fixed-Parameter Tractable Algorithms for Abstract Argumentation

DBAI-TR-2011-74

Wolfgang Dvořák **Reinhard Pichler**
Stefan Woltran

DBAI TECHNICAL REPORT
2011

Institut für Informationssysteme
Abteilung Datenbanken und
Artificial Intelligence
Technische Universität Wien
Favoritenstr. 9
A-1040 Vienna, Austria
Tel: +43-1-58801-18403
Fax: +43-1-58801-18493
sekret@dbai.tuwien.ac.at
www.dbai.tuwien.ac.at

TU WIEN
TECHNISCHE UNIVERSITÄT WIEN
Vienna University of Technology

DBAI TECHNICAL REPORT
DBAI TECHNICAL REPORT DBAI-TR-2011-74, 2011

Towards Fixed-Parameter Tractable Algorithms for Abstract Argumentation

Wolfgang Dvořák [1] **Reinhard Pichler** [1] **Stefan Woltran** [1]

Abstract. Abstract argumentation frameworks have received a lot of interest in recent years. Most computational problems in this area are intractable but several tractable fragments have been identified. In particular, Dunne showed that many problems can be solved in linear time for argumentation frameworks of bounded tree-width. However, these tractability results, which were obtained via Courcelle's Theorem, do not directly lead to efficient algorithms. The goal of this paper is to turn the theoretical tractability results into efficient algorithms and to explore the potential of directed notions of tree-width for defining larger tractable fragments. As a by-product, we will sharpen some known complexity results.

[1]Institute of Information Systems 184/2, Technische Universität Wien, Favoritenstrasse 9-11, 1040 Vienna, Austria. E-mail: {dvorak, pichler, woltran }@dbai.tuwien.ac.at

Acknowledgements: This work was supported by the Vienna Science and Technology Fund (WWTF) under grant ICT08-028 and by the Austrian Science Fund (FWF) under grant P20704-N18. A short version of this article appeared in the Proceedings of the 12th International Conference on Knowledge Representation and Reasoning (KR 2010), AAAI Press, 2010.

1 Introduction

Argumentation has evolved as an important field in AI with abstract argumentation frameworks (AFs, for short) as introduced by Dung [15] being its most popular formalization. Meanwhile, a wide range of semantics for AFs has been proposed (for an overview see [3]) and their complexity has been analyzed in depth. In fact, most computational problems in this area are intractable (see e.g. [12, 19, 20]), but the importance of efficient algorithms for tractable fragments has been clearly recognized (see e.g. [13]). Such tractable fragments are, for instance, symmetric argumentation frameworks [8] or bipartite argumentation frameworks [17].

An interesting approach to dealing with intractable problems comes from parameterized complexity theory and is based on the following observation: Many hard problems become tractable if some problem parameter is bounded by a fixed constant. This property is referred to as *fixed-parameter tractability* (FPT). One important parameter of graphs is the tree-width, which measures the "tree-likeness" of a graph. Indeed, Dunne [17] showed that many problems in the area of argumentation can be solved in linear time for argumentation frameworks of bounded tree-width. This FPT-result was shown via a seminal result by Courcelle [9]. However, as stated in [17], "rather than synthesizing methods indirectly from Courcelle's Theorem, one could attempt to develop practical *direct* methods". The primary goal of this paper is therefore to present new, direct algorithms for certain reasoning tasks in abstract argumentation.

Clearly, the quest for FPT-results in argumentation should not stop at the tree-width, and further parameters have to be analyzed. This may of course also lead to negative results. For instance, considering as parameter the degree of an argument (i.e., the number of incoming and outgoing attacks), Dunne [17] showed that reasoning remains intractable, even if decision problems are given over AFs with at most two incoming and two outgoing attacks. A number of further parameters is however, still unexplored. Hence, the second major goal of this paper is to explore the potential of further parameters for identifying tractable fragments of argumentation. In particular, since AFs are directed graphs, it is natural to consider directed notions of width to obtain larger classes of tractable AFs. To this end, we investigate the effect of bounded cycle-rank [22] on reasoning in AFs. We show that reasoning remains intractable even if we only consider AFs of cycle-rank 2. Actually, many further directed notions of width exist in the literature. However, it has been recently shown [4, 25, 24] that problems which are hard for bounded cycle-rank remain hard when several other directed variants of the tree-width are bounded. A notable exception is the related notion of clique-width [10] which (in contrast to tree-width) can be directly extended to directed graphs. Moreover, meta-theorems for clique-width [11] show that Dunne's result on tractability with respect to bounded tree-width extend to AFs of bounded clique-width (for details, we refer to [21]).

Still, the main focus of this paper is on novel algorithms for decision problems defined over the so-called preferred semantics of AFs. Roughly speaking, the preferred extensions of an AF are maximal admissible sets of arguments, where admissible means that the selected arguments defend themselves against attacks. To be more precise, we present here algorithms for the following three decision problems.

- Credulous Acceptance: deciding whether a given argument is contained in at least one pre-

ferred extension of a given AF.

- Skeptical Acceptance: deciding whether a given argument is contained in all preferred extensions of a given AF.
- Ideal Acceptance: deciding whether a given argument is contained in an admissible set which itself is a subset of each preferred extension of a given AF.

The problem of ideal acceptance is better known as ideal semantics [16]. To the best of our knowledge, FPT results for ideal semantics have not been established yet, thus the algorithm that we present in the paper provides such a result as a by-product (one could alternatively use Courcelle's meta-theorem to obtain that result). By its very nature, the running times of our novel algorithms will heavily depend on the tree-width of the given AF, but are linear in the size of the AF. Thus for AFs of small tree-width, these algorithms are expected to be preferable over standard algorithms from the literature (see e.g. [14, 29]).

One reason why we have chosen the preferred semantics for our work here is that it is widely used. Moreover, admissibility and maximality are prototypical properties common in many other semantics, for instance complete and stable [15], stage [32], and semi-stable [6] semantics. Hence, we expect that the methods developed here can also be extended to other semantics.

Summary of results

- We first prove some negative results: we show that reasoning remains intractable in AFs of bounded cycle-rank [22]. As has been mentioned above, this negative result carries over to many other directed notions of width. We also show that the problem of skeptical acceptance is coNP-complete for AFs of cycle-rank 1.
- We develop a dynamic programming approach to characterize admissible sets of AFs. The time complexity of our algorithm is linear in the size of the AFs (as expected by Courcelle's Theorem) with a multiplicative constant that is *single* exponential in the tree-width (which is in great contrast to algorithms derived via Courcelle's Theorem). This algorithm can be directly used to decide the problem of credulous acceptance.
- This dynamic programming algorithm is then extended so as to cover also the preferred semantics, and thus to decide skeptical acceptance.
- We finally show how to further adapt this algorithm to decide ideal acceptance.

Structure of the paper In Section 2, we recall some basic notions and results on AFs and discuss some width-measures for graphs. We then show in Section 3 some negative results for reasoning in AFs where some parameters of directed graphs are bounded. In Section 4.1, we first develop a dynamic programming approach for credulous acceptance in AFs of bounded tree-width. This algorithm is then extended to cover also preferred semantics in Section 4.2 and adapted to ideal acceptance in Section 4.3. Section 5 provides some final conclusions as well as pointers to related and future work.

2 Background

In this section, we first introduce argumentation frameworks and then some graph measures we want to investigate for such frameworks.

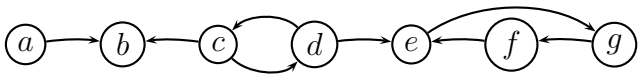
2.1 Argumentation Frameworks

We start by introducing (abstract) argumentation frameworks [15], and then recall the preferred as well as the ideal semantics for such frameworks. Afterwards, we highlight some known complexity results for typical decision problems associated to such frameworks.

Definition 1. *An* argumentation framework (AF) *is a pair $F = (A, R)$ where A is a set of arguments and $R \subseteq A \times A$ is the attack relation. We sometimes use the notation $a \rightarrowtail b$ instead of $(a, b) \in R$, in case no ambiguity arises. Further, for $S \subseteq A$ and $a \in A$, we write $S \rightarrowtail a$ (resp. $a \rightarrowtail S$) iff there exists $b \in S$, such that $b \rightarrowtail a$ (resp. $a \rightarrowtail b$). An argument $a \in A$ is* defended *by a set $S \subseteq A$ iff for each $b \in A$, such that $b \rightarrowtail a$, also $S \rightarrowtail b$ holds.*

An AF can naturally be represented as a directed graph.

Example 1. Let $F = (A, R)$ with $A = \{a, b, c, d, e, f, g\}$ and $R = \{(a, b), (c, b), (c, d), (d, c), (d, e), (e, g), (f, e), (g, f)\}$. The graph representation of F is given as follows.



We continue with a few basic concepts and the definition of preferred extensions as introduced in Dung's seminal paper [15] as well as the concept of ideal sets as proposed by Dung, Mancarella and Toni [16].

Definition 2. *Let $F = (A, R)$ be an AF. A set $S \subseteq A$ is* conflict-free (in F)*, iff there are no $a, b \in S$, such that $(a, b) \in R$. A set $S \subseteq A$ is* admissible *for F, if S is conflict-free in F and each $a \in S$ is defended by S in F. We denote the collection of all admissible extensions of F by $adm(F)$.*

Definition 3. *Let $F = (A, R)$ be an AF. A set $S \subseteq A$ is a* preferred extension *of F, iff S is a maximal (wrt. subset inclusion) admissible set for F. We denote the collection of all preferred extensions of F by $pref(F)$.*

Definition 4. *Let $F = (A, R)$ be an AF. A set $S \subseteq A$ is called* ideal *for F, if $S \in adm(F)$ and S is a subset of all preferred extensions (i.e., $S \subseteq \bigcap_{P \in pref(F)} P$). We denote the collection of all ideal sets of F by $ideal(F)$.*

An admissible set S is called complete, if each argument defended by S is contained in S. It was shown in [16] that each AF F possesses a unique maximal ideal set (called the *ideal extension* of F) and that this set is also a complete extension of F.

Example 2. For the AF F in Example 1, we get as admissible sets $\{\}$, $\{a\}$, $\{c\}$, $\{d\}$, $\{d, g\}$, $\{a, c\}$, $\{a, d\}$, and $\{a, d, g\}$. Consequently, $pref(F) = \{\{a, c\}, \{a, d, g\}\}$, and moreover, $ideal(F) = \{\{\}, \{a\}\}$. Thus, $\{a\}$ is the ideal extension of F. $\diamond$

Next, we recall the complexity of reasoning over preferred and ideal extensions. To this end, we define the decision problems of credulous acceptance (CA), skeptical acceptance (SA) and ideal acceptance (ID) which have as input an AF $F = (A, R)$ and an argument $a \in A$:

- CA: Is a contained in some $S \in pref(F)$?
- SA: Is a contained in each $S \in pref(F)$?
- ID: Is a contained in some $S \in ideal(F)$?

Note that the problem ID is equivalent to deciding whether a is contained in the ideal extension of F.

It is known that CA is NP-complete, while SA is Π_2^P-complete (see [12, 19]). The reason why CA is located on a lower level of the polynomial hierarchy compared to SA, is the fact that it is sufficient to check whether a is contained in at least one admissible set for the given AF F. Then a is also contained in a preferred extension of F. In other words, the maximality requirement of preferred extensions does not come into play for CA. For SA, the situation is different, and maximality has to be taken into account, leading to an additional source of complexity. The exact complexity of ID is still an open problem but for the lower bound it is known that ID is coNP-hard and as an upper bound membership in Θ_2^P has been shown (see [18]). Hence, under usual complexity-theoretic assumptions SA is harder to decide than CA and ID. Moreover, the analysis in [18] suggests that ID might be mildly harder than CA. As we will see later, these theoretical observations are to some extent mirrored by the running-times of our algorithms.

2.2 Parameters for Graphs

We review several notions of parameters for graphs (both directed and undirected). One of the most important concepts for fixed-parameter tractability on graphs is the tree-width, which was introduced by Robertson and Seymour [30].

To start with, we recall the concept of an induced subgraph: given a graph $G = (V, E)$ and a set A, we write $G|_A = (V \cap A, E \cap (A \times A))$ for the subgraph of G induced by A.

Definition 5. *Let $G = (V, E)$ be an undirected graph. A* tree decomposition *of G is a pair $(\mathcal{T}, \mathcal{X})$ where $\mathcal{T} = (V_\mathcal{T}, E_\mathcal{T})$ is a tree and $\mathcal{X} = (X_t)_{t \in V_\mathcal{T}}$ is a set of so-called bags, which has to satisfy the following conditions:*

1. *$\bigcup_{t \in V_\mathcal{T}} X_t = V$, i.e. $\mathcal{X}$ is a cover of V,*
2. *for each $v \in V$, $\mathcal{T}|_{\{t | v \in X_t\}}$ is connected,*
3. *for each $\{v_i, v_j\} \in E$, $\{v_i, v_j\} \subseteq X_t$ for some $t \in V_\mathcal{T}$.*

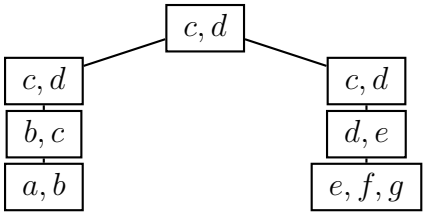


Figure 1: A tree decomposition of the graph in Example 1.

The width of such a tree decomposition is given by $\max\{card(X_t) \mid t \in V_{\mathcal{T}}\} - 1$. *The* tree-width *of a graph* G *is the minimum width over all tree decompositions of* G.

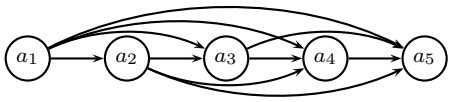
It was shown by Bodlaender [5] that, for fixed $w \geq 1$, it can be decided in linear time whether a graph has tree-width at most w. Moreover, in case of a positive answer, a tree decomposition of width w can be computed in linear time. Figure 1 shows a tree decomposition of width 2 for the AF from Example 1 (when considered as an undirected graph).

Many NP-hard problems on graphs have been shown to be linear time computable on graphs of bounded tree-width. In particular, Courcelle's Theorem [9] provides a powerful tool to obtain such results. It states that any property over graphs which can be expressed in Monadic Second-Order Logic, can be decided in linear time (wrt. to the size of the graph) for graphs which have bounded tree-width. Dunne [17] used this result to show fixed-parameter tractability of the problems CA and SA for the parameter tree-width.

However, there is a certain problem when using tree-width in the area of directed graphs. In fact, there are many digraphs which we intuitively consider as simply structured but already have high tree-width. As an example consider the acyclic digraphs of the form ($n \geq 1$)

$$G_n = \big(\{a_1, \ldots, a_n\}, \{(a_i, a_j) \mid 1 \leq i < j \leq n\}\big). \tag{1}$$

For $n = 5$, G_n looks as follows



Seen as undirected graph, each G_n turns into a clique of size n. Thus, the tree-width of the graphs G_n (with increasing n) cannot be bounded by a constant.

As AFs are directed graphs, it seems natural to consider parameters exclusively defined for digraphs. Indeed, many such measures exist like directed tree-width [27], DAG-width [4] or Kelly-width [25]. An old but particularly interesting parameter, which we shall focus on here, is cycle-rank [22]. One reason why there are many different such notions is due to the fact that, so far, no analogue to Courcelle's Theorem which is comparably general has been found for digraph problems.[1]

[1] As mentioned in the introduction, (directed) clique-width is a notable exception; we again refer to [21] for a more detailed discussion.

Before giving the definition of cycle-rank, we recall some basic definitions: we call a graph *acyclic*, if it does not contain a cycle going through distinct vertices. In other words, self-loops are not considered as cycles. A directed graph is *strongly connected*, if each vertex is reachable from any other vertex in this graph. Finally, a *strongly connected component (SCC)* of a graph G is an induced subgraph $G|_S$ of G such that S is maximal with the property that $G|_S$ is strongly connected.

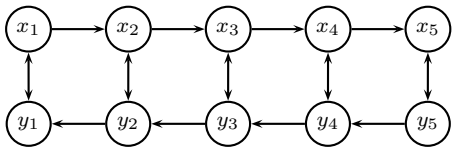
Definition 6. *Let $G = (V, E)$ be a directed graph. The cycle-rank $r(.)$ of G is defined as follows: an acyclic graph has $r(G) = 0$; if G is strongly connected then $r(G) = 1 + min_{v \in V} r(G|_{V \setminus \{v\}})$. If G is not strongly connected, then $r(G)$ is the maximum cycle-rank among all SCCs of G.*

Intuitively, the cycle-rank corresponds to the maximum recursion depth of a procedure which – in each call – eliminates one node per SCC until we have an acyclic graph. Note that the graphs G_n of the form (1) are acyclic and, thus, have cycle-rank 0 for any n.

The cycle-rank is of particular interest because recent results [4, 24, 25] showed that problems which are hard for bounded cycle-rank also remain hard when some of the other aforementioned parameters are bounded. Indeed, in Section 3 we shall prove several intractability results for AFs with bounded cycle-rank. These intractability results thus immediately carry over to the other parameters for directed graphs.

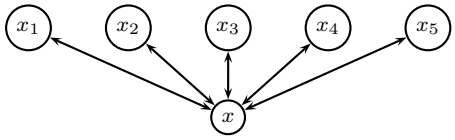
For a similar intractability result, Dunne [17] has recently shown that CA and SA remain intractable for AFs with bounded in- and out-degree. The following example illustrates that the class of graphs with bounded cycle-rank is incomparable with the class of graphs with bounded in- and out-degree.

Example 3. Let $(H_n = (V_n, E_n))_{n \geq 1}$ be a family of directed graphs with $V_n = \{x_1, \ldots, x_n, y_1, \ldots, y_n\}$ and $E_n = \{(x_i, y_i), (y_i, x_i) \mid 1 \leq i \leq n\} \cup \{(x_i, x_{i+1}), (y_{i+1}, y_i) \mid 1 \leq i \leq n-1\}$. As an example, the graph H_5 looks as follows



It is easy to see that the in- and out-degrees of these graphs are bounded by 2, but that these graphs are of arbitrary cycle-rank.

As another example, let $(I_n = (V_n, E_n))_{n \geq 1}$ be the family of directed graphs with $V_n = \{x_1, \ldots, x_n, x\}$ and $E_n = \{(x, x_i), (x_i, x) \mid 1 \leq i \leq n\}$. The graph I_5 looks as follows



Each graph I_n has cycle-rank 1, but there are graphs of form I_n which have arbitrary in- and out-degree. ◇

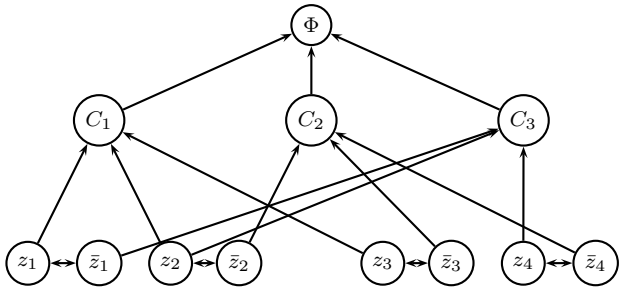


Figure 2: AF F_Φ for CNF formula Φ in Example 4.

3 Parameters for Directed Graphs – Negative Results

3.1 Bounded Cycle-Rank

We continue to prove that NP-hardness for CA holds, even if we restrict ourselves to AFs with bounded cycle-rank. We employ the reduction from [12] which maps each instance (i.e. a CNF formula) of the NP-hard problem SAT to an argumentation framework.

Definition 7. *Given a CNF formula $\Phi = \bigwedge_{j=1}^{m} C_j$ with C_j being clauses over variables Z, define $F_\Phi = (A, R)$ with*

$$\begin{aligned} A &= \{\Phi, C_1, \ldots, C_m\} \cup Z \cup \bar{Z} \\ R &= \{(C_j, \Phi) \mid 1 \leq j \leq m\} \cup \\ &\quad \{(z, \bar{z}), (\bar{z}, z) \mid z \in Z\} \cup \\ &\quad \{(z, C_j) \mid z \text{ occurs in } C_j, 1 \leq j \leq m\} \cup \\ &\quad \{(\bar{z}, C_j) \mid \neg z \text{ occurs in } C_j, 1 \leq j \leq m\} \end{aligned}$$

where $\bar{Z} = \{\bar{z} \mid z \in Z\}$ is a set of fresh arguments.

Example 4. Consider the CNF formula

$$\Phi = (z_1 \vee z_2 \vee z_3) \wedge (\neg z_2 \vee \neg z_3 \vee \neg z_4) \wedge (\neg z_1 \vee z_2 \vee z_4).$$

Figure 2 illustrates the corresponding AF F_Φ. ◇

For any CNF formula Φ, F_Φ can be constructed in polynomial time, and Φ is satisfiable iff argument Φ is credulously accepted in F_Φ. This gives the NP-hardness for CA, first shown by Dimopoulos and Torres [12] and later rephrased in terms of AFs by Dunne and Bench-Capon [19]. We strengthen this result as follows.

Theorem 1. CA *is* NP*-hard, even if the problem is restricted to AFs of cycle-rank* 1.

Proof. As discussed above, AFs F of the form given in Definition 7 provide us with a valid reduction from SAT to CA. To prove the assertion it is thus sufficient to show that for each CNF formula

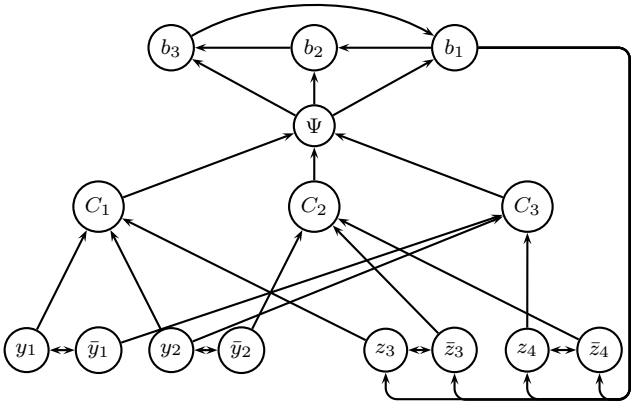


Figure 3: AF G_Ψ for QBF Ψ in Example 5.

Φ, the corresponding AF F has at most cycle-rank 1. Indeed, such an AF F has the following SCCs: $F|_{\{z,\bar{z}\}}$ for each $z \in Z$ and the singletons $C_1, \ldots, C_m$, and Φ. Obviously, components $F|_{\{z,\bar{z}\}}$ have cycle-rank 1 and all other components have cycle-rank 0. Hence, each F constructed following Definition 7 has cycle-rank 1. □

We now turn our attention to the Π_2^P-hard problem SA. The following reduction from QBFs to AFs is used in [19].

Definition 8. *Given a QBF $\Psi = \forall Y \exists Z \bigwedge_{j=1}^{m} C_j$ over variables $X = Y \cup Z$. We define the AF $G_\Psi = (A, R)$ with*

$$\begin{aligned} A &= \{\Psi, C_1, \ldots, C_m\} \cup X \cup \bar{X} \cup \{b_1, b_2, b_3\} \\ R &= \{(C_j, \Psi) \mid 1 \leq j \leq m\} \cup \\ &\quad \{(x, \bar{x}), (\bar{x}, x) \mid x \in X\} \cup \\ &\quad \{(x, C_j) \mid x \text{ occurs in } C_j, 1 \leq j \leq m\} \cup \\ &\quad \{(\bar{x}, C_j) \mid \neg x \text{ occurs in } C_j, 1 \leq j \leq m\} \cup \\ &\quad \{(\Psi, b_1), (\Psi, b_2), (\Psi, b_3)\} \cup \\ &\quad \{(b_1, b_2), (b_2, b_3), (b_3, b_1)\} \cup \\ &\quad \{(b_1, z), (b_1, \bar{z}) \mid z \in Z\} \end{aligned}$$

where $\bar{X} = \{\bar{x} \mid x \in X\}$ is a set of fresh arguments.

Example 5. Consider the QBF

$$\Psi = \forall y_1 y_2 \exists z_3 z_4 (y_1 \vee y_2 \vee z_3) \wedge (\neg y_2 \vee \neg z_3 \vee \neg z_4) \wedge (\neg y_1 \vee y_2 \vee z_4).$$

In Figure 3, we depict the corresponding AF G_Ψ. ◇

As shown by Dunne and Bench-Capon [19], the following holds for each QBF Ψ of the above form: Ψ is valid iff argument Ψ is contained in each $S \in \mathit{pref}(G_\Psi)$. Since G_Ψ can be constructed from Ψ in polynomial time, this showed Π_2^P-hardness of the problem SA. We strengthen this result as follows.

