Noname manuscript No.
(will be inserted by the editor)

Argumentation frameworks as Constraint Satisfaction Problems

Leila Amgoud · Caroline Devred

the date of receipt and acceptance should be inserted later

Abstract *Argumentation* is a promising approach for defeasible reasoning. It consists of justifying each plausible conclusion by *arguments*. Since the available information may be inconsistent, a conclusion and its negation may both be justified. The arguments are thus said to be conflicting. The main issue is how to evaluate the arguments. Several *semantics* were proposed for that purpose. The most important ones are: stable, preferred, complete, grounded and admissible. A semantics is a set of criteria that should be satisfied by a set of arguments, called *extension*, in order to be acceptable. Different decision problems related to these semantics were defined (like whether an argumentation framework has a stable extension). It was also shown that most of these problems are computationally costly.
This paper studies how to encode the problem of computing the extensions of an argumentation framework (under each of the previous semantics) as a constraint satisfaction problem (CSP). Such encoding is of great importance since it makes it possible to use the very efficient solvers (developed by the CSP community) for computing the extensions. We focus on three families of argumentation frameworks: Dung's abstract framework, its constrained version and preference-based argumentation framework.

Keywords Default Reasoning, Argumentation, CSP

1 Introduction

Argumentation is a reasoning model based on the construction and evaluation of interacting arguments. An argument is a reason for believing in a statement, doing an action, pursuing a goal, etc. Argumentation theory is gaining an increasing interest in Artificial Intelligence, namely for reasoning about defeasible/uncertain information, making decisions under uncertainty, learning concepts, and modeling agents' interactions (see [1] for a survey of the various works done on argumentation in AI and [29,

Leila Amgoud
IRIT - CNRS
118, route de Narbonne, 31062 Toulouse Cedex 9, France
E-mail: amgoud@irit.fr

5, 21] for some works on complexity issues).

The most abstract argumentation framework was proposed in the seminal paper [19] by Dung. It consists of a set of (equally important) *arguments*, a *binary relation* representing *attacks* among arguments, and *semantics* for evaluating the arguments. A semantics describes when a set of arguments, called *extension*, is acceptable without bothering neither on the origin nor on the structure of the arguments/attacks. The most important semantics are grounded (which ensures only one extension), admissible, complete, stable and preferred semantics which may lead to more than one extension. It is worth mentioning that generally an argumentation framework is represented by a graph whose nodes are the arguments of the framework and the edges are the attacks among arguments.

Dung's framework was extended in different ways in the literature. In [2,4], arguments are assumed to not have the same strength. Thus, in addition to the attack relation, another binary relation on the set of arguments is available. It represents preferences between arguments. In [13], arguments are assumed to have the same strength, but a constraint on arguments may be available. This constraint represents a desirable property that should be satisfied by the extensions, i.e., by the acceptable sets of arguments. It is worth mentioning that in both works, Dung's semantics are used to evaluate arguments, thus to compute the extensions.

Different decision problems related to the implementation of the previous semantics were identified (like for instance testing whether a framework has a stable extension, or whether an argument belongs to one (or every) extension, or whether a given set of arguments is an extension under a given semantics). The computational complexity of each identified problem was studied (see for instance [13,17,18,23]). The results are disappointing since they show that the most important decision problems are very costly. For instance, checking whether a set of arguments is a preferred extension of an argumentation framework is CO-NPcomplete, and checking whether a framework has any stable extension is NPcomplete. Some algorithms that compute extensions under some semantics were developed, for instance in [11,16,24]. However, the efficiency of those algorithms was not proved. They are neither tested on benchmarks nor compared to other algorithms developed for the same purpose.

In [6], Besnard and Doutre defined Dung's semantics as satisfiability problems. Indeed, for each semantics, the argumentation framework is translated into a set of propositional formulas. The models of this set correspond exactly to the extensions of the framework under the chosen semantics. This work is of great importance since it makes it possible to use the efficient SAT solvers for computing the extensions of argumentation frameworks.

