Proceedings of the Twelfth International Conference on the Principles of Knowledge Representation and Reasoning (KR 2010)

Towards Fixed-Parameter Tractable Algorithms for Argumentation*

Wolfgang Dvořák and **Reinhard Pichler** and **Stefan Woltran**
Institute of Information Systems, Vienna University of Technology, A-1040 Vienna, Austria.
email: {dvorak, pichler, woltran}@dbai.tuwien.ac.at

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.

Introduction

Argumentation has evolved as an important field in AI with abstract argumentation frameworks (AFs, for short) as introduced by Dung (1995) being its most popular formalization. Meanwhile, many semantics for AFs have been proposed (for an overview see (Baroni and Giacomin 2009)). Most computational problems in this area are intractable (see e.g. (Dimopoulos and Torres 1996; Dunne and Bench-Capon 2002)), but the importance of efficient algorithms for tractable fragments has been clearly recognized (see e.g. (Dix et al. 2009)). Such tractable fragments are, for instance, symmetric argumentation frameworks (Coste-Marquis, Devred, and Marquis 2005) or bipartite argumentation frameworks (Dunne 2007).

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 (2007) 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 (1990). However, as stated in (Dunne 2007), "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 (skeptical and credulous) reasoning.

Clearly, the quest for FPT-results in argumentation should not stop at the tree-width, and further parameters should be analyzed. This may of course also lead to negative results. For instance, if we consider as parameter the degree of an argument (i.e., the number of incoming and outgoing attacks), Dunne (2007) showed that reasoning remains intractable, even if we restrict ourselves to 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 (Eggan 1963) (a precise definition will be given below) 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 in (Berwanger et al. 2006; Hunter and Kreutzer 2008; Gruber 2008) that problems which are hard for bounded cycle-rank remain hard when several other directed variants of the tree-width are bounded. Hence, in the current state of research, bounded tree-width seems to be the most general parameter to obtain FPT.

Due to lack of space, we have to restrict ourselves here to the preferred semantics. 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. One reason for choosing the preferred semantics is that it is widely used. Moreover, admissibility and maximality are prototypical properties common in many other semantics, for instance complete and stable (Dung 1995), stage (Verheij 1996), semi-stable (Caminada 2006), and ideal semantics (Dung, Mancarella, and Toni 2007). Hence, we expect that the methods developed here can also be extended to other semantics.

Structure of the paper and summary of results.

• After recalling some basic notions and results on AFs and width-measures for graphs, we show that reasoning remains

*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.

intractable in AFs with bounded cycle-rank (Eggan 1963). As has been mentioned above, this negative result carries over to many other directed notions of width.

• A dynamic programming approach is developed 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).

• In case of credulous reasoning, the algorithm for admissible sets also applies to the preferred semantics. For skeptical reasoning, we have to extend this algorithm so as to cover also the preferred semantics. Finally, we outline some directions of future research – notably the further extension of our algorithms to other semantics.

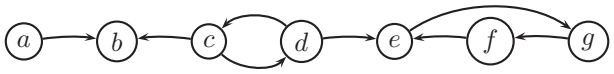
Argumentation Frameworks

In this section we introduce (abstract) argumentation frameworks (Dung 1995), recall the preferred semantics for such frameworks, and highlight some known complexity results.

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.



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.*

Definition 3. *Let $F = (A, R)$ be an AF. A set S 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 $pref(F)$.*

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, $pref(F) = \{\{a, c\}, \{a, d, g\}\}$.

Next, we recall the complexity of reasoning over preferred extensions. To this end, we define the decision problems of credulous acceptance (CA) and skeptical acceptance (SA), 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 $S \in pref(F)$?

It is known that CA is NP-complete, while SA is Π_2^P-complete (see (Dimopoulos and Torres 1996; Dunne and Bench-Capon 2002)). 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.

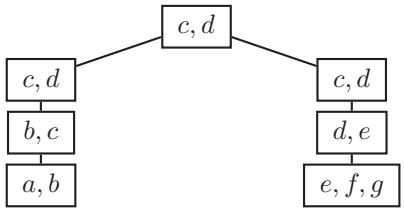


Figure 1: A tree decomposition of the graph in Example 1.

Parameters for Graphs

In this section, we review several notions of parameters for graphs (both directed and undirected). One of the most important concepts for FPT on graphs is the tree-width, which was introduced by Robertson and Seymour (1986).

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 4. *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 $t \in V_\mathcal{T}$.*

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 (1996) that, for fixed $w \geq 1$, it can be decided in linear time if 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).