Theorem 2. SA *is* Π_2^P*-hard, even if the problem is restricted to AFs of cycle-rank* 2*.*

Proof. We can proceed similarly as in the proof of Theorem 1. Moreover, we are allowed to restrict ourselves to QBFs Φ of the form $\forall Y \exists Z \bigwedge_{j=1}^{m} C_j$ where each C_j contains at least one occurrence of an atom from Z; the validity problem for such QBFs obviously remains Π_2^P-hard. Each AF G according to Definition 8 has the following SCCs:

- $G|_{\{y,\bar{y}\}}$ for each $y \in Y$;
- $G|_S$ for $S = \{z, \bar{z} \mid z \in Z\} \cup \{C_1, \ldots, C_m, \Phi, b_1, b_2, b_3\}$.

Components $G|_{\{y,\bar{y}\}}$ have cycle-rank 1, and $H = G|_S$ has cycle-rank 2. This can be seen as follows: Removing Φ leads to SCCs $H|_{\{z,\bar{z}\}}$ (for each $z \in Z$), $H|_{\{b_1,b_2,b_3\}}$, and singletons C_1 ,..., C_m. All these have cycle-rank 1 or 0. □

We now turn our attention to the coNP-hard problem ID. The following reduction from CNF formulas to AFs is a slightly modified version of that given by Dunne [18].

Definition 9. *Given a formula* $\Phi = \bigwedge_{j=1}^{m} C_j$ *over variables* Z*. We define the AF* $H_\Phi = (A, R)$ *with*

$$\begin{aligned} A &= \{\Phi, C_1, \ldots, C_m\} \cup Z \cup \bar{Z} \cup \{\Psi\} \\ R &= \{(C_j, \Phi) \mid 1 \leq j \leq m\} \cup \\ &\quad \{(z, \bar{z}), (\bar{z}, z) \mid z \in Z\} \cup \\ &\quad \{(z, C_j) \mid z \text{ occurs in } C_j, 1 \leq j \leq m\} \cup \\ &\quad \{(\bar{z}, C_j) \mid \neg z \text{ occurs in } C_j, 1 \leq j \leq m\} \cup \\ &\quad \{(\Psi, \Phi), (\Phi, \Psi)\} \end{aligned}$$

where $\bar{Z} = \{\bar{Z} \mid z \in Z\}$ *is a set of fresh arguments.*

Example 6. Recall the CNF formula Φ from Example 4, i.e.

$$\Phi = (z_1 \vee z_2 \vee z_3) \wedge (\neg z_2 \vee \neg z_3 \vee \neg z_4) \wedge (\neg z_1 \vee z_2 \vee z_4),$$

Figure 4 illustrates the corresponding AF H_Φ. ◇

As shown by Dunne [18], the following holds for each formula Φ of the above form: Φ is unsatisfiable iff the argument Ψ is contained in the ideal extension[2]. Since H_Φ can be constructed from Φ in polynomial time, coNP-hardness of the problem ID follows. We strengthen this result as follows.

Theorem 3. ID *is* coNP*-hard, even if the problem is restricted to AFs of cycle-rank* 1*.*

[2]We note that the proof given by Dunne also works for our slightly modified reduction.

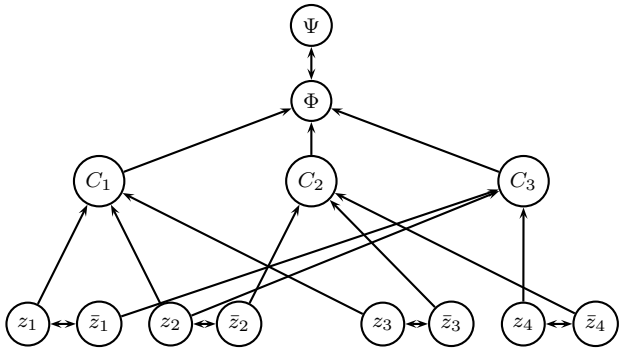


Figure 4: AF H_Φ for CNF formula Φ in Example 6.

Proof. We can proceed similar as in the proofs of Theorems 1 and 2. Let H be an arbitrary AF which follows Definition 9. Then H has the following SCCs: $H|_{\{z,\bar{z}\}}$ for each $z \in Z$, the singletons $C_1, \ldots, C_m$, and $H|_{\{\Phi,\Psi\}}$. Each of these components either has cycle-rank 1 or cycle-rank 0 and thus each H constructed following Definition 9 has cycle-rank 1. □

Theorems 1–3 show that the parameter cycle-rank is not applicable for fixed-parameter tractability of the considered problems. However, these theorems leave some room for potential tractable fragments. First, consider the class of AFs of cycle-rank 0. By definition this is the class of acyclic AFs and it is well known that the acceptance problems under consideration are tractable for acyclic AFs (there is a single preferred extension which coincides with the so-called *grounded extension* – for details see [15]). It thus remains to classify the complexity of skeptical acceptance for AFs of cycle-rank 1. Using the reduction to the AF H_Φ from Definition 9, we immediately get that this problem is still coNP-hard, i.e. it does not form a tractable fragment in the usual sense. Nevertheless, we next show that also coNP membership holds for skeptical acceptance when restricted to AFs of cycle-rank 1. Hence, this fragment turns out to be computationally easier, bearing in mind that SA is Π_2^P-complete in general.

Theorem 4. SA *is in* coNP *for AFs of cycle-rank* 1.

Proof. To prove SA $\in$ coNP we provide a polynomial-time algorithm for verifying that a given set is a preferred extension. Then one can build a coNP-algorithm for SA by deciding its complement by a standard guess and check approach. To verify whether a set E is a preferred extension of an AF $F = (A, R)$ we first compute the SCCs and build a linear order $S_1, \ldots S_m$ of the SCCs which respects the partial order given by the attacks between different components, i.e. for $i < j$ we have that $S_j \not\rightarrow S_i$. Note that both the identification of SCCs and obtaining such a linear order can be done in polynomial time by depth-first search. Now one can decide the verification problem by considering each SCC separately starting with S_1 and then following the linear ordering. Therefore, we use a multi-labeling $\mathcal{M} : V_S \rightarrow 2^{\{in,def,undec\}}$ which maps vertices to sets of labels, as well as ordinary labelings $\mathcal{L} : V_S \rightarrow \{in, def, undec\}$ (see [7]). Intuitively such a labeling corresponds to an extension in the following way: an argument is labeled in if it is in the extension. An argument is labeled def if it is not in the extension and attacked by some argument in the extension.

Intuitively, the label def indicates that the extension is "defended" against potential attacks from this argument. Finally, an argument is labeled $undec$ if it is neither in the extension nor attacked by an argument in the extension. Intuitively, the label $undec$ indicates that the status of this argument is in a sense "undecided" yet.

The multi-labeling will be used as a certain form of initialization of the currently considered SCC S_j (for $j > 1$ this might take results from SCCs S_i with $i < j$ into account); ordinary labelings are then obtained from $\mathcal{M}$ by taking a designated argument as a starting point and are finally compared to the candidate E.

The verification algorithm (see also Example 7 below for illustration) for a given AF $F = (A, R)$ with linearly ordered SCCs $S_1, \ldots, S_m$ and a set of arguments E is as follows and loops over j with $1 \leq j \leq m$.

1. First, initialize a multi-labeling $\mathcal{M}_j$ with $\mathcal{M}_j(a) = \{in, def, undec\}$, for all vertices a in S_j. For each attack (a, b) in F with $a \in S_i$, $b \in S_j$ and $i < j$, we set

$$\begin{array}{ll} \mathcal{M}_j(b) := \mathcal{M}_j(b) \setminus \{in, undec\} & \text{if } a \in E \\ \mathcal{M}_j(b) := \mathcal{M}_j(b) \setminus \{in\} & \text{if } a \notin E \wedge E \not\rightarrowtail a \end{array}$$

2. Identify an argument $x \in S_j$ such that $S_j \setminus \{x\}$ is acyclic.

3. Compute a labeling $\mathcal{L}_j^l$ for each label $l \in \mathcal{M}_j(x)$ as follows: $\mathcal{L}_j^l(x) = l$ and for all vertices $a \neq x$ in S_j:

$$\mathcal{L}_j^l(a) = \begin{cases} in & \text{if } in \in \mathcal{M}_j(a) \ \wedge \ \forall b \in S_j : b \rightarrowtail a \Rightarrow \mathcal{L}_j^l(b) = def \\ def & \text{if } \mathcal{M}_j(a) = \{def\} \text{ or } \exists b \in S_j : \mathcal{L}_j^l(b) = in \ \wedge \ b \rightarrowtail a \\ undec & \text{otherwise} \end{cases}$$

4. Verify the status of the selected argument x in labelings $\mathcal{L}_j^l$:

   - $\mathcal{L}_j^{in}$ is valid iff $\forall b \in S_j : b \rightarrowtail x \Rightarrow \mathcal{L}_j^l(b) = def$
   - $\mathcal{L}_j^{def}$ is valid iff $\mathcal{M}_j(x) = \{def\}$ or $\exists b \in S_j : \mathcal{L}_j^{def}(b) = in \ \wedge \ b \rightarrowtail x$
   - $\mathcal{L}_j^{undec}$ is valid anyway

   Let L_j be the set of valid labelings for S_j.

5. Define

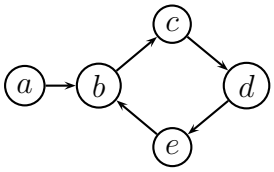
$$L_j^* = \begin{cases} L_j & \text{if } L_j = \{\mathcal{L}_j^{undec}\} \\ L_j \setminus \{\mathcal{L}_j^{undec}\} & \text{otherwise} \end{cases}$$

6. Verification: Reject, if there is no $\mathcal{L} \in L_j^*$ such that for all vertices a in S_j it holds that $\mathcal{L}(a) = in$ iff $a \in E$; otherwise continue with the next SCC.

If the above algorithm terminates without rejecting E, then E is a preferred extension.

A few words about the correctness of this algorithm are in order. In the first step, we simply initialize the multi-labeling. If arguments from the current SCC S_j are attacked from a different SCC S_i, note that than $i < j$ holds and we incorporate the effects of these attacks with respect to the candidate E (which has already been verified at least "up to" S_i). In the next step we use the fact that, by definition, $cr(F) \leq 1$ iff for each SCC $S = (A_S, R_S)$ of F, there is an argument $x \in A_S$, such that $S|_{A_S \setminus \{x\}}$ is acyclic. We note that this can be easily done in polynomial time. In the third step we compute all possible labelings for the current SCC which can be obtained from the multi-labeling (thus respecting attacks from outside). Given the label of the selected argument x, note that we can compute the labels of all other arguments in S_j by a finite recursion (due to the fact that the SCC without x is acyclic). In the next step we verify the computed labelings with respect to x. For the case where we labeled x with in, we have to check whether x is defended with respect to labelling $\mathcal{L}_j^{in}$. Similar, for $l = def$, we have to check whether x is attacked. In the fifth step, we have to take into account that a labelling $\mathcal{L}_j^{undec}$, i.e. where x is labeled $undec$, only corresponds to a (part of the) preferred extension if the other labelings $\mathcal{L}_j^{in}$ and $\mathcal{L}_j^{def}$ are both not valid. Finally a preferred extension for the whole framework has to coincide with a valid labeling for the current SCC. If this is not the case, we stop the loop and reject. □

Example 7. For illustration of this algorithm consider the AF $F = (\{a, b, c, d, e\}, \{(a, b), (b, c), (c, d), (d, e), (e, b)\})$ and the set $E = \{a, c\}$.



We have two SCCs $S_1 = F|_{\{a\}}$ and $S_2 = F|_{\{b,c,d,e\}}$. First we apply our algorithm to S_1. Since S_1 is an initial SCC, its multi-labeling is given by $\mathcal{M}_1(a) = \{in, def, undec\}$. S_1 has only one argument, we thus select $x = a$ in Step 2 and get the following three labelings $\mathcal{L}_1^{in}(a) = in$, $\mathcal{L}_1^{def}(a) = def$ and $\mathcal{L}_1^{undec}(a) = in$ in Step 3. As there is no argument attacking a, $\mathcal{L}_1^{def}(a)$ is not valid (Step 4). In Step 5, we obtain $L_1^* = \{\mathcal{L}_1^{in}, \mathcal{L}_1^{undec}\} \setminus \{\mathcal{L}_1^{undec}\} = \{\mathcal{L}_1^{in}\}$. As $a \in E$ and $\mathcal{L}_1^{in}(a) = in$, we now have that E is valid on S_1 and we thus continue the algorithm with SCC S_2.

For the multi-labeling $\mathcal{M}_2$ we have that $\mathcal{M}_2(c) = \mathcal{M}_2(d) = \mathcal{M}_2(e) = \{in, def, undec\}$ and $\mathcal{M}_2(b) = \{def\}$. The latter equality holds because $a \in E$ and $a \rightarrowtail b$. In the next step we have four options for argument x to make S_2 acyclic. Let us consider $x = d$. We compute three labelings $\mathcal{L}_2^{in}$, $\mathcal{L}_2^{def}$ and $\mathcal{L}_2^{undec}$. They are given as follows:

$$\begin{array}{cccc} \mathcal{L}_2^{in}(b) = def & \mathcal{L}_2^{in}(c) = in & \mathcal{L}_2^{in}(d) = in & \mathcal{L}_2^{in}(e) = def; \\ \mathcal{L}_2^{def}(b) = def & \mathcal{L}_2^{def}(c) = in & \mathcal{L}_2^{def}(d) = def & \mathcal{L}_2^{def}(e) = in; \\ \mathcal{L}_2^{undec}(b) = def & \mathcal{L}_2^{undec}(c) = in & \mathcal{L}_2^{undec}(d) = undec & \mathcal{L}_2^{undec}(e) = undec. \end{array}$$

The labeling $\mathcal{L}_2^{in}$ is not valid, because of the fact that $c \rightarrowtail d$ and $\mathcal{L}_2^{in}(c) = in$. Hence we have that $L_2^* = \{\mathcal{L}_2^{def}\}$. Now, since $\mathcal{L}_2^{def}(e) = in$ but $e \notin E$, E is rejected by the algorithm.

It is easy to see that $\{a, c, e\}$ is the only set that would be accepted by the algorithm, which mirrors the fact that $\{a, c, e\}$ is the only preferred extension of F. ◇

3.2 Further directed graph measures

In this section we extend our hardness results to some popular parameters for directed graphs, namely for directed tree-width, dag-width, Kelly-width, and directed path-width. We first review the definition of directed tree-width [27]. Hereby, a so-called arboreal decomposition is built similarly to a tree decomposition: an arboreal decomposition also consists of a tree and "bags", i.e., sets of vertices of the graph that we want to decompose. However, in contrast to tree decompositions of undirected graphs, the bags in an arboreal decomposition can be used both as vertex labels *and* as edge labels in the tree. The bags used as vertex labels build a partition of the vertices of the original graph. The bags used as edge labels are vertex-sets that concern the subtree rooted at the target vertex of the associated edge; in particular they isolate the subgraph induced by the union of the vertex bags in this subtree. That is that every path which starts and ends – but eventually leaves the subgraph – comes across a vertex which is in the edge-bag.

The following definition make these concepts formal. It is convenient to introduce the following notation first. For $t \in V_{\mathcal{T}}$ and $e = (u, v) \in E_{\mathcal{T}}$ we write $t > e$ iff $v = t$ or there exists a path from v to t in $\mathcal{T}$. Moreover, for $e = (u, v) \in E_{\mathcal{T}}$, we use the term $e \sim t$ to denote that either $u = t$ or $v = t$ holds.

Definition 10. *Let $G = (V, E)$ be a directed graph. An* arboreal decomposition *of G is a tuple $(\mathcal{T}, \mathcal{X}, \mathcal{Y})$ where $\mathcal{T} = (V_{\mathcal{T}}, E_{\mathcal{T}})$ is a directed tree with a unique root and $\mathcal{X} = (X_t)_{t \in V_{\mathcal{T}}}$, $\mathcal{Y} = (Y_e)_{e \in E_{\mathcal{T}}}$ are families of subsets of V, such that*

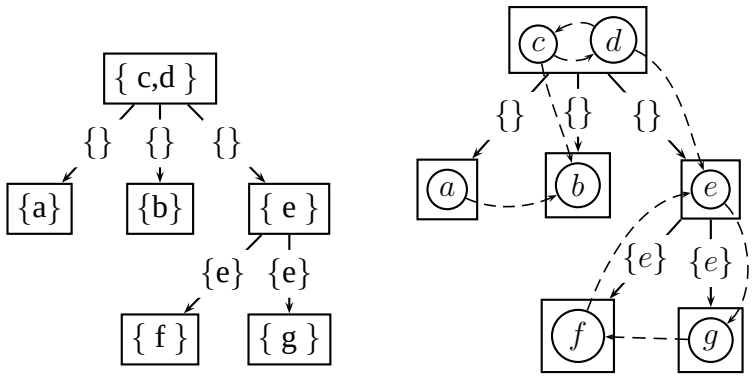
1. *$\mathcal{X}$ is a partition of V_G into non-empty sets;*
2. *for each $e \in E_T$, there is no directed path in $G|_{V \setminus Y_e}$ with first and last vertex in $X_{\geq e} := \bigcup\{X_t \mid t > e\}$ that contains a vertex in $V \setminus (Y_e \cup X_{\geq e})$.*

The width of an arboreal decomposition is $\max_{t \in V_t}\{card(X_t \cup \bigcup_{e \sim t} Y_e)\} - 1$. The directed tree-width *of G (denoted as $dtw(G)$) is the smallest width of any arboreal decomposition.*

Example 8. Recall the AF F from Example 1. Below, we define an arboreal decomposition $(\mathcal{T}, \mathcal{X}, \mathcal{W})$ of the corresponding graph interpretation. For simplicity, we identify each node $t \in \mathcal{T}$ with the corresponding bag X_t. The arboreal decomposition is as follows:

- $V_{\mathcal{T}} = \mathcal{X} = \{\{c, d\}, \{a\}, \{b\}, \{e\}, \{f\}, \{g\}\}$
- $E_{\mathcal{T}} = \{(\{c, d\}, \{a\}), (\{c, d\}, \{b\}), (\{c, d\}, \{e\}), (\{e\}, \{f\}), (\{e\}, \{g\})\}$
- $\mathcal{Y} = (Y_y)_{y \in E_{\mathcal{T}}}$ with $Y_{(\{e\},\{f\})} = \{e\}$, $Y_{(\{e\},\{g\})} = \{e\}$ and $Y_y = \emptyset$ for the other edges in $E_{\mathcal{T}}$.

For an illustration of the decomposition, see Figure 5. We mention that this decomposition has width 1. For example, consider the cycle e, g, f of the AF. In our decomposition the cycle is partitioned in three nodes such that the nodes $\{f\}, \{g\}$ are sucessors of $\{e\}$. Now given $\{f\}$, we have a path in F, namely f, e, g, f, which starts and ends in the bag $\{f\}$ but eventually leafs the bag. Thus to fulfill condition (2) for being an arboreal decomposition, one has to add either e or g to the edge bag of $(\{e\}, \{f\})$. For similar reasons we have to add either e or f to the edge bag of



(a) Arboreal decomposition

(b) Arboreal decomposition, with induced subframeworks

Figure 5: An arboreal decomposition for the AF in Example 1

$(\{e\}, \{g\})$. In both cases we decided to add e as it is already contained in the predecessors vertex bag, i.e. in $\{e\}$, and thus does not increase the width generated by the predecessor node, while in any case it increases the width generated by the node $\{f\}$ resp. $\{g\}$. $\diamond$

We do not require explicitly the definitions of the other three graph parameters mentioned in this paper. Instead, we only provide a summary of results compiled together from [4, 25, 24]; the reader is referred to [1, 4, 25] for formal definitions of the parameters of dag-width, Kelly-width and directed path-width.

Proposition 1. *Let be $G = (V, E)$ a directed graph then the following holds:*

$$(dtw(G) - 1)/3 \leq dagw(G) \leq dpw(G) + 1 \leq cr(G) + 1$$
$$(dtw(G) + 2)/6 \leq kellyw(G) \leq dpw(G) + 1 \leq cr(G) + 1$$

where $dagw(G)$, $kellyw(G)$, and $dpw(G)$ denote the dag-width, Kelly-width, and directed path-width of G, respectively.

Indeed, this proposition allows us to obtain hardness results for directed tree-width, dag-width, Kelly-width and directed path-width from the corresponding hardness result for cycle-rank, which we have obtained in the previous subsection. Figure 6 illustrates how a hardness result for one of the above mentioned graph measures can be propagated to the other graph measures. The following corollaries exploit this result, but use a distinguished argumentation for the parameter of directed tree-width.

Corollary 1. CA *is* NP*-hard even for*

- *AFs of directed path-width* 1
- *AFs of DAG-width* 2
- *AFs of Kelly-width* 2

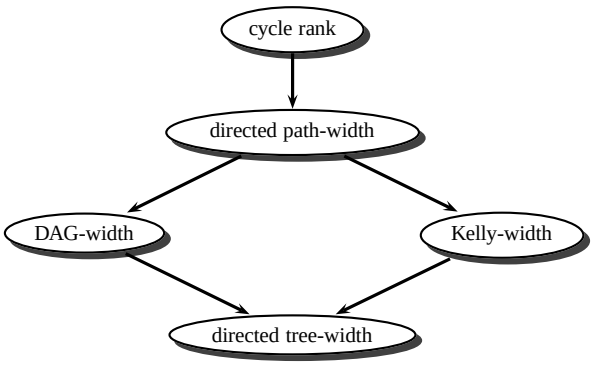
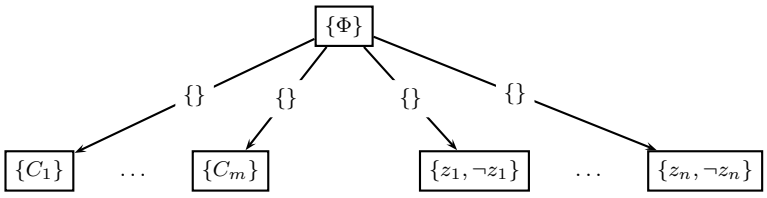


Figure 6: Propagation of Hardness Results for directed graph measures (see Proposition 1)

- *AFs of directed tree-width* 1.

Proof. While the results for directed path-width, DAG-width and Kelly-width follow directly from Proposition 1, we give an explicit proof for coNP-hardness of CA over AFs with directed tree-width 1 (Proposition 1 only provides hardness for AFs of directed tree-width 7). To this end, we construct arboreal decompositions $(\mathcal{T}, \mathcal{X}, \mathcal{W})$ for frameworks of the form $\mathcal{F}_\Phi$ as given in Definition 7.

- $V_\mathcal{T} = \mathcal{X} = \{\{z_1, \neg z_1\}, \ldots, \{z_n, \neg z_n\}, \{C_1\}, \ldots, \{C_m\}, \{\Phi\}\}$
- $E_\mathcal{T} = \{(\{\Phi\}, \{z_i, \neg z_i\}) \mid 1 \leq i \leq n\} \cup \{(\{\Phi\}, \{C_i\}) \mid 1 \leq i \leq m\}$
- $\mathcal{Y} = (Y_e)_{e \in E_\mathcal{T}}$ with $Y_e = \emptyset$ for all $e \in E_\mathcal{T}$



This arboreal decomposition has width 1 and by the fact that $\mathcal{F}_\Phi$ contains a clique of size 2, namely $\{z, \bar{z}\}$, we conclude that $dtw(\mathcal{F}_\Phi) = 1$. □

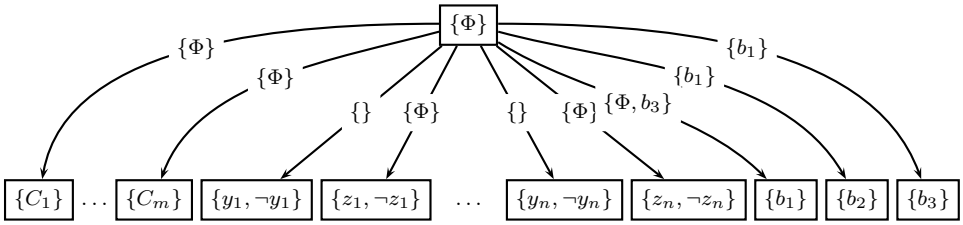
Corollary 2. SA *is* Π_2^P*-hard even for*

- *AFs of directed path-width* 2
- *AFs of DAG-width* 3
- *AFs of Kelly-width* 3
- *AFs of directed tree-width* 2.

Proof. As before the results for directed path-width, DAG-width and Kelly-width follow directly from Proposition 1 and we have to construct appropriate arboreal decompositions for AFs of the form $\mathcal{G}_\Psi$ as given in Definition 8 to show the desired hardness result for AFs of directed tree-width 2.

