Valued-Based Argumentation for Tree-like Value Graphs

Eun Jung Kim, Sebastian Ordyniak

Valued-Based Argumentation for Tree-like Value Graphs

Eun Jung KIM [a] and Sebastian ORDYNIAK [b]

[a] *LAMSADE-CNRS, Université de Paris-Dauphine, 75775 Paris, France*

[b] *Institute of Information Systems, Vienna University of Technology, 1040 Vienna, Austria*

Abstract. We consider value-based argumentation frameworks (VAFs) introduced by Bench-Capon (*J. Logic Comput.* 13, 2003), which has been established as a fruitful model to study abstract argumentation systems. It takes into account the relative importance among arguments which reflects the value system of an audience. The central issue in the study of VAFs is the decision problems of subjective acceptance and objective acceptance: an argument is subjectively (objectively, resp.) accepted if it is accepted with respect to one audience (all possible audiences, resp.) An important limitation for using VAFs in real-world applications is the computational intractability of the acceptance problems. We identify nontrivial classes in terms of structural restrictions on the underlying graph structure of VAFs and present a polynomial-time algorithm in the spirit of dynamic programming. We supplement the tractability by the hardness result. This extends and generalize the results of Dunne (*COMMA* 2010) and Kim et al. (*Artificial Intelligence* 175, 2011).

Keywords. Value-based argumentation frameworks, treewidth, NP-hardness, polynomial-time tractability, subjective and objective acceptance

1. Introduction

The study of arguments as abstract entities and their interaction as introduced by Dung [7] has become one of the most active research branches within Artificial Intelligence and Reasoning [3,4,16]. Some domains such as mathematics require rigorous reasoning as a way to convince and a conflict between arguments are not allowed. Most argumentation, however, in human activities involve practical reasoning such as disputes in law and politics. For example, upon a case being handled in court, one party tries to effectively attack the other party's arguments and defend its own. As it is impossible to rigorously demonstrate one side's point of view, now the main issue is to find a strategy to effectively appeal its standpoint. The study of arguments takes into account possible conflicts, called the *attacks*, among the arguments. A suitable selection of arguments which is coherent and which defends itself from the attack of other arguments is the central problem (*acceptance problem*) in the study of argumentation system. Abstract argumentation provides concepts and formalisms to study, represent, and process various reasoning problems most prominently in defeasible reasoning (see, e.g., [15,6]) and agent interaction (see, e.g.,[14]).

In practice, there may be more than one suitable selection of arguments and the jury may advocate one standpoint over the others. Such a preference can be the result of the value system on which the jury base its decision. An attack of the argument B using the argument A can be considered vain if A is based on higher value than B in the value system of the jury. *Value-based argumentation*, introduced by Bench-Capon [2], extends the argumentation framework in an attempt to capture the aspect of preferences as a result of values. In this extended setting, each argument is allocated to a *value* and one value system, as a ranking of the values, is construed as an *audience*. The feasibility of a standpoint is formalized as the acceptance of an argument with respect to one audience (*subjective acceptance*) or to all possible audiences (*objective acceptance*).

An important limitation for using valued-based argumentation systems in real-world applications is the *computation intractability* of the two basic acceptance problems: deciding whether a given argument is subjectively accepted is NP-hard, deciding whether it is objectively accepted is co-NP-hard [10]. Therefore it is important to identify classes of value-based systems that are still useful and expressible, but allow polynomial-time tractable acceptance decision.

Previous Studies and Our Contribution: In this paper, we identify nontrivial classes of value-based systems for which the acceptance problems are tractable. We distinguish the classes in terms of the following notions. The *value-width* of a value-based system is the largest number of arguments of the same value. The *extended graph structure* of a value-based system has as nodes the arguments of the value-based system and two arguments are joined by an edge if either one attacks the other or both share the same value. The *value graph* of a value-based system has as vertices the values of the system and two values v_1 and v_2 are joined by a directed edge if some argument of value v_1 attacks an argument of value v_2 [8].

In [9], the acceptance problem was shown to be tractable when the value graph is a tree such that both the degree and the number of branching nodes are bounded. In contrast to the tractability result, it was also proved that once the treewidth of the value graph becomes two, the acceptance problems are (co-)NP-complete even if the witnessing tree is a path. Notice that both the degree and the number of branching nodes are very small in a path. At first glance, this appears to imply that the tractability cannot be extended beyond trees. Our main algorithmic result states that the acceptance problems can be solved in uniform polynomial time (i.e. the order of the polynomial depends neither on w nor or Δ) if both the treewidth w of the value graph and Δ are bounded. While the definition of Δ is deferred to a later section, we remark that both Δ and w are bounded when the treewidth of the extended graph structure is bounded. Due to this fact, our algorithmic result generalizes the main tractable classes discussed in [12]: value-based systems whose extended graph structure has bounded treewidth and value-based systems of bounded value-width whose value graphs have bounded treewidth. Our nontrivial algorithm explores the *tree decomposition* of the value graph in the spirit of dynamic programming.

We supplement the tractability by the hardness result such as: the acceptance problems remain (co-)NP-hard for value-based systems whose value graphs are trees. This strengthens the hardness result of [9]. It is interesting to note that in the construction of the proof, the value graph has exactly one node which has (a) unbounded number of children, and (b) Δ is unbounded as well. This curves out sharply both the tractable classes in [9] and in this paper.

2. Preliminaries

2.1. Abstract Argumentation

An *abstract argumentation system* or *argumentation framework* (*AF*, for short) is a pair (X, A) where X is a finite set of elements called *arguments* and $A \subseteq X \times X$ is a binary relation called the *attack relation*. If $(x, y) \in A$ we say that x *attacks* y.

Next we define commonly used semantics of AFs as introduced by Dung [7] (for a discussion of other semantics and variants, see e.g., Baroni and Giacomin's survey [1]). Let $F = (X, A)$ be an AF and $S \subseteq X$.

1. S is *conflict-free* in F if there is no $(x, y) \in A$ with $x, y \in S$.
2. S is *acceptable* in F if whenever an argument in S is attacked by an argument $y \in X \setminus S$ then there is an argument in S that attacks y.
3. S is *admissible* in F if it is conflict-free and acceptable.
4. S is a *preferred extension* of F if S is admissible in F and there is no admissible set S' of F that properly contains S.

In the following we provide some specific notation that we will need for the presentation of our algorithm. For an acyclic AF $F = (X, A)$, two sets $X_1, X_2 \subseteq X$, and a new argument $s \notin X$,

we define $\sigma(F, X_1, s) = \sigma(F, X_1)$ to be the AF $F' = (X', A')$ such that $X' = X \cup \{s\}$ and $A' = A \cup \{ (s, x) \mid x \in X_1 \}$. Furthermore, we define $\alpha_F(X_1, X_2) = \{ x \in X_2 \mid (y, x) \in A \text{ and } y \in X_1 \}$, $\delta_F^-(X_1) = \{ x \in X_1 \mid (x, y) \in A \text{ and } y \in X \setminus X_1 \}$, and $\delta_F^+(X_1) = \{ x \in X_1 \mid (y, x) \in A \text{ and } y \in X \setminus X_1 \}$.

2.2. Value-Based Argumentation

A *value-based argumentation framework* (VAF) is a tuple $F = (X, A, V, \eta)$ where (X, A) is an argumentation framework, V is a set of *values* and η is a mapping $X \to V$ such that the graph $(\eta^{-1}(v), \{ (x, y) \in A \mid x, y \in \eta^{-1}(v) \})$ is acyclic for all $v \in V$. An *audience* $\leq$ for a VAF is a partial ordering $\leq$ on the set of values of F. Given a VAF $F = (X, A, V, \eta)$ and an audience $\leq$ for F, we define the AF $F_\leq = (X, A_\leq)$ by setting $A_\leq = \{ (x, y) \in A \mid \neg(\eta(x) < \eta(y)) \})$. An audience $\leq$ is *specific* if it is a total ordering on V. For an audience $\leq$ we also define $<$ in the obvious way, i.e. $x < y$ if and only if $x \leq y$ and $x \neq y$. Note that if $\leq$ is a specific audience, then $F_\leq = (X, A_\leq)$ is an acyclic digraph and thus, has a unique preferred extension [3]. For a VAF $F = (X, A, V, \eta)$ and a set $V' \subseteq V$ we denote by $F - V'$ the VAF obtained from F after deleting all arguments with value $v \in V'$ and all attacks involving these arguments. We also define $F[V']$ to be the VAF $F - (V \setminus V')$ and $\eta^{-1}(V')$ to be the set of arguments with value $v \in V'$.