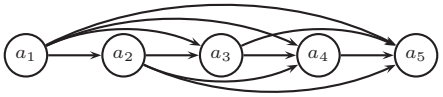
Many NP-hard problems on graphs have been shown to be linear time computable on graphs of bounded tree-width. In particular, Courcelle's Theorem (Courcelle 1990) provides a powerful tool to obtain such results. It states that any property over graphs which can be expressed in Monadic Second-Order Logic, can be decided in linear time (wrt. to the size of the graph) for graphs which have bounded tree-width. Dunne (2007) used this result to show the FPT of the problems CA and SA wrt. the 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 ($n \geq 1$)

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

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 (Johnson et al. 2001), DAG-width (Berwanger et al. 2006), and Kelly-width (Hunter and Kreutzer 2008). An old but particularly interesting parameter, which we shall focus on here, is cycle-rank (Eggan 1963). One reason why there are many different such notions is due to the fact that, so far, no analogue to Courcelle's Theorem has been found for digraph problems.

Definition 5. *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 strongly connected components (SCCs) of G.*

Note that the graphs G_n introduced above are acyclic and, thus, have cycle-rank 0 for any n. The cycle-rank is of particular interest because recent results (Berwanger et al. 2006; Hunter and Kreutzer 2008; Gruber 2008) showed that problems which are hard for bounded cycle-rank also remain hard when some of the other aforementioned parameters are bounded. Further, the class of graphs with bounded cycle-rank is incomparable with the class of graphs with bounded in- and out-degree. The latter was analyzed in terms of AFs by Dunne (2007), who showed that CA and SA remain intractable for AFs with bounded in- and out-degree.

Some Negative Results for Directed Graphs

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 (Dimopoulos and Torres 1996) which maps each instance (i.e. a CNF) of the NP-hard problem SAT to an argumentation framework.

Definition 6. *Given a CNF $\Phi = \bigwedge_{j=1}^{m} C_j$ 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 2. For $\Phi = (z_1 \vee z_2 \vee z_3) \wedge (\neg z_2 \vee \neg z_3 \vee \neg z_4) \wedge (\neg z_1 \vee z_2 \vee z_4)$, Figure 2 illustrates the AF F_Φ. $\diamond$

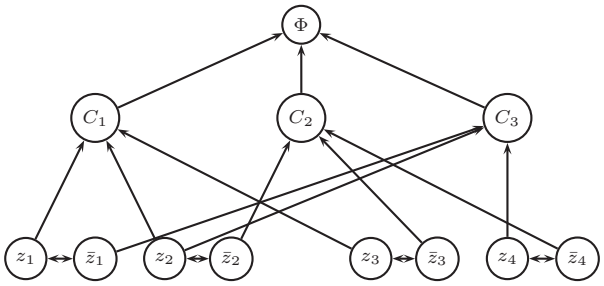


Figure 2: AF F_Φ for example CNF Φ.

For any CNF 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 (1996) and later rephrased in terms of AFs by Dunne and Bench-Capon (2002). We strengthen this result as follows.

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

Proof. As discussed above, AFs F of the form as given in Definition 6 provide us with a valid reduction from SAT to CA. To prove the assertion it is thus sufficient to show that for each CNF 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 6 has cycle-rank 1. $\square$

We now turn our attention to the Π_2^P-hard problem SA. The following reduction from QBFs to AFs is used by Dunne and Bench-Capon (2002).

Definition 7. *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 3. Consider $\Psi = \forall y_1 y_2 \exists z_3 z_4 (y_1 \vee y_2 \vee z_3) \wedge (\neg y_2 \vee \neg z_3 \vee \neg z_4) \wedge (\neg y_1 \vee y_2 \vee z_4)$. Figure 3 illustrates the corresponding AF G_Ψ. $\diamond$

As shown by Dunne and Bench-Capon (2002), the following holds for each QBF Ψ of the above form: Ψ is valid iff argument Ψ is contained in each $S \in pref(G_\Psi)$. 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.