- $V_\mathcal{T} = \mathcal{X} = \{\{y_1, \neg y_1\}, \{z_1, \neg z_1\}, \ldots, \{y_n, \neg y_n\}, \{z_n, \neg z_n\}, \{C_1\}, \ldots, \{C_m\}, \{\Phi\}, \{b_1\}, \{b_2\}, \{b_3\}\}$
- $E_\mathcal{T} = \{(\{\Phi\}, \{y_i, \neg y_i\})\} \cup \{(\{\Phi\}, \{z_i, \neg z_i\})\} \cup \{(\{\Phi\}, \{C_i\})\} \cup \{(\Phi, b_i)\}$
- $\mathcal{Y} = (Y_e)_{e \in E_\mathcal{T}}$ with

$$Y_e = \begin{cases} \{\Phi\} & \text{for } e = (\{\Phi\}, \{C_i\}) \text{ or } (\{\Phi\}, \{z_i, \neg z_i\}) \\ \{b_1\} & \text{for } e = (\{\Phi\}, \{b_2\}) \text{ or } e = (\{\Phi\}, \{b_3\}) \\ \{\Phi, b_3\} & \text{for } e = (\{\Phi\}, \{b_1\}) \\ \emptyset & \text{otherwise} \end{cases}$$



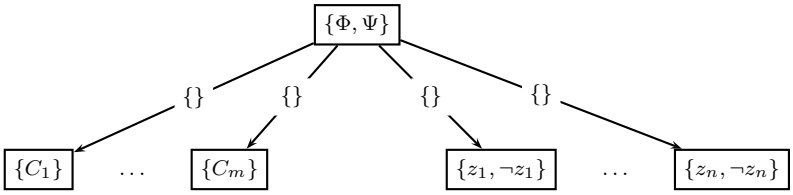
One can see that this arboreal decomposition has width 2 and further one can show that $dtw(\mathcal{F}_\Phi) = 2$. □

Corollary 3. ID *is* coNP*-hard even for*

- *AFs of directed path-width* 1
- *AFs of DAG-width* 2
- *AFs of Kelly-width* 2
- *AFs of directed tree-width* 1.

Proof. Once more the results for directed path-width, DAG-width and Kelly-width follow directly from Proposition 1. To show hardness for AFs of directed tree-width 1, we give arboreal decompositions for AFs of the form $\mathcal{H}_\Phi$ (see Definition 9).

- $V_\mathcal{T} = \mathcal{X} = \{\{z_1, \neg z_1\}, \ldots, \{z_n, \neg z_n\}, \{C_1\}, \ldots, \{C_m\}, \{\Phi, \Psi\}\}$
- $E_\mathcal{T} = \{(\{\Phi\}, \{z_i, \neg z_i\}) : 1 \leq i \leq n\} \cup \{(\{\Phi\}, \{C_i\}) : 1 \leq i \leq m\}$
- $\mathcal{Y} = (Y_e)_{e \in E_\mathcal{T}}$ with $Y_e = \emptyset$ for all $e \in E_\mathcal{T}$



One can see that this arboreal decomposition has width 1 and by the presence of cliques of size, $dtw(\mathcal{H}_\Phi) = 1$ follows. □

To summarize, we have shown that none of the parameters cycle-rank, directed tree-width, DAG-width, Kelly-width, and directed path-width is applicable for fixed-parameter tractability results. Hence, we observe that directed graph measures generalizing tree-width are not well suited for the reasoning problems CA, SA, and ID on AFs.

An explanation for this obstacle is that argumentation semantics are based on conflict-freeness of the extension which is an undirected property, i.e. the orientation of the attacks does not play a role, and thus cannot be captured well by the measures discussed in this subsection. As the standard reasoning tasks are computationally easy for (maximal) conflict-free sets one might expect that conflict-freeness does not harm when searching for tractable fragments. But there are also problems which are hard for conflict-free sets. For instance, the problem of counting for maximal conflict-free sets is $\#\mathrm{P}$-hard, this was originally shown by Valiant [31] (in terms of counting maximal cliques) and was recently applied to abstract argumentation [2]. Thus the above $\#\mathrm{P}$-hardness-result is an evidence against the existence of tractability results based on directed graph measures.

The above results leave some space for tractable fragments when bounding directed tree-width, DAG-width, Kelly-width or directed path-width to 0, 1 or 2. Taking the minimal bounds into account (by definition, 0 for directed path-width and directed tree-width and resp. 1 for DAG-width and Kelly-width) leads to acyclic AFs which are trivially tractable. Moreover, using Corollary 3, we can show that skeptical acceptance is still coNP-hard for AFs of directed path-width 1, DAG-width 2, Kelly-width 2 or directed tree-width 1. We leave the exact complexity classification for these particular fragments as open problems.

We now come back to the parameter of tree-width, which, in contrast to the parameters discussed in this section, allows for fixed-parameter tractability results. In fact, in the next section we present novel algorithms which exploit this feature.

4 Dynamic Programming for Argumentation

Before we introduce our algorithms, we need some more notation for tree decompositions. In particular, it is useful to reduce the number of different node types and to identify a root node. The following concept serves this purpose.

Definition 11. *A tree decomposition $(\mathcal{T}, \mathcal{X})$ of a graph G is called* nice *if $\mathcal{T}$ is a rooted tree and if each node*[3] *$t \in \mathcal{T}$ is of one of the following types:*

[3]For $\mathcal{T} = (V_\mathcal{T}, E_\mathcal{T})$ we often write $t \in \mathcal{T}$ instead of $t \in V_\mathcal{T}$.

1. *LEAF:* t *is a leaf of* $\mathcal{T}$
2. *FORGET:* t *has only one child* t' *and* $X_t = X_{t'} \setminus \{v\}$ *for some* $v \in X_{t'}$
3. *INSERT:* t *has only one child* t' *and* $X_t = X_{t'} \cup \{v\}$ *for some* $v \notin X_{t'}$
4. *JOIN:* t *has two children* t', t'' *and* $X_t = X_{t'} = X_{t''}$

Kloks [28] showed that a tree decomposition $(\mathcal{T}, \mathcal{X})$ of a graph G where $\mathcal{T}$ has n nodes, can be transformed in time $O(n)$ into a nice tree decomposition $(\mathcal{T}', \mathcal{X}')$ of G which has the same width as $(\mathcal{T}, \mathcal{X})$ and where $\mathcal{T}'$ has $O(n)$ nodes.

As already mentioned, the concept of tree-width is defined for undirected graphs but can also be applied to directed graphs and thus to AFs.

Definition 12. *Let* $F = (A, R)$ *be an AF. A tree decomposition of the undirected graph* (A, R') *where* R' *contains the edges of* R *without orientation is called a* tree decomposition *of* F*. The tree-width of an AF* F *is given by the minimum width over all tree decompositions of* F*.*

Next, we have to introduce a few more technical notions which allow us direct access to some objects associated with certain nodes in a tree decomposition.

Definition 13. *For a tree decomposition* $(\mathcal{T}, \mathcal{X})$ *of an AF* F *and* $t \in \mathcal{T}$*, let* $X_{\geq t}$ *be the union of all bags* $X_s \in \mathcal{X}$ *such that* s *occurs in the subtree of* $\mathcal{T}$ *rooted at* t*. Moreover,* $X_{>t}$ *denotes* $X_{\geq t} \setminus X_t$*. We also use the following terminology:*

- $F_t = F|_{X_t}$ *is the* subframework in t*;*
- $F_{\geq t} = F|_{X_{\geq t}}$ *is the* subframework induced by (the subtree rooted at) t*.*

Note that the subframework induced by the root of such a decomposition of an AF F is F itself.

Example 9. For the AF F from Example 1, we have already depicted a tree decomposition in Figure 1. To obtain a nice tree decomposition, we have to introduce some further nodes. For instance, between the nodes with bags $\{a, b\}$ and $\{b, c\}$, we insert a further node with bag $\{b\}$, etc. We also have added two forget-nodes above the $\{c, d\}$-node in order to have an empty root. The resulting nice tree decomposition of F is illustrated in Figure 7, which has to be read as follows. In each node t, the bag X_t contains the arguments in (solid) cycles. In addition, we depict in each node t the AF F_t, i.e. the subframework in t; by adding the dotted parts of the graph, we obtain $F_{\geq t}$, the subframework induced by t. ◇

In what follows we restrict ourselves to nice tree decompositions where the bag of the root is empty. Unless stated otherwise, we thus assume below that $(\mathcal{T}, \mathcal{X})$ always denotes a nice tree decomposition (with empty root bag) for some given AF F.

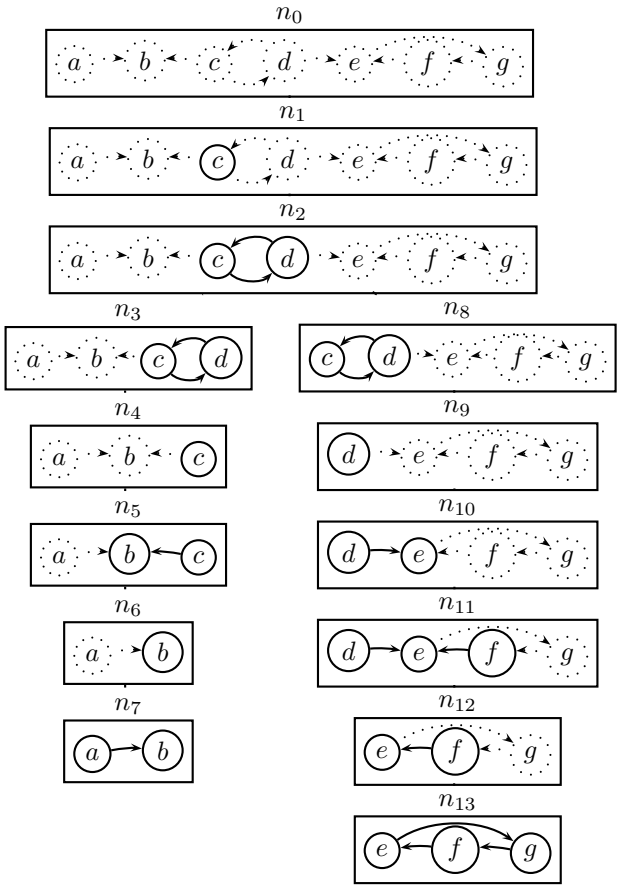


Figure 7: Tree decomposition of F with subframeworks.

4.1 Characterizing admissible sets

We first introduce a relativization of admissible sets to a given set B of arguments.

Definition 14. *Let $F = (A, R)$ be an AF and B a set of arguments. A set $S \subseteq A$ is a* B-restricted admissible set for F*, if S is conflict-free in F and S defends itself in F against all $a \in A \cap B$.*

Example 10. Let us consider the AF

$$(\{e, f, g\}, \{(e, g), (g, f), (f, e)\})$$

which is a subframework of our running example. Indeed, in the tree decomposition in Figure 7, it is the subframework in n_{13} and also the subframework induced by n_{12}. The $\{g\}$-restricted admissible sets of this AF are $\emptyset$, $\{e\}$, and $\{g\}$. In fact, $\{f\}$ is not $\{g\}$-restricted admissible here, since $g \rightarrowtail f$ but f does not defend itself against g. $\diamond$

Note that for $A \subseteq B$, B-restricted admissible sets of AFs (A, R) are just the standard admissible sets; for $A \cap B = \emptyset$, B-restricted admissible sets are just the conflict-free sets.

We now introduce the underlying "data structure" of our dynamic programming algorithm for characterizing admissible sets. The idea hereby is to store at each node t in the tree decomposition $(\mathcal{T}, \mathcal{X})$ for a given AF F, a set of candidates which are represented solely via the elements in the bag X_t. More precisely, we assign to each node $t \in \mathcal{T}$ a certain set of mappings $C\colon X_t \to \{in, out, att, def\}$. We call such mappings also colorings for t. The rationale behind a coloring for t is as follows: explicitly, a coloring characterizes the set

$$[C] = \{a \mid C(a) = in\}$$

and the values out, att, def tell us about the relationship between $[C]$ and the remaining arguments $X_t \setminus [C]$. In fact, att will denote arguments which attack $[C]$ but are not attacked by $[C]$, def denotes arguments attacked by $[C]$, and out are those which are in no relation with arguments from $[C]$. However, we will define colorings in such a way that they characterize sets over $X_{\geq t}$, rather than over X_t as sketched above. Formally, this intuition is captured as follows:

Definition 15. *Let $(\mathcal{T}, \mathcal{X})$ be a tree decomposition of an AF F and $t \in \mathcal{T}$. Given a coloring C for a node $t \in \mathcal{T}$, we define $e_t(C)$ as the collection of $X_{>t}$-restricted admissible sets S for $F_{\geq t}$ which satisfy the following conditions for each $a \in X_t$:*

(i) $C(a) = in$ *iff* $a \in S$;

(ii) $C(a) = def$ *iff* $S \rightarrowtail a$;

(iii) $C(a) = att$ *iff* $S \not\rightarrowtail a$ *and* $a \rightarrowtail S$;

(iv) $C(a) = out$ *iff* $S \not\rightarrowtail a$ *and* $a \not\rightarrowtail S$.

If $e_t(C) \neq \emptyset$, C is called a valid coloring *for t. The set of valid colorings for t is denoted by $\mathcal{C}_t$.*

Example 11. Consider the node $t = n_{11}$ of our example tree decomposition with $X_t = \{d, e, f\}$ (see the right-hand side of the tree in Figure 7) and the coloring C with $C(d) = in$ and $C(e) = C(f) = def$. We have $F_{\geq t} = (\{d, e, f, g\}, \{(d, e), (e, g), (g, f), (f, e)\})$ and $X_{>t} = \{g\}$. The only set which is $X_{>t}$-restricted admissible for $F_{\geq t}$ and satisfies the conditions from Definition 15 is $\{d, g\}$. $S = \{d\}$ would also be $X_{>t}$-restricted admissible but violates Condition (ii), since $C(f) = def$ and $S \not\rightarrowtail f$. In summary, this particular C is valid for $t = n_{11}$ and we have $e_t(C) = \{\{d, g\}\}$. ◇

Our ultimate goal is to efficiently compute the set $\mathcal{C}_r$ of valid colorings for the root node r of a given tree decomposition for an AF $F = (A, R)$. The reason for this is the fact that $\bigcup_{C \in \mathcal{C}_t} e_t(C)$ gives exactly the set of $X_{>t}$-restricted admissible sets for $F_{\geq t}$ (as we show next). Since the root r has an empty bag, and thus $X_{>r} = A$, we obtain that $\mathcal{C}_r$ characterizes the admissible sets of F.

By definition, each element in $e_t(C)$ is an $X_{>t}$-restricted admissible set for $F_{\geq t}$. Next, we show that also the opposite direction holds.

Lemma 1. *Let $(\mathcal{T}, \mathcal{X})$ be a tree decomposition of an AF F, $t \in \mathcal{T}$, and S an $X_{>t}$-restricted admissible set for $F_{\geq t}$. Then, there is a coloring $C \in \mathcal{C}_t$ such that $S \in e_t(C)$.*

Proof. Since S is an $X_{>t}$-restricted admissible set for $F_{\geq t}$, each argument $a \in X_t$ satisfies one of the following conditions: (i) $a \in S$, (ii) $S \rightarrowtail a$, (iii) $S \not\rightarrowtail a$ and $a \rightarrowtail S$, or (iv) $S \not\rightarrowtail a$ and $a \not\rightarrowtail S$. For these four cases, we define C as follows:

In case (i): $C(a) = in$,
in case (ii): $C(a) = def$,
in case (iii): $C(a) = att$, and
in case (iv): $C(a) = out$.

By the construction of C, the set S satisfies conditions (i) – (iv) in Definition 15 and, since S is $X_{>t}$-restricted admissible for $F_{\geq t}$, it holds that $S \in e_t(C)$. □

Moreover, different colorings for a node t characterize different $X_{>t}$-restricted admissible sets for $F_{\geq t}$.

Lemma 2. *Let $(\mathcal{T}, \mathcal{X})$ be a tree decomposition of an AF F and let C, C' be different colorings for a node $t \in \mathcal{T}$. Then, $e_t(C) \cap e_t(C') = \emptyset$.*

Proof. Suppose to the contrary that there is a set $S \in e_t(C) \cap e_t(C')$, where C and C' are different colorings for t. Then there exists an argument $a \in X_t$ such that $C(a) \neq C'(a)$. It remains to inspect all possible pairs of values of $C(a)$ and $C'(a)$ and to derive a contradiction in each case. First let us consider the case where $C(a) = in$ and $C'(a) \in \{def, att, out\}$. By Definition 15, $C(a) = in$ implies $a \in S$ and further $C'(a) \in \{def, att, out\}$ implies $a \notin S$, a contradiction. We continue with the case where $C(a) = def$ and $C'(a) \in \{att, out\}$. By Definition 15, $C(a) = def$ implies $S \rightarrowtail a$. On the other hand, $C'(a) \in \{att, out\}$ implies $S \not\rightarrowtail a$, a contradiction. Finally, in case $C(a) = att$ and $C'(a) = out$, we get a contradiction by the fact that $C(a) = att$ implies $a \rightarrowtail S$ and $C'(a) = out$ implies $a \not\rightarrowtail S$. The remaining cases follow by symmetry, i.e. by interchanging the roles of C, C' in the above arguments. □

To guarantee fixed-parameter tractability with respect to tree-width, we want to compute the sets $\mathcal{C}_t$ in a bottom-up manner along the tree decomposition *without* an explicit computation of $e_t(\cdot)$. Therefore, we recursively define the concept of vcolorings which we afterwards show to be equivalent to valid colorings.

Definition 16. *Let $t \in \mathcal{T}$ be a node in a tree decomposition $(\mathcal{T}, \mathcal{X})$ of an AF F and let t', t'' be the possible children of t. The operations $(C - a)$, $(C + a)$, $(C \dot{+} a)$, and $(C \bowtie D)$ used below are defined in Figure 8. Depending on the node type of t, we define a* vcoloring *for t as follows:*

- *LEAF-node: Each coloring $X_t \to \{in, out, att, def\}$ where*

$$C(x) = in \Rightarrow C(y) \in \{att, def\} \text{ for all } y \rightarrowtail x;$$
$$C(x) = att \Rightarrow \exists y : C(y) = in \text{ and } x \rightarrowtail y;$$
$$C(x) = def \Leftrightarrow \exists y : C(y) = in \text{ and } y \rightarrowtail x;$$

  holds for all $x \in X_t$, is a vcoloring for t.

$$(C - a)(b) = C(b) \text{ for each } b \in A \setminus \{a\}$$

$$(C + a)(b) = \begin{cases} C(b) \text{ if } b \in A \\ def \text{ if } b = a \text{ and } [C] \rightarrowtail a \\ att \text{ if } b = a, [C] \not\rightarrowtail a \text{ and } a \rightarrowtail [C] \\ out \text{ otherwise} \end{cases}$$

$$(C \dot{+} a)(b) = \begin{cases} in \text{ if } b = a \text{ or } C(b) = in \\ def \text{ if } a \neq b \text{ and } ((a,b) \in F_t \text{ or } C(b) = def) \\ out \text{ if } a \neq b, C(b) = out, (a,b) \notin F_t, (b,a) \notin F_t \\ att \text{ otherwise} \end{cases}$$

$$(C \bowtie D)(b) = \begin{cases} in \text{ if } C(b) = D(b) = in \\ out \text{ if } C(b) = D(b) = out \\ def \text{ if } C(b) = def \text{ or } D(b) = def \\ att \text{ otherwise} \end{cases}$$

Figure 8: Operations for Colorings $C, D : A \rightarrow \{in, out, att, def\}$.

- *FORGET-node: If C is a vcoloring for t', $X_t = X_{t'} \setminus \{a\}$, and $C(a) \neq att$, then $C - a$ is a vcoloring for t.*
- *INSERT-node: If C is a vcoloring for t' and $X_t = X_{t'} \cup \{a\}$, then $C + a$ is a vcoloring for t; if $a \not\rightarrowtail a$, $[C] \not\rightarrowtail a$, and $a \not\rightarrowtail [C]$ hold, then $C \dot{+} a$ is also a vcoloring for t.*
- *JOIN-node: If C is a vcoloring for t', D is a vcoloring for t'', and $[C] = [D]$, then $C \bowtie D$ is a vcoloring for t.*

In what follows, we show the adequacy of vcolorings (i.e., that they match the concept of valid colorings) and also illustrate their functioning on our running example. We will do this step-by-step distinguishing between the different node types.

Example 12. Recall the AF from Example 1 and its tree decomposition in Figure 7. Figure 9 illustrates the bottom-up computation of the vcolorings for all nodes in the tree decomposition. More precisely, for each node t we give a table representing all vcolorings of t, whereby each row gives one such vcoloring. Below we will discuss some of the transitions from children to parent nodes (as defined by vcolorings, cf. Definition 16) in detail.

For the moment, let us just describe a few further aspects in Figure 9. For a better understanding we also added the # column in Figure 9 to show the cardinalities of the sets $e_t(C)$, i.e. the number of $X_{>t}$-restricted admissible sets for $F_{\geq t}$ characterized by vcoloring C in t. In particular, we see in the root that we end up with 8 such sets which refer to the admissible sets from our example AF (see Example 1). An explanation for the ✓ symbol in the tables of Figure 9 follows later in Example 17, when we shall illustrate how to decide CA using the concept of vcolorings. ◇

Let us start with *LEAF* nodes. We first give the desired result and then provide for illustration the computation of the vcolorings for a leaf node in our running example.

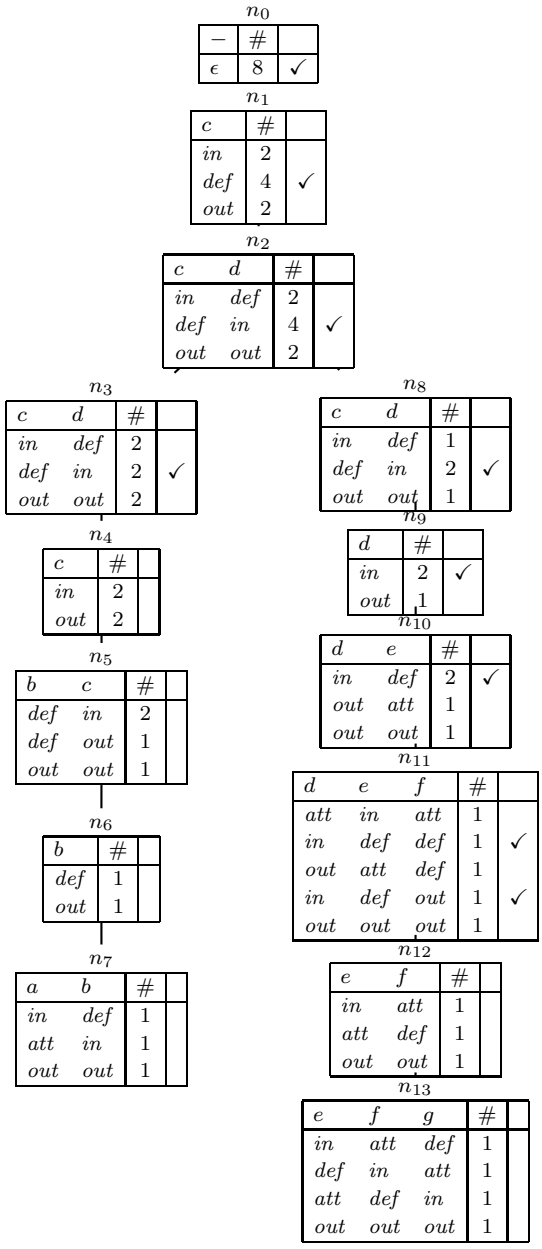


Figure 9: Computation of vcolorings for the example AF.

Lemma 3. *For any* LEAF *node in a tree decomposition of an AF, valid colorings and vcolorings coincide.*

Proof. Let $(\mathcal{T}, \mathcal{X})$ be a tree decomposition of F and t a leaf in $\mathcal{T}$. We have $X_{>t} = \emptyset$; therefore, the $X_{>t}$-restricted admissible sets for $F_{\geq t}$ coincide with the conflict-free sets.

First, let C be a vcoloring for t. We have to show that then C is a valid coloring for t. Suppose to the contrary that it is not, i.e., either $[C]$ is not conflict-free in $F_t = F_{\geq t}$ or C violates one of the

conditions (ii) – (iv) in Definition 15. It is easy to check that, in both cases, one of the conditions for C being a vcoloring is violated. For instance, if there is a conflict in $[C]$, then there exist arguments $x, y \in X_t$ with $x \rightarrowtail y$ and $C(x) = C(y) = in$. Hence, the first condition in Definition 16 for vcolorings at a *LEAF* node is violated, a contradiction.

Now suppose that C is a valid coloring for t, i.e., C satisfies the conditions (i) – (iv) of a coloring (see Definition 15) and $[C]$ is conflict-free in $F_{\geq t}$. Then C satisfies the condition of a vcoloring for a *LEAF* node according to Definition 16. For instance, let $x, y \in X_t$ with $C(x) = in$ and $y \rightarrowtail x$. Then, since C is a coloring, either case (ii) or case (iii) of Definition 15 applies and, thus, $C(y) \in \{att, def\}$ holds. □

Example 13. Consider, for instance, the *LEAF* node n_{13} in Figure 9 with bag $\{e, f, g\}$. We have here four vcolorings for n_{13} which correspond to the conflict-free (and thus to the $\emptyset$-restricted admissible) sets for $F_{\geq n_{13}} = (\{e, f, g\}, \{(e, g), (g, f), (f, e)\})$, namely $\{e\}$, $\{f\}$, $\{g\}$, and $\emptyset$. ◇

We proceed with the *FORGET* nodes.