Let $F = (X, A, V, \eta)$ be a VAF. We say that an argument $x_1 \in X$ is *subjectively accepted in* F if there exists a specific audience $\leq$ such that x_1 is in the unique preferred extension of $F_\leq$. Similarly, we say that an argument $x_1 \in X$ is *objectively accepted in* F if x_1 is contained in the unique preferred extension of $F_\leq$ for every specific audience $\leq$.

We consider the following decision problems.

SUBJECTIVE ACCEPTANCE
Instance: A VAF $F = (X, A, V, \eta)$ and an argument $x_1 \in X$.
Question: Is x_1 subjectively accepted in F?

OBJECTIVE ACCEPTANCE
Instance: A VAF $F = (X, A, V, \eta)$ and an argument $x_1 \in X$.
Question: Is x_1 objectively accepted in F?

Considering an instance (F, x_1) of SUBJECTIVE/OBJECTIVE ACCEPTANCE, we shall refer to the argument x_1 as the *query argument*.

Let $F = (X, A, V, \eta)$ be a VAF. We define the *value-width* of F, denoted by $vw(F)$, as the largest number of arguments with the same value, i.e., $vw(F) = \max_{v \in V} |\eta^{-1}(v)|$. The *value graph* of F is the directed graph $G_F^{\text{val}} = (V, E)$ whose vertices are the values of F and where two values u, v are joined by a directed edge from u to v (in symbols $(u, v) \in E$) if and only if there exist some argument $x \in X$ with $\eta(x) = u$, some argument $y \in X$ with $\eta(y) = v$, and $(x, y) \in A$. The *extended graph structure* of F is the (undirected) graph $G_F^{\text{ext}} = (X, E)$ whose vertices are the arguments of F and where two arguments x, y are joined by an edge if an only if $(x, y) \in A$ or $\eta(x) = \eta(y)$. Furthermore, for a VAF $F = (X, A, V, \eta)$, $X' \subseteq X$, $V' \subseteq V$, and a specific audience $\leq$ on V' we define $\alpha_\leq(X', V')$ to be $\alpha_{F_\leq}(X', \eta^{-1}(V'))$.

2.3. Labelings

Let X be a set of arguments. A *partial labeling* λ of X is a function $\lambda : Y \to \{IN, OUT\}$ where $Y \subseteq X$. For a partial labeling λ we set $IN(\lambda) = \{ x \mid \lambda(x) = IN \}$, $OUT(\lambda) = \{ x \mid \lambda(x) = OUT \}$ and $DEF(\lambda) = Y$. Let X and Y be two sets of arguments and λ_X be a partial labeling of X and λ_Y a partial labeling of Y. We say λ_X is *compatible* with λ_Y if $\lambda_X(x) = \lambda_Y(x)$ for every $x \in DEF(\lambda_X) \cap DEF(\lambda_Y)$. If λ_X is compatible with λ_Y we define the partial labeling $\lambda_X \cup \lambda_Y$ of $X \cup Y$ by setting $(\lambda_X \cup \lambda_Y)(x) = \lambda(x)$ if $x \in DEF(\lambda_X)$ and $(\lambda_X \cup \lambda_Y)(x) = \lambda_Y(x)$ if $x \in DEF(\lambda_Y)$. We also define the partial labeling $\lambda_X \cap \lambda_Y$ by setting $(\lambda_X \cap \lambda_Y)(x) = \lambda_X(x)$ for every $x \in DEF(\lambda_X) \cap DEF(\lambda_Y)$.

Let $F = (X, A)$ be an acyclic AF. We define the propagation of F, denoted $\Lambda(F)$, to be the labeling obtained by the following simple labeling procedure. Repeatedly apply the following two rules to the arguments in X until each of them is either labeled IN or OUT:

L1 An argument x is labeled IN if all arguments that attack x are labeled OUT (in particular, if x is not attacked by any argument).

L2 An argument x is labeled OUT if it is attacked by some argument that is labeled IN.

It is well-known that the unique preferred extension $\mathrm{GE}(F)$ equals $IN(\Lambda(F))$ if F is acyclic.

Proposition 1 ([7]). *Let $F = (X, A)$ be an acyclic AF. Then $IN(\Lambda(F))$ is the unique preferred extension of F. Furthermore, $\Lambda(F)$ can be computed in time $O(|X| + |A|)$.*

The following lemma is central to our dynamic programming algorithm. Informally, it allows us to split an AF into two parts that can be labeled separately as long as we take care of the interactions between the two parts.

Lemma 1. *Let $F_L = (X_L, A_L)$ be an acyclic AF, $X_1, X_2 \subseteq X_L$ such that $X_L = X_1 \cup X_2$ and $X_1 \cap X_2 = \emptyset$, $I_L \subseteq \delta^-_{F_L}(X_2)$, $O_L \subseteq \delta^+_{F_L}(X_2)$, $F_1 = \sigma(F_L[X_1], \alpha_{F_L}(I_L, X_1), s_1)$, and $F_2 = \sigma(F_L[X_2], O_L, s_2)$. If $O_L = \alpha_{F_L}(IN(\Lambda(F_1)), X_2)$ and $I_L = IN(\Lambda(F_2)) \cap \delta^-_{F_L}(X_2)$ or $I_L = IN(\Lambda(F_L)) \cap \delta^-_{F_L}(X_2)$ then $\Lambda(F_L)$ is compatible with $\Lambda(F_1) \cup \Lambda(F_2)$.*

Proof. Let $O_F = (x_1, \ldots, x_n)$ be an acyclic ordering of the arguments of F_L. We show by induction over n that $\Lambda(F_L)$ is compatible with $\Lambda(F_1) \cup \Lambda(F_2)$. Let $1 \leq i \leq n$ and assume that the claim has already been shown for every $j < i$. We distinguish four cases depending on whether x_i is contained in F_1 or F_2 and depending on whether x_i is labeled IN or OUT by the labeling $\Lambda(F_L)$.

So suppose that x_i is contained in F_1 and $\Lambda(F_L)(x_i) = IN$. According to rule L1 it holds that $\Lambda(F_L)(y) = OUT$ for every attacker y of x_i in F_L. By the induction hypothesis it follows that $\Lambda(F_L)(y) = \Lambda(F_2)(y) = OUT$ for every attacker $y \in X_2$ of x_i in F_L. Consequently, y is neither contained in $IN(\Lambda(F_2)) \cap \delta^-_{F_L}(X_2)$ nor in $IN(\Lambda(F_L)) \cap \delta^-_{F_L}(X_2)$ and thus all the attackers of x_i in F_1 are contained in X_1. By the induction hypothesis we have that $\Lambda(F_1)(y) = \Lambda(F_L)(y) = OUT$ for every attacker y of x_i in F_1. Hence, $\Lambda(F_1)(x_i) = IN$ by rule L1 and the lemma holds for this case.

We now consider the case that x_i is contained in F_1 and $\Lambda(F_L)(x_i) = OUT$. According to rule L2 x_i is attacked by some y in F_L such that $\Lambda(F_L)(y) = IN$. If $y \in X_1$ then y is also an attacker of x_i in F_1 and $\Lambda(F_1)(y) = \Lambda(F_L)(y) = IN$ by the induction hypothesis and by rule L2 $\Lambda(F_1)(x_i) = OUT$. So suppose that $y \in X_2$. Then $\Lambda(F_2)(y) = \Lambda(F_L)(y) = IN$ because of the induction hypothesis and hence y is contained in both $IN(\Lambda(F_2) \cap \delta^-_{F_L}(X_2)$ and $IN(\Lambda(F_L) \cap \delta^-_{F_L}(X_2)$ implying that y is contained in I_L. It follows that s_1 attacks x_i in F_1 and because $\Lambda(F_1)(s_1) = IN$ we obtain $\Lambda(F_1)(x_i) = OUT$ from rule L2 and the lemma holds for this case.