Besides, there is a huge literature on *Constraints Satisfaction Problems* (CSP) since many real-world problems can be described as CSPs (e.g. [12,33,15,26,28,30,32]. A CSP consists of a set of *variables*, a (generally finite) *domain* for each variable and a set of *constraints*. Each constraint is defined over a subset of variables and limits the combination of values that the variables in this subset can take. The goal is to find an assignment to the variables which satisfies all the constraints. In some problems, the goal is to find all such assignments. Solving a constraint satisfaction problem on a finite domain is an NP-complete problem in general. In order to be solved in a reasonable time, different *solvers* were developed. They use a form of search based on variants of backtracking, constraint propagation and local search [28].

CSP and SAT are considered as two independent threads of research. They have a lot in common as evidenced by similar ideas underlying the branch and prune algorithms that are most successful at solving both kinds of problems. They also exhibit differences in the way they are used to state and solve problems (see [8] for a survey of the similarities and differences between CSP and SAT). It is well known that CSP is more expressive for the modeling phase. In an argumentation context, CSP is more appropriate for encoding complex semantics like grounded and preferred. It is also possible for users to add constraints as in [13] into CSP while in SAT there is usually little room for this parametrization.

In this paper, we extend the work of Besnard and Doutre [6] in three ways: first we study all Dung's semantics including preferred and grounded semantics which were not considered in [6]. Second, we study the semantics in three families of frameworks: Dung's framework [19] as in [6] but also in constrained argumentation framework [13] and preference-based argumentation framework [2]. Finally, we encode the various semantics as CSPs. We show that each semantics is captured by one particular CSP. In each CSP, the arguments of the framework play the role of variables whose domains are all binary. For those semantics that are studied in [6] we take advantage of the possible translation from SAT to CSP in order to define their corresponding CSPs.

This paper is organized as follows: Section 2 recalls the basic concepts of a CSP. Section 3 recalls Dung's argumentation framework and shows how to compute its extensions (under the studied semantics) by CSPs. Section 4 presents the constrained version of Dung's framework and proposes different CSPs that compute the extensions under the same semantics of such a framework. Section 5 recalls the last family of argumentation frameworks (preference-based argumentation frameworks) as well as their encoding as CSPs. In Section 6, we compare our approach to existing works on the topic. The last section is devoted to concluding remarks and perspectives.

2 Constraint satisfaction problems (CSPs)

A *constraint satisfaction problem* (or CSP) is expressed as follows: a set of *variables*, finite sets of possible values that can be assigned to the variables (called *domains*), and a list of *constraints*. Each constraint involves some subset of the variables and specifies the allowable combinations of values for that subset. The idea is to find the possible combinations of values of the variables that satisfy every constraint. It is worth mentioning that many real-world problems such as scheduling fall into this framework (see [30] for some applications). Another example of CSP is the timetabling problem in which courses are given to different classes, in different rooms, and by different lecturers. The same lecturer cannot give two courses at the same time, and two courses cannot take place at the same time, etc.

Definition 1 (CSP) A *CSP* instance is a triple $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ where:

– $\mathcal{X} = \{x_1, \ldots, x_n\}$ is a set of variables,
– $\mathcal{D} = \{\mathcal{D}_1, \ldots, \mathcal{D}_n\}$ where $\mathcal{D}_i$ is a finite domain for the variable $x_i \in \mathcal{X}$, and
– $\mathcal{C} = \{c_1, \ldots, c_m\}$ is a set of constraints.

Each constraint c_i is a pair (h_i, H_i) where

- $h_i = (x_{i1}, \ldots, x_{ik})$ where $x_{ij} \in \mathcal{X}$
- $H_i \subseteq \mathcal{D}_{i1} \times \ldots \times \mathcal{D}_{ik}$ where $\mathcal{D}_{ij}$ is the domain of $x_{ij} \in h_i$.

Note that H_i is a k-ary relation over $\mathcal{D}$. It is a subset of all allowed combinations of values for the variables in h_i. A state of the problem is defined by an *assignment* of values to some or all of the variables.

Definition 2 (Assignment) An *assignment* v for a CSP instance $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ is a mapping that assigns to every variable $x_i \in \mathcal{X}$ an element $v(x_i) \in \mathcal{D}_i$. An assignment v satisfies a constraint $((x_{i1}, \ldots, x_{ik}), H_i) \in \mathcal{C}$ iff $(v(x_{i1}), \ldots, v(x_{ik})) \in H_i$.