Lemma 4. *For any* FORGET *node t in a tree decomposition of an AF, valid colorings and vcolorings coincide, if they coincide in the child node t' of t.*

Proof. Let $(\mathcal{T}, \mathcal{X})$ be a tree decomposition of $F = (A, R)$, t a *FORGET* node in $\mathcal{T}$, and t' the child node of t. By definition, $X_t = X_{t'} \setminus \{a\}$, for some $a \in A$. Moreover, we get $X_{\geq t} = X_{\geq t'}$ and $X_{>t} = X_{>t'} \cup \{a\}$.

Let C be a valid coloring for t. We show that there exists a valid coloring C' for t' with $C'(a) \neq att$ and $C = C' - a$. We define C' as follows: For all $b \in X_t = X_{t'} \setminus \{a\}$, we set $C'(b) = C(b)$. Hence, no matter which value of $\{in, def, out\}$ we assign to $C'(a)$, we have $C = C' - a$. In order to define $C'(a)$, we consider an arbitrary set $S \in e_t(C)$ and distinguish two cases:

1) If $a \in S$, then we set $C'(a) = in$. Since S is $X_{>t}$-restricted admissible for $F_{\geq t}$, it is also $X_{>t'}$-restricted admissible for $F_{\geq t'} = F_{\geq t}$. Moreover, $S \in e_{t'}(C')$, i.e., C' is a valid coloring for t' (this can be seen by just using the chosen S in the conditions (i) – (iv) in Definition 15). Hence, by assumption, C' is a vcoloring for t' and, therefore, also $C = C' - a$ is a vcoloring for t, by definition.

2) Now let $a \notin S$. If $S \rightarrowtail a$, we set $C'(a) = def$. If $S \not\rightarrowtail a$ and $a \not\rightarrowtail S$, we set $C'(a) = out$. In both cases, $S \in e_{t'}(C')$. Note that the case $S \not\rightarrowtail a$ and $a \rightarrowtail S$ cannot occur since, by assumption, S is $X_{>t}$-restricted admissible for $F_{\geq t}$. By the same reasoning as above, C' (and thus also C) is a vcoloring for t' (resp. for t).

Now let C be a vcoloring for t, i.e., there exists a vcoloring C' for t' such that $C'(a) \neq att$ and $C = C' - a$. By assumption, C' is a valid coloring for t'. Hence, there exists $S \in e_{t'}(C')$, i.e., S is $X_{>t'}$-restricted admissible for $F_{\geq t'} = F_{\geq t}$. Since $C'(a) \neq att$, it cannot happen that both $a \rightarrowtail S$ and $S \not\rightarrowtail a$ hold. But then S is also $X_{>t}$-restricted admissible for $F_{\geq t}$ and $S \in e_t(C)$. Thus, $C \in \mathcal{C}_t$. □

Example 14. Let us continue the running example which we started above by computing the vcolorings for node n_{13}. The next node n_{12} above n_{13} is of type *FORGET* and removes argument g. Thus $X_{>n_{12}} = \{g\}$. The vcolorings for n_{12} are obtained from the vcolorings for n_{13} with the exception of the coloring C with $[C] = \{f\}$. Here we have $C(g) = att$, which violates the construction for the *FORGET* node. Intuitively, $[C] = \{f\}$ is not further propagated because $\{f\}$ is attacked by the argument g which is no longer present in $X_{n_{12}}$. Hence, by properties (2) and (3) of tree decompositions, g is not attacked by any argument outside $X_{\geq n_{12}}$. Therefore, $[C] = \{f\}$ cannot be extended to an admissible set along the bottom-up traversal, i.e., any extension of $[C] = \{f\}$ to arguments outside $X_{\geq n_{12}}$ will not defend itself against this attack from g against f. The vcolorings for n_{12} are now in accordance with the $X_{>n_{12}}$-restricted admissible sets for $F_{\geq n_{12}} = F_{\geq n_{13}}$ (see also Example 10 where we already analyzed exactly this situation). ◇

The next nodes we want to consider are those of type *INSERT*.

Lemma 5. *For any* INSERT *node t in a tree decomposition of an AF, valid colorings and vcolorings coincide, if they coincide in the child node t' of t.*

Proof. Let $(\mathcal{T}, \mathcal{X})$ be a tree decomposition of $F = (A, R)$, t an *INSERT* node in $\mathcal{T}$, and t' the child node of t. Let us assume $X_t = X_{t'} \cup \{a\}$ and $a \notin X_{t'}$. Thus, we have that $X_{\geq t} = X_{\geq t'} \cup \{a\}$ and $X_{>t} = X_{>t'}$. By properties (2) and (3) of tree decompositions, we know that there are no attacks between the new argument a and arguments in $X_{>t}$.

Let C be a valid coloring for t, i.e., there exists an $X_{>t}$-restricted admissible set $S \in e_t(C)$ for $F_{\geq t}$. By $X_{>t} = X_{>t'}$, S is also $X_{>t'}$-restricted admissible for $F_{\geq t}$. Moreover, since a cannot attack any argument in $X_{>t'}$, also $S \setminus \{a\}$ is $X_{>t'}$-restricted admissible for $F_{\geq t'}$ (of course, if $a \notin S$, then $S \setminus \{a\} = S$ and the latter admissibility property is trivial). As in the proof of Lemma 1, we construct a coloring C' for t' with $S \setminus \{a\} \in e_{t'}(C')$ as follows. For arbitrary $b \in X_{t'}$, we define:

$C'(b) = in$ if $b \in S \setminus \{a\}$,
$C'(b) = def$ if $b \notin S$ and $S \setminus \{a\} \rightarrowtail b$,
$C'(b) = att$ if $b \notin S$, $b \rightarrowtail S \setminus \{a\}$, and $S \setminus \{a\} \not\rightarrowtail b$,
$C'(b) = out$ if $b \notin S$, $b \not\rightarrowtail S \setminus \{a\}$, and $S \setminus \{a\} \not\rightarrowtail b$.

Thus, $C' \in \mathcal{C}_{t'}$, and by assumption, a vcoloring for t'. Moreover, it is easy to check that either $C = C' + a$ holds (if $a \notin S$) or $C = C' \dot{+} a$ holds (if $a \in S$). Hence, C is a vcoloring for t.

Now let C be a vcoloring for t, i.e., there exists a vcoloring C' for t' with either $C = C' + a$ or $C = C' \dot{+} a$. By assumption, C' is a valid coloring, i.e. there exists an $X_{>t'}$-restricted (and, hence, $X_{>t}$-restricted) admissible set $S \in e_{t'}(C')$ of $F_{\geq t'}$. It is easy to check that then $S \in e_t(C' + a)$. Moreover if the set $S \cup \{a\}$ is conflict-free in $F_{\geq t}$, then $S \cup \{a\} \in e_t(C' \dot{+} a)$ as well. Thus, C (which is either $C' + a$ or $C' \dot{+} a$) is a valid coloring for t. □

Example 15. We continue our running example: the next node n_{11} is of type *INSERT* and adds d. Consider the coloring C' for n_{12} with $C'(e) = att$ and $C'(f) = def$. We have two possibilities to add d. In case we want d to be in the set, we obtain the coloring C with $C(d) = in$, $C(e) = def$, $C(f) = def$ (note that e changes its color since it is now a "defeated attacker"); we have seen this coloring already in Example 11. The other possibility is to have d not in the set, resulting in the coloring C'' with $C''(d) = out$, $C''(e) = att$, $C''(f) = def$. ◇

Lemma 6. *For any* JOIN *node t in a tree decomposition of an AF, valid colorings and vcolorings coincide, if they coincide also for both child nodes of t.*

Proof. Let $(\mathcal{T}, \mathcal{X})$ be a tree decomposition of $F = (A, R)$ and t a *JOIN* node in $\mathcal{T}$ with successors t' and t''. Then $X_t = X'_t = X''_t$ and $X_{\geq t'} \cap X_{\geq t''} = X_t$ and $X_{\geq t} = X_{\geq t'} \cup X_{\geq t''}$. So we can partition $X_{\geq t}$ into three disjoint sets $X_{>t'}$, $X_{>t''}$ and X_t. Thus every set $S \subseteq X_{\geq t}$ can be seen as the union of two sets $S_1 \subseteq X_{\geq t'}$ and $S_2 \subseteq X_{\geq t''}$ with $S_1 \cap X_t = S_2 \cap X_t$. The following lemmas identify important properties of these sets S_1 and S_2.

Lemma 7. *Let $S_1 \subseteq X_{\geq t'}$ and $S_2 \subseteq X_{\geq t''}$, such that*

1. *S_1 is $X_{>t'}$-restricted admissible for $F_{\geq t'}$*
2. *S_2 is $X_{>t''}$-restricted admissible for $F_{\geq t''}$*
3. *$S_1 \cap X_t = S_2 \cap X_t$.*

Then $S = S_1 \cup S_2$ is an $X_{>t}$-restricted admissible set for $F_{\geq t}$.

Proof. By properties (2) and (3) of tree decompositions, there are no attacks between the argument sets $X_{>t'}$ and $X_{>t''}$. In order to show that $S = S_1 \cup S_2$ is $X_{>t}$-restricted admissible, we have to prove that (a) S is conflict-free in the AF $F_{\geq t}$; and (b) S defends itself against all attacks from arguments in $X_{>t} = X_{>t'} \cup X_{>t''}$ in $F_{\geq t}$.

(a) Suppose to the contrary that there is a conflict $a \rightarrowtail b$ with $a, b \in S$. Then either $a, b \in X_{\geq t'}$ (resp. $a, b \in X_{\geq t''}$) or $a \in X_{\geq t'}$ while $b \in X_{\geq t''}$ (or vice versa). In the case $a, b \in X_{\geq t'}$, we get $a, b \in S_1$ and, therefore, S_1 is not conflict-free in $F_{\geq t'}$, a contradiction to assumption 1 (the same argument applies to the case $a, b \in X_{\geq t''}$). Thus assume $a \in X_{\geq t'}$ while $b \in X_{\geq t''}$ (or vice versa). Since there is an attack between an argument from $X_{>t'}$ and an argument from $X_{>t''}$, it must hold that $a \in X_t$ or $b \in X_t$. Hence, $\{a, b\} \subseteq X_{\geq t'}$ or $\{a, b\} \subseteq X_{\geq t''}$ holds. Assuming $S_1 \cap X_t = S_2 \cap X_t$, this means that there is a conflict in either S_1 or S_2, yielding a contradiction to assumption 1 or 2.

(b) We show that all arguments in S_1 are defended by S against arguments from $X_{>t}$ in $F_{\geq t}$. The analogous result for S_2 then follows by symmetry. In total, every argument in S is then defended by S against arguments from $X_{>t}$. Together with the result from (a), we thus derive the desired result, i.e. that S is an $X_{>t}$-restricted admissible set for $F_{\geq t}$.

By assumption, S_1 defends itself against $X_{>t'}$ in $F_{\geq t'}$ and thus against $X_{>t'}$ in $F_{\geq t}$. Moreover, there are no attacks from $X_{>t''}$ against $X_{>t'}$ in $F_{\geq t}$ by the properties of tree decompositions. So $X_{>t''}$ can only attack arguments in $S_1 \cap X_t$. Thus, S_2 defends S_1 against $X_{>t''}$ since, $S_1 \cap X_t = S_2 \cap X_t$ and by assumption, S_2 defends itself against all attacks from $X_{>t''}$ in $F_{\geq t'}$ and thus also in $F_{\geq t}$. Putting this together, we have that $S = S_1 \cup S_2$ defends S_1 against $X_{>t}$ in $F_{\geq t}$. □

Lemma 8. *Let S be an $X_{>t}$-restricted admissible set for $F_{\geq t}$, $S_1 = S \cap X_{\geq t'}$ and $S_2 = S \cap X_{\geq t''}$. Then,*

1. S_1 *is* $X_{>t'}$*-restricted admissible for* $F_{\geq t'}$
2. S_2 *is* $X_{>t''}$*-restricted admissible for* $F_{\geq t''}$
3. $S_1 \cap X_t = S_2 \cap X_t$.

Proof. Let S be an $X_{>t}$-restricted admissible set for $F_{\geq t}$. 3. is immediate by the fact that $X_{\geq t'} \cap X_{\geq t''} = X_t$. Moreover, since S is conflict-free in $F_{\geq t}$, each subset of S is conflict-free in any subframework of $F_{\geq t}$, in particular $S_1 = S \cap X_{\geq t'}$ is conflict-free in $F_{\geq t'}$ and $S_2 = S \cap X_{\geq t''}$. is conflict-free in $F_{\geq t''}$. It remains to show that S_1 (resp. S_2) defends itself against all attacks from $X_{>t'}$ (resp. from $X_{>t''}$) in $F_{\geq t'}$ (resp. in $F_{\geq t''}$). Suppose to the contrary that there exists $a \in X_{>t'}$ such that $a \rightarrowtail S_1$ and $S_1 \not\rightarrow a$ in $F_{\geq t'}$. Since S is $X_{>t}$-restricted admissible in $F_{\geq t}$, we know that $S \rightarrowtail a$ in $F_{\geq t}$. Hence, there has to exist an argument $b \in S \setminus S_1 = S \cap X_{>t''}$ such that $b \rightarrowtail a$ in $F_{\geq t}$. But, as already observed earlier, there are no attacks between $X_{>t'}$ and $X_{>t''}$, a contradiction. By symmetry, also S_2 is $X_{>t''}$-restricted admissible for $F_{\geq t''}$. □

Proof of Lemma 6 continued. We now show that valid colorings and vcolorings for a *JOIN* node t coincide. First, let C be a vcoloring for t, i.e., $C = C' \bowtie C''$, where C' (resp. C'') is a vcoloring for t' (resp. t'') and $[C'] = [C'']$. By assumption, C' and C'' are valid colorings for the respective nodes t' and t''. Hence, there exists $S_1 \in e_{t'}(C')$ and $S_2 \in e_{t''}(C'')$. Moreover, by $[C'] = [C'']$, we have $S_1 \cap X_t = S_2 \cap X_t$. Thus, by Lemma 7, $S = S_1 \cup S_2$ is $X_{>t}$-restricted admissible. It remains to show that $S \in e_t(C)$. To this end, we check that the conditions (i) – (iv) in Definition 15 are satisfied for every $a \in X_t$:

(i) By the definition of the $\bowtie$-operator in Figure 8, we have $C(a) = in$ iff $C'(a) = in$ and $C''(a) = in$. This, in turn, is equivalent to $a \in S_1$ and $a \in S_2$. In total, we have $C(a) = in$ iff $a \in S$.

(ii) $C(a) = def$ iff $C'(a) = def$ or $C''(a) = def$ (see Figure 8) iff $S_1 \rightarrowtail a$ or $S_2 \rightarrowtail a$ iff $S \rightarrowtail a$.

(iii) By the "otherwise" branch in Figure 8, we have $C(a) = att$ iff ($C'(a) = att$ or $C''(a) = att$) and ($C'(a) \neq def$ and $C''(a) \neq def$). This, in turn, is equivalent to ($a \rightarrowtail S_1$ or $a \rightarrowtail S_2$) and ($S_1 \not\rightarrow a$ and $S_2 \not\rightarrow a$). In total, we have $C(a) = att$ iff $a \rightarrowtail S$ but $S \not\rightarrow a$.

(iv) $C(a) = out$ iff $C'(a) = out$ and $C''(a) = out$ (see Figure 8) iff $a \not\rightarrow S_1$, $a \not\rightarrow S_2$, $S_1 \not\rightarrow a$ and $S_2 \not\rightarrow a$ iff $a \not\rightarrow S$ and $S \not\rightarrow a$.

Now assume that C is a valid coloring for t, i.e., there exists $S \in e_t(C)$. We define $S_1 = S \cap X_{\geq t'}$ and $S_2 = S \cap X_{\geq t''}$. Then, by Lemma 8, S_1 is $X_{>t'}$-restricted admissible for $F_{\geq t'}$, S_2 is $X_{>t''}$-restricted admissible for $F_{\geq t''}$, and $S_1 \cap X_t = S_2 \cap X_t$. As in the proof of Lemma 1, we can define a coloring C' at t' and a coloring C'' at t'', such that $S_1 \in e_{t'}(C')$ and $S_2 \in e_{t''}(C'')$. Then C' and C'' are valid colorings for the respective nodes t' and t'', and, therefore, by assumption they are also vcolorings for their node. Now define the vcoloring $C^* = C' \bowtie C''$ for node t. We claim that

$C^* = C$ holds. To prove this claim, we have to show that $C^*(a) = C(a)$ for every $a \in X_t$. This equality is shown by distinguishing the four possible values $\{in, def, att, out\}$ and by exploiting the conditions (i) – (iv) in Definition 15 as well as the definition of the $\bowtie$ operator in Figure 8. We only work out the case of "in"-nodes here. The remaining cases are treated analogously. Inspecting the definition of $\bowtie$ in Figure 8, shows that $C^*(a) = in$ iff $C'(a) = in$ and $C''(a) = in$. This, in turn, is equivalent to $a \in S_1$ and $a \in S_2$. On the other hand, by condition (i) of Definition 15, we have $C(a) = in$ iff $a \in S$. By the definition of S_1 and S_2, this is equivalent to $a \in S_1$ and $a \in S_2$. In total, we thus have $C^*(a) = in$ iff $C(a) = in$. □

Example 16. The only node of type JOIN in our example is the node n_2, which combines the subframeworks $F_{\geq n_3}$ and $F_{\geq n_8}$. Consider the coloring C' for node n_3 and the coloring C'' for node n_8 with $C'(c) = C''(c) = in$ and $C'(d) = C''(d) = out$. As $[C'] = [C'']$, i.e. the extensions coincide on the intersection $X_{\geq n_3} \cap X_{\geq n_8}$, we can join these colorings without causing any conflict. Thus we obtain C with $C(c) = in$ and $C(d) = out$ for the node n_{12}. Now let us consider the coloring D'' for node n_8 with $D''(c) = def$ and $D''(d) = in$. We have that $[C'] \neq [D'']$ and one can see that the set $[C'] \cup [D''] = \{c, d\}$ has a conflict. Hence the pair C', D'' does not lead to a vcoloring for the node n_{12}. ◇

Lemmas 3–6 show that vcolorings provide us with exactly the same information as valid colorings. The following result thus immediately follows by structural induction over a given tree decomposition.

Theorem 5. *Let $(\mathcal{T}, \mathcal{X})$ be a tree decomposition of an AF $F = (A, R)$. Then, for each coloring C for a node $t \in \mathcal{T}$, it holds that C is a valid coloring for t iff C is a vcoloring for t.*

Let us now describe how credulous acceptance can be performed via vcolorings: We just have to mark each coloring which assigns the value in to the argument we are interested in and accordingly pass this mark up to the root. In other words, we mark a coloring if it is constructed by using at least one marked coloring. If the coloring of the root has the mark, then we know that credulous acceptance for this argument holds.

Example 17. Recall the computation from Example 12 in Figure 9. We now consider the problem of deciding if the argument d is credulously accepted. The argument d is introduced in the nodes n_3 and n_{11} thus we mark all their vcolorings C satisfying $C(d) = in$ and illustrate this with a ✓in the last column of the table. Consider, for instance, the node n_8 with the colorings $C_1(c) = in, C_1(d) = def$, $C_2(c) = def, C_2(d) = in$ and $C_3(c) = out, C_3(d) = out$. The child node n_9 has colorings $C'_1(d) = in$ and $C'_2(d) = out$, the first marked for credulous acceptance. As C_2 is constructed via the marked C'_1 ($C_2 = C'_1 + \{c\}$) it is also marked and as C_1 and C_3 are both constructed via C'_2 ($C_1 = C'_2 \dot{+} \{c\}$, $C_3 = C'_2 + \{c\}$) they are not marked. ◇

Since vcolorings can be computed efficiently (for bounded bag size) we obtain the following result for such an algorithm, assuming that AFs come together with a nice tree decomposition of suitable width. The upper bound on the time complexity is obtained by considering the maximum number of vcolorings per node and assuming a straightforward method (e.g., nested loops) for computing a node's vcolorings from the vcolorings at the child node(s).

Theorem 6. *Deciding* CA *for an AF* $F = (A, R)$ *of tree-width* $k-1$ *can be done in time* $O(10^k \cdot k \cdot |A|)$.

Proof. Let $(\mathcal{T}, \mathcal{X})$ be a tree decomposition of an AF $F = (A, R)$. First, we observe that the number of colorings for each node $t \in \mathcal{T}$ is bounded by 4^k, since there at most k arguments in $X_t \in \mathcal{X}$ and there are only 4 colors $\{in, out, def, att\}$ to assign to these arguments. We assume that the set of vcolorings for a node t is stored in a table with 4^k rows. Each row contains a coloring plus an additional bit which indicates if this coloring is a vcoloring. We assume that, given a coloring C, we can find the corresponding row in this table within time $O(k)$. We have to show that computing the vcolorings at each node $t \in \mathcal{T}$ is feasible in time $O(10^k \cdot k)$ in a single bottom-up traversal of $\mathcal{T}$. Since the number of nodes of $\mathcal{T}$ may be assumed to be bounded by $O(|A|)$, the desired upper bound of the theorem follows immediately. We prove the upper bound $O(10^k \cdot k)$ for the time needed at each node $t \in \mathcal{T}$ by distinguishing the four types of nodes in a nice tree decomposition.

At a *LEAF* node t, we inspect each coloring C in the table at t and check in time $O(k^2)$ if C is a vcoloring, i.e., conflict-free. To this end, we simply consider all pairs of arguments in the bag. This yields the bound $O(4^k \cdot k^2)$.

For a *FORGET* node t, we iterate over all vcolorings C' for the successor node t' and check for each such C' if $C'(a) \neq att$. If this is the case, we compute the coloring $C = C' - a$ in time $O(k)$. Then we access in time $O(k)$ the coloring C in the table at t and set the vcoloring-bit. In total, we can compute the vcoloring-table at t in time $O(4^k \cdot k)$. An *INSERT* node t is treated similarly.

In a *JOIN* node t, the vcolorings are computed by combining two colorings of the successors t' and t''. In a naive implementation, up to $4^k \cdot 4^k = 4^{2k} = 16^k$ pairs exist. However, we show that only 10^k pairs have to be considered. By using appropriate data structures, we can implement the join such that we only consider pairs (C', C'') with $[C'] = [C'']$. For instance, we can sort the colorings in the tables at t' and t'' in lexicographical order by treating in as 1 and the other values (i.e., def, att, out) as 0. In the sorted table, the colorings D, D' with $[D] = [D']$ are in contiguous rows. This sorting requires time $O(4^k \cdot k)$.

Let C be a coloring over k arguments with $m \leq k$ arguments mapped to in. Then, for each argument with $C(a) \neq in$, we can choose any color in $\{out, def, att\}$ without effecting the set $[C]$. Thus there exist at most 3^{k-m} different colorings C' such that $[C] = [C']$. For every m, there are $\binom{k}{m}$ different choices of m arguments and thus there are $\binom{k}{m} \cdot 3^{k-m}$ colorings C in the first table mapping m arguments to in. Each of these colorings can be combined with 3^{k-m} colorings from the second table. Hence we have at most $\binom{k}{m} 3^{k-m} \cdot 3^{k-m}$ join pairs produced by colorings that map m arguments to in. The sum over all possible m yields the desired upper bound for the total number of join pairs: $\sum_{m=0}^{k} \binom{k}{m} \cdot 3^m \cdot 3^m = \sum_{m=0}^{k} \binom{k}{m} \cdot 9^m = 10^k$. The latter equality follows from the combinatorial identity $\sum_{i=0}^{n} \binom{n}{i} \cdot (l)^i = (l+1)^n$. Each joinable pair (C', C'') can be handled in time $O(k)$ (for computing $C = C' \bowtie C''$ and setting the vcoloring-bit of C). In total, the vcolorings for a *JOIN* node can thus be computed in time $O(10^k \cdot k)$. □

As hinted at in Example 12, our dynamic programming approach can be easily extended so as to *count the number of admissible sets*. In fact, we just need to add the computation of the # column to our algorithm (which is straightforward due to Lemma 2). Finally, we also emphasize

the possibility of *enumerating (with linear delay) all admissible sets* (using a second top-down pass of the tree similar as sketched in [26]).