Suppose now that x_i is contained in F_2 and $\Lambda(F_L)(x_i) = IN$. Then, according to rule L1 it holds that $\Lambda(F_L)(y) = OUT$ for every attacker y of x_i in F_L. By the induction hypothesis it follows that $\Lambda(F_L)(y) = \Lambda(F_1)(y) = OUT$ for every attacker $y \in X_1$ of x_i in F_L. Consequently, $x_i \notin O_L$ and all the attackers of x_i in F_2 are contained in X_2. By the induction hypothesis we have that $\Lambda(F_L)(y) = \Lambda(F_2)(y) = OUT$ for every attacker y of x_i in F_2. Hence, $\Lambda(F_2)(x_i) = IN$ by rule L1 and the lemma holds for this case.

It remains to show the lemma for the case that x_i is contained in F_2 and $\Lambda(F_L)(x_i) = OUT$. According to rule L2 x_i is attacked by some y in F_L such that $\Lambda(F_L)(y) = IN$. If $y \in X_2$ then y also attacks x_i in F_2 and $\Lambda(F_2)(y) = \Lambda(F_L)(y) = IN$ by the induction hypothesis. It follows from rule L2 that $\Lambda(F_2)(x_i) = OUT$. So suppose that $y \in X_1$. Then $\Lambda(F_1)(y) = \Lambda(F_L)(y) = IN$ because of the induction hypothesis and consequently $x_i \in O_L$. It follows that s_2 attacks x_i in F_2 and because $\Lambda(F_2)(s_2) = IN$ we obtain $\Lambda(F_2)(x_i) = OUT$ which completes the proof. □

2.4. Tree Decompositions

Treewidth is an important graph parameter that indicates in a certain sense the "tree-likeness" of a graph.

The treewidth of a graph $G = (V, E)$ is defined via the following notion of decomposition: a *tree decomposition* of G is a pair (T, χ) where T is a tree and χ is a labeling function with $\chi(t) \subseteq V$ for every tree node t, such that the following conditions hold:

1. Every vertex of G occurs in $\chi(t)$ for some tree node t.
2. For every edge $\{u, v\}$ of G there is a tree node t such that $u, v \in \chi(t)$.
3. For every vertex v of G, the tree nodes t with $v \in \chi(t)$ induce a connected subtree of T.

The *width* of a tree decomposition (T, χ) is the size of a largest set $\chi(t)$ minus 1 among all nodes t of T. A tree decomposition of smallest width is *optimal*. The *treewidth* of a graph G, denoted $tw(G)$, is the width of an optimal tree decomposition of G.

Given G with n vertices and a constant w, it is possible to decide whether G has treewidth at most w, and if so, to compute an optimal tree decomposition of G in time $O(n)$ [5]. Furthermore there exist powerful heuristics to compute tree decomposition of small width in a practically feasible way [11].

When designing algorithms on tree decompositions it is convenient to consider tree decompositions in the following normal form [13]: A triple (T, χ, r) is a *nice tree decomposition* of a graph G if (T, χ) is a tree decomposition of G, the tree T is rooted at node r, and each node of T is of one of the following four types:

1. a *leaf node*: a node having no children;
2. a *join node*: a node t having exactly two children t_1, t_2, and $\chi(t) = \chi(t_1) = \chi(t_2)$;
3. an *introduce node*: a node t having exactly one child t', and $\chi(t) = \chi(t') \cup \{v\}$ for a vertex v of G;
4. a *forget node*: a node t having exactly one child t', and $\chi(t) = \chi(t') \setminus \{v\}$ for a vertex v of G.

For a nice tree decomposition (T, χ, r) we define $\chi^*(t)$ to be the union of all the sets $\chi(t')$ where t' is contained in the subtree of T rooted at t. Furthermore, we define the set N_t of *forgotten nodes* to be $N_t = \chi^*(t) \setminus \chi(t)$.

The following facts follow easily from the definition of a (nice) tree decomposition and will be used in the sequel.

Proposition 2. *Let t be a join node with children t_1 and t_2. Then $N_{t_1} \cap N_{t_2} = \emptyset$ and there is no edge between a vertex $u \in N_{t_1}$ and a vertex $v \in N_{t_2}$ in G.*

Proposition 3. *Let t be an introduce node with child t' such that $\chi(t) = \chi(t') \cup \{v_0\}$. Then there is no edge from v_0 to a vertex $v \in F_t$. Furthermore, $v_0 \notin F_{t'}$.*

Proposition 4. *Let t be a forget node with child t' such that $\chi(t) = \chi(t') \setminus \{v_0\}$. Then there is no edge from v_0 to a vertex $v \in V(G) \setminus \chi^*(t)$.*

Given a tree decomposition of a graph G of width w, and a vertex $r \in V(G)$, one can effectively obtain in time $O(|V(G)|)$ a nice tree decomposition (T, χ, r) with $O(|V(G)|)$ nodes and of width at most w such that $\chi(r) = \{v\}$ [13].

3. Hardness for Trees

In this section we show that both the problems SUBJECTIVE ACCEPTANCE and OBJECTIVE ACCEPTANCE are intractable for value-based systems even when their value graphs are trees.

Theorem 1. (A) SUBJECTIVE ACCEPTANCE remains NP-hard for value-based systems whose value graphs are trees. (B) OBJECTIVE ACCEPTANCE remains co-NP-hard for value-based systems whose value graphs are trees.

Proof. We start by showing part (A) of the theorem by devising a polynomial reduction from 3-SAT. Let Φ be a 3-CNF formula with clauses $C_1, \ldots, C_m$ and variables $x_1, \ldots, x_n$. In the following we construct a value-based system $F = (X, A, V, \eta)$ whose value graph is a tree such that the query argument $\Phi \in X$ is subjectively accepted in F if and only if the formula Φ is satisfiable.

The set X contains the following arguments:

- The query argument Φ;
- Seven arguments v_i, v_i^n, v_i^p, x_i^n, x_i^p, y_i^n, and y_i^p for every $1 \leq i \leq n$;
- One argument C_j for every $1 \leq j \leq m$.

The set A contains the following attacks:

- Three attacks (v_i, Φ), (v_i^p, v_i), and (v_i^n, v_i) for every $1 \leq i \leq n$;
- One attack (C_j, Φ) for every $1 \leq j \leq m$;
- Four attacks (x_i^p, v_i^p), (x_i^n, v_i^n), (y_i^p, x_i^p), and (y_i^n, x_i^n) for every $1 \leq i \leq n$;
- One attack (x_i^p, C_j) for every clause C_j where the variable x_i occurs positively;
- One attack (x_i^n, C_j) for every clause C_j where the variable x_i occurs negatively.

The set V contains the following values:

- One value c for the arguments in $X \setminus \{ x_i^n, x_i^p, y_i^n, y_i^p \mid 1 \leq i \leq n \}$, i.e., $\eta(x) = c$ if and only if $x \in X \setminus \{ x_i^n, x_i^p, y_i^n, y_i^p \mid 1 \leq i \leq n \}$;
- Four values x_i^n, x_i^p, y_i^n, y_i^p for every $1 \leq i \leq n$ such that $\eta(x) = x$ if and only if $x \in \{ x_i^n, x_i^p, y_i^n, y_i^p \mid 1 \leq i \leq n \}$.