Finally, a solution of a CSP is an assignment that satisfies its constraints.

Definition 3 (Solution) A *solution* of a CSP instance $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ is an assignment v that satisfies all the constraints in $\mathcal{C}$ and in which all the variables of $\mathcal{X}$ are assigned a value. We write $(v(x_1), \ldots, v(x_n))$ to denote the solution.

3 Abstract argumentation framework

This section recalls Dung's argumentation framework and presents the different corresponding CSPs which return its extensions under various semantics.

3.1 Dung's framework

In [19], Dung has developed the most abstract argumentation framework in the literature. It consists of a set of arguments and an attack relation between them.

Definition 4 (Argumentation framework) An *argumentation framework* (AF) is a pair $\mathcal{F} = (\mathcal{A}, \mathcal{R})$ where $\mathcal{A}$ is a set of arguments and $\mathcal{R}$ is an attack relation ($\mathcal{R} \subseteq \mathcal{A} \times \mathcal{A}$). The notations $a\mathcal{R}b$ or $(a, b) \in \mathcal{R}$ mean that the argument a *attacks* the argument b.

Different *acceptability semantics* for evaluating arguments were proposed in the same paper [19]. A semantics is a set of criteria that should be satisfied by a set of arguments, called *extension*, in order to be acceptable. Some semantics ensure only one extension to every argumentation framework whereas others may lead to several extensions. Dung's semantics have in common two basic properties: *conflict-freeness* and *defence*.

Definition 5 Let $\mathcal{F} = (\mathcal{A}, \mathcal{R})$ be an AF, $a \in \mathcal{A}$ and $\mathcal{B} \subseteq \mathcal{A}$.

- $\mathcal{B}$ is *conflict-free* iff $\nexists\, a, b \in \mathcal{B}$ s.t. $a\mathcal{R}b$.
- $\mathcal{B}$ *defends* an argument a iff for all $b \in \mathcal{A}$ s.t. $b\mathcal{R}a$, there exists $c \in \mathcal{B}$ s.t. $c\mathcal{R}b$.

The following definition recalls all the acceptability semantics proposed in [19].

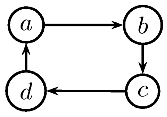
Definition 6 (Acceptability semantics) Let $\mathcal{F} = (\mathcal{A}, \mathcal{R})$ be an AF and $\mathcal{B} \subseteq \mathcal{A}$.

- $\mathcal{B}$ is an *admissible* set iff it is conflict-free and defends its elements.
- $\mathcal{B}$ is a *preferred* extension iff it is a maximal (for set $\subseteq$) admissible set.
- $\mathcal{B}$ is a *stable* extension iff it is conflict-free and attacks any argument in $\mathcal{A} \setminus \mathcal{B}$.

– $\mathcal{B}$ is a *complete* extension iff it is conflict-free and contains all arguments it defends.
– $\mathcal{B}$ is a *grounded* extension iff it is a minimal (for set $\subseteq$) complete extension.

It was shown in [19] that any argumentation framework has a unique grounded extension (which may be empty). It may have several admissible, complete, stable and preferred extensions. However, unlike other semantics, stable extensions do not always exist. Finally, it can be checked that any stable extension is a preferred one but the reverse is not always true.

Example 1 Let us consider the framework $\mathcal{F}_1 = (\mathcal{A}_1, \mathcal{R}_1)$ where $\mathcal{A}_1 = \{a, b, c, d\}$ and let $\mathcal{R}_1$ be as depicted in the figure below.



The framework $\mathcal{F}_1$ has two preferred extensions which are also stable: $\mathcal{B}_1 = \{a, c\}$ and $\mathcal{B}_2 = \{b, d\}$. The sets $\mathcal{B}_1$, $\mathcal{B}_2$ and $\mathcal{B}_3 = \{\}$ are the admissible and complete extensions of the framework, whereas $\mathcal{B}_3$ is its grounded extension.