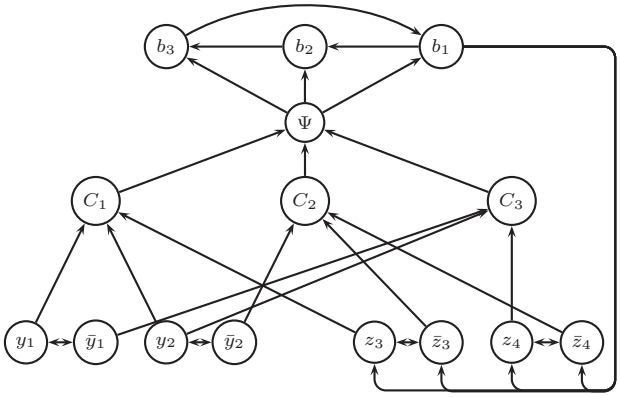


Figure 3: AF G_Ψ for example QBF Ψ.

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

Proof. We can proceed similar 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 which follows Definition 7 has 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 $C_1, \ldots, C_m$. All these have cycle-rank 1 or 0. □

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 8. *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*[1] *$t \in \mathcal{T}$ is of one of the following types:*

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

Here the operator $\dot\cup$ denotes the disjoint union of two sets and v denotes an arbitrary vertex in G.

Kloks (1994) 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.

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

Definition 9. *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.*

Definition 10. *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 such that s occurs in the subtree of $\mathcal{T}$ rooted at t. Moreover, let $X_{>t}$ denote $X_{\geq t} \setminus X_t$. We also use the following terminology: $F_t = F|_{X_t}$ is the* sub-framework in *t; $F_{\geq t} = F|_{X_{\geq t}}$ is the* sub-framework induced by (the subtree rooted at) *t.*

Note that the sub-framework induced by the root of such a decomposition of an AF F is F itself. W.l.o.g., we may restrict ourselves to nice tree decompositions where the bag of the root is empty. Unless stated otherwise, we assume below that $(\mathcal{T}, \mathcal{X})$ denotes a nice tree-decomposition for some given AF F.

Example 4. For the AF F from Example 1, we 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. Moreover, we add 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 4, which has to be read as follows. In each node t, the bag X_t contains the arguments in (solid) cycles. In addition, we depicted in each node t the AF F_t, i.e. the sub-framework in t; adding the dotted parts of the graphs, we obtain $F_{\geq t}$, the sub-framework induced by t. ◇

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

Definition 11. *Let $F = (A, R)$ be an AF and B a set of arguments. A set $E \subseteq A$ is a B-*restricted admissible set for *F, if E is conflict-free in F and E defends itself in F against all $a \in A \cap B$.*

Example 5. An AF $(\{e, f, g\}, \{(e, g), (g, f), (f, e)\})$ has $\{g\}$-restricted admissible sets $\emptyset$, $\{e\}$ and $\{g\}$. ◇

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 are now prepared to present the dynamic programming algorithm. Therefore, 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 12. *Given a coloring C for a node $t \in \mathcal{T}$, 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 $a \in$*

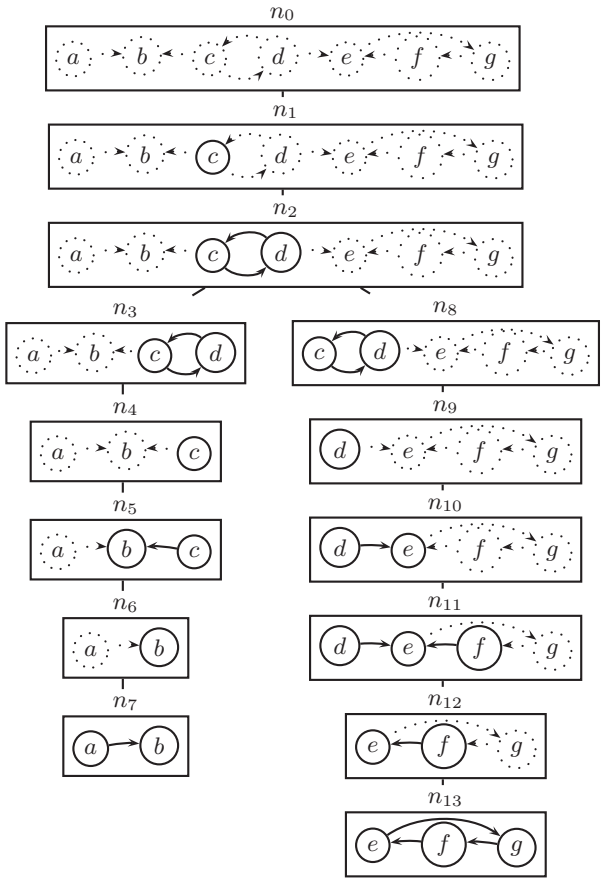


Figure 4: Tree decomposition of F with sub-frameworks.