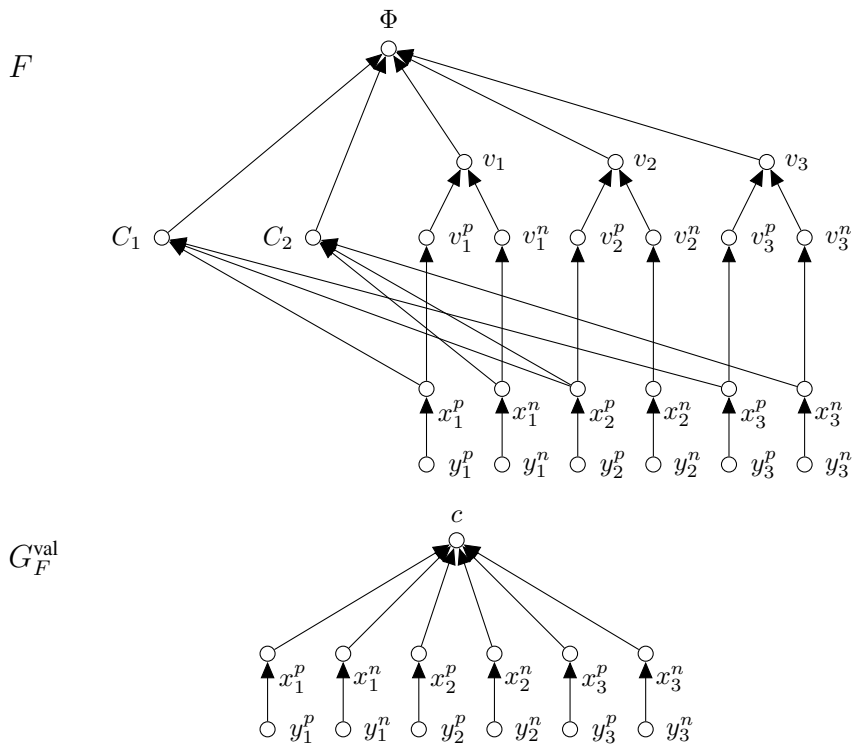

Figure 1. Illustration for the reduction used in the proof of Theorem 1, showing the AF F together with it's value graph G_F^{val}, obtained from the 3-CNF formula $\phi = C_1 \wedge C_2$ with $C_1 = x_1 \vee x_2 \vee x_3$ and $C_2 = \neg x_1 \vee x_2 \vee \neg x_3$.

An illustration for the VAF F and it's value graph is given in Figure 1. It is not difficult to see that the value graph G_F^{val} of the constructed value-based system is a tree and that the VAF F can be constructed from Φ in polynomial time.

To establish part (A) of the theorem it remains to show that the formula Φ is satisfiable if and only if the argument Φ is subjectively accepted in F.

Suppose that the formula Φ is satisfied by an assignment β. We consider a specific audience $\leq$ defined as:

- $v > c$ for every $v \in V \setminus \{c\}$;

- $\eta(y_i^p) < \eta(x_i^p)$ and $\eta(y_i^n) > \eta(x_i^n)$ whenever $\beta(x_i) =$ true;
- $\eta(y_i^p) > \eta(x_i^p)$ and $\eta(y_i^n) < \eta(x_i^n)$ whenever $\beta(x_i) =$ false.

Because $F_\leq$ is acyclic it has a unique preferred extension GE($F_\leq$) which using Proposition 1 is equal to $IN(\Lambda(F_\leq))$. We need to show that the argument Φ is labeled IN by the labeling $\Lambda(F_\leq)$. Because β assigns a value to every variable of the formula Φ it follows that for every $1 \leq i \leq n$ one of the arguments x_i^p and x_i^n is labeled IN and the other is labeled OUT, and $\Lambda(F_\leq)(x_i^p) = IN$ if and only if $\beta(x_i) =$ true. Hence, for every $1 \leq i \leq n$ one of the arguments v_i^p and v_i^n is labeled IN by $\Lambda(F_\leq)$ ($\Lambda(F_\leq)(v_i^p) = IN$ if and only if $\Lambda(F_\leq)(x_i^p) = OUT$) and hence the argument v_i is labeled OUT by $\Lambda(F_\leq)$. Furthermore, because the assignment β satisfies the formula Φ we know that every argument C_j is attacked by some argument $x \in \{ x_i^p, x_i^n \mid 1 \leq i \leq n \}$ such that $\Lambda(F_\leq)(x) = IN$ and hence $\Lambda(F_\leq)(C_j) = OUT$ for every $1 \leq j \leq m$. But then the argument Φ is labeled IN by $\Lambda(F_\leq)$ and $\Phi \in$ GE($F_\leq$).

To see the reverse direction suppose that the argument Φ is subjectively accepted in F and let $\leq$ be a specific audience of F witnessing this. Again using Proposition 1 we know that an argument is contained in the unique preferred extension of $F_\leq$ if and only if the argument is labeled IN by $\Lambda(F_\leq)$. We will first show that for every $1 \leq i \leq n$ there is at most one argument $x \in \{x_i^n, x_i^p\}$ such that $\eta(x) > c$ and $\Lambda(F_\leq)(x) = IN$. Suppose this is not the case, i.e., there is some $1 \leq i \leq n$ such that $\eta(x_i^n), \eta(x_i^p) > c$ and $\Lambda(F_\leq)(x_i^n) = \Lambda(F_\leq)(x_i^p) = IN$. It follows that both arguments v_i^n and v_i^p are labeled OUT by $\Lambda(F_\leq)$ and hence $\Lambda(F_\leq)(v_i) = IN$. But then $\Lambda(F_\leq)(\Phi) = OUT$ contradicting the assumption that the argument Φ is contained in the unique preferred extension of $F_\leq$. We will now show that the assignment β with $\beta(x_i) =$ true if $\eta(x_i^p) > c$ and $\Lambda(F_\leq)(x_i^p) = IN$ and $\beta(x_i) =$ false otherwise is a satisfying assignment for the formula Φ. Again suppose not. Then there is a clause C_j that is not satisfied by the assignment β. Let x_i be a variable contained in C_j. If x_i is contained positively in C_j then $\beta(x_i) =$ false and consequently either $\eta(x_i^p) < c$ or $\Lambda(F_\leq)(x_i^p) = OUT$. Similarly, if x_i is contained negatively in C_j then $\beta(x_i) =$ true and consequently either $\eta(x_i^n) < c$ or $\Lambda(F_\leq)(x_i^n) = OUT$. It follows that either $\eta(x) < c$ or $\Lambda(F_\leq)(x) = OUT$ for every argument that could attack the argument C_j in $F_\leq$ and hence $\Lambda(F_\leq)(C_j) = IN$. Hence, $\Lambda(F_\leq)(\Phi) = OUT$ contradicting the assumption that the argument Φ is contained in the unique preferred extension of $F_\leq$.

To prove part (B) of the theorem, we modify F into F' by adding the argument Φ' and the attack (Φ, Φ') to F. The value of Φ' in F' is set to c. Since Φ is the only attacker of Φ', the target argument Φ' is objectively accepted in F' if and only if Φ is not subjectively accepted in F'. Because of the construction of F' we have that the argument Φ is subjectively accepted in F' if and only if it is subjectively accepted in F. Hence (using part (A) of the theorem) it follows that the argument Φ' is objectively accepted in F' if and only if the formula Φ is not satisfiable. This completes the proof. □

4. An Algorithm for Bounded Treewidth

In this section we present the dynamic programming algorithm and establish our tractability results. Let $F = (X, A, V, \eta)$ be a VAF, $x_1 \in X$, and (T, χ, r) a nice tree decomposition of width at most w of G_F^{val}. For $t \in V(T)$, let X_t denote the set $\eta^{-1}(\chi(t))$. Allowing a slight abuse of notations, we denote the set $\delta_F^-(X_t) := \{ x \in X_t \mid (y, x) \in A \text{ and } y \in N_t \}$ by $\delta^-(t)$ and call it the *incoming border* of t. Likewise we denote the set $\delta_F^+(X_t)$ by $\delta^+(t)$, which is called the *outgoing border* of t. The *border* $\delta(t)$ of t is the union $\delta(t) = \delta^+(t) \cup \delta^-(t)$. *incoming border* of t, denoted $\delta^+(t)$, to be $\delta^+(t) = \{ x \in \eta^{-1}(\chi(t)) \mid (y, x) \in A \text{ and } y \in N_t \}$. The *outgoing border* $\delta^-(t)$ of t is the set $\delta^-(t) = \{ x \in \eta^{-1}(\chi(t)) \mid (x, y) \in A \text{ and } y \in N_t \}$. The *border* $\delta(t)$ of t is the union $\delta(t) = \delta^+(t) \cup \delta^-(t)$. Our main tractability result states that for each fixed value $\Delta := \max_{t \in V(T)} |\delta(t)|$ and fixed treewidth w, both ACCEPTANCE problems can be solved in uniformly polynomial time.