3.2 Computing Dung's semantics by CSPs

In this section, we propose different mappings of Dung's argumentation framework into CSP instances. The idea is: starting from an argumentation framework, we define a CSP instance whose solutions are the extensions of the framework under a given acceptability semantics. In all the instances, arguments play the role of variables that is, a variable is associated to each argument. Each variable may take two values 0 or 1 meaning that the corresponding argument is rejected or accepted. Thus, the domains of the variables are all *binary*. Things are different with the constraints. We show that according to the semantics that is studied, the definition of a constraint changes.

Let us start with a CSP instance that computes the conflict-free sets of arguments. The idea here is that a set of arguments cannot contain two conflicting arguments. Thus, each attack $(a, b) \in \mathcal{R}$ gives birth to a constraint which says that the two variables a and b cannot take value 1 at the same time. This constraint has the following form: $((a, b), ((0, 0), (0, 1), (1, 0)))$. The combination $(0, 0)$ means that the two arguments do not belong to the set. It is also worth noticing that the combinations $(0, 0), (0, 1), (1, 0)$ are exactly the models of the propositional formula $a \Rightarrow \neg b$ (or $b \Rightarrow \neg a$). Both formulas $a \Rightarrow \neg b$ and $b \Rightarrow \neg a$ can be used since in conflict-freeness, the orientation of the attack relation is not important.

Notations: Throughout the paper, $\lceil x \rceil$ denotes the models of a propositional formula x, and $\mathtt{Atoms}(x)$ the set of atoms that are involved in x.

Definition 7 (Free CSP) Let $\mathcal{F} = (\mathcal{A}, \mathcal{R})$ be an argumentation framework. A *free CSP* associated with $\mathcal{F}$ is a tuple $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ where $\mathcal{X} = \mathcal{A}$, $\forall \mathcal{D}_i \in \mathcal{D}$, $\mathcal{D}_i = \{0, 1\}$ and $\mathcal{C} = \{(h, H) \mid h = (a, b)$ where $(a, b) \in \mathcal{R}$ and $H = \lceil a \Rightarrow \neg b \rceil\}$.

It can be checked that there are as many constraints as attacks in the argumentation framework.

Property 1 Let $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ be the free CSP instance associated with the AF $\mathcal{F} = (\mathcal{A}, \mathcal{R})$. It holds that $|\mathcal{C}| = |\mathcal{R}|$.

Proof This follows immediately from Definition 7.

The following result shows that the solutions of this CSP are the conflict-free sets of arguments of the corresponding AF.

Theorem 1 *Let $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ be the free CSP instance associated with the AF $\mathcal{F} = (\mathcal{A}, \mathcal{R})$. The tuple $(v(x_1), \ldots, v(x_n))$ is a solution of the CSP iff the set $\{x_j, \ldots, x_k\} \subseteq \mathcal{X}$ s.t. $v(x_i) = 1$ is conflict-free (with $i = j \ldots, k$).*

Proof Let $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ be the free CSP associated with the argumentation framework $\mathcal{F} = (\mathcal{A}, \mathcal{R})$. Assume that $(v(x_1), \ldots, v(x_n))$ is a solution of the CSP. Let $X = \{x_j, \ldots, x_k\}$ be such that $v(x_i) = 1$. Assume also that X is not conflict-free, thus $\exists x, x' \in X$ such that $x\mathcal{R}x'$. Thus, $\exists c \in \mathcal{C}$ such that $c = ((x, x'), \lceil x \Rightarrow \neg x' \rceil)$. Since $(v(x_1), \ldots, v(x_n))$ is a solution, then it satisfies the constraint c. However, $v(x) = 1$ and $v(x') = 1$, then $(1, 1) \notin \lceil x \Rightarrow \neg x' \rceil)$. Contradiction. Assume now a tuple $(v(x_1), \ldots, v(x_n))$ such that $X = \{x_j, \ldots, x_k\}$ (with $v(x_i) = 1$) is conflict-free. Assume that $(v(x_1), \ldots, v(x_n))$ is not a solution. Thus, there exists a constraint $c = ((x, x'), \lceil x \Rightarrow \neg x' \rceil)$ which is violated. But since X is conflict-free, then from the definition of Free CSP, this constraint cannot exist between x and x'.

Let us illustrate this kind of CSP by considering the argumentation framework $\mathcal{F}_1$ given in Example 1.