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

Example 6. Consider the node n_{11} of our example tree decomposition with $X_{n_{11}} = \{d, e, f\}$ (see the rhs of the tree in Figure 4) 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 12 is $\{d, g\}$. $S = \{d\}$ would also be $X_{>t}$-restricted admissible but violates Condition (ii), since $C(f) = def$ and $S \not\rightarrowtail f$. $\diamond$

Our ultimate goal is to efficiently compute $\mathcal{C}_r$ for the root node 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, $\mathcal{C}_r$ thus characterizes the admissible sets of F.

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

Lemma 1. *Let S be 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. Let S be an $X_{>t}$-restricted admissible set for $F_{\geq t}$. Then, for each argument $a \in X_t$, one of the following conditions hold: (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 12 and, since S is $X_{>t}$-restricted admissible for $F_{\geq t}$, it holds that $S \in e_t(C)$. □

Moreover, different colorings characterize different extensions.

Lemma 2. *Let C, C' be different colorings for $t \in \mathcal{T}$. 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. For example, let $C(a) = def$ and $C'(a) = att$. By Definition 12, $C(a) = def$ implies $S \rightarrowtail a$. On the other hand, $C'(a) = att$ implies $S \not\rightarrowtail a$, a contradiction. Similar arguments hold for the other combinations of colors. □

To guarantee tractability, 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. The operations on colorings used in the next definition are depicted in Figure 5.

Definition 13. *Let $t \in \mathcal{T}$ be a node and let t', t'' its possible children. Depending on the node type of t, we define a* vcoloring *for t as follows:*

LEAF: Each coloring $X_t \rightarrow \{in, out, att, def\}$ where

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

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

FORGET: 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: If C is a vcoloring for t' and $X_t = X_{t'} \cup \{a\}$, then $C + a$ is a vcoloring for t; if also $a \not\rightarrowtail a$, $[C] \not\rightarrowtail a$, and $a \not\rightarrowtail [C]$ hold, then $C \dot{+} a$ is a vcoloring for t as well.

JOIN: 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.

Let us illustrate this idea on our running example.

Example 7. Recall the AF from Example 1 and its tree decomposition in Figure 4. Figure 6 illustrates the bottom-up computation of the vcolorings. Consider, for instance, the leaf node n_{13} 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 $F_{\geq n_{13}} = (\{e, f, g\}, \{(e, g), (g, f), (f, e)\})$. 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

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

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

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

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

Figure 5: Operations for $C, D : A \to \{\mathit{in}, \mathit{out}, \mathit{att}, \mathit{def}\}$.

with $[C] = \{f\}$. Here we have $C(g) = \mathit{att}$, which violates the construction for the *FORGET* node. 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 5 where we already analyzed exactly this situation). The next node n_{11} is of type *INSERT* and adds d. Consider the coloring C' for n_{12} with $C'(e) = \mathit{att}$ and $C'(f) = \mathit{def}$. We have two possibilities to add d. In case we want d to be in the set, we obtain the coloring C with $C(d) = \mathit{in}$, $C(e) = \mathit{def}$, $C(f) = \mathit{def}$ (note that e changes its color since it is now a "defeated attacker"); we have seen this coloring already in Example 6. The other possibility is to have d not in the set, resulting in the coloring C'' with $C''(d) = \mathit{out}$, $C''(e) = \mathit{att}$, $C''(f) = \mathit{def}$.

Due to lack of space, we have to omit a full discussion of the computation here. However, for a better understanding we also added the # column in Figure 6 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 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). ◇

Vcolorings provide us with exactly the same information as valid colorings.

Theorem 3. *For each coloring C for a node t, it holds that C is a valid coloring for t iff C is a vcoloring.*

Proof. We proceed by structural induction. For the induction begin, we have to show that vcolorings and valid colorings coincide on *LEAF* nodes. For the induction step, we show this property for *FORGET*, *INSERT*, and *JOIN* nodes.