Theorem 2. *Given a VAF $F = (X, A, V, \eta)$, an argument $x_1 \in X$, and a nice tree decomposition (T, χ, r) of width at most w for G_F^{val} such that $\chi(r) = \{\eta(x_1)\}$. Then the problems* SUBJECTIVE

ACCEPTANCE *and* OBJECTIVE ACCEPTANCE *can be solved in time* $O(2^{\Delta} \cdot (2^{\Delta}(w+1)!)^2 \cdot (vw(F) + aw(F)) \cdot |V|)$.

We establish the above theorem with a dynamic programming algorithm over the given nice tree decomposition. The algorithm computes a set of so-called records (compact representations of possibly preferred extensions) for each node of the tree decomposition in a bottom-up manner. A *record* of a tree node $t \in V(T)$ is a triple $R = (I, O, \leq)$ such that:

1. $I \subseteq \delta^-(t)$;
2. $O \subseteq \delta^+(t)$;
3. $\leq$ is a specific audience for $F[\chi(t)]$.

Let $t \in V(T)$, $R = (I, O, \leq)$ a record of t and $\leq_t$ a specific audience for $F[\chi^*(t)]$ that is compatible with $\leq$. We define the *propagation* of R as $\Lambda(R) = \Lambda(\sigma(F_{\leq}[\chi(t)], O))$ and the propagation of R with respect to $\leq_t$ as $\Lambda_{\leq_t}(R) = \Lambda(\sigma(F_{\leq_t}[N_t], \alpha_{\leq_t}(I, N_t)))$. We say that $R = (I, O, \leq)$ *represents* a specific audience $\leq_t$ for $F[\chi^*(t)]$ if $\leq_t$ is compatible with $\leq$ and $O = \alpha_{\leq_t}(IN(\Lambda_{\leq_t}(R)), \chi(t))$. We say that $R = (I, O, \leq)$ is *valid* if there is an audience $\leq_t$ for $F[\chi^*(t)]$ that is represented by R. With each tree node $t \in V(T)$ we associate the set $\mathcal{R}(t)$ of all valid records for t. Informally, if R is a valid record that represents a specific audience $\leq_t$ then the set O contains exactly those arguments that are attacked by an argument that is labeled IN by $\Lambda_{\leq_t}(R)$ and the set I contains exactly those arguments that are labeled IN by $\Lambda(R)$ and attack arguments in N_t.

Lemma 2. *Let $t \in V(T)$. Then for every $I \subseteq \delta^-(t)$, and every specific audience $\leq_t$ for $F[\chi^*(t)]$ there is a record $R = (I, O, \leq) \in \mathcal{R}(t)$ that represents $\leq_t$.*

Proof. The Lemma follows immediately from the definition of $\mathcal{R}(t)$. □

The following four lemmas show that we can compute the set $\mathcal{R}(t)$ of all valid records for a tree node t from the sets $\mathcal{R}(t')$ of all valid records for every child t' of t in T.

Lemma 3 (leaf node)**.** *Let $t \in V(T)$ be a leaf node of (T, χ). Then $R = (I, O, \leq) \in \mathcal{R}(t)$ if and only if $I = \emptyset$, $O = \emptyset$ and $\leq$ is a specific audience of $F[\chi(t)]$. Furthermore, the set $\mathcal{R}(t)$ can be computed in time $O((w+1)!)$.*

Proof. The Lemma follows because $\delta(t) = \emptyset$ for every leaf node $t \in V(T)$. □

Lemma 4 (join node)**.** *Let $t \in V(T)$ be a join node of (T, χ) and t_1, t_2 be the children of t in T. Then $R = (I, O, \leq) \in \mathcal{R}(t)$ if and only if there are records $R_1 = (I_1, O_1, \leq_1) \in \mathcal{R}(t_1)$ and $R_2 = (I_2, O_2, \leq_2) \in \mathcal{R}(t_2)$ such that the following conditions hold:*

J1 $I = I_1 \cup I_2$;
J2 $O = O_1 \cup O_2$;
J3 $\leq = \leq_1 = \leq_2$;
J4 $I_1 \cap \delta^-(t_1) \cap \delta^-(t_2) = I_2 \cap \delta^-(t_1) \cap \delta^-(t_2)$

Furthermore, the set $\mathcal{R}(t)$ can be computed from the sets $\mathcal{R}(t_1)$ and $\mathcal{R}(t_2)$ in time $O(|\mathcal{R}(t_1)| \cdot |\mathcal{R}(t_2)| \cdot |\delta(t)|$.

Proof. We start with the following claim.

Claim 1. *Let $\leq_t$ be a specific audience for $F[\chi^*(t)]$, $\leq_{t_1}$ a specific audience for $F[\chi^*(t_1)]$, and $\leq_{t_2}$ a specific audience for $F[\chi^*(t_2)]$ such that $\leq_t$ is compatible with $\leq_{t_1}$ and $\leq_{t_2}$. Furthermore, let $R = (I, O, \leq)$ be a record in $\mathcal{R}(t)$ that represents $\leq_t$, $R_1 = (I_1, O_1, \leq_1)$ be a record in $\mathcal{R}(t_1)$ that represents $\leq_{t_1}$, and $R_2 = (I_2, O_2, \leq_2)$ be a record in $\mathcal{R}(t_2)$ that represents $\leq_{t_2}$. If R, R_1, and R_2 satisfy conditions J1, J3, and J4, then they also satisfy condition J2.*

We first prove the equation $IN(\Lambda_{\leq_t}(R)) \cap \eta^{-1}(N_{t_i}) = IN(\Lambda_{\leq_{t_i}}(R_i)) \cap \eta^{-1}(N_{t_i})$ for $i = 1, 2$ and w.l.o.g. let $i = 1$. Note that the condition J4 implies $I \cap \delta^-(t_1) = I_1 \cap \delta^-(t_1)$. Due to Proposition 2, it follows that the two propagations $\Lambda_{\leq_t}(R)$ and $\Lambda_{\leq_{t_1}}(R_1)$ are identical on $\eta^{-1}(N_{t_i})$. The equation is an immediate consequence.

Observe that for a valid record $R = (I, O, \leq)$, $x \in O$ if and only there exists $y \in IN(\Lambda_{\leq_t}(R)) \cap \eta^{-1}(N_t)$. Let $x \in O$ and let $y \in IN(\Lambda_{\leq_t}(R)) \cap \eta^{-1}(N_t)$. Due to the above equation, we have $y \in IN(\Lambda_{\leq_t}(R)) \cap \eta^{-1}(N_{t_1} \cup N_{t_2}) = \bigcup_{i=1}^{2}[IN(\Lambda_{\leq_{t_i}}(R_i)) \cap \eta^{-1}(N_{t_i})]$. The validity of the record $R_i = (I_i, O_i, \leq_i)$ implies $x \in O_1 \cup O_2$. For the opposite direction, let $x \in O_1$ w.l.o.g and let $y \in IN(\Lambda_{\leq_{t_1}}(R_1)) \cap \eta^{-1}(N_{t_1})$. The above equations indicates $IN(\Lambda_{\leq_{t_1}}(R_1)) \cap \eta^{-1}(N_{t_1}) \subseteq IN(\Lambda_{\leq_t}(R)) \cap \eta^{-1}(N_t)$ and it follows that $x \in O$. This completes the proof of the claim.