4.2 Characterizing preferred extensions

So far, we have solved the credulous acceptance problem via a certain characterization for the admissible sets. For skeptical reasoning, we have to characterize preferred extensions rather than the admissible sets. We thus need a more complicated data structure. Instead of colorings for nodes t we shall use pairs (C, Γ) where C is a coloring for t and Γ is a set of colorings for t. The set Γ of "certificates" contains further colorings which characterize $X_{>t}$-restricted admissible sets strictly larger than the $X_{>t}$-restricted admissible sets characterized by C. Intuitively, Γ represents those $X_{>t}$-restricted admissible sets which may ultimately keep the elements in $e_t(C)$ from being maximal.

Definition 17. *Let $(\mathcal{T}, \mathcal{X})$ be a tree decomposition of an AF F, $t \in \mathcal{T}$, and (C, Γ) a pair with C being a coloring for t and Γ being a set of colorings for t. We call (C, Γ) simply a* pair *for t and define $e_t(C, \Gamma)$ as the collection of sets S which satisfy the following conditions:*

(i) $S \in e_t(C)$*;*

(ii) for all $C' \in \Gamma$, there is an $E \in e_t(C')$ such that $S \subset E$;

(iii) for all $X_{>t}$-restricted admissible (for $F_{\geq t}$) sets E with $S \subset E$, there exists some $C' \in \Gamma$ with $E \in e_t(C')$.

If $e_t(C, \Gamma) \neq \emptyset$, (C, Γ) is a valid pair for *t.*

The following technical lemmas mirror Lemma 1 and Lemma 2.

Lemma 9. *Let $(\mathcal{T}, \mathcal{X})$ be a tree decomposition of an AF F, $t \in \mathcal{T}$, and S an $X_{>t}$-restricted admissible set for $F_{\geq t}$. Then, there is a pair (C, Γ) for t such that $S \in e_t(C, \Gamma)$.*

Proof. Let S be an $X_{>t}$-restricted admissible set for $F_{\geq t}$. By Lemma 1, there exists a coloring C with $S \in e_t(C)$. Now let $\mathcal{E} = \{E \mid E \text{ is } X_{>t}\text{-restricted admissible for } F_{\geq t} \text{ and } S \subset E\}$. Moreover, let $\Gamma = \{C' \mid \exists E \in \mathcal{E}, \text{ s.t. } E \in e_t(C')\}$. We claim that $S \in e_t(C, \Gamma)$. To prove this, we check the conditions (i) – (iii) from Definition 17: (i) $S \in e_t(C)$ by the selection of C. (ii) For all $C' \in \Gamma$, there exists $E \in e_t(C')$ with $S \subset E$; this follows by the construction of Γ from $\mathcal{E}$. (iii) For all $X_{>t}$-restricted sets E that are admissible in $F_{\geq t}$ with $S \subset E$, there exists $C' \in \Gamma$ with $E \in e_t(C')$; again this follows by the construction of Γ from $\mathcal{E}$. □

Lemma 10. *Let $(\mathcal{T}, \mathcal{X})$ be a tree decomposition of an AF F, $t \in \mathcal{T}$, and let $(C, \Gamma), (C', \Gamma')$ be different pairs for t (but not necessarily $C \neq C'$). Then, $e_t(C, \Gamma) \cap e_t(C', \Gamma') = \emptyset$.*

Proof. If $C \neq C'$ then, by Lemma 2, $e_t(C) \cap e_t(C') = \emptyset$ and our claim follows. Thus, it remains to consider pairs $(C, \Gamma), (C, \Gamma')$ with $\Gamma \neq \Gamma'$. W.l.o.g., we assume that there exists a coloring $\bar{C}$ for

t such that $\bar{C} \in \Gamma$ but $\bar{C} \notin \Gamma'$. In order to show that $e_t(C,\Gamma) \cap e_t(C,\Gamma') = \emptyset$, we prove that none of the sets $S \in e_t(C,\Gamma)$ is contained in $e_t(C,\Gamma')$.

Let S be an arbitrary set in $e_t(C,\Gamma)$. Suppose to the contrary that S is also contained in $e_t(C,\Gamma')$. By Definition 17 (applied to $e_t(C,\Gamma)$), there exists an $X_{>t}$-restricted admissible set $E \in e_t(\bar{C})$ for $F_{\geq t}$ such that $S \subset E$. By Definition 17 (applied to $e_t(C,\Gamma')$), there exists a coloring $C^* \in \Gamma'$ such that $E \in e_t(C^*)$. By Lemma 2, the colorings $\bar{C}$ and C^* coincide. Thus, $\bar{C} \in \Gamma'$, a contradiction. □

Hence, each element $S \in e_t(C,\Gamma)$ is an $X_{>t}$-restricted admissible set for $F_{\geq t}$ and each $X_{>t}$-restricted admissible set for $F_{\geq t}$ is characterized by some valid pair for t.

Now that we have augmented valid colorings with sets of valid colorings, we can identify the preferred extensions of F in the root node. Recall that the root node r of $\mathcal{T}$ has an empty bag, thus there are only two possible pairs for r, namely $(\epsilon, \emptyset)$ and $(\epsilon, \{\epsilon\})$, where ϵ is the empty coloring. Only the first pair corresponds to preferred extensions (see Definition 17) and we have the following relationship.

Proposition 2. *Let r be the root of a tree decomposition $(\mathcal{T}, \mathcal{X})$ of an AF F. Then, $e_r(\epsilon, \emptyset) = pref(F)$.*

Proof. We recall that $e_r(\epsilon) = adm(F)$. To show the set inclusion $e_r(\epsilon, \emptyset) \subseteq pref(F)$, let S be an arbitrary set such that $S \in e_r(\epsilon, \emptyset)$. By Definition 17 (i) we obtain that S is admissible for $F_{\geq r} = F$. Further by (iii) and the fact that $\Gamma = \emptyset$ we conclude that there is no proper superset of S being admissible for F, i.e. S is a preferred extension of F. It remains to show that $e_r(\epsilon, \emptyset) \supseteq pref(F)$. Thus let $S \in pref(F)$ be an arbitrary preferred extension of F. By Lemma 9 and Lemma 10 we get that there exists a unique pair (C, Γ) such that $S \in e_t(C, \Gamma)$. Since the root node has an empty bag, $C = \epsilon$ and further, by Definition 17 (ii) and the fact that S is a $\subseteq$-maximal admissible set for F, we conclude that $\Gamma = \emptyset$ has to hold as well. □

Thus, our pairs have the desired property to characterize preferred extensions. It remains to find an efficient way to compute them. As we did for admissible sets, we shall employ vcolorings for this purpose. However, the bottom-up computation now has to be applied to certificates as well, which makes the definition more involved. To handle the certificates, we have to extend the definition of the operators for vcolorings (see Figure 8) to *sets of vcolorings*. By slight abuse of notation, we overload the operators $-$, $+$, $\dot{+}$, and $\bowtie$ as follows:

$$\Gamma - a = \{C - a \mid C \in \Gamma \text{ and } C(a) \neq att\}$$
$$\Gamma + a = \{C + a \mid C \in \Gamma\}$$
$$\Gamma \dot{+} a = \{C \dot{+} a \mid C \in \Gamma, a \not\rightarrowtail a, [C] \not\rightarrowtail a \text{ and } a \not\rightarrowtail [C]\}$$
$$\Gamma \bowtie \Delta = \{C \bowtie D \mid C \in \Gamma, D \in \Delta, \text{ and } [C] = [D]\}$$

We observe that if Γ is a set of vcolorings for a node t' having t as its parent node, applying an operator corresponding to the node type of t results in a set of vcolorings for t. For the Join operator we additionally have to assume that Δ is a set of vcolorings for a node t'' which shares t as a parent node with t'. As an analogue to vcolorings we formally define now the concept of *vpairs* as follows.

Definition 18. *Let $(\mathcal{T}, \mathcal{X})$ be a tree decomposition of an AF F and let $t \in \mathcal{T}$ be a node with t', t'' its possible children. Depending on the node type of t we define a* vpair *for t as follows:*

- *LEAF: Each (C, Γ) where $C \in \mathcal{C}_t$ and $\Gamma = \{C' \in \mathcal{C}_t \mid [C] \subset [C']\}$ is a vpair for t.*
- *FORGET: If (C', Γ') is a vpair for t', $X_t = X_{t'} \setminus \{a\}$, and $C'(a) \neq att$, then*
  - *$(C' - a, \Gamma' - a)$ is a vpair for t.*
- *INSERT: If (C', Γ') is a vpair for t' and $X_t = X_{t'} \cup \{a\}$, then*
  - *$(C' + a, (\Gamma' + a) \cup (\Gamma' \dot{+} a) \cup (\{C'\} \dot{+} a))$ is a vpair for t;*
  - *if $C' \dot{+} a$ is a vcoloring then $(C' \dot{+} a, \Gamma' \dot{+} a)$ is a vpair for t as well.*
- *JOIN: If (C', Γ') is a vpair for t', (C'', Γ'') is a vpair for t'', and $[C'] = [C'']$, then*
  - *$(C' \bowtie C'', (\Gamma' \bowtie \Gamma'') \cup (\{C'\} \bowtie \Gamma'') \cup (\Gamma' \bowtie \{C''\}))$ is a vpair for t.*

A few words about the certificates of $C' + a$ in the above definition are in order. We consider here a new argument a but do not add it to $[C]$. Now each certificate $E' \in \Gamma'$ may give rise to two certificates of $C' + a$. First, if we do not add a to $[E']$, we get that $E' + a$ is still a certificate for $C' + a$. But we possibly also get a certificate for $C' + a$ if we do add a to $[E]$, namely $E' \dot{+} a$ – hence the union with $(\Gamma' \dot{+} a)$. Finally, we may also get a new certificate of $C' + a$ if we take C' itself and add a to it – hence the union with $\{C'\} \dot{+} a$.

Similar considerations underly the certificates of $C' \bowtie C''$. Here we combine vcolorings C, C', of different subframeworks $F_{\geq t'}, F_{\geq t''}$ to a vcoloring for the union of these subframeworks. Now let D be a certificate of C and D' a certificate of C' then clearly $D \bowtie D'$ is a certificate for $C \bowtie C'$. But further we have that $C \bowtie D'$ and $D \bowtie C''$ are also certificates for $C \bowtie C'$. This relies on the fact that for a $\subset$-relationship in the combined AF, it suffices to have a $\subset$-relationship in one of the subframeworks and a $\subseteq$-relationship in the other.

Example 18. Recall the AF from Example 9. The computation of vpairs for nodes t is illustrated in Figure 10. As before we use the #-column to notate the cardinality of the sets $e_t(C, \Gamma)$ for better readability. Furthermore, we use the $\lightning$ symbol to illustrate how to decide SA – a detailed explanation of this concept follows in Example 23. Also observe that we indeed have pairs (C, Γ) and (C, Γ') with $\Gamma \neq \Gamma'$ for the same node. An example is node n_5 with bag $\{b, c\}$ on the left branch and the coloring C_1 with $C_1(b) = def$ and $C_1(c) = in$, i.e. $[C_1] = \{c\}$. We have that $e_t(C_1) = \{\{c\}, \{a, c\}\}$. However, $e_t(C_1, \{C_1\}) = \{\{c\}\}$ (since we have $\{a, c\}$ as certificate), while $e_t(C_1, \emptyset) = \{\{a, c\}\}$. ◇

In what follows, we show that vpairs match the concept of valid pairs and thus are appropriate for our purposes. Similarly as for vcolorings, we will do this step-by-step distinguishing between the different node types. We start with the nodes of type *LEAF*:

Lemma 11. *For any* LEAF *node in a tree decomposition of an AF, its vpairs coincide with its valid pairs.*

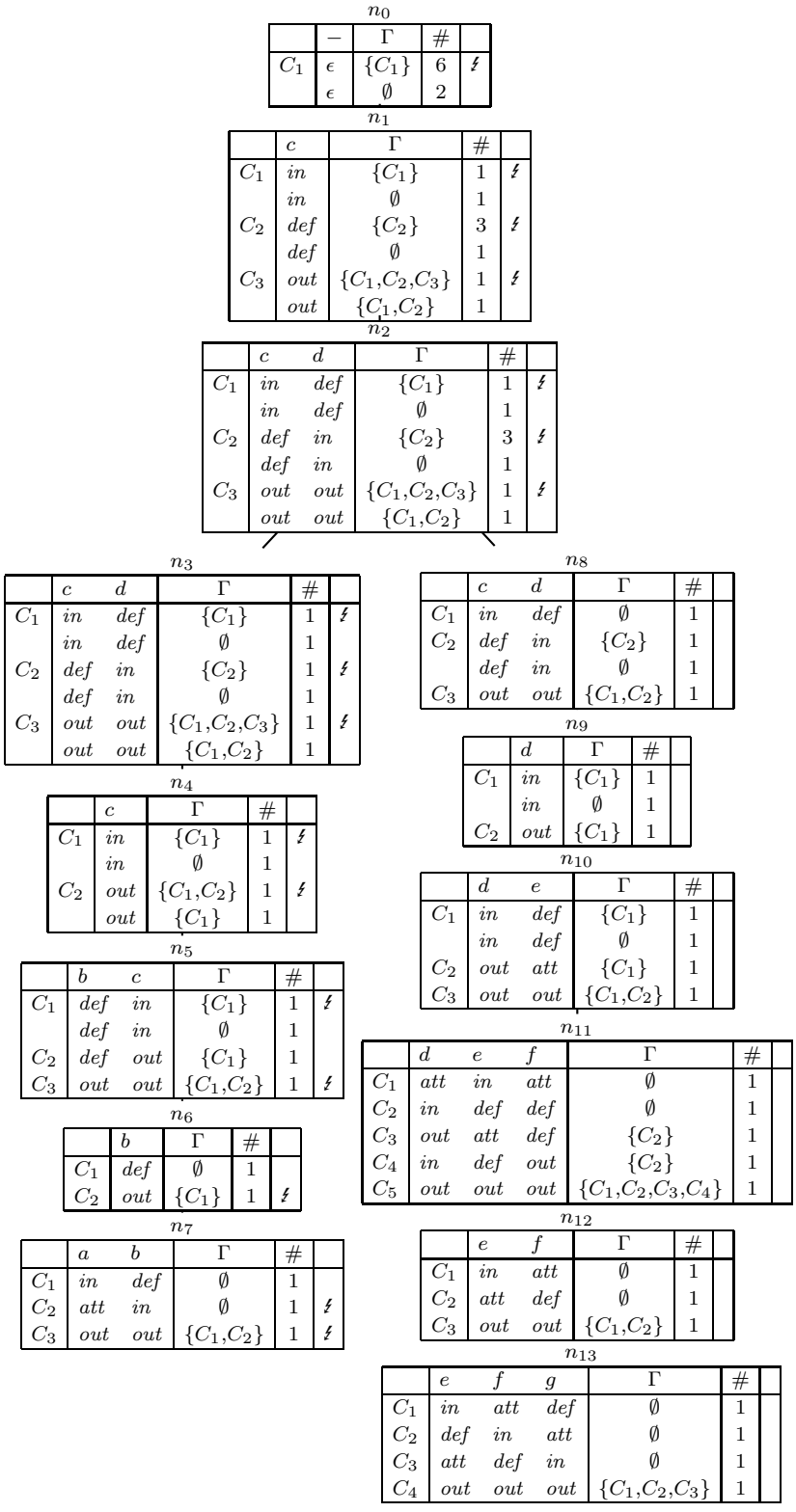


Figure 10: Computation of vpairs for the example AF.

Proof. Let $(\mathcal{T}, \mathcal{X})$ be a tree decomposition of F and t a leaf node in $\mathcal{T}$. The $X_{>t}$-restricted admissible sets for $F_{\geq t}$ coincide with the sets $[C]$ for the valid colorings $C \in \mathcal{C}_t$. Moreover, the valid colorings and vcolorings for t coincide by Lemma 3. Now let (C, Γ) be a valid pair for t. Then, by Definition 17, $[C] \in e_t(C, \Gamma)$. Hence, by Definition 18, (C, Γ) is a vpair for t. Conversely, let (C, Γ) be a vpair for t and let $S = [C]$. By Definition 16, S is $X_{>t}$-restricted admissible for $F_{\geq t}$. Hence, by Definition 17 and Definition 18, $S \in e_t(C, \Gamma)$. (C, Γ) is thus a valid pair for node t. □

Example 19. Consider, for instance, the *LEAF* node n_{13} in Figure 10. As mentioned before we have four valid colorings C_1, C_2, C_3, C_4 that correspond to the $\emptyset$-restricted admissible sets $\{e\}, \{f\}, \{g\}, \emptyset$ of $F_{\geq n_{13}}$. One can see that the first three sets are $\subseteq$-maximal in being $\emptyset$-restricted admissible for $F_{\geq n_{13}}$ and thus correspond to the vpairs $(C_1, \emptyset), (C_2, \emptyset), (C_3, \emptyset)$ of n_{13}. On the other hand, $\emptyset$ has three such supersets, namely $\{e\}, \{f\}, \{g\}$, and thus the corresponding vpair is $(C_4, \{C_1, C_2, C_3\})$. ◇

Next we consider *FORGET* nodes:

Lemma 12. *For any* FORGET *node t in a tree decomposition of an AF, vpairs and valid pairs coincide, if they coincide in the child node t' of t.*

Proof. Let $(\mathcal{T}, \mathcal{X})$ be a tree decomposition of $F = (A, R)$, t a *FORGET* node in $\mathcal{T}$, and t' the child node of t. We have that $X_t = X_{t'} \setminus \{a\}$, for some argument $a \in X_{t'}$.

First we show that every valid pair for t is also a vpair for t. Thus let (C, Γ) be a valid pair for t. Then there exists a set $S \in e_t(C, \Gamma)$. In particular, S is $X_{>t}$-restricted admissible for $F_{\geq t}$, and hence, also $X_{>t'}$-restricted admissible for $F_{\geq t'} = F_{\geq t}$. Thus, by Lemma 9, there exists a valid pair (C', Γ') for t' with $S \in e_{t'}(C', \Gamma')$. By assumption, (C', Γ') is a vpair for t'. Since S is $X_{>t}$-restricted admissible for $F_{\geq t}$ and $S \in e_{t'}(C')$, we have $C'(a) \neq att$. Then $(C' - a, \Gamma' - a)$ is a vpair for t. We claim that $(C' - a, \Gamma' - a) = (C, \Gamma)$ holds. The equality $C' - a = C$ is shown as in the proof of Lemma 4.

To show $\Gamma' - a = \Gamma$, we first consider the inclusion $\Gamma' - a \subseteq \Gamma$: Let $D' \in \Gamma'$ with $D'(a) \neq att$. By condition (ii) of Definition 17, there exists an $X_{>t'}$-restricted admissible set E for $F_{\geq t'}$ with $S \subset E$ and $E \in e_{t'}(D')$. By $D'(a) \neq att$, we know that E is also $X_{>t}$-restricted admissible. Hence, by condition (iii) of Definition 17, there exists $D \in \Gamma$ with $E \in e_t(D)$. As in the proof of Lemma 4, we thus have $D = D' - a$. Hence, $\Gamma' - a \subseteq \Gamma$.

Now consider an arbitrary set D in Γ. By condition (ii) of Definition 17, there exists an $X_{>t}$-restricted admissible set E for $F_{\geq t}$ with $S \subset E$ and $E \in e_t(D)$. By condition (iii) of Definition 17 and since E is also $X_{>t'}$-restricted admissible for $F_{\geq t'}$, there exists $D' \in \Gamma'$ with $E \in e_{t'}(D')$. As in the proof of Lemma 4, we thus have $D = D' - a$. Hence, $\Gamma \subseteq \Gamma' - a$.

We now show that every vpair for the *FORGET* node t is also valid pair for t. Let (C, Γ) be a vpair for t, i.e., there exists a vpair (C', Γ') for node t' with $C'(a) \neq att$ and $(C, \Gamma) = (C' - a, \Gamma' - a)$. By assumption, (C', Γ') is a valid pair for t'. Hence, there exists $S \in e_{t'}(C', \Gamma')$. We claim that also $S \in e_t(C, \Gamma)$ holds. As in the proof of Lemma 4, $S \in e_t(C)$ holds since $C = C' - a$. It remains to show that also conditions (ii) and (iii) of Definition 17 are fulfilled.

To show condition (ii), let $D \in \Gamma$, i.e., D is of the form $D = D' - a$ for some $D' \in \Gamma'$ with $D'(a) \neq att$. Since $S \in e_{t'}(C', \Gamma')$, there exists $E \in e_{t'}(D')$ with $S \subset E$. As in the proof of

Lemma 4, then also $E \in e_t(D' - a)$. To show condition (iii), let E be $X_{>t}$-restricted admissible for $F_{\geq t}$ with $S \subset E$. Then E is also $X_{>t'}$-restricted admissible for $F_{\geq t'}$, and therefore, there exists $D' \in \Gamma'$ with $E \in e_{t'}(D')$. Since E is $X_{>t}$-restricted admissible, we have $D'(a) \neq att$. But then, as in the proof of Lemma 4, also $E \in e_t(D' - a)$. □

Example 20. As an example for a *FORGET* node, consider the node n_{12} in Figure 10. which removes argument g from its child node n_{13}. The vpairs of n_{12} are obtained from the vpairs of n_{13} with the exception of the vpair $(C'_2, \emptyset)$ with $[C'_2] = \{f\}$. This is due to the fact that C'_2 is not further propagated as a vcoloring; thus also the vpair $(C'_2, \emptyset)$ is not propagated by definition (since $C'_2(g) = att$). For the same reason, we also have to eliminate C'_2 from the certificates of the vpair $(C'_4, \{C'_1, C'_2, C'_3\})$ of n_{13} which leads to the vpair $(C_3, \{C_1, C_2\})$ for n_{12}. ◇

We continue with nodes of type *INSERT*.

Lemma 13. *For any* INSERT *node t in a tree decomposition of an AF, vpairs and valid pairs coincide, if they coincide in the child node t' of t.*

Proof. Let $(\mathcal{T}, \mathcal{X})$ be a tree decomposition of $F = (A, R)$, t an *INSERT* node in $\mathcal{T}$, and t' the child node of t. Hence we have that $X_t = X_{t'} \cup \{a\}$ for some argument $a \in A$.

First we show that every valid pair for t is also a vpair for t. Thus let (C, Γ) be a valid pair for t. Then there exists $S \in e_t(C, \Gamma)$, which is $X_{>t}$-restricted admissible for $F_{\geq t}$ and further the set $S' = S \setminus \{a\}$ is $X_{>t'}$-restricted admissible for $F_{\geq t'}$. Thus, by Lemma 9, there exists a valid pair (C', Γ') for t' with $S' \in e_{t'}(C', \Gamma')$. By assumption, (C', Γ') is a vpair for t'. Then $(C' + a, \Gamma_1)$ with $\Gamma_1 = (\Gamma' + a) \cup (\Gamma' \dot{+} a) \cup (\{C'\} \dot{+} a)$ is a vpair for t and further if $[C'] \cup a$ is conflict-free in $F_{t'}$, then also $(C' \dot{+} a, \Gamma' \dot{+} a)$ is a vpair. We claim that either $(C' + a, \Gamma_1) = (C, \Gamma)$ or $(C' \dot{+} a, \Gamma' \dot{+} a) = (C, \Gamma)$ holds. As shown in the proof of Lemma 5 we have that either $C = C' + a$ (if $a \notin S$) or $C = C' \dot{+} a$ (if $a \in S$) holds. In the following we show that also the respective sets of certificates coincide. To this end we distinguish between the two mentioned cases, namely $a \notin S$ and $a \in S$, respectively:

1) Assume $a \notin S$: To derive $\Gamma_1 = \Gamma$, we first show the relation $\Gamma_1 \subseteq \Gamma$; this can be split up into the following three statements:

$$(\alpha)\ \Gamma' + a \subseteq \Gamma, \quad (\beta)\ \Gamma' \dot{+} a \subseteq \Gamma, \quad \text{and} \quad (\gamma)\ \{C'\} \dot{+} a \subseteq \Gamma.$$

To show (α) and (β), consider $D' \in \Gamma'$. By condition (ii) of Definition 17, there exists an $X_{>t'}$-restricted admissible set E' for $F_{\geq t'}$ with $S' \subset E'$ and $E' \in e_{t'}(D')$. As we have here $S = S'$, we obtain $S \subset E = E'$ for (α), and $S \subset E = E' \cup \{a\}$ for (β). In the first case we have that E is conflict-free in $F_{\geq t}$ by definition, and further as $X_{>t} = X_{>t'}$ and $a \notin X_{>t}$ it is also an $X_{>t}$-restricted admissible set for $F_{\geq t}$. In the latter case E is conflict-free in $F_{\geq t}$ iff the set $[D'] \cup \{a\}$ is conflict-free. This is due to the definition of tree decompositions which ensures that there are no attacks between the set $X_{>t}$ and the new argument a. Using that a is not attacked by $X_{>t}$ we get that if E is conflict-free in $F_{\geq t}$ then E is also an $X_{>t}$-restricted admissible set for $F_{\geq t}$.