LEAF. For a *LEAF* node t, we have $X_{>t} = \emptyset$ and, 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 or $[C]$ violates one of the conditions (ii) – (iv) in Definition 12. 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

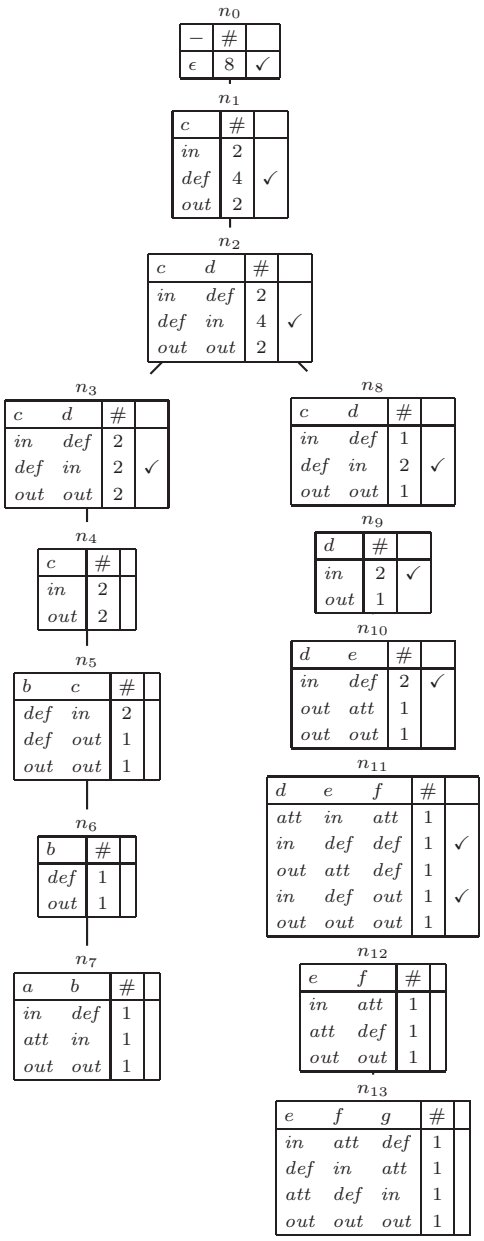


Figure 6: Computation of vcolorings for the example AF.

$C(x) = C(y) = \mathit{in}$. Hence, the first condition in Definition 13 for vcolorings at a *LEAF* node is violated, a contradiction.

Now suppose that C is valid coloring for t, i.e., C satisfies the conditions (i) – (iv) of a coloring (see Definition 12) and C is conflict-free in $F_{\geq t}$. Then C satisfies the condition of a vcoloring for a *LEAF* node according to Definition 13. For instance, let $x, y \in X_t$ with $C(x) = \mathit{in}$ and $y \rightarrowtail x$. Then, since C is a coloring, either case (ii) or case (iii) of Definition 12 applies and, thus, $C(y) \in \{\mathit{att}, \mathit{def}\}$ holds.

FORGET. Let t be a *FORGET* node with successor t', such that $X_t = X_{t'} \setminus \{a\}$. Of course, then also $X_{\geq t} = X_{\geq t'}$ and $X_{>t} = X_{>t'} \cup \{a\}$ hold.

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 $F_{\geq t'}$. Moreover, $S \in e_{t'}(C')$, i.e., C' is a valid coloring for t'. Hence, by the induction hypothesis, C' is a vcoloring for t' and, therefore, also $C = C' - a$ is a vcoloring for t.

(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 $F_{\geq t}$. As above, we may apply the induction hypothesis to conclude that C' (and thus also C) is a vcoloring.

Now let C be a vcoloring for t, i.e., there exists a vcoloring C' for t' with $C'(a) \neq att$ and $C = C' - a$. By the induction hypothesis, 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, $C \in \mathcal{C}_t$.

INSERT. Let t be an *INSERT* node with successor t', such that $X_t = X_{t'} \cup \{a\}$. 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 the induction hypothesis, 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 $a \in S$).

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 the induction hypothesis, C' is a valid coloring, i.e. there exists an $X_{>t'}$-restricted (and, hence, $X_{>t}$-restricted) admissible set S $e_{t'}(C')$ for $F_{\geq t'}$. It is easy to check that then $S \in e_t(C' + a)$. Moreover if the set $S \cup \{a\}$ is conflict-free 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.

JOIN. Due to the lack of space, we give only a sketch for this case.

Let t be a *JOIN* node 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 $S_1 \cap X_t = S_2 \cap X_t$.

To show that each vcoloring C for t is also a valid coloring for t, we proceed as follows: by definition, we have vcolorings C' for t' and respectively C'' for t'', such that $C = C' \bowtie C''$ and $[C'] = [C'']$. By the induction hypothesis, C' and C'' are also valid colorings for the respective nodes. Hence, there exists an $X_{>t'}$-restricted admissible set $S_1 \in e_{t'}(C')$ for $F_{\geq t'}$ and an $X_{>t''}$-restricted admissible set $S_2 \in e_{t''}(C'')$ for $F_{\geq t''}$. Using $[C'] = [C'']$, one can now show that $S = S_1 \cup S_2$ is then $X_{>t}$-restricted admissible for $F_{\geq t}$. It then remains to show that $S \in e_t(C)$, which can be done by checking that S satisfies the conditions (i) – (iv) in Definition 12 for every $a \in X_t$.