We now proceed to show the lemma. So suppose that $R = (I, O, \leq) \in \mathcal{R}(t)$. It follows that there is a specific audience $\leq_t$ for $F[\chi^*(t)]$ that is represented by R. For every $1 \leq i \leq 2$, we set $I_i = I \cap \delta^-(t_i)$, $\leq_i = \leq$, and $\leq_{t_i}$ to be the unique specific audience for $F[\chi^*(t_i)]$ that is compatible with $\leq_t$. Because of Lemma 2, there are records $R_i = (I_i, O_i, \leq_i) \in \mathcal{R}(t_i)$ that represent $\leq_{t_i}$. Observe that R, R_1, and R_2 satisfy the conditions J1, J3, and J4. Using Claim 1 we obtain that R, R_1, and R_2 also satisfy the condition J2.

To see the reverse direction let $1 \leq i \leq 2$ and suppose that there are records $R_i = (I_i, O_i, \leq_i) \in \mathcal{R}(t_i)$ that satisfy the conditions J3 and J4. It follows that each R_i represents a specific audiences $\leq_{t_i}$ for $F[\chi^*(t_i)]$. Let $I = I_1 \cup I_2$, $\leq = \leq_1$ and $\leq_t$ be the unique specific audience for $F[\chi^*(t)]$ that is compatible with $\leq_{t_1}$ and $\leq_{t_2}$. Because of Lemma 2 it holds that there is a record $R = (I, O, \leq) \in \mathcal{R}(t)$ that is represented by $\leq_t$. Observe that the records R, R_1 and R_2 already satisfy the conditions J1, J3, and J4. It now follows from Claim 1 that they also satisfy condition J2.

It remains to show that $\mathcal{R}(t)$ can be computed from $\mathcal{R}(t_1)$ and $\mathcal{R}(t_2)$ in time $O(|\mathcal{R}(t_1)| \cdot |\mathcal{R}(t_2)| \cdot |\delta(t)|)$. Using the first statement of the lemma it follows that $\mathcal{R}(t)$ can be computed from $\mathcal{R}(t_1)$ and $\mathcal{R}(t_2)$ by applying the following steps to every pair $(R_1 = (I_1, O_1, \leq_1) \in \mathcal{R}(t_1), R_2 = (I_2, O_2, \leq_2) \in \mathcal{R}(t_2))$ of records:

1. Check whether R_1 and R_2 satisfy condition J4 and $\leq_1$ is compatible with $\leq_2$. If not then reject the pair (R_1, R_2), if they do, proceed to step 2.
2. Compute the record R from R_1 and R_2 according to the conditions J1–J3 and store R in $\mathcal{R}(t)$.

Because the steps 1. and 2. can be carried out in time $O(|\delta(t)|)$ it follows that the overall running time to compute $\mathcal{R}(t)$ from $\mathcal{R}(t_1)$ and $\mathcal{R}(t_2)$ is $O(|\mathcal{R}(t_1)| \cdot |\mathcal{R}(t_2)| \cdot |\delta(t)|)$. □

Lemma 5 (introduce node)**.** *Let $t \in V(T)$ be an introduce node of (T, χ) with child t', such that $\chi(t) = \chi(t') \cup \{v_0\}$. Then $R = (I, O, \leq) \in \mathcal{R}(t)$ if and only if there is a record $R' = (I, O, \leq') \in \mathcal{R}(t')$ such that $\leq'$ is compatible with $\leq$. Furthermore, the set $\mathcal{R}(t)$ can be computed from the set $\mathcal{R}(t')$ in time $O(|\mathcal{R}(t')| \cdot (w + 1))$.*

Proof. So suppose that $R = (I, O, \leq) \in \mathcal{R}(t)$. It follows that R represents a specific audience $\leq_t$ for $F[\chi^*(t)]$. Let $\leq'$ be the unique specific audience for $F[\chi(t')]$ that is compatible with $\leq_t$. Because of Lemma 2 it follows that there is a record $R' = (I, O', \leq') \in \mathcal{R}(t')$ that represents $\leq_t$. Using Proposition 3 we obtain $O = O'$.

To see the reverse direction suppose that there is a record $R' = (I', O', \leq') \in \mathcal{R}(t')$. It follows that R' represents a specific audience $\leq_{t'}$ for $F[\chi^*(t')]$. Let $\leq$ be any specific audience for $F[\chi(t)]$ that is compatible with $\leq_{t'}$ and let $\leq_t$ be any specific audience for $F[\chi^*(t)]$ that is compatible with both $\leq_{t'}$ and $\leq$. Because of Lemma 2 it follows that there is a record $R = (I', O, \leq) \in \mathcal{R}(t')$ that represents $\leq_t$. Using Proposition 3 we obtain $O = O'$.

It remains to show that $\mathcal{R}(t)$ can be computed from $\mathcal{R}(t')$ in time $O(|\mathcal{R}(t_1)| \cdot (w + 1))$. Using the first statement of the lemma it follows that we can compute $\mathcal{R}(t)$ from $\mathcal{R}(t')$ by computing one record $R = (I', O', \leq)$ for every record $R' = (I', O', \leq') \in \mathcal{R}(t')$ and every specific audience $\leq$ for $F[\chi(t)]$ that is compatible with $\leq'$. Observe that there are at most $w + 1$ specific audiences $\leq$ for $F[\chi(t)]$ that are compatible with a specific audience $\leq'$ for $F[\chi(t')]$ and the record R can be

computed from R' and $\leq$ in time $O(1)$. It follows that the overall running time to compute $\mathcal{R}(t)$ from $\mathcal{R}(t')$ is $O(|\mathcal{R}(t')| \cdot (w+1))$. $\square$

Lemma 6 (forget node)**.** *Let t be a forget node of (T, χ) with child t' such that $\chi(t) = \chi(t') \setminus \{v_0\}$. Then $R = (I, O, \leq) \in \mathcal{R}(t)$ if and only if there is a set $S \subseteq \delta^-(t) \setminus \delta^-(t')$ and a record $R' = (I', O', \leq') \in \mathcal{R}(t')$ such that:*

F1 $I = (I' \setminus P) \cup S$;
F2 $P = IN(\Lambda(F_{v_0})) \cap \delta^-(t')$;
F3 $O = (\delta^+(t) \cap O') \cup \alpha_{\leq'}(IN(\Lambda(F_{v_0})), \chi(t))$;
F4 $\leq$ *is compatible with* $\leq'$;

where $P = I' \cap \eta^{-1}(v_0)$ and $F_{v_0} = \sigma(F[v_0], (\eta^{-1}(v_0) \cap O') \cup \alpha_{\leq'}(I, v_0))$. Furthermore, the set $\mathcal{R}(t)$ can be computed from the set $\mathcal{R}(t')$ in time $O(2^{|\delta^-(t) \setminus \delta^-(t')|} \cdot |\mathcal{R}(t')| \cdot (|\eta^{-1}(v_0)| + |A(F[v_0])| + \Delta))$.

Proof. So suppose that $R = (I, O, \leq) \in \mathcal{R}(t)$. It follows that R represents a specific audience $\leq_t$ for $F[\chi^*(t)] = F[\chi^*(t')]$. Let $S = I \cap (\delta^-(t) \setminus \delta^-(t'))$, $P = IN(\Lambda_{\leq_t}(R)) \cap \eta^{-1}(v_0) \cap \delta^-(t')$, $I' = I \setminus S \cup P$, and $\leq'$ be a specific audience for $F[\chi(t')]$ that is compatible with $\leq_t$. It follows from Lemma 2 that there is a record $R' = (I', O', \leq') \in \mathcal{R}(t')$ that represents $\leq_t$. Consequently, the records R, R', and S satisfy F1 and F4. Because $\Lambda(F_{v_0})$ is compatible with $\Lambda_{\leq_t}(R)$ we obtain that R, R' and S also satisfy F2 and F3.