Now by condition (iii) of Definition 17 there exists $D \in \Gamma$ such that $E \in e_t(D)$. As shown in the proof of Lemma 5, it holds that in case (α) $D = D' + a$ and in case (β) $D = D' \dot{+} a$.

Next we prove statement (γ). To do this let us consider the set $(\{C'\}\dot{+}a)$. If $(\{C'\}\dot{+}a) = \emptyset$ statement (γ) is trivially true. Otherwise we have that $S \cup \{a\} \in e_t(C'\dot{+}a)$ and as $S \subset S \cup \{a\}$ that $C'\dot{+}a \in \Gamma$. Hence, $\{C'\}\dot{+}a \subseteq \Gamma$ and finally $\Gamma_1 \subseteq \Gamma$.

To prove $\Gamma \subseteq \Gamma_1$, consider an arbitrary $D \in \Gamma$. By condition (ii) of Definition 17, there exists an $X_{>t}$-restricted admissible set E for $F_{\geq t}$ with $S \subset E$ and $E \in e_t(D)$. By the assumption $a \notin S$, i.e. $S = S'$, we have that for $E' = E \setminus \{a\}$ either $S' \subset E'$ or $E = S \cup \{a\}$ (i.e. $E' = S$) holds. In both cases we have that E' is $X_{>t'}$-restricted admissible for $F_{\geq t'}$ and thus there exists $D' \in \Gamma'$ with $E' \in e_{t'}(D')$. For the case $S' \subset E'$ we can use the proof of Lemma 5, to show that that either $D = D' + a$ or $D = D'\dot{+}a$. In the other case, we also can use the proof of Lemma 5, but now to show that $D = C'\dot{+}a$. Hence, $\Gamma \subseteq \Gamma_1$.

2) Assume $a \in S$: To show $\Gamma'\dot{+}a = \Gamma$, we first consider the inclusion $\Gamma'\dot{+}a \subseteq \Gamma$: Consider $D' \in \Gamma'$. By condition (ii) of Definition 17, there exists an $X_{>t'}$-restricted admissible set E' for $F_{\geq t'}$ with $S' \subset E'$ and $E' \in e_{t'}(D')$. As by assumption $S = S' \cup \{a\}$ we have that $S \subset E = E' \cup \{a\}$. Further as in case (1) if $[D'] \cup \{a\}$ is conflict-free then E is $X_{>t}$-restricted admissible for $F_{\geq t}$. In this case we get by Definition 17 that there exists $D \in \Gamma$ such that $E \in e_t(D)$. As shown in the proof of Lemma 5, it holds that $D = D'\dot{+}a$. Hence, $\Gamma'\dot{+}a \subseteq \Gamma$.

   To prove $\Gamma \subseteq \Gamma_1$, consider an arbitrary D in Γ. By condition (ii) of Definition 17, there exists an $X_{>t}$-restricted admissible set E for $F_{\geq t}$ with $S \subset E$ and $E \in e_t(D)$. We have that $S' \subset E' = E \setminus \{a\}$ and further that E' is $X_{>t'}$-restricted admissible for $F_{\geq t'}$. Thus there exists $D' \in \Gamma'$ with $E' \in e_{t'}(D')$. As in the proof of Lemma 5, we get that $D = D'\dot{+}a$. Hence, $\Gamma \subseteq \Gamma'\dot{+}a$ holds.

It remains to show that every vpair for an *INSERT* node is also a valid pair. Thus let (C, Γ) be a vpair for t, i.e., there exists a vpair (C', Γ') for node t' such that either (1) $(C, \Gamma) = (C' + a, \Gamma_1)$ (Γ_1 defined as above) or (2) $(C, \Gamma) = (C'\dot{+}a, \Gamma'\dot{+}a)$ with $[C] \cup \{a\}$ being conflict-free in F_t. By assumption, (C', Γ') is a valid pair for t' and thus there exists $S' \in e_{t'}(C', \Gamma')$. To show that (C, Γ) is a valid pair for t we distinguish the cases (1) and (2) as follows:

1) As in the proof of Lemma 5, $S = S' \in e_t(C)$ holds since $C = C' + a$. It remains to show that also conditions (ii) and (iii) of Definition 17 are fulfilled. To show condition (ii), consider an arbitrary $D \in \Gamma$, i.e., D is either of the form

   (a) $D = D' + a$,

   (b) $D = D'\dot{+}a$ with $[D] \cup \{a\}$ conflict-free in F_t or

   (c) $D = C'\dot{+}a$ with $[C'] \cup \{a\}$ conflict-free in F_t

   Since $S \in e_{t'}(C', \Gamma')$, there exists $E' \in e_{t'}(D')$ with $S \subset E'$. In case (a), we follow the proof of Lemma 5, and obtain $E' \in e_t(D' + a)$. For case (b), we get by the construction of D that $E = E' \cup \{a\}$ is conflict-free in $F_{\geq t}$. Once more we can use the fact $X_t \not\rightarrow a$ to obtain that E is an $X_{>t}$-restricted admissible set for $F_{\geq t}$. Further $S \subset E$ and as in the proof

of Lemma 5, then also $E \in e_t(D'\dot{+}a)$. Finally, for (c) the construction of $D = C'\dot{+}a$ yields that $E = S \cup \{a\}$ is conflict-free in $F_{\geq t}$ and thus as before E is an $X_{>t}$-restricted admissible set for $F_{\geq t}$. Hence, as in the proof of Lemma 5, also $E \in e_t(C'\dot{+}a)$ holds. Further, as $a \notin S$, we have that $S \subset E$.

To show condition (iii), consider an arbitrary $X_{>t}$-restricted admissible set E for $F_{\geq t}$ such that $S \subset E$. Then $E' = E \setminus \{a\}$ is $X_{>t'}$-restricted admissible for $F_{\geq t'}$. If $E' = S$ then C' is the unique vcoloring such that $E \in e_{t'}(C')$. Otherwise if $E' \neq S$ it holds that $S \subset E'$ and thus, there exists $D' \in \Gamma'$ with $E \in e_{t'}(D')$. Since E is $X_{>t}$-restricted admissible for $F_{\geq t}$, we have that there is a unique vcoloring D such that $E \in e_t(D)$. But then, as in the proof of Lemma 5, either $D = C'\dot{+}a$, $D = D' + a$ or $D = D'\dot{+}a$ holds.

2) By the assumption $C = C'\dot{+}a$ we have that $S = S' \cup \{a\} \in e_t(C)$. It remains to show that the vpair (C, Γ) also satisfies conditions (ii) and (iii) of Definition 17. To show condition (ii), consider $D \in \Gamma$, i.e., D is of the form $D = D'\dot{+}a$ with $[D'] \cup \{a\}$ conflict-free in F_t. Since $S' \in e_{t'}(C', \Gamma')$, there exists $E' \in e_{t'}(D')$ with $S' \subset E'$. By the construction of D we have that $E = E' \cup \{a\}$ is conflict-free and thus that E is an $X_{>t}$-restricted admissible set for $F_{\geq t}$. By definition of E it holds that $S \subset E$ and further, as in the proof of Lemma 5, we get that $E \in e_t(D'\dot{+}a)$. To show condition (iii) of Definition 17, let E be an $X_{>t}$-restricted admissible set for $F_{\geq t}$ such that $S \subset E$. Then $E' = E \setminus \{a\}$ is $X_{>t'}$-restricted admissible for $F_{\geq t'}$ and $S \setminus \{a\} = S' \subset E'$. Thus, there exists $D' \in \Gamma'$ with $E' \in e_{t'}(D')$. Since E is $X_{>t}$-restricted admissible, we have that there is a unique vcoloring D such that $E \in e_t(D)$. But then, as in the proof of Lemma 5, $D = D'\dot{+}a$ holds.

□

Example 21. Consider the *INSERT*-node n_{11} in Figure 10, which adds the argument d. Let us illustrate how vpairs of n_{11} are obtained from the vpairs of n_{12}. For instance, consider the vpair $(C'_3, \{C'_1, C'_2\})$ of n_{12}, with $[C'_3] = \emptyset$. There are two ways to incorporate the argument d for the resulting vpairs of n_{11}. We first consider adding the argument d to the extensions, i.e. we set $C(d) = in$. As the certificates represent supersets we have to extend them in the same way, otherwise the $\subset$-relation would be violated. In our example we have to consider $C'_1\dot{+}d$ and $C'_2\dot{+}d = C_2$. In the first case we have that the set $[C'_1] \cup d$ contains a conflict and thus it is neither a vcoloring nor a certificate. But the set $[C'_2] \cup d$ is conflict-free and thus we obtain C_2 as certificate and we end up with the vpair $(C_4, \{C_2\})$.

Now let us consider not adding d to the vpair $(C'_3, \{C'_1, C'_2\})$. This results in the vcoloring $C_5 = C'_3 + d$, with $C_5(d) = out, C_5(e) = out, C_5(f) = out$. Now both adding d or not adding d to the certificates $\{C'_1, C'_2\}$ preserve the $\subset$-relation. Thus we have both $\{C'_1, C'_2\} + d \subseteq \Gamma$ and $\{C'_1, C'_2\}\dot{+}d \subseteq \Gamma$. The first leads to $\{C_1, C_3\} \subseteq \Gamma$ and the latter as we already have seen to $\{C_2\} \subseteq \Gamma$ Further, as $C_4 = C'\dot{+}d$ represents supersets of $C' + d$, we also get $C_4 \in \Gamma$. In total, we obtain the vpair $(C_5, \{C_1, C_2, C_3, C_4\})$. ◇

Lemma 14. *For any* JOIN *node t in a tree decomposition of an AF, the vpairs coincide with the valid pairs if they coincide on the successors t' and t'' of t.*

Proof. Let $(\mathcal{T}, \mathcal{X})$ be a tree decomposition of $F = (A, R)$ and t a *JOIN* node in $\mathcal{T}$ with successors t' and t''.

First consider an arbitrary valid pair (C, Γ) for t. We show that (C, Γ) is also a vpair. As (C, Γ) is valid there exists an $X_{>t}$-restricted admissible set S for $F_{\geq t}$ such that $S \in e_t(C, \Gamma)$. As in the proof of Lemma 6 we have that there exist unique sets $S_1 \subseteq X_{\geq t'}$ and $S_2 \subseteq X_{\geq t''}$, such that $S_1 \cap X_t = S_2 \cap X_t$ and $S = S_1 \cup S_2$. Further, there exist valid colorings C', C'' such that $S_1 \in e_{t'}(C')$, $S_2 \in e_{t''}(C'')$ and $C = C' \bowtie C''$. Thus there are valid pairs (C', Γ') and (C'', Γ''), such that $S_1 \in e_{t'}(C', \Gamma')$ and $S_2 \in e_{t''}(C'', \Gamma'')$. By assumption these valid pairs are also vpairs.

Now we turn our attention to the set Γ. We first have to show that $\Gamma \subseteq \Gamma^*$ with $\Gamma^* = (\Gamma' \bowtie \Gamma'') \cup (\{C'\} \bowtie \Gamma'') \cup (\Gamma' \bowtie \{C''\})$. For every $D \in \Gamma$ there exists an $X_{>t}$-restricted admissible set $E \in e_t(D)$ such that $S \subset E$. We define $E = E_1 \cup E_2$ analogously to S_1, S_2. Now we have that $S \subset E$ holds iff either

$$\text{(i) } S_1 \subset E_1 \wedge S_2 \subset E_2, \quad \text{(ii) } S_1 = E_1 \wedge S_2 \subset E_2 \quad \text{or} \quad \text{(iii) } S_1 \subset E_1 = S_2 \subset E_2$$

holds. We discuss these three cases separately:

(i) As E_1 is $X_{>t'}$-restricted admissible for $F_{\geq t'}$ and E_2 is $X_{>t''}$-restricted admissible for $F_{\geq t''}$, there exist sets $D' \in \Gamma'$ and $D'' \in \Gamma''$, such that $E_1 \in e_{t'}(D')$ and $E_2 \in e_{t''}(D'')$. By the proof of Lemma 6 we have that $D = D' \bowtie D''$ and thus $D \in \Gamma' \bowtie \Gamma''$.

(ii) As E_2 is $X_{>t''}$-restricted admissible there exists $D'' \in \Gamma''$ such that $E_2 \in e_{t''}(D'')$. By the proof of Lemma 6 we have that $D = C' \bowtie D''$ and thus $D \in (\{C'\} \bowtie \Gamma'')$.

(iii) By the symmetry to case (ii) we get that $D \in (\Gamma' \bowtie \{C''\})$.

Thus we have that $\Gamma \subseteq \Gamma^*$. It remains to show that $\Gamma^* \subseteq \Gamma$ which is equivalent to showing each of the following inclusions

$$\text{(i) } \Gamma' \bowtie \Gamma'' \subseteq \Gamma, \quad \text{(ii) } \{C'\} \bowtie \Gamma'' \subseteq \Gamma \quad \text{and} \quad \text{(iii) } \Gamma' \bowtie \{C''\} \subseteq \Gamma.$$

This can be done as follows:

(i) Consider arbitrary $D' \in \Gamma'$ and $D'' \in \Gamma''$ with $[D'] = [D'']$, $E_1 \in e_{t'}(D')$ and $E_2 \in e_{t''}(D'')$. By Definition 17 we have that $S_1 \subset E_1$ and $S_2 \subset E_2$. We conclude that $S \subset E$ and by the proof of Lemma 6 that $D = D' \bowtie D''$ is the unique coloring such that $E \in e_t(D)$. Therefore $D' \bowtie D'' \in \Gamma$ and thus $\Gamma' \bowtie \Gamma'' \subseteq \Gamma$.

(ii) Consider an arbitrary $D'' \in \Gamma''$ with $[C'] = [D'']$ and $E_2 \in e_{t''}(D'')$. We have that $S \subset E = S_1 \cup E_2$ and that $D = C' \bowtie D''$ is the unique coloring such that $E \in e_t(D)$. Thus $\{C'\} \bowtie \Gamma'' \subseteq \Gamma$.

(iii) By symmetry to (ii).

This shows $\Gamma = \Gamma^*$ and thus every valid pair (C, Γ) is also a vpair.

Now we show that every vpair for t is also a valid pair for t. Thus let (C, Γ) be a vpair for t, i.e., there exists a vpair (C', Γ') for node t' and a vpair (C'', Γ'') for node t'' with $[C'] = [C'']$ such that $(C, \Gamma) = (C' \bowtie C'', \Gamma^*)$ (Γ^* defined as above). By assumption (C', Γ') and (C'', Γ'') are valid pairs. Hence, there exist sets $S_1 \in e_{t'}(C', \Gamma')$ and $S_2 \in e_{t''}(C'', \Gamma')$. As in the proof of Lemma 6, $S = S_1 \cup S_2 \in e_t(C)$ holds since $[C''] = [C']$

It remains to show that (C, Γ) also fulfills conditions (ii) and (iii) of Definition 17. To show condition (ii), consider $D \in \Gamma$, i.e., D is of one of the following forms:

(a) $D = D' \bowtie D''$ for some $D' \in \Gamma'$, $D'' \in \Gamma''$ with $[D'] = [D'']$

(b) $D = C' \bowtie D''$ for some $D'' \in \Gamma''$ with $[C'] = [D'']$

(c) $D = D' \bowtie C''$ for some $D' \in \Gamma'$ with $[C''] = [D']$

We only discuss case (a) here as the cases (b) and (c) are similar: Since $S_1 \in e_{t'}(C', \Gamma')$ and $S_2 \in e_{t''}(C'', \Gamma'')$, there exist $E_1 \in e_{t'}(D')$ and $E_2 \in e_{t''}(D'')$ with $S \subset E_1$, $S \subset E_2$ and $E_1 \cap X_t = E_2 \cap X_t$. As in the proof of Lemma 6, then also $E = E_1 \cup E_2 \in e_t(D' \bowtie D'')$ and $S \subset E$.

To show condition (iii), let E be $X_{>t}$-restricted admissible for $F_{\geq t}$ with $S \subset E$. Then E_1 is $X_{>t'}$-restricted admissible for $F_{\geq t'}$ and E_2 is $X_{>t''}$-restricted admissible for $F_{\geq t''}$. Hence there exist sets D' and D'' with $E_1 \in e_{t'}(D')$, $E_2 \in e_{t''}(D'')$, $E_1 \cap X_t = E_2 \cap X_t$, and either

(a) $D' \in \Gamma', D'' \in \Gamma''$ (b) $D' = C', D'' \in \Gamma''$ or (c) $D' \in \Gamma', D'' = C''$

holds. But then, as in the proof of Lemma 6, also $E = E_1 \cup E_2 \in e_t(D' \bowtie D'')$. □

Example 22. To give an example consider the *JOIN*-node n_2 in Figure 10. Let us have a look at the pair $(C'_1, \{C'_1\})$ of n_3 with $C'_1(c) = in$, $C'_1(d) = def$, and pair $(C''_1, \emptyset)$ of n_8 with $C''_1(c) = in$, $C''_1(d) = def$. As $[C'_1] = [C''_1]$ we combine these vpairs using the $\bowtie$ operations. The join $C'_1 \bowtie C''_1$ leads to the vcoloring C_1 with $C_1(c) = in$ and $C_1(d) = def$. To update the certificates we have to consider the sets $\Gamma' \bowtie \Gamma''$, $\{C'\} \bowtie \Gamma''$, and $\Gamma' \bowtie \{C''\}$. The first two sets are empty as $\Gamma'' = \emptyset$ and the third one leads to the certificate C_1. In this way, we have obtained the vpair $(C_1, \{C_1\})$ for n_2. ◇

Theorem 7. *Let $(\mathcal{T}, \mathcal{X})$ be a tree decomposition of an AF $F = (A, R)$. Then, for each pair (C, Γ) for a node t, it holds that (C, Γ) is a valid pair for t iff (C, Γ) is a vpair for t.*

Proof. As in Theorem 5, the proof proceeds by structural induction. For the induction base, we have to show that vpairs and valid pairs coincide on *LEAF* nodes, which is the case due to Lemma 11. For the induction step, we have to show this property for the remaining nodes. Indeed, this is captured by Lemmas 12, 13 and 14. □

Thus, we now have a handle to efficiently decide skeptical acceptance for bounded tree-width. We just have to mark all pairs (C, Γ) where the considered argument a satisfies $C(a) \neq in$ and pass this mark accordingly towards the root node. If $(\epsilon, \emptyset)$ carries this mark, then we know that skeptical acceptance does not hold.

Example 23. Let us now consider the problem of deciding if the argument a is skeptically accepted in our example AF. In Figure 10 we illustrate the vpairs which are marked as contradictory for skeptical acceptance with a $\lightning$ in the last column of the table. Note that for a vpair (C, Γ) to be marked it is sufficient that for one set $S \in e_t(C, \Gamma)$ it holds that $a \notin S$. The counter $\#$ in Figure 10 still refers to *all* $X_{>t}$-admissible sets (for $F_{\geq t}$) in $e_t(\cdot, \cdot)$. Thus, the number of such sets $S \in e_t(\cdot, \cdot)$ with $a \notin S$ is, in general, smaller. $\diamond$

Theorem 8. *Deciding* SA *for an AF* $F = (A, R)$ *of tree-width* $k-1$ *can be done in time* $O(2^{2^{2k+1}+8k} \cdot |A|)$.

Proof. Recall that the number of colorings for each node is bounded by 4^k. In order to maintain the vpairs for each node, we consider all possible pairs (C, Γ), where C is a coloring and Γ is a set of colorings. Hence, we have to consider at most $4^k \cdot 2^{4^k} = 2^n$ pairs at each node, where $n = 2^{2k} + 2k$ (we use abbreviation n throughout the proof). Analogously to the proof of Theorem 6, we can store the vpairs for a node t in a table with one row per possible pair (C, Γ). In an additional bit we indicate if this row represents a vpair. Given a pair (C, Γ), we can find the corresponding row in time $O(n)$.

We have to show that computing the vpairs at each node $t \in \mathcal{T}$ is feasible in time $O(2^{2^{2k+1}+8k})$ in a single bottom-up traversal of $\mathcal{T}$. Since the number of nodes of $\mathcal{T}$ may be assumed to be bounded by $O(|A|)$, the desired upper bound of the theorem follows immediately. We prove the upper bound $O(2^{2^{2k+1}+8k})$ for the time needed at each node $t \in \mathcal{T}$ by distinguishing the four types of nodes. As in the proof of Theorem 6, the computationally most expensive node type is the *JOIN* node, which is the one we shall focus on below. The other node types are treated similarly.

Let t be a *JOIN* node with successors t' and t''. To compute the table of vpairs for t, we iterate in a nested loop over all pairs (C', Γ') in the table at t' and all pairs (C'', Γ'') in the table at t'' and do the following: check if (C', Γ') is a vpair and (C'', Γ'') is a vpair and $[C'] = [C'']$. If this is the case, we compute the vpair $(C, \Gamma) = (C' \bowtie C'', (\Gamma' \bowtie \Gamma'') \cup (\Gamma' \bowtie \{C''\}) \cup (\{C'\} \bowtie \Gamma''))$ and set the vpair-bit in the row corresponding to (C, Γ) in the table at node t.. As in the proof of Theorem 6, the join-operation can be carried out in time $O(10^k \cdot k)$. The access to the appropriate row in the table at node t is feasible in time $O(2^{2k} \cdot k)$. In total, we have to process at most $(2^n)^2$ combinations of vpairs (C, Γ) and (C', Γ'). Moreover, the action required for each such combination of vpairs fits into $O(10^k \cdot k + 2^{2k} \cdot k) = O(2^{4k})$. We thus end up with the upper bound $O((2^{2^{2k}+2k})^2 \cdot 2^{4k}) = O((2^{2^{2k+1}+4k}) \cdot 2^{4k}) = O(2^{2^{2k+1}+8k})$. $\square$

4.3 Characterizing Ideal Sets

So far, we have solved the credulous and the skeptical acceptance problems. For the first problem we used colorings to characterize admissible sets and for the latter problem we extended our data structure by certificates Γ, handling the subset maximality, to characterize preferred extensions. Here, we will reuse the concept of certificates to characterize ideal sets. But instead of storing supersets in the certificates we store certain witnesses against being an ideal set. Such witnesses have been identified by Dunne [18] as follows.

Proposition 3 ([18]). *Let $F = (A, R)$ be an AF and $S \subseteq A$ a set of arguments. S is an ideal set of F iff the following conditions hold:*

- *S is admissible in F;*
- *for every argument $p \in S^- = \{x \in A \mid (x, s) \in R$ for some $s \in S\}$, no admissible set of F contains p.*

Intuitively, S is an ideal set of an AF F if S is admissible and S is not attacked by any other admissible set. Therefore, a certificate Γ of a pair (C, Γ) should represent all restricted admissible sets that attack some $S \in e_t(C)$. This is formally defined next.

Definition 19. *Given a tree decomposition $(\mathcal{T}, \mathcal{X})$ of an AF F and a pair (C, Γ) for $t \in T$, i.e. where C is a coloring for t and Γ is a set of colorings for t, define $e_t^{\mathsf{ID}}(C, \Gamma)$ as the collection of sets S which satisfy the following conditions:*

(i) $S \in e_t(C)$*;*

(ii) for all $C' \in \Gamma$, there exists a set $E \in e_t(C')$, such that $E \rightarrowtail S$;

(iii) for all $X_{>t}$-restricted admissible (for $F_{\geq t}$) sets E such that $E \rightarrowtail S$ there is a $C' \in \Gamma$ with $E \in e_t(C')$.

If $e_t^{\mathsf{ID}}(C, \Gamma) \neq \emptyset$, we call (C, Γ) an ID-pair for t.

The following lemmas are analogous to the Lemmas 1 and 2 (resp. Lemmas 9 and 10).