Example 1 (Cont): The free CSP corresponding to $\mathcal{F}_1$ is $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ s.t.

– $\mathcal{X} = \{a, b, c, d\}$,
– $\mathcal{D} = \{\{0, 1\}, \{0, 1\}, \{0, 1\}, \{0, 1\}\}$,
– $\mathcal{C} = \{((a, b), \lceil a \Rightarrow \neg b \rceil), ((b, c), \lceil b \Rightarrow \neg c \rceil), ((c, d), \lceil c \Rightarrow \neg d \rceil), ((d, a), \lceil d \Rightarrow \neg a \rceil)\}$.

This CSP has the following solutions:

$(0, 0, 0, 0)$	$(1, 0, 0, 0)$
$(0, 1, 0, 0)$	$(0, 0, 1, 0)$
$(0, 0, 0, 1)$	$(1, 0, 1, 0)$
$(0, 1, 0, 1)$	

Thus, the sets $\{\}$, $\{a\}$, $\{b\}$, $\{c\}$, $\{d\}$, $\{a, c\}$ and $\{b, d\}$ are conflict-free.

Let us now study the case of stable semantics. Stable extensions are computed by a CSP which considers that an argument and its attackers cannot have the same value.

Definition 8 (Stable CSP) Let $\mathcal{F} = (\mathcal{A}, \mathcal{R})$ be an argumentation framework. A *stable CSP* associated with $\mathcal{F}$ is a tuple $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ where $\mathcal{X} = \mathcal{A}$, $\forall \mathcal{D}_i \in \mathcal{D}$, $\mathcal{D}_i = \{0, 1\}$ and $\mathcal{C} = \{(h, H) \mid h = (a, b_1, \ldots, b_n)$ s.t. $(b_i, a) \in \mathcal{R}$ and $H = \lceil a \Leftrightarrow \bigwedge \neg b_i \rceil, a \in \mathcal{A}\}$, with the convention that $\bigwedge_{x \in X} x = \top$ if $X = \emptyset$.

Unlike free CSP, the number of constraints is equal to the number of arguments of the framework.

Property 2 Let $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ be the stable CSP instance associated with the AF $\mathcal{F} = (\mathcal{A}, \mathcal{R})$. It holds that $|\mathcal{C}| = |\mathcal{A}|$.

Proof This follows immediately from Definition 8.

It is worth mentioning that the previous definition is inspired from [14]. The following result shows the correspondence between the solutions of a stable CSP and the stable extensions of its associated argumentation framework.

Theorem 2 *Let $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ be a stable CSP associated with $\mathcal{F} = (\mathcal{A}, \mathcal{R})$. The tuple $(v(x_1), \ldots, v(x_n))$ is a solution of the CSP iff the set $\{x_j, \ldots, x_k\}$ s.t. $v(x_i) = 1$ is a stable extension of $\mathcal{F}$.*

Proof Let $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ be a stable CSP associated with $\mathcal{F} = (\mathcal{A}, \mathcal{R})$. In [6], it was shown that a set $\mathcal{E} \subseteq \mathcal{A}$ is a stable extension of the argumentation framework $\mathcal{F}$ iff $\mathcal{E}$ is a model of the formula $\bigwedge_{a \in \mathcal{A}} (a \Leftrightarrow \bigwedge_{b:(b,a) \in \mathcal{R}} \neg b)$, thus a model of the set $\{a \Leftrightarrow \bigwedge_{b:(b,a) \in \mathcal{R}} \neg b, a \in \mathcal{A}\}$. Consequently, $\mathcal{E}$ is a stable extension iff it satisfies all the constraints of $\mathcal{C}$, thus iff it is a solution of the stable CSP.

Let us illustrate the notion of stable CSP on our running Example 1.