To see the reverse direction suppose that there is a record $R' = (I', O', \leq') \in \mathcal{R}(t')$, a set $S \subseteq (\delta^-(t) \setminus \delta^-(t'))$ and a record $R = (I, O, \leq)$ of t that satisfy F1–F4. It follows that there is a specific audience $\leq_{t'}$ for $F[\chi^*(t')] = F[\chi^*(t)]$. We show that R represents $\leq_{t'}$ and hence $R \in \mathcal{R}(t)$. Let $F_L = \sigma(\sigma(F_{\leq_{t'}}[N_t], \alpha_{\leq_{t'}}(I, N_{t'}), s_1), \alpha_{\leq_{t'}}(I, v_0), s_2)$,, $X_1 = N_{t'} \cup \{s_1\}$, $X_2 = \eta^{-1}(v_0) \cup \{s_2\}$, $F_1 = \sigma(F_L[X_1], \alpha_{F_L}(I_L, X_1), s_1)$, $F_2 = \sigma(F_L[X_2], \alpha_{F_L}(O_L, X_2), s_2)$, $O_L = O' \cap \eta^{-1}(v_0)$, $I_L = P$. Observe that by construction $\Lambda(F_L)$ is compatible with $\Lambda_{\leq_{t'}}(R)$, $\Lambda(F_1)$ is compatible with $\Lambda_{\leq_{t'}}(R')$, and $\Lambda(F_2)$ is compatible with $\Lambda(F_{v_0})$. Hence, $\alpha_{F_L}(IN(\Lambda(F_1)), X_2) = O' \cap \eta^{-1}(v_0) = O_L$ and because of F2 also $IN(\Lambda(F_2)) \cap \delta^-_{F_L}(X_2) = P = I_L$. It now follows from Lemma 1 that $\Lambda(F_L) = \Lambda_{\leq_{t'}}(R)$ is compatible with $\Lambda(F_1) \cup \Lambda(F_2) = \Lambda_{\leq_{t'}}(R') \cup \Lambda(F_{v_0})$. Consequently,

$$
\begin{aligned}
&\alpha_{\leq_{t'}}(IN(\Lambda_{\leq_{t'}}(R)), \chi(t)) \\
=&\alpha_{\leq_{t'}}(IN(\Lambda_{\leq_{t'}}(R') \cup \Lambda(F_{v_0})), \chi(t)) \\
=&\alpha_{\leq_{t'}}(IN(\Lambda_{\leq_{t'}}(R'), \chi(t)) \cup \alpha_{\leq_{t'}}(IN(\Lambda(F_{v_0}))), \chi(t)) \\
=&(\delta^+(t) \cap O') \cup \alpha_{\leq_{t'}}(IN(\Lambda(F_{v_0})), \chi(t)) \\
=&(\delta^+(t) \cap O') \cup \alpha_{\leq'}(IN(\Lambda(F_{v_0})), \chi(t)) \\
=&O
\end{aligned}
$$

It follows that R represents $\leq_{t'}$.

It remains to show that $\mathcal{R}(t)$ can be computed from $\mathcal{R}(t')$ in time $O(2^{|\delta^-(t) \setminus \delta^-(t')|} \cdot |\mathcal{R}(t')| \cdot (|\eta^{-1}(v_0)| + |A(F[v_0])| + \Delta))$ Because of the first statement of this lemma it follows that we can compute $\mathcal{R}(t)$ from $\mathcal{R}(t')$ using the following procedure:

1. For every $S \subseteq (\delta^-(t) \setminus \delta^-(t'))$ and
   $R' = (I', O', \leq') \in \mathcal{R}(t')$ do
2. $\quad P = I' \cap \eta^{-1}(v_0)$
3. $\quad P' = IN(\Lambda(F_{v_0}))$
4. $\quad$ if $P == P' \cap \delta^-(t')$ then
5. $\quad\quad I = (I' \setminus P) \cup S$
6. $\quad\quad O = (\delta^+(t) \cap O') \cup \alpha_{\leq'}(P', \chi(t))$
7. $\quad\quad \leq$ is the unique specific audience for $F[\chi(t)]$
   $\quad\quad$ that is compatible with $\leq'$
8. $\quad\quad$ return $R = (I, O, \leq)$

Because there are at most $2^{|\delta^-(t)\setminus\delta^-(t')|}$ sets $S \subseteq (\delta^-(t) \setminus \delta^-(t'))$ and exactly $|\mathcal{R}(t')|$ records in $\mathcal{R}(t')$ we obtain that everything inside the initial for-loop is executed at most $2^{|\delta^-(t)\setminus\delta^-(t')|} \cdot |\mathcal{R}(t')|$ times. We obtain the following worst-case running time for the lines 2.–9.:

2. $O(|\eta^{-1}(v_0)|)$;
3. Because $|X(F_{v_0})| \leq |\eta^{-1}(v_0) + 1|$ and $|A(F_{v_0})| \leq |A(F[v_0])| + |\eta^{-1}(v_0)|$ it follows from Proposition 1 that P' can be computed in time $O(|\eta^{-1}(v_0)| + (|A(F[v_0])| + |\eta^{-1}(v_0)|)) \subseteq O(|\eta^{-1}(v_0)| + |A(F[v_0])|)$;
4. Because $|P| \leq |\eta^{-1}(v_0)|$ we can check whether P is equal to $P' \cap \delta^+(t')$ in time $O(|\eta^{-1}(v_0)|)$;
5. $O(|\delta^-(t)|) \subseteq O(\Delta)$;
6. $O(|\delta^+(t')| + |\delta^+(t)|) \subseteq O(\Delta)$;
7. $O(1)$;
8. $O(1)$.

It follows that the overall running time to compute $\mathcal{R}(t)$ from $\mathcal{R}(t')$ is $O(2^{|\delta^-(t)\setminus\delta^-(t')|} \cdot |\mathcal{R}(t')| \cdot (|\eta^{-1}(v_0)| + |A(F[v_0])| + \Delta))$. □

We say that a record $R = (I, O, \leq) \in \mathcal{R}(r)$ is *realizable* if $I = (IN(\Lambda(R)) \cap \delta^-(r))$. We say that a record $R = (I, O, \leq) \in \mathcal{R}(r)$ is *compatible* with a specific audience $\leq_F$ of F if $\Lambda(R)$ is compatible with $\Lambda(F_{\leq_F})$. The following lemma shows that the set of all realizable records in $\mathcal{R}(t)$ is a complete representation of all specific audiences of F.

Lemma 7. *Every specific audience $\leq_F$ of F is compatible with some realizable record $R \in \mathcal{R}(r)$. Furthermore, every realizable record $R \in \mathcal{R}(r)$ is compatible with some specific audience $\leq_F$ of F.*

Proof. Suppose there is a specific audience $\leq_F$ for F. Let $I = IN(\Lambda(F_{\leq_F})) \cap \delta^-(t)$ and $\leq$ a specific audience for $F[\chi(r)]$ that is compatible with $\leq_F$. It follows from Lemma 2 that there is a record $R = (I, O, \leq) \in \mathcal{R}(r)$ that is represented by $\leq_F$ and $O = \alpha_{\leq_F}(IN(\Lambda_{\leq_F}(R)), \chi(r))$. We first show that R is compatible with $\leq_F$ with the help of Lemma 1. Let $F_L = F_{\leq_F}$, $X_1 = \eta^{-1}(N_r)$, $X_2 = \eta^{-1}(\chi(r))$, $I_L = I$, and $O_L = O$ and let F_1 and F_2 be the AFs as defined in the statement of Lemma 1. Because R represents $\leq_F$ we have $O_L = \alpha_{\leq_F}(IN(\Lambda(F_{\leq_F}[N_r])), \chi(r)) = \alpha_{F_L}(IN(\Lambda(F_1)), X_2)$ and we also know that $I_L = IN(\Lambda(F_{\leq_F})) \cap \delta^-(t) = IN(\Lambda(F_L)) \cap \delta^-_{F_L}(X_2)$. Consequently, all pre-requisites of Lemma 1 are fulfilled and $\Lambda(F_L) = \Lambda(F_{\leq_F})$ is compatible with $\Lambda(F_2) = \Lambda(R)$. But then R is also realizable because $I = IN(\Lambda(F_{\leq_F})) \cap \delta^-(r) = IN(\Lambda(R)) \cap \delta^-(r)$ as required.