Lemma 15. *Let $(\mathcal{T}, \mathcal{X})$ be a tree decomposition of an AF F, $t \in \mathcal{T}$, and S an $X_{>t}$-restricted admissible set for $F_{\geq t}$. Then there is a pair (C, Γ) with $S \in e_t^{\mathsf{ID}}(C, \Gamma)$.*

Proof. Let S be an $X_{>t}$-restricted admissible set for $F_{\geq t}$. By Lemma 1, there exists a coloring C with $S \in e_t(C)$. Now let $\mathcal{E} = \{E \mid E$ is $X_{>t}$-restricted admissible for $F_{\geq t}$ and $E \rightarrowtail S\}$. Moreover, let $\Gamma = \{C' \mid \exists E \in \mathcal{E}$, s.t. $E \in e_t(C')\}$. We claim that $S \in e_t^{\mathsf{ID}}(C, \Gamma)$. To prove this, we check the conditions (i) – (iii) from Definition 19: (i) $S \in e_t(C)$ by the selection of C. (ii) For all $C' \in \Gamma$, there exists $E \in e_t(C')$ with $E \rightarrowtail S$; this follows by the construction of Γ from $\mathcal{E}$. (iii) For all E being $X_{>t}$-restricted sets admissible in $F_{\geq t}$ with $E \rightarrowtail S$, there exists $C' \in \Gamma$ such that $E \in e_t(C')$; again this follows by the construction of Γ from $\mathcal{E}$. □

Lemma 16. *Let $(\mathcal{T}, \mathcal{X})$ be a tree decomposition of an AF F and let (C, Γ), (C', Γ') be different pairs for $t \in \mathcal{T}$. Then, $e_t^{\mathsf{ID}}(C, \Gamma) \cap e_t^{\mathsf{ID}}(C', \Gamma') = \emptyset$.*

Proof. If $C \neq C'$ then, by Lemma 2, $e_t(C) \cap e_t(C') = \emptyset$ and our claim follows. Thus, it remains to consider pairs $(C, \Gamma), (C, \Gamma')$ with $\Gamma \neq \Gamma'$. W.l.o.g., we assume that there exists a coloring $\bar{C}$ for t such that $\bar{C} \in \Gamma$ but $\bar{C} \notin \Gamma'$. In order to show that $e_t^{\mathsf{ID}}(C, \Gamma) \cap e_t^{\mathsf{ID}}(C, \Gamma') = \emptyset$, we prove that none of the sets $S \in e_t^{\mathsf{ID}}(C, \Gamma)$ is contained in $e_t^{\mathsf{ID}}(C, \Gamma')$.

Let S be an arbitrary set in $e_t^{\mathsf{ID}}(C, \Gamma)$. Suppose to the contrary that S is also contained in $e_t^{\mathsf{ID}}(C, \Gamma')$. By Definition 19 (applied to $e_t^{\mathsf{ID}}(C, \Gamma)$), there exists an $X_{>t}$-restricted admissible set $E \in e_t(\bar{C})$ for $F_{\geq t}$ such that $E \rightarrowtail S$. By Definition 19 (applied to $e_t^{\mathsf{ID}}(C, \Gamma')$), there exists a coloring $C^* \in \Gamma'$ such that $E \in e_t(C^*)$. By Lemma 2, the colorings $\bar{C}$ and C^* coincide. Thus, $\bar{C} \in \Gamma'$, a contradiction. □

In summary, we again conclude that each element $S \in e_t^{\mathsf{ID}}(C, \Gamma)$ is an $X_{>t}$-restricted admissible set for $F_{\geq t}$ and each $X_{>t}$-restricted admissible set for $F_{\geq t}$ is characterized by a unique ID-pair for t.

Proposition 4. *Let r be the root of a tree decomposition $(\mathcal{T}, \mathcal{X})$ of an AF F. Then, $e_r^{\mathsf{ID}}(\epsilon, \emptyset) = ideal(F)$.*

Proof. We recall that $e_r(\epsilon) = adm(F)$. To show the set inclusion $e_r^{\mathsf{ID}}(\epsilon, \emptyset) \subseteq ideal(F)$, let S be an arbitrary set such that $S \in e_r^{\mathsf{ID}}(\epsilon, \emptyset)$. By Definition 19 (i), we obtain that S is $X_{>r}$-restricted admissible in $F_{\geq r}$, i.e. (since the root has an empty bag) S is an admissible set for F. Further by (iii) and the fact that $\Gamma = \emptyset$ we conclude that there is no admissible set E such that $E \rightarrowtail S$. By Proposition 3, S is thus an ideal extension of F.

It remains to show that $e_r^{\mathsf{ID}}(\epsilon, \emptyset) \supseteq ideal(F)$. Thus let $S \in ideal(F)$ be an arbitrary ideal extension. By Lemma 15 and Lemma 16 we get that there exists a unique ID-pair (C, Γ) such that $S \in e_t^{\mathsf{ID}}(C, \Gamma)$. Since the root has an empty bag we have $C = \epsilon$ and further by Definition 19 (ii) and the fact that there is no admissible set E of F such that $E \rightarrowtail S$ (again using Proposition 3) we conclude that $\Gamma = \emptyset$. □

Thus our pairs have the desired property to characterize ideal extensions. As in the previous subsections, we give now an alternative definition of such pairs which allows for an efficient computation (as long as the tree-width of the given AF is small).

Definition 20. *Let $t \in \mathcal{T}$ be a node in a tree decomposition $(\mathcal{T}, \mathcal{X})$ of an AF and t', t'' the possible children of t. Depending on the node type of t we define an* ID-vpair *for t as follows:*

- *LEAF: Each (C, Γ) where $C \in \mathcal{C}_t$ and $\Gamma = \{C' \in \mathcal{C}_t \mid [C'] \rightarrowtail [C]\}$, is an ID-vpair for t.*
- *FORGET: If (C', Γ') is an ID-vpair for t', $X_t = X_{t'} \setminus \{a\}$, and $C'(a) \neq att$, then*
  - *$(C' - a, \Gamma' - a)$ is an ID-vpair for t.*
- *INSERT: If (C', Γ') is an ID-vpair for t' and $X_t = X_{t'} \cup \{a\}$, then*
  - *$(C' + a, \Gamma)$ with $\Gamma = (\Gamma' + a) \cup (\Gamma' \dot{+} a) \cup \{C \in \mathcal{C}_t \mid [C] \rightarrowtail [C' + a]\}$ is an ID-vpair for t;*
  - *if $C' \dot{+} a$ is a vcoloring then $(C' \dot{+} a, \Gamma)$ with $\Gamma = (\Gamma' + a) \cup (\Gamma' \dot{+} a) \cup \{C \in \mathcal{C}_t \mid [C] \rightarrowtail [C' \dot{+} a]\}$ is an ID-vpair for t as well.*
- *JOIN: If (C', Γ') is an ID-vpair for t', (C'', Γ'') is an ID-vpair for t'', and $[C'] = [C'']$, then*
  - *$(C' \bowtie C'', (\Gamma' \bowtie \mathcal{C}_{t''}) \cup (\mathcal{C}_{t'} \bowtie \Gamma''))$ is an ID-vpair for t.*

Let us comment on the construction of the set of certificates for pairs with colorings $C' + a$, $C' \dot{+} a$ and $C' \bowtie C''$. By the nature of certificates for ID-pairs, we have to consider *all* vcolorings that can be constructed from the certificates in the successor nodes (in the case of vpairs in the previous subsection we could restrict ourselves to a certain superset relation). Let us first explain

the construction for the $C' + a$ operation. Here we consider a new argument a but do not add it to $[C]$. Now each certificate $E' \in \Gamma'$ may give rise to two certificates of $C' + a$, namely $E' + a$ and (possibly) $E' \dot{+} a$. Further we may also get new certificates of $C' + a$ from the vcolorings of the current node. If $a \rightarrowtail [C' + a]$ then all vcolorings E with $E(a) = in$ are certificates for $C' + a$. This is why Γ in the above definition contains the set $\{C \in \mathcal{C}_t \mid [C] \rightarrowtail [C' + a]\}$. Similar considerations underly the certificates of $C' \dot{+} a$, but here the set $\{C \mid C \in \mathcal{C}_t, [C] \rightarrowtail [C' \dot{+} a]\}$ captures the colorings E with $[E] \rightarrowtail a$. Next let us consider the certificates of $C' \bowtie C''$. A certificate $E' \in \Gamma'$ may give rise to several certificates of $C' \bowtie C''$. The certificate E' is combined with each vcoloring D'' of t'' such that $[E'] = [D'']$. Similarly a certificate $E'' \in \Gamma''$ is combined with each vcoloring D' of t' such that $[E''] = [D']$.

Example 24. Recall the AF from Example 9. The computation of ID-vpairs for the nodes of the tree decomposition for this AF is illustrated in Figure 11. The symbol ✓ is now used to mark ID-vpairs that correspond to at least one $X_{>t}$-restricted admissible set containing the argument a. ◇

In the following we show that the concept of ID-vpairs coincides with the concept ID-pairs and thus is appropriate for efficiently deciding the problem of ideal acceptance. As before we do this separately for each node type starting with *LEAF*-nodes:

Lemma 17. *For any* LEAF *node t in a tree decomposition of an AF, the* ID*-vpairs of t coincide with the* ID*-pairs of t.*

Proof. Let $(\mathcal{T}, \mathcal{X})$ be a tree decomposition of an AF F and t a leaf in $\mathcal{T}$. The $X_{>t}$-restricted admissible sets for $F_{\geq t}$ coincide with the sets $[C]$ for the valid colorings $C \in \mathcal{C}_t$. Moreover, the valid colorings and vcolorings for t coincide by Lemma 3. Now let (C, Γ) be an ID-pair for t. Then, by Definition 19, $[C] \in e_t^{\mathsf{ID}}(C, \Gamma)$. Hence, by Definition 20, (C, Γ) is an ID-vpair for t. Conversely, let (C, Γ) be an ID-vpair for t and let $S = [C]$. By Definition 16, S is $X_{>t}$-restricted admissible for $F_{\geq t}$. Hence, by Definition 19 and Definition 20, $S \in e_t^{\mathsf{ID}}(C, \Gamma)$. Thus, (C, Γ) is an ID-pair for t. □

Example 25. As an example consider the *LEAF*-node n_{13} in Figure 11. The vcolorings C_1, C_2, C_3, C_4 correspond to the $\emptyset$-restricted admissible sets $\{e\}$, $\{f\}$, $\{g\}$ and $\emptyset$. As in our example $f \rightarrowtail e$ we have that C_2 is a certificate for C_1. For similar reasons we get that C_3 is a certificate for C_2 and that C_1 is a certificate for C_3. As $\emptyset$ has no attackers, the set of certificates for C_4 is empty. ◇

We proceed with nodes of type *FORGET*:

Lemma 18. *For any* FORGET *node t in a tree decomposition of an AF, the* ID*-vpairs and* ID*-pairs coincide, if they coincide in the child node t' of t.*

Proof. Let $(\mathcal{T}, \mathcal{X})$ be a tree decomposition of $F = (A, R)$, t a *FORGET* node in $\mathcal{T}$, and t' the child node of t. It holds that $X_t = X_{t'} \setminus \{a\}$ for some argument $a \in X_{t'}$. First we show that every ID-pair for t is also an ID-vpair for t. Thus let (C, Γ) be an ID-pair for t and $S \in e_t^{\mathsf{ID}}(C, \Gamma)$. In particular, S is $X_{>t}$-restricted admissible for $F_{\geq t}$ and, hence, also $X_{>t'}$-restricted admissible for $F_{\geq t'} = F_{\geq t}$. Thus, by Lemma 15, there exists an ID-pair (C', Γ') for t' with $S \in e_{t'}^{\mathsf{ID}}(C', \Gamma')$. By assumption, (C', Γ') is an ID-vpair for t'. Since S is $X_{>t'}$-restricted admissible for $F_{\geq t'}$ and

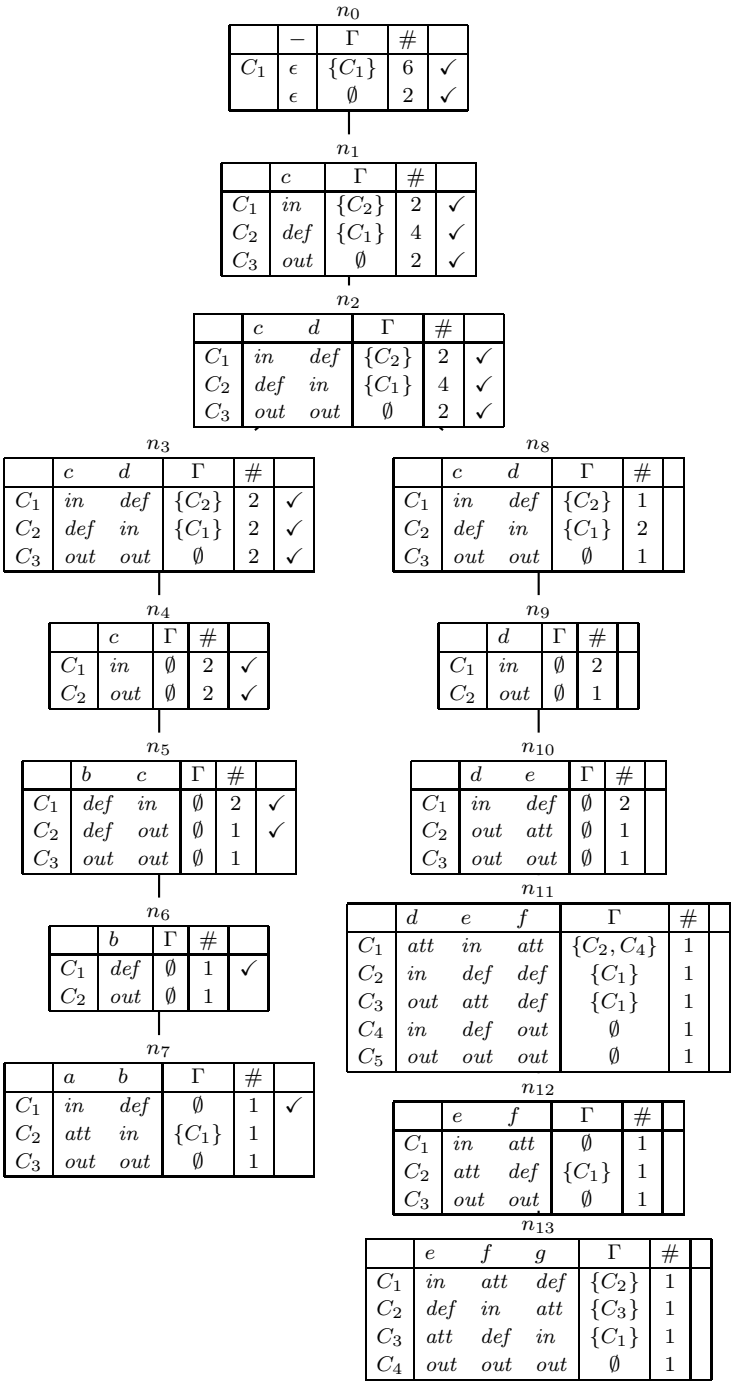


Figure 11: Computation of ID-vpairs for the example AF.

$S \in e^{\mathsf{ID}}_{t'}(C')$, we have $C'(a) \neq att$. Then $(C' - a, \Gamma' - a)$ is an ID-vpair for t. We claim that $(C' - a, \Gamma' - a) = (C, \Gamma)$ holds. The equality $C' - a = C$ is shown as in the proof of Lemma 4.

To show $\Gamma' - a = \Gamma$, we first consider the inclusion $\Gamma' - a \subseteq \Gamma$: Thus consider $D' \in \Gamma'$ with $D'(a) \neq att$. By condition (ii) of Definition 19, there exists an $X_{>t'}$-restricted admissible set E for $F_{\geq t'}$ with $E \rightarrowtail S$ and $E \in e_{t'}(D')$. By $D'(a) \neq att$, we know that E is also $X_{>t}$-restricted admissible for $F_{\geq t}$. Hence, by condition (iii) of Definition 19, there exists $D \in \Gamma$ with $E \in e_t(D)$. As in the proof of Lemma 4, we thus have $D = D' - a \in \Gamma$. Hence, $\Gamma' - a \subseteq \Gamma$.

Now consider an arbitrary $D \in \Gamma$. By condition (ii) of Definition 19, there exists an $X_{>t}$-restricted admissible set E for $F_{\geq t}$ with $E \rightarrowtail S$ and $E \in e_t(D)$. By condition (iii) of Definition 19 and since E is also $X_{>t'}$-restricted admissible for $F_{\geq t'} = F_{\geq t}$, there exists $D' \in \Gamma'$ with $E \in e_{t'}(D')$. As in the proof of Lemma 4, we thus have $D = D' - a \in \Gamma' - a$. Hence, $\Gamma \subseteq \Gamma' - a$.

It remains to show that every ID-vpair for t is an ID-pair for t. Therefore consider an ID-vpair (C, Γ) for t. By definition there exists an ID-vpair (C', Γ') for node t' with $C'(a) \neq att$ and $(C, \Gamma) = (C' - a, \Gamma' - a)$. By assumption, (C', Γ') is also an ID-pair for t'. Hence, there exists $S \in e_{t'}^{\mathsf{ID}}(C', \Gamma')$. We claim that also $S \in e_t^{\mathsf{ID}}(C, \Gamma)$ holds. As in the proof of Lemma 4, $S \in e_t(C)$ holds since $C'(a) \neq att$ and $C = C' - a$. It remains to show that (C, Γ) also satisfies conditions (ii) and (iii) of Definition 19.

To show condition (ii), consider $D \in \Gamma$, i.e., D is of the form $D = D' - a$ for some $D' \in \Gamma'$ with $D'(a) \neq att$. Since $S \in e_{t'}^{\mathsf{ID}}(C', \Gamma')$, there exists $E \in e_{t'}(D')$ with $E \rightarrowtail S$. As in the proof of Lemma 4, then also $E \in e_t(D' - a)$. To show condition (iii), let E be an $X_{>t}$-restricted admissible set for $F_{\geq t}$ with $E \rightarrowtail S$. Then E is also $X_{>t'}$-restricted admissible for $F_{\geq t'}$ and, therefore, there exists $D' \in \Gamma'$ with $E \in e_{t'}(D')$. Since E is $X_{>t}$-restricted admissible, we have $D'(a) \neq att$. But then, as in the proof of Lemma 4, also $E \in e_t(D' - a)$. □

Example 26. Consider the *FORGET*-node n_{12} in Figure 11 where argument g is removed. The ID-vpairs for n_{12} are obtained from the ID-vpairs of n_{13}, but (as for vcolorings) with one exception. As discussed in Section 4.1, the vcoloring C_2' of n_{13} with $C(f) = in$ is not a vcoloring for n_{12}. Thus we exclude the ID-vpair based on this vcoloring and further exclude C_2' from all certificate-sets. ◇

Next we consider *INSERT*-nodes:

Lemma 19. *For any* INSERT *node t in a tree decomposition of an AF, the* ID*-vpairs and* ID*-pairs coincide, if they coincide in the child node t' of t.*

Proof. Let $(\mathcal{T}, \mathcal{X})$ be a tree decomposition of $F = (A, R)$, t an *INSERT* node in $\mathcal{T}$, and t' the child node of t. Then there exists an argument $a \in A$ such that $X_t = X_{t'} \cup \{a\}$. First we show that every ID-pair for t is also an ID-vpair for t. Thus consider an ID-pair (C, Γ) for t. Then there exists an $X_{>t}$-restricted admissible set S for $F_{\geq t}$ such that $S \in e_t^{\mathsf{ID}}(C, \Gamma)$. Moreover also the set $S' = S \setminus \{a\}$ is $X_{>t'}$-restricted admissible for $F_{\geq t'}$. Thus, by Lemma 15, there exists an ID-pair (C', Γ') for t' with $S' \in e_{t'}^{\mathsf{ID}}(C', \Gamma')$ and by assumption, (C', Γ') is also an ID-vpair for t'.

Then $(C' + a, \Gamma_1)$ with $\Gamma_1 = (\Gamma' + a) \cup (\Gamma' \dot{+} a) \cup \{C \in \mathcal{C}_t \mid [C] \rightarrowtail C' + a\}$ is an ID-vpair for t and further if $[C'] \cup a$ is conflict-free in F_t then also $(C' \dot{+} a, \Gamma_2)$ with $\Gamma_2 = (\Gamma' + a) \cup (\Gamma' \dot{+} a) \cup \{C \in \mathcal{C}_t \mid [C] \rightarrowtail C' \dot{+} a\}$ is an ID-vpair for t. This follows by the same arguments based on properties of a tree decomposition as we have used earlier, e.g. in the proof of Lemma 13. We claim that either $(C' + a, \Gamma_1) = (C, \Gamma)$ or $(C' \dot{+} a, \Gamma_2) = (C, \Gamma)$ holds. As shown in the proof of Lemma 5 we have that either $C = C' + a$ (if $a \notin S$) or $C = C' \dot{+} a$ (if $a \in S$) holds. To show the equality for the certificates, i.e. $\Gamma = \Gamma_1$ or $\Gamma = \Gamma_2$ we distinguish two cases:

1) Assume $a \notin S$: To show $\Gamma_1 = \Gamma$, we first prove the inclusion $\Gamma_1 \subseteq \Gamma$: For the inclusion $(\Gamma' + a) \cup (\Gamma' \dot{+} a) \subseteq \Gamma$, consider an arbitrary $D' \in \Gamma'$. By condition (ii) of Definition 19, there exists an $X_{>t'}$-restricted admissible set E' for $F_{\geq t'}$ with $E' \rightarrowtail S'$ and $E' \in e_{t'}(D')$.

   As by assumption $a \notin S$ we have that

   (i) $E' \rightarrowtail S$ and (ii) $E = E' \cup \{a\} \rightarrowtail S$, respectively.

   In the first case we have that E' is conflict-free in $F_{\geq t}$; in the latter case, E is conflict-free in $F_{\geq t}$ if $[D'] \cup \{a\}$ is so. Further if E is conflict-free in $F_{\geq t}$ then it is also $X_{>t}$-restricted admissible for $F_{\geq t}$ (using the fact that there are no attacks between arguments from $X_{>t}$ and a which holds by properties of tree decompositions). Thus, by Definition 19, there exists a set $D \in \Gamma$ such that $E \in e_t(D)$. As shown in the proof of Lemma 5, in case (i), we have $D = D' + a$ and in case (ii), we have $D = D' \dot{+} a$. This concludes the proof that $(\Gamma' + a) \cup (\Gamma' \dot{+} a) \subseteq \Gamma$ holds.

   It remains to show $\{C^* \in \mathcal{C}_t \mid [C^*] \rightarrowtail [C' + a]\} \subseteq \Gamma$. Thus let us consider such a coloring C^* and an arbitrary set $E \in e_t(C^*)$. As $[C^*] \rightarrowtail [C' + a]$ it follows that for each $S \in e_t(C)$, $E \rightarrowtail S$ and thus $C^* \in \Gamma$ must hold. Hence, $\Gamma_1 \subseteq \Gamma$.

   Now consider an arbitrary vcoloring $D \in \Gamma$. By condition (ii) of Definition 19, there exists an $X_{>t}$-restricted admissible set $E \in e_t(D)$ for $F_{\geq t}$ such that $E \rightarrowtail S$. Using the assumption $a \notin S$ we conclude that for $E' = E \setminus \{a\}$ one of the following conditions hold.

   (i) $E' \rightarrowtail S'$ or (ii) $E' \not\rightarrowtail S'$ but $a \rightarrowtail S$

   In both cases we have that E' is $X_{>t'}$-restricted admissible for $F_{\geq t'}$ and thus there exists $D' \in \Gamma'$ with $E \in e_{t'}(D')$. In case (i) we can use the proof of Lemma 5, to show that either $D = D' + a$ or $D = D' \dot{+} a$. In case (ii) we use that $a \rightarrowtail S$ iff $[D] \rightarrowtail [C]$ iff $D \in \{C^* \in \mathcal{C}_t \mid [C^*] \rightarrowtail [C' + a]\}$. Hence, $\Gamma \subseteq \Gamma_1$.

2) Assume $a \in S$: To show $\Gamma_2 = \Gamma$, we first consider the inclusion $\Gamma_2 \subseteq \Gamma$: For the inclusion $(\Gamma' + a) \cup (\Gamma' \dot{+} a) \subseteq \Gamma$, consider an arbitrary $D' \in \Gamma'$. By condition (ii) of Definition 19, there exists an $X_{>t'}$-restricted admissible set $E' \in e_{t'}(D')$ for $F_{\geq t'}$ with $E' \rightarrowtail S'$. As $X_{>t} = X_{>t'}$ we have that E' is also an $X_{>t}$-restricted admissible set for $F_{\geq t}$ and further if $[D'] \cup \{a\}$ is conflict-free in F_t then also $E = E' \cup \{a\}$ is $X_{>t}$-restricted admissible for $F_{\geq t}$. By Definition 19 there exists a $D \in \Gamma$ such that $E' \in e_t(D)$ or $E \in e_t(D)$, respectively. As shown in the proof of Lemma 5, it holds that $D = D' + a$ or $D = D' \dot{+} a$, respectively. It remains to show that $\{C^* \in \mathcal{C}_t \mid [C^*] \rightarrowtail [C' \dot{+} a]\} \subseteq \Gamma$. As $[C^*] \rightarrowtail [C]$ it follows that for each $S \in e_t(C)$, $E \rightarrowtail S$ and thus $C^* \in \Gamma$ must hold. Hence, $\Gamma_2 \subseteq \Gamma$.