Now assume that $C \in \mathcal{C}_t$, i.e., there exists an $X_{>t}$-restricted admissible set $S \in e_t(C)$ for $F_{\geq t}$. We define $S_1 = S \cap X_{\geq t'}$ and $S_2 = S \cap X_{\geq t''}$. One can now show that 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' for t' and a coloring C'' for t'', such that $S_1 \in e_{t'}(C')$ and $S_2 \in e_{t''}(C'')$. Then $C' \in \mathcal{C}_{t'}$, $C'' \in \mathcal{C}_{t''}$, and, therefore, by the induction hypothesis, C' and C'' are vcolorings. 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 4 possible values $\{in, def, att, out\}$ and by exploiting the conditions (i) – (iv) in Definition 12 as well as the definition of the $\bowtie$ operator in Figure 5. □

Let us briefly describe how credulous acceptance can now 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. If the coloring of the root has the mark, then we know that credulous acceptance for this argument holds

Example 8. Recall the computation from Example 7 in Figure 6. 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. In the remaining nodes of the tree-decomposition we mark a coloring iff it is constructed using at least one marked coloring. 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 successor 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 straightfor-

ward method (e.g., nested loops) for computing a node's vcolorings from the vcolorings at the child node(s).

Theorem 4. *Deciding* CA *for an AF $F = (A, R)$ of treewidth $k-1$ can be done in time $O(10^k \cdot k \cdot |A|)$.*

Proof. First, we observe that the number of colorings for each bag is bounded by 4^k, since there at most k arguments in a bag 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 7, 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).

Characterizing preferred extensions. So far, we have solved the credulous acceptance problem via 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 14. *Given $t \in \mathcal{T}$ and a pair (C, Γ) for t, define $e_t(C, \Gamma)$ as the collection of sets S which satisfy the following conditions: (i) $S \in e_t(C)$; (ii) $\forall C' \in \Gamma\, \exists E \in e_t(C')$ such that $S \subset E$; (iii) for all $X_{>t}$-restricted admissible (for $F_{\geq t}$) sets E such that $S \subset E$ it holds that $\exists C' \in \Gamma$ 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 3. *Let S be an $X_{>t}$-restricted admissible set for $F_{\geq t}$, then there is a pair (C, Γ) with $S \in e_t(C, \Gamma)$.*

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 14: (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 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 4. *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 14 (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 14 (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 is in relation to preferred extensions (see Definition 14) and we have $e_r(\epsilon, \emptyset) = pref(F)$. 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 5) to *sets of vcolorings*. By slight abuse of notation, we overload the operators $-$, $+$, $\dot{+}$, and $\bowtie$ as follows:

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

Definition 15. *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 Γ the set of all $C' \in \mathcal{C}_t$ with $[C] \subset [C']$, 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 $C' \bowtie C''$.

Example 9. Recall the AF from Example 4. The computation of vpairs for nodes t is illustrated in Figure 7. 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 lhs branch and the coloring C_1 with $C_1(b) = def$ and $C_1(c) = in$, i.e. $[C_1] = \{c\}$. Note that $e_t(C_1) = \{\{c\},\{a,c\}\}$. However, $e_t(C_1,\{C_1\}) = \{\{c\}\}$ (since we have $\{a,c\}$ as certificate), while $e_t(C_1,\emptyset) = \{\{a,c\}\}$. $\diamond$