Example 1 (Cont): The stable CSP corresponding to $\mathcal{F}_1$ is $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ s.t. $\mathcal{X} = \{a, b, c, d\}$, $\mathcal{D} = \{\{0,1\}, \{0,1\}, \{0,1\}, \{0,1\}\}$, and $\mathcal{C} = \{((a,d), \lceil a \Leftrightarrow \neg d \rceil)$, ((b,a), $\lceil b \Leftrightarrow \neg a \rceil)$, ((c,b), $\lceil c \Leftrightarrow \neg b \rceil)$, ((d,c), $\lceil d \Leftrightarrow \neg c \rceil\})$. This CSP has two solutions: $(1,0,1,0)$ and $(0,1,0,1)$. The sets $\{a, c\}$ and $\{b, d\}$ are the two stable extensions of $\mathcal{F}_1$.

Let us now consider the case of an argumentation framework which has no stable extensions.

Example 2 Let $\mathcal{F}_2 = (\mathcal{A}_2, \mathcal{R}_2)$ be an argumentation framework where $\mathcal{A}_2 = \{a, b\}$ and $\mathcal{R}_2 = \{(b, b)\}$. Note that this framework has no stable extension. The corresponding stable CSP is $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ where $\mathcal{X} = \mathcal{A}_2$ and $\mathcal{C} = \{((a), \lceil a \rceil), ((b), \lceil b \wedge \neg b \rceil)\}$. This CSP has no solution since the constraint $b \wedge \neg b$ is not satisfiable. Consequently, $\mathcal{F}_2$ has no stable extensions.

The two previous CSPs are simple since attacks are directly transformed into constraints. The notion of defence is not needed in both cases. However, things are not so obvious with admissible semantics. Indeed, a CSP that computes admissible extensions contains, for each argument, two kinds of constraints: constraints that ensure that this argument cannot be simultaneously accepted with any of its attackers, and constraints that ensure that the argument is defended by other arguments. Note that the first kind of constraints ensures conflict-freeness whereas the second ensures the defence requirement of the semantics.

Definition 9 (Admissible CSP) Let $\mathcal{F} = (\mathcal{A}, \mathcal{R})$ be an argumentation framework. An *admissible CSP* associated with $\mathcal{F}$ is a tuple $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ where $\mathcal{X} = \mathcal{A}$, $\forall \mathcal{D}_i \in \mathcal{D}$, $\mathcal{D}_i = \{0,1\}$ and $\mathcal{C} = \mathcal{C}_1 \cup \mathcal{C}_2$ where

- $\mathcal{C}_1 = \{(h, H) \mid h = (a, b_1, \ldots, b_m), H = \left\lceil a \Rightarrow \bigwedge_{(b_i,a)\in\mathcal{R}} \neg b_i \right\rceil \mid a \in \mathcal{A}\}$
- $\mathcal{C}_2 = \{(h, H) \mid h = (a, c_1, \ldots, c_n), H = \left\lceil a \Rightarrow \bigwedge_{b:(b,a)\in\mathcal{R}} (\bigvee_{(c_i,b)\in\mathcal{R}} c_i) \right\rceil \mid a \in \mathcal{A}\}$

It is worth mentioning that the two kinds of constraints can be combined in a unique class, where each constraint is a conjunction of a formula in $\mathcal{C}_1$ and a formula of $\mathcal{C}_2$ for the same argument. However, such encoding would not be optimal. It makes Generalized Arc Consistency possible but not polynomial [31].

The number of constraints is at most the number of arguments of the CAF plus one (the constraint of the CAF).

Property 3 Let $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ be the admissible CSP instance associated with the AF $\mathcal{F} = (\mathcal{A}, \mathcal{R})$. It holds that $|\mathcal{C}| \leq 2 * |\mathcal{A}|$.

Proof This follows immediately from the previous definition.

The following result shows that the solutions of an admissible CSP provide the admissible extensions of the corresponding argumentation framework.

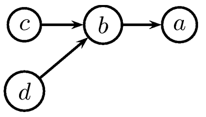
Theorem 3 *Let $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ be an admissible CSP associated with an AF $\mathcal{F}$. The tuple $(v(x_1), \ldots, v(x_n))$ is a solution of this CSP iff the set $\{x_j, \ldots, x_k\}$ s.t. $v(x_i) = 1$ is an admissible set of $\mathcal{F}$.*

Proof Let $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ be an admissible CSP associated with an AF $\mathcal{F}$. In [6], it was shown that a set $\mathcal{E} \subseteq \mathcal{A}$ is an admissible extension of the argumentation framework $\mathcal{F}$ iff $\mathcal{E}$ is a model of the formula $\bigwedge_{a\in\mathcal{A}} ((a \Rightarrow \bigwedge_{(b_i,a)\in\mathcal{R}} \neg b_i) \wedge (a \Rightarrow \bigwedge_{b:(b,a)\in\mathcal{R}} (\bigvee_{(c_i,b)\in\mathcal{R}} c_i))$. Consequently, $\mathcal{E}$ is an admissible extension iff it satisfies all the constraints of $\mathcal{C}$, thus iff it is a solution of the admissible CSP.

Let us illustrate the notion of admissible CSP with a simple example.

Example 3 Let us consider the framework $\mathcal{F}_3 = (\mathcal{A}_3, \mathcal{R}_3)$ where $\mathcal{A}_3 = \{a, b, c, d\}$ and let $\mathcal{R}_3$ be as depicted in figure below:

$$\begin{tikzcd} c \arrow[r] & b \arrow[r] & a \\ d \arrow[ru] & & \end{tikzcd}$$



The admissible CSP associated with $\mathcal{F}_3$ is $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ where: $\mathcal{X} = \mathcal{A}_3$, $\mathcal{D} = \{\{0,1\}, \{0,1\}, \{0,1\}, \{0,1\}\}$ and $\mathcal{C} =$
$\{((d), \lceil d \Rightarrow \top\rceil),$
$((c), \lceil c \Rightarrow \top\rceil),$
$((b, c, d), \lceil b \Rightarrow \neg c \wedge \neg d\rceil),$
$((a, b), \lceil a \Rightarrow \neg b)\rceil,$
$((a, c, d), \lceil a \Rightarrow c \vee d\rceil)\}.$

This CSP has the following solutions: $(0,0,0,0)$, $(0,0,1,0)$, $(0,0,0,1)$, $(0,0,1,1)$ $(1,0,1,0)$, $(1,0,0,1)$, $(1,0,1,1)$. These solutions return the admissible sets of $\mathcal{F}_3$, that is: $\{\}$, $\{c\}$, $\{d\}$, $\{c, d\}$, $\{a, c\}$, $\{a, d\}$ and $\{a, c, d\}$.

As preferred extensions are maximal (for set inclusion) admissible sets, then they are computed by an admissible CSP.

Theorem 4 *Let $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ be an admissible CSP associated with an AF $\mathcal{F}$. Each maximal (for set inclusion) set $\{x_j, \ldots, x_k\}$, s.t. $v(x_i) = 1$ and $(v(x_1), \ldots, v(x_n))$ is a solution of the CSP, is a preferred extension of $\mathcal{F}$.*

Proof This follows directly from the definition of a preferred extension and Theorem 3.

Let us now come back to Example 3 and check the preferred extensions of the framework.

Example 3 (Cont): The solution $(1, 0, 1, 1)$ returns the only preferred extension of $\mathcal{F}_3$, i.e. $\{a, c, d\}$.

Complete extensions are also computed by a CSP which takes into account the notion of defence in the constraints.

Definition 10 (Complete CSP) Let $\mathcal{F} = (\mathcal{A}, \mathcal{R})$ be an argumentation framework. A *complete CSP* associated with $\mathcal{F}$ is a tuple $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ where $\mathcal{X} = \mathcal{A}$, for each $\mathcal{D}_i \in \mathcal{D}$, $\mathcal{D}_i = \{0, 1\}$ and $\mathcal{C} = \mathcal{C}_1 \cup \mathcal{C}_2$ where

- $\mathcal{C}_1 = \{(h, H) \mid h = (a, b_1, \ldots, b_m), H = \left[a \Rightarrow \bigwedge_{(b_i, a) \in \mathcal{R}} \neg b_i \right] \mid a \in \mathcal{A}\}$
- $\mathcal{C}_2 = \{(h, H) \mid h = (a, c_1, \ldots, c_n), H = \left[a \Leftrightarrow \bigwedge_{b:(b,a) \in \mathcal{R}} (\bigvee_{(c_i, b) \in \mathcal{R}} c_i) \right] \mid a \in \mathcal{A}\}$

Note that there is a slight difference between the constraints of an admissible CSP and those of a complete CSP. Since a complete extension should contain all the arguments it defends, then an argument and all its defenders should be in the same set. However, the only requirement on an admissible set is that it defends its arguments. This is encoded by a simple logical implication.