   Now let $D \in \Gamma$. By condition (ii) of Definition 19, there exists an $X_{>t}$-restricted admissible set $E \in e_t(D)$ for $F_{\geq t}$ with $E \rightarrowtail S$. By assumption $a \in S$ and thus we have that $E' = E \setminus \{a\} \rightarrowtail S$. Further we have that E' is $X_{>t'}$-restricted admissible for $F_{\geq t'}$ and thus there exists $D' \in \Gamma'$ with $E' \in e_{t'}(D')$. Now either $E' \rightarrowtail S' = S \setminus \{a\}$ or $E' \rightarrowtail a$. In the first case we have that either $D = D' + a$ or $D = D' \dot{+} a$ holds (cf. the proof of Lemma 5). Thus $D \in (\Gamma' + a) \cup (\Gamma' \dot{+} a)$ holds. In the latter case it holds that $[D] \rightarrowtail [C]$ (as $a \in [C]$) and $D \in \mathcal{C}_t$ (as $D \in \Gamma$). Thus $D \in \{C^* \in \mathcal{C}_t \mid [C^*] \rightarrowtail [C' \dot{+} a]\}$ holds. Hence, $\Gamma \subseteq \Gamma_2$.

It remains to show that every ID-vpair for t is also an ID-pair for t. Thus let (C, Γ) be an ID-vpair for t. By definition there exists an ID-vpair (C', Γ') for node t' such that either $(C, \Gamma) = (C' + a, \Gamma_1)$ or, in case $[C'] \cup \{a\}$ is conflict-free in F_t, $(C, \Gamma) = (C' \dot{+} a, \Gamma_2)$ (Γ_1, Γ_2 defined as above).

By assumption, (C', Γ') is an ID-pair for t'. Hence, there exists $S' \in e^{\mathsf{ID}}_{t'}(C', \Gamma')$. We claim that also $S \in e^{\mathsf{ID}}_t(C, \Gamma)$ holds, where S is defined as follows: $S = S'$ if $C = C' + a$ and $S = S' \cup \{a\}$ if $C = C' \dot{+} a$. As in the proof of Lemma 5, $S \in e_t(C)$ holds in both cases. It remains to show that also conditions (ii) and (iii) of Definition 19 are fulfilled.

To show condition (ii), let $D \in \Gamma$, i.e., D is either of the form

(a) $D = D' + a$, (b) $D = D' \dot{+} a$ or (c) $[D] \rightarrowtail [C]$

for some $D' \in \Gamma'$. We prove for each of these cases that condition (ii) holds, i.e., there exists a set $E \in e_t(D)$, such that $E \rightarrowtail S$;

a) By $S' \in e^{\mathsf{ID}}_{t'}(C', \Gamma')$, there exists $E' \in e_{t'}(D')$ with $E' \rightarrowtail S'$. Thus, by $S' \subseteq S$, also $E' \rightarrowtail S$ holds. Moreover, as in the proof of Lemma 5, we also have $E' \in e_t(D' + a)$.

b) Again, by $S' \in e^{\mathsf{ID}}_{t'}(C', \Gamma')$, there exists an $E' \in e_{t'}(D')$ with $E' \rightarrowtail S'$ and, therefore, also $E' \rightarrowtail S$. By the construction of $D = D' \dot{+} a$ we know that $E = E' \cup \{a\}$ is conflict-free in $F_{\geq t}$. By the usual arguments exploiting the definition of tree decompositions, we obtain that E is an $X_{>t}$-restricted admissible set for $F_{\geq t}$. Following the proof of Lemma 5, we get $E \in e_t(D' \dot{+} a)$. Moreover, $E \rightarrowtail S$ follows from $E' \rightarrowtail S$ and $E' \subseteq E$.

c) As D is a valid coloring there exists an $X_{>t}$-restricted admissible set $E \in e_t(D)$. From $[D] \rightarrowtail [C]$ and $S \in e_t(C)$ it follows that $E \rightarrowtail S$.

To show condition (iii), let E be $X_{>t}$-restricted admissible for $F_{\geq t}$ such that $E \rightarrowtail S$. Further let D be the unique coloring such that $E \in e_t(D)$. We claim that $D \in \Gamma$. Clearly, $E' = E \setminus \{a\}$ is $X_{>t'}$-restricted admissible for $F_{\geq t'}$. If $E' \rightarrowtail S \setminus \{a\}$ then D' denotes the unique vcoloring such that $E' \in e_{t'}(D')$ and $D' \in \Gamma'$. Then, as in the proof of Lemma 5, either $D = D' + a$ or $D = D' \dot{+} a$ holds and therefore $D \in \Gamma$. Otherwise if $E' \not\rightarrowtail S$ it must hold that either (i) $a \in E$ and $a \rightarrowtail S$ or (ii) $a \in S$ and $E \rightarrowtail a$. But both (i) and (ii) imply that $[D] \rightarrowtail [C]$ and hence $D \in \Gamma$. □

Example 27. One example for an *INSERT*-node is the node n_{11} in Figure 11, where the argument d is added. The ID-vpairs of n_{11} are obtained from the ID-vpairs of n_{12}. For instance consider the ID-vpair $(C'_1, \emptyset)$ of n_{12}. For the vcoloring $C_1 = C'_1 + d$, observe that $[C_2] \rightarrowtail [C_1]$ and $[C_4] \rightarrowtail [C_1]$; hence we derive the ID-vpair $(C_1, \{C_2, C_4\})$ for n_{11}.

Now consider the ID-vpair $(C'_2, \{C'_1\})$ of n_{12}. We get the vcoloring $C_2 = C'_2 \dot{+} d$ and the certificate $C_1 = C'_1 + d$. $C'_1 \dot{+} d$ is not a vcoloring and C_1 is the only vcoloring with $[C_1] \rightarrowtail [C_2]$. Thus we obtain the ID-vpair $(C_2, \{C_1\})$ ◇

Finally we discuss *JOIN*-nodes.

Lemma 20. *For a* JOIN *node t in a tree decomposition of an AF with successors t', t'', the* ID*-vpairs of t coincide with the* ID*-pairs of t if they coincide for t' as well as for t''.*

Proof. Let $(\mathcal{T}, \mathcal{X})$ be a tree decomposition of $F = (A, R)$ and t a *JOIN* node in $\mathcal{T}$ with successors t' and t''. Recall that we have $X_t = X_{t'} = X_{t''}$. To show that every ID-pair for t is also an ID-vpair for t, consider an arbitrary ID-pair (C, Γ) for t. Then, there exists an $X_{>t}$-restricted admissible set $S \in e_t^{\mathsf{ID}}(C, \Gamma)$ for $F_{\geq t}$. As in the proof of Lemma 6 we have that there exist unique sets $S_1 \subseteq X_{\geq t'}$ and $S_2 \subseteq X_{\geq t''}$ such that $S_1 \cap X_t = S_2 \cap X_t$ and $S = S_1 \cup S_2$. Moreover there exist vcolorings C', C'' such that $S_1 \in e_{t'}(C')$, $S_2 \in e_{t''}(C'')$ and $C = C_1 \bowtie C_2$. Hence there exist ID-pairs (C', Γ'), (C'', Γ'') with $S_1 \in e_{t'}^{\mathsf{ID}}(C', \Gamma')$ and $S_2 \in e_{t''}^{\mathsf{ID}}(C'', \Gamma'')$, which, by assumption are also ID-vpairs.

Now we turn our attention to the certificates. We have to show $\Gamma = (\Gamma' \bowtie \mathcal{C}_{t''}) \cup (\mathcal{C}_{t'} \bowtie \Gamma'')$. We first prove the inclusion $\Gamma \subseteq (\Gamma' \bowtie \mathcal{C}_{t''}) \cup (\mathcal{C}_{t'} \bowtie \Gamma'')$. So, let $D \in \Gamma$. By Definition 19 there exists an $X_{>t}$-restricted admissible set E for $F_{\geq t}$ such that $E \rightarrowtail D$. We define $E = E_1 \cup E_2$ analogously to S_1, S_2. As we mentioned in the proof of Lemma 6, there are no attacks between the argument sets $X_{>t'}$ and $X_{>t''}$, because of the properties (2) and (3) of tree decompositions. Thus we have that $E \rightarrowtail S$ holds iff either

$$\text{(i) } E_1 \rightarrowtail S_1 \quad \text{or} \quad \text{(ii) } E_2 \rightarrowtail S_2$$

holds. As both cases are symmetric it suffices to consider case (i). As E is $X_{>t}$-restricted admissible for $F_{\geq t}$, we have that also E_1 is $X_{>t'}$-restricted admissible for $F_{\geq t'}$, and likewise, E_2 is $X_{>t''}$-restricted admissible for $F_{\geq t''}$. Thus there exists $D'' \in \mathcal{C}_{t''}$ such that $E_2 \in e_{t''}(D'')$. Moreover by (i) we have that there exists $D' \in \Gamma'$ with $E_1 \in e_{t'}(D')$. By the proof of Lemma 6 we have that $D = D' \bowtie D''$ and thus $D \in \Gamma' \bowtie \mathcal{C}_{t''}$. Hence $\Gamma \subseteq (\Gamma' \bowtie \mathcal{C}_{t''}) \cup (\mathcal{C}_{t'} \bowtie \Gamma'')$.

It remains to show that $(\Gamma' \bowtie \mathcal{C}_{t''}) \cup (\mathcal{C}_{t'} \bowtie \Gamma'') \subseteq \Gamma$ which is equivalent to

$$\text{(i) } \Gamma' \bowtie \mathcal{C}_{t''} \subseteq \Gamma \quad \text{and} \quad \text{(ii) } \mathcal{C}_{t'} \bowtie \Gamma'' \subseteq \Gamma.$$

As before, by symmetry, we may restrict ourselves to case (i). Thus let $D' \in \Gamma'$ and $D'' \in \mathcal{C}_{t''}$ with $[D'] = [D'']$, $E_1 \in e_{t'}(D')$ and $E_2 \in e_{t''}(D'')$. By Definition 19 we have that $E_1 \rightarrowtail S_1$ and therefore that $E = E_1 \cup E_2 \rightarrowtail S$. Further by the proof of Lemma 6, $D = D' \bowtie D''$ is the unique coloring such that $E \in e_t(D)$. We thus obtain the desired result $D' \bowtie D'' \in \Gamma$. Hence $(\Gamma' \bowtie \mathcal{C}_{t''}) \cup (\mathcal{C}_{t'} \bowtie \Gamma'') \subseteq \Gamma$. In summary, we have proved $\Gamma = (\Gamma' \bowtie \mathcal{C}_{t''}) \cup (\mathcal{C}_{t'} \bowtie \Gamma'')$ and thus every ID-pair (C, Γ) is also an ID-vpair.

In the second part of the proof we show that every ID-vpair for t is an ID-pair for t. Thus let (C, Γ) be an ID-vpair for t. By definition there is an ID-vpair (C', Γ') for node t' and an ID-vpair (C'', Γ'') for node t'' such that $(C, \Gamma) = (C' \bowtie C'', (\Gamma' \bowtie \mathcal{C}_{t''}) \cup (\mathcal{C}_{t'} \bowtie \Gamma'')$ and $[C'] = [C'']$. By assumption, (C', Γ') and (C', Γ') are also ID-pairs and thus there are sets S_1, S_2 such that $S_1 \in e_{t'}^{\mathsf{ID}}(C', \Gamma')$ and $S_2 \in e_{t''}^{\mathsf{ID}}(C'', \Gamma')$. As in the proof of Lemma 6, $S = S_1 \cup S_2 \in e_t(C)$ holds since $[C''] = [C']$

It remains to show that also conditions (ii) and (iii) of Definition 19 are fulfilled:

(ii) To show condition (ii), consider $D \in \Gamma$. Then D is either of the form

- (a) $D = D' \bowtie D''$ for some $D' \in \Gamma'$, $D'' \in \mathcal{C}_{t''}$ with $[D'] = [D'']$ or
- (b) $D = D' \bowtie D''$ for some $D' \in \mathcal{C}_{t'}$, $D'' \in \Gamma''$ with $[D'] = [D'']$.

By symmetry, it suffices to consider case (a). As $S_1 \in e^{\mathsf{ID}}_{t'}(C', \Gamma')$ there exists $E_1 \in e_{t'}(D')$ such that $E_1 \rightarrowtail S_1$. Further by $D'' \in \mathcal{C}_{t''}$ and $[D'] = [D'']$, there exists $E_2 \in e_{t''}(D'')$ such that $E_1 \cap X_t = E_2 \cap X_t$. Now, using the proof of Lemma 6, it holds that $E = E_1 \cup E_2 \in e_t(D' \bowtie D'')$ and $E \rightarrowtail S$.

(iii) To show condition (iii), let E be $X_{>t}$-restricted admissible for $F_{\geq t}$ with $E \rightarrowtail S$. Then E_1 is $X_{>t'}$-restricted admissible for $F_{\geq t'}$ and E_2 is $X_{>t''}$-restricted admissible for $F_{\geq t''}$. Moreover as $E \rightarrowtail S$ either $E_1 \rightarrowtail S_1$ or $E_2 \rightarrowtail S_2$ holds. Thus there exist D', D'' with $E_1 \in e_{t'}(D')$, $E_2 \in e_{t''}(D'')$, $E_1 \cap X_t = E_2 \cap X_t$, and either

(a) $D' \in \Gamma', D'' \in \mathcal{C}_{t''}$ or (b) $D' \in \mathcal{C}_{t'}, D'' \in \Gamma''$

holds. But then, as in the proof of Lemma 6, also $E = E_1 \cup E_2 \in e_t(D' \bowtie D'')$.

□

Example 28. The only *JOIN* node in our example is n_2 in Figure 11. For instance consider joining the ID-vpair $(C'_1, \{C'_2\})$ of n_3 with the ID-vpair $(C''_1, \{C''_2\})$ of n_8. As $[C'_1] = [C''_1]$ these two pairs can be combined to a pair (C_1, Γ). Further as for C'_2 the only appropriate join partner in n_8 is C''_2 and vice versa we only get one certificate $C_2 = C'_2 \bowtie C''_2$. This leads to the ID-vpair $(C_1, \{C_2\})$. ◇

Theorem 9. *Let $(\mathcal{T}, \mathcal{X})$ be a tree decomposition of an AF $F = (A, R)$. Then, for each pair (C, Γ), it holds that (C, Γ) is an* ID*-pair for t iff (C, Γ) is an* ID*-vpair for t.*

Proof. The proof makes use of the above lemmas and is the same as for the corresponding theorems in the previous sections. □

To decide whether an argument a is ideally accepted we now can proceed as for credulous acceptance: We have to mark ID-vpairs which assign the value in to the argument a and pass this mark up to the root. If the ID-vpair $(\epsilon, \emptyset)$ at the root has the mark, then we can conclude that the argument a is ideally accepted. Otherwise if $(\epsilon, \emptyset)$ is not marked then the argument a is not ideally accepted.

Example 29. Recall the computation in Figure 11. Now we consider the problem of deciding whether the argument a is ideally accepted. The argument a first appears in the node n_7 and thus we mark the ID-vpair with $C_1(a) = in$ and as before we illustrate this with a ✓ in the corresponding row of the table. Now consider node n_5; here we have that the ID-vpairs $(C_1, \emptyset)$ and $(C_2, \emptyset)$ are constructed from the marked ID-vpair $(C'_1, \emptyset)$ of n_6 and thus they are marked. We mention that the ID-vpair $(C_1, \emptyset)$ can also be built from $(C'_2, \emptyset)$, but this does not affect the mark. On the other hand the only way to build the ID-vpair $(C_3, \emptyset)$ is via the ID-vpair $(C'_2, \emptyset)$ and thus it is not marked. Inspecting the root shows that a is ideally accepted, which indeed holds since $\{a\}$ is an ideal set of our running example (see Example 2). ◇

Theorem 10. *Deciding* ID *for an AF $F = (A, R)$ of tree-width $k-1$ can be done in time $O(2^{2^{2k+1}+8k} \cdot |A|)$.*

Proof. Recall the proof of Theorem 6. We have that the number of pairs (C, Γ) in each node is bounded by 2^n with $n = 2^{2k} + 2k$. Further we store these pairs in tables such that we can find a given pair in time $O(n)$.

We have to show that computing the ID-vpairs at each node $t \in \mathcal{T}$ is feasible in time $O(2^{2^{2k+1}+8k})$ in a single bottom-up traversal of $\mathcal{T}$. Since the number of nodes of $\mathcal{T}$ may be assumed to be bounded by $O(|A|)$, the desired upper bound of the theorem follows immediately. We prove the upper bound $O(2^{2^{2k+1}+8k})$ for the time needed at each node $t \in \mathcal{T}$ by distinguishing the four types of nodes. As in the proof of Theorem 6, the computationally most expensive node type is the *JOIN* node, which is the one we shall focus on below. The other node types are treated similarly.

Let t be a *JOIN* node with successors t' and t''. To compute the table of ID-vpairs for t, we iterate in a nested loop over all pairs (C', Γ') in the table at t' and all pairs (C'', Γ'') in the table at t'' and do the following: check if (C', Γ') is an ID-vpair and (C'', Γ'') is an ID-vpair and $[C'] = [C'']$. If this is the case, we compute the vpair $(C, \Gamma) = (C' \bowtie C'', (\Gamma' \bowtie \mathcal{C}_{t''}) \cup (\mathcal{C}_{t'} \bowtie \{C''\}))$ and set the ID-vpair-bit in the row corresponding to (C, Γ) in the table at node t. As in the proof of Theorem 8, this can be done in time $O(2^{2^{2k+1}+8k})$. □

5 Conclusion

In this paper, we have turned several theoretical tractability results for argumentation frameworks of bounded tree-width into efficient algorithms. All these algorithms are based on a dynamic programming approach which uses a single bottom-up traversal of a tree decomposition of the given argumentation framework. For the basic algorithm, we introduced vcolorings as the crucial data structure to be maintained along this bottom-up traversal. We proved that this data structure allows us to succinctly represent the admissible sets and thus to efficiently decide credulous acceptance. For succinctly representing the preferred extensions and thus deciding skeptical acceptance we had to extend our basic data structure to vpairs – consisting of a vcoloring plus a set of certificates, which are themselves vcolorings. Finally, we modified these vpairs to so-called ID-pairs, which allowed us to design an efficient algorithm for ideal acceptance in argumentation frameworks of bounded tree-width. Moreover, we have shown that some further graph parameters (which, in contrast to tree-width, apply to directed graphs), do not lead to similar tractability results. The key to this collection of intractability results was the intractability for argumentation frameworks of bounded cycle-rank.

Several algorithms for the problems discussed in this paper have been presented in the literature. We mention the work by Doutre and Mengin [14] here which relies on set-enumeration techniques exploring a binary tree. Although this tree is conceptually different from the tree decompositions we use, a number of short-cuts for accelerating the enumeration is provided, which could be applied to our algorithms as well.

Recall that our algorithms rely on the concept of colorings. They look similar to labelings (see [7, 29]). However, labelings are defined for complete frameworks, while we require here a concept which also applies to subframeworks (recall that for our complexity results in Theorems

6, 8 and 10, it was essential that colorings are defined over a small number of arguments); in other words, we do not know in advance, whether an argument will eventually be defended; this also explains why we need four colors, whereas the number of labels is usually three. Nonetheless, known results about relations between labelings for different semantics might help us in extending our algorithms to other semantics, which is indeed a major topic for future work.

Further ongoing and future work is as follows:

- We plan to adapt our algorithms to other semantics, such as complete, stable, stage, and semi-stable. As we have already mentioned, we expect no major obstacles in extending the methods developed here to such other semantics.

- Another important aspect of future work is to analyze if typical argumentation scenarios naturally lead to AFs of low tree-width. Note that graphs containing big cliques have high tree-width. However, for argumentation scenarios we would rather expect graphs with small cliques or cycles, which are harmless as far as the tree-width is concerned.

- A first prototype system which implements the algorithms from this paper is available under

  `www.dbai.tuwien.ac.at/research/project/argumentation/dynpartix`.

  We are currently comparing our implementation with existing systems, for instance, the ASPARTIX system [23] which relies on reduction to logic programs, or to similar algorithms which however are designed along different parameters [21].

References

[1] J. Barát. Directed path-width and monotonicity in digraph searching. *Graphs and Combinatorics*, 22(2):161–172, 2006.

[2] P. Baroni, P. E. Dunne, and M. Giacomin. On extension counting problems in argumentation frameworks. In P. Baroni, F. Cerutti, M. Giacomin, and G. R. Simari, editors, *Proceedings of the 3rd Conference on Computational Models of Argument (COMMA'10)*, volume 216 of *Frontiers in Artificial Intelligence and Applications*, pages 63–74. IOS Press, 2010.

[3] P. Baroni and M. Giacomin. Semantics of abstract argument systems. In I. Rahwan and G. Simari, editors, *Argumentation in Artificial Intelligence*, pages 25–44. Springer, 2009.

[4] D. Berwanger, A. Dawar, P. Hunter, and S. Kreutzer. DAG-width and parity games. In *Proceedings of the 23rd Annual Symposium on Theoretical Aspects of Computer Science (STACS 2006)*, pages 524–536, 2006.

[5] H. L. Bodlaender. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM J. Comput.*, 25(6):1305–1317, 1996.

[6] M. Caminada. Semi-stable semantics. In P. E. Dunne and T. J. M. Bench-Capon, editors, *Proceedings of the 1st Conference on Computational Models of Argument (COMMA'06)*, volume 144 of *Frontiers in Artificial Intelligence and Applications*, pages 121–130. IOS Press, 2006.

[7] M. Caminada and D. M. Gabbay. A logical account of formal argumentation. *Studia Logica*, 93(2-3):109–145, 2009.

[8] S. Coste-Marquis, C. Devred, and P. Marquis. Symmetric argumentation frameworks. In L. Godo, editor, *Proceedings of the 8th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty* (*ECSQARU 2005*), volume 3571 of *LNCS*, pages 317–328. Springer, 2005.

[9] B. Courcelle. The monadic second-order logic of graphs. I. Recognizable sets of finite graphs. *Inf. Comput.*, 85(1):12–75, 1990.

[10] B. Courcelle, J. Engelfriet, and G. Rozenberg. Handle-rewriting hypergraph grammars. *J. Comput. Syst. Sci.*, 46(2):218–270, 1993.

[11] B. Courcelle, J. A. Makowsky, and U. Rotics. Linear time solvable optimization problems on graphs of bounded clique-width. *Theory Comput. Syst.*, 33(2):125–150, 2000.

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

[13] J. Dix, S. Parsons, H. Prakken, and G. R. Simari. Research challenges for argumentation. *Computer Science - R&D*, 23(1):27–34, 2009.

[14] S. Doutre and J. Mengin. Preferred extensions of argumentation frameworks: Query answering and computation. In R. Goré, A. Leitsch, and T. Nipkow, editors, *Proceedings of the 1st International Joint Conference on Automated Reasoning (IJCAR 2001)*, volume 2083 of *LNCS*, pages 272–288. Springer, 2001.

[15] 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.

[16] P. M. Dung, P. Mancarella, and F. Toni. Computing ideal sceptical argumentation. *Artif. Intell.*, 171(10-15):642–674, 2007.

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

[18] P. E. Dunne. The computational complexity of ideal semantics. *Artif. Intell.*, 173(18):1559–1591, 2009.

[19] P. E. Dunne and T. J. M. Bench-Capon. Coherence in finite argument systems. *Artif. Intell.*, 141(1/2):187–203, 2002.

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

[21] 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, editors, *Proceedings of the 3rd Conference on Computational Models of Argument (COMMA'10)*, volume 216 of *Frontiers in Artificial Intelligence and Applications*, pages 219–230. IOS Press, 2010.

[22] L. C. Eggan. Transition graphs and the star height of regular events. *Michigan Math. J.*, 10:385–397, 1963.

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

[24] H. Gruber. Digraph complexity measures and applications in formal language theory. *Proceedings of the 4th Workshop on Mathematical and Engineering Methods in Computer Science (MEMICS 2008)*, pages 60–67, 2008.

[25] P. Hunter and S. Kreutzer. Digraph measures: Kelly decompositions, games, and orderings. *Theor. Comput. Sci.*, 399(3):206–219, 2008.

[26] M. Jakl, R. Pichler, and S. Woltran. Answer-set programming with bounded treewidth. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI 2009)*, pages 816–822, 2009.

[27] T. Johnson, N. Robertson, P. D. Seymour, and R. Thomas. Directed tree-width. *J. Comb. Theory, Ser. B*, 82(1):138–154, 2001.

[28] T. Kloks. *Treewidth, Computations and Approximations*, volume 842 of *Lecture Notes in Computer Science*. Springer, 1994.

[29] S. Modgil and M. Caminada. Proof theories and algorithms for abstract argumentation frameworks. In I. Rahwan and G. Simari, editors, *Argumentation in Artificial Intelligence*, pages 105–129. Springer, 2009.

[30] N. Robertson and P. D. Seymour. Graph minors. II. Algorithmic aspects of tree-width. *J. Algorithms*, 7(3):309–322, 1986.

[31] L. G. Valiant. The complexity of enumeration and reliability problems. *SIAM J. Comput.*, 8(3):410–421, 1979.

[32] B. Verheij. Two approaches to dialectical argumentation: Admissible sets and argumentation stages. In J. Meyer and L. van der Gaag, editors, *Proceedings of the 8th Dutch Conference on Artificial Intelligence (NAIC'96)*, pages 357–368, 1996.