Theorem 5. *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.*

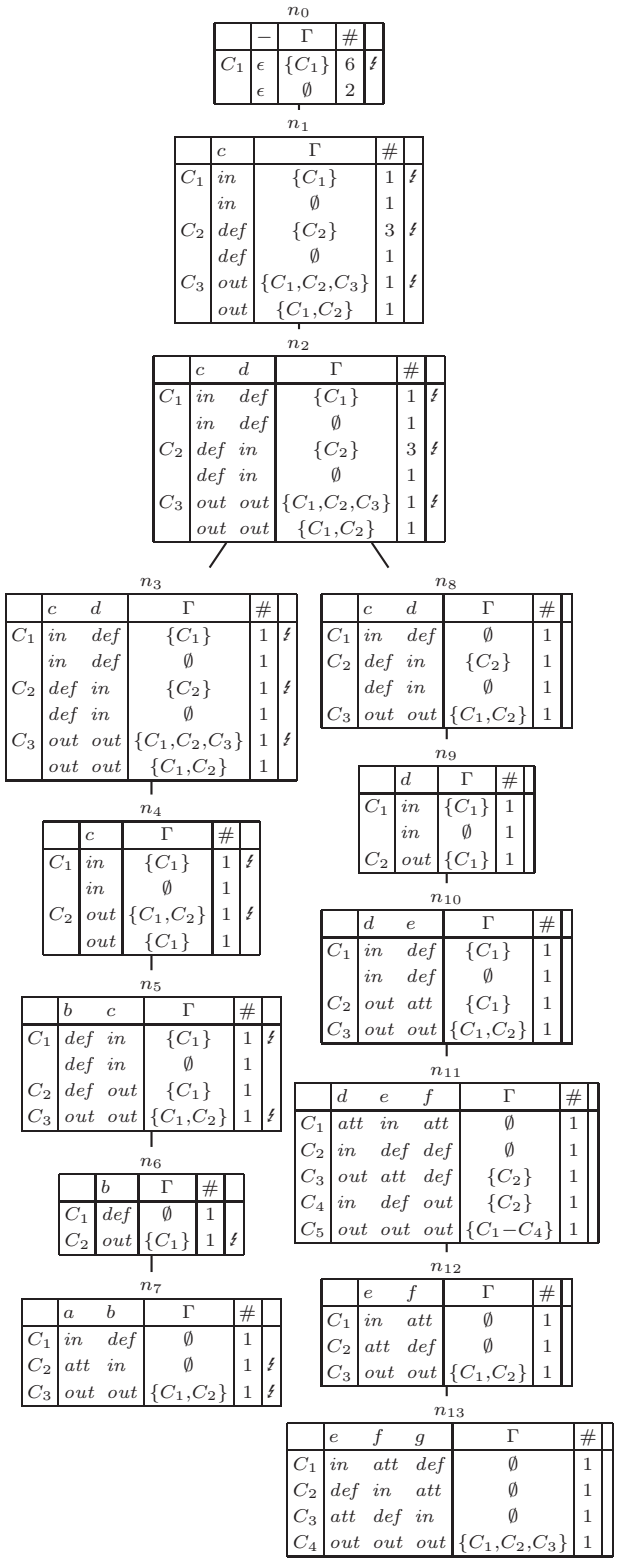


Figure 7: Computation of vpairs for the example AF.

Proof. As in Theorem 3, the proof proceeds by structural induction. For the induction begin, we have to show that vpairs and valid pairs coincide on *LEAF* nodes. For the induction step, we have to show this property for the remaining nodes. We give the details for *LEAF* and *FORGET* nodes. *INSERT* and *JOIN* nodes can be treated analogously.

LEAF. For a *LEAF* node t, the $X_{>t}$-restricted admissible sets for $F_{\geq t}$ coincide with the sets $[C]$ for the valid colorings C for t. Moreover, the valid colorings and vcolorings for t coincide by Theorem 3. Now let (C, Γ) be a valid pair for t. Then, by Definition 14, $[C] \in e_t(C, \Gamma)$. Hence, by Definition 15, (C, Γ) is a vpair for t. Conversely, let (C, Γ) be a vpair for t and let $S = [C]$. By Definition 13, S is $X_{>t}$-restricted admissible for $F_{\geq t}$. Hence, by Definition 14 and Definition 15, $S \in e_t(C, \Gamma)$. (C, Γ) is thus a valid pair for t.

FORGET. Let t be a *FORGET* node with successor t' such that $X_t = X_{t'} \setminus \{a\}$. Let (C, Γ) be a valid pair for t. Then there exists $S \in e_t(C, \Gamma)$. 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 3, there exists a valid pair (C', Γ') for t' with $S \in e_{t'}(C', \Gamma')$. By the induction hypothesis, (C', Γ') is a vpair for t'. Since S is $X'_{>t}$-restricted admissible 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 Theorem 3.

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 14, 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 14, there exists $D \in \Gamma$ with $E \in e_t(D)$. As in the proof of Theorem 3, we thus have $D = D' - a$. Hence, $\Gamma' - a \subseteq \Gamma$.