Property 4 Let $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ be the complete CSP instance associated with the AF $\mathcal{F} = (\mathcal{A}, \mathcal{R})$. It holds that $|\mathcal{C}| \leq 2 * |\mathcal{A}|$.

Proof This follows immediately from the previous definition.

The next result confirms that a complete CSP returns all the complete extensions of an AF.

Theorem 5 *Let $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ be a complete CSP associated with an AF $\mathcal{F}$. The tuple $(v(x_1), \ldots, v(x_n))$ is a solution of the CSP iff the set $\{x_j, \ldots, x_k\}$, s.t. $v(x_i) = 1$ is a complete extension of $\mathcal{F}$.*

Proof Let $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ be a complete CSP associated with an AF $\mathcal{F}$. In [6], it was shown that a set $\mathcal{E} \subseteq \mathcal{A}$ is a complete extension of the argumentation framework $\mathcal{F}$ iff $\mathcal{E}$ is a model of the formula $\bigwedge_{a \in \mathcal{A}} ((a \Rightarrow \bigwedge_{(b_i, a) \in \mathcal{R}} \neg b_i) \wedge (a \Leftrightarrow \bigwedge_{b:(b,a) \in \mathcal{R}} (\bigvee_{(c_i, b) \in \mathcal{R}} c_i))$. Consequently, $\mathcal{E}$ is a complete extension iff it satisfies all the constraints of $\mathcal{C}$, thus iff it is a solution of the complete CSP.

Example 3 (Cont): The complete CSP associated with $\mathcal{F}_3$ is $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ where: $\mathcal{X} = \mathcal{A}_3$, $\mathcal{D} = \{\{0,1\}, \{0,1\}, \{0,1\}, \{0,1\}\}$ and $\mathcal{C} =$
$\{((d), \lceil d \Rightarrow \top \rceil),$
$((c), \lceil c \Rightarrow \top \rceil),$
$((b,c,d), \lceil b \Rightarrow \neg c \wedge \neg d \rceil),$
$((a,b), \lceil a \Rightarrow \neg b) \rceil,$
$((a,c,d), \lceil a \Leftrightarrow c \vee d \rceil)\}$.
This CSP has one solution which is $(1,0,1,1)$. Thus, the set $\{a,c,d\}$ is the unique complete extension of $\mathcal{F}_3$.

Since grounded extension is a minimal (for set inclusion) complete extension, then it is computed by a complete CSP as follows.

Theorem 6 *Let $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ be a complete CSP associated with an AF $\mathcal{F}$. The grounded extension of $\mathcal{F}$ is the minimal (for set inclusion) set $\{x_j, \ldots, x_k\}$ s.t. $v(x_i) = 1$ and $(v(x_1), \ldots, v(x_n))$ is a solution of the CSP.*

Proof This follows from the definition of a grounded extension and Theorem 5.

Example 3 (Cont): The grounded extension of $\mathcal{F}_3$ is $\{a,c,d\}$ which is returned by the unique solution of the complete CSP corresponding to $\mathcal{F}_3$.

4 Constrained framework

This section recalls the constrained version of Dung's argumentation framework [13] and proposes various CSPs that compute its extensions under Dung's semantics.

4.1 Basic definitions

The basic argumentation framework of Dung was extended in [13] by adding a *constraint* on arguments. This constraint should be satisfied by Dung's extensions (under a given semantics). For instance, in Example 1, one may imagine a constraint which requires that the two arguments a and c belong to the same extension. Note that this constraint is satisfied by $\mathcal{B}_1$ but not by $\mathcal{B}_2$. Thus, $\mathcal{B}_1$ would be the only extension of the framework. The constrained version of Dung's system may be useful in some AI problems, like practical reasoning [3], where a filter needs to be applied on the extensions of the argumentation framework in order to select only the ones that contain for each desire, an argument justifying the desire and another argument ensuring its feasibility.

In this framework, the constraint is a formula of a propositional language $\mathcal