To see the reverse direction suppose that $R = (I, O, \leq) \in \mathcal{R}(r)$ is realizable. Because R is valid it follows that there is a specific audience $\leq_F$ for F that is represented by R. We show that R is compatible with $\leq_F$ with the help of Lemma 1. Let $F_L = F_{\leq_F}$, $X_1 = \eta^{-1}(N_r)$, $X_2 = \eta^{-1}(\chi(r))$, $I_L = I$, and $O_L = O$ and let F_1 and F_2 be the AFs as defined in the statement of Lemma 1. Because R represents $\leq_F$ we have $O_L = \alpha_{\leq_F}(IN(\Lambda(F_{\leq_F}[N_r])), \chi(r)) = \alpha_{F_L}(IN(\Lambda(F_1)), X_2)$ and because R is realizable it follows that $I_L = IN(\Lambda(R)) \cap \delta^-(t) = IN(\Lambda(F_2)) \cap \delta^-_{F_L}(X_2)$. Consequently, all pre-requisites of Lemma 1 are fulfilled and $\Lambda(F_L) = \Lambda(F_{\leq_F})$ is compatible with $\Lambda(F_2) = \Lambda(R)$. □

Lemma 8. *Let $R \in \mathcal{R}(r)$. Then we can decide whether R is realizable and whether $x_1 \in \Lambda(R)$ in time $O(|X(F[\eta(x_1)]| + |A(F[\eta(x_1)])| + \Delta)$.*

Proof. Let $R = (I, O, \leq) \in \mathcal{R}(r)$. To decide whether R is realizable we need to compute $\Lambda(R) = \Lambda(\sigma(F_{\leq}[\chi(r)], O) = \Lambda(\sigma(F[\eta(x_1)], O))$. Because of Proposition 1 this can be done in time $O(|\eta^{-1}(\eta(x_1))| + |A[\eta(x_1)]| + \Delta)$. Once we have computed $\Lambda(R)$ we can check whether R is realizable and whether $x_1 \in IN(\Lambda(R))$ in time $O(\Delta)$. Hence, both problems can be decided in time $O(|\eta^{-1}(\eta(x_1))| + |A[\eta(x_1)]| + \Delta)$. □

Proof of Theorem 2. Let $F = (X, A, V, \eta)$ be the given VAF, $x_1 \in X$ the query argument, and (T, χ, r) the nice tree decomposition of width at most w for G_F^{val} such that $|T| = O(|V|)$ and $\chi(r) = \{\eta(x_1)\}$. Starting from the leaf nodes of the nice tree decomposition the algorithm computes the set of all valid records for each tree node $t \in V(T)$ in a bottom-up manner. Denote by $\mathcal{R}_{\max} = \max_{t \in V(T)} |\mathcal{R}(t)|$. Combining the running times for a leaf node of $O((w+1)!)$ (Lemma 3), for a join node of $O(|\mathcal{R}(t_1)| \cdot |\mathcal{R}(t_2)| \cdot |\delta(t)|) \subseteq O((\mathcal{R}_{\max})^2 \cdot \Delta)$ (Lemma 4), for an introduce node of $O(|\mathcal{R}(t')| \cdot (w+1)) \subseteq O(\mathcal{R}_{\max} \cdot (w+1))$ (Lemma 5), and for a forget node of $O(2^{|\delta^-(t) \setminus \delta^-(t')|} \cdot |\mathcal{R}(t')| \cdot (|\eta^{-1}(v_0)| + |A(F[v_0])| + \Delta)) \subseteq O(2^{\Delta} \cdot \mathcal{R}_{\max} \cdot (vw(F) + aw(F) + \Delta))$ (Lemma 6) we obtain $O(2^{\Delta} \cdot (\mathcal{R}_{\max})^2 \cdot (w+1)! \cdot (vw(F) + aw(F) + \Delta))$ as the worst-case running time of the algorithm for a tree node $t \in V(T)$. Because there at most $O(|V|)$ tree nodes in T we obtain $O(2^{\Delta} \cdot (\mathcal{R}_{\max})^2 \cdot (w+1)! \cdot (vw(F) + aw(F) + \Delta) \cdot |V|)$ as the running time that our algorithm needs to compute the sets of all valid records for every tree node $t \in V(T)$.

It follows from Lemma 7 that the argument x_1 is subjectively accepted in F if and only if there is a realizable record $R \in \mathcal{R}(r)$ such that $x_1 \in IN(\Lambda(R))$. For the same reason, the argument x_1 is objectively accepted in F if and only if for all realizable records $R \in \mathcal{R}(r)$ it holds that $x_1 \in IN(\Lambda(R))$. Because there are at most 2^{Δ} records in $\mathcal{R}(r)$ and for each such record R we can check whether R is realizable and whether $x_1 \in IN(\Lambda(R))$ in time $O(|X(F[\eta(x_1)]| + |A(F[\eta(x_1)])| + \Delta) \subseteq O(vw(F) + aw(F) + \Delta)$ (see Lemma 8) it follows that given $\mathcal{R}(r)$, we can decide whether x_1 is subjectively or objectively accepted in F in time $O(2^{\Delta} \cdot (vw(F) + aw(F) + \Delta))$. Combining these two steps leads to an overall running time of the algorithm of $O(2^{\Delta} \cdot (\mathcal{R}_{\max})^2 \cdot (w+1)! \cdot (vw(F) + aw(F) + \Delta) \cdot |V|)$. It remains to observe that $|\mathcal{R}_{\max}| \leq 2^{\Delta} \cdot (w+1)!$ $\square$

References

[1] 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 Verlag, 2009.

[2] T. J. M. Bench-Capon. Persuasion in practical argument using value-based argumentation frameworks. *J. Logic Comput.*, 13(3):429–448, 2003.

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

[4] P. Besnard and A. Hunter. *Elements of Argumentation*. The MIT Press, 2008.

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

[6] A. Bondarenko, P. M. Dung, R. A. Kowalski, and F. Toni. An abstract, argumentation-theoretic approach to default reasoning. *Artificial Intelligence*, 93(1-2):63–101, 1997.

[7] P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, 77(2):321–357, 1995.

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

[9] P. E. Dunne. Tractability in value-based argumentation. In P. Baroni, F. Cerutti, M. Giacomin, and G. R. Simari, editors, *Computational Models of Argumentation, Proceedings of COMMA 2010*, volume 216 of *Frontiers in Artificial Intelligence and Applications*, pages 195–206. IOS, 2010.

[10] P. E. Dunne and T. J. M. Bench-Capon. Complexity in value-based argument systems. In J. J. Alferes and J. A. Leite, editors, *Logics in Artificial Intelligence, 9th European Conference, JELIA 2004, Lisbon, Portugal, September 27-30, 2004, Proceedings*, volume 3229 of *Lecture Notes in Computer Science*, pages 360–371. Springer Verlag, 2004.

[11] V. Gogate and R. Dechter. A complete anytime algorithm for treewidth. In *Proceedings of the Proceedings of the Twentieth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-04)*, pages 201–208, Arlington, Virginia, 2004. AUAI Press.

[12] E. J. Kim, S. Ordyniak, and S. Szeider. Algorithms and complexity results for persuasive argumentation. *Artificial Intelligence*, 175:1722–1736, 2011.

[13] T. Kloks. *Treewidth: Computations and Approximations*. Springer Verlag, Berlin, 1994.

[14] S. Parsons, M. Wooldridge, and L. Amgoud. Properties and complexity of some formal inter-agent dialogues. *J. Logic Comput.*, 13(3):347–376, 2003.

[15] J. L. Pollock. How to reason defeasibly. *Artificial Intelligence*, 57(1):1–42, 1992.

[16] I. Rahwan and G. R. Simari, editors. *Argumentation in Artificial Intelligence*. Springer Verlag, 2009.