Now consider an arbitrary D in Γ. By condition (ii) of Definition 14, 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 14 and since E is also $X_{>t'}$-restricted admissible, there exists $D' \in \Gamma'$ with $E \in e_{t'}(D')$. As in the proof of Theorem 3, we thus have $D = D' - a$. Hence, $\Gamma \subseteq \Gamma' - a$.

We now show that every vpair for a *FORGET* node is a valid pair. 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 the induction hypothesis, (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 Theorem 3, $S \in e_t(C)$ holds since $C = C' - a$. It remains to show that also conditions (ii) and (iii) of Definition 14 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 Theorem 3, 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 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 Theorem 3, also $E \in e_t(D' - a)$. □

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 10. Let us now consider the problem of deciding if the argument a is skeptically accepted in our example AF. In Figure 7 we illustrate the vpairs which are marked as contradictory for skeptical acceptance with a ↯ 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 7 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. ◇

Theorem 6. *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 4, 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 4, the computationally most expensive node type is the *JOIN* node, which is the one we shall focus below. 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 4, 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 combinations of vpairs fits into $O(10^k \cdot k + 2^{2k} \cdot k) = O(2^{4k})$. We thus end up with the upper bound $O((2^{2^{2k}+2k})^2 \cdot 2^{4k}) = O((2^{2^{2k+1}+4k}) \cdot 2^{4k}) = O(2^{2^{2k+1}+8k})$. □

Conclusion

In this paper, we turned some theoretical tractability results for argumentation frameworks of bounded tree-width into efficient algorithms. Moreover, we showed that some further

graph parameters (which, in contrast to tree-width, apply to directed graphs), do not lead to similar tractability results. In future work, we will investigate clique-width (Courcelle and Olariu 2000) which can be considered as more general than tree-width since there are sets of graphs of bounded clique-width but arbitrarily high tree-width, while sets of graphs of bounded tree-width also have bounded clique-width.

Several algorithms for the problems discussed in this paper have been presented in the literature. We mention the work by Doutre and Mengin (2001) 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 relied on the concept of colorings. They look similar to labellings (see (Caminada and Gabbay 2009; Modgil and Caminada 2009)). However, labellings are defined for complete frameworks, while we require here a concept which also applies to sub-frameworks (recall that for our complexity results in Theorem 4 and Theorem 6, 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 labellings for different semantics might help us in extending our algorithms to other semantics, which is indeed a major topic for future work. In fact, we plan to adapt our algorithms for complete, stable, stage, semi-stable, and ideal 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.

Finally, we plan to implement the presented algorithms. Note that our description of the actions required along the bottom-up traversal of a tree-decomposition is already quite close to an implementation in a functional programming language. For instance, see (Jakl, Pichler, and Woltran 2009) for such a realization in the area of answer-set programming via Haskell.

References

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

Berwanger, D.; Dawar, A.; Hunter, P.; and Kreutzer, S. 2006. DAG-width and parity games. In *Proc. STACS'06*, volume 3884 of *LNCS*, 524–536. Springer.

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

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

Caminada, M. 2006. Semi-stable semantics. In *Proc. COMMA'06*, volume 144 of *Frontiers in Artificial Intelligence and Applications*, 121–130. IOS Press.

Coste-Marquis, S.; Devred, C.; and Marquis, P. 2005. Symmetric argumentation frameworks. In *Proc. ECSQARU'05*, volume 3571 of *LNCS*, 317–328. Springer.

Courcelle, B., and Olariu, S. 2000. Upper bounds to the clique-width of graphs. *Discr. Appl. Math.* 101(1-3):77–114.

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

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

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

Doutre, S., and Mengin, J. 2001. Preferred extensions of argumentation frameworks: Query answering and computation. In *Proc. IJCAR'01*, volume 2083 of *LNCS*, 272–288. Springer.

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

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

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

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

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

Gruber, H. 2008. Digraph complexity measures and applications in formal language theory. In *Proc. MEMICS'08*, 60–67.

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

Jakl, M.; Pichler, R.; and Woltran, S. 2009. Answer-set programming with bounded treewidth. In *Proc. IJCAI'09*, 816–822. AAAI Press.

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

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

Modgil, S., and Caminada, M. 2009. Proof theories and algorithms for abstract argumentation frameworks. In *Argumentation in Artificial Intelligence*. Springer. 105–129.

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

Verheij, B. 1996. Two approaches to dialectical argumentation: Admissible sets and argumentation stages. In *Proc. NAIC'96*, 357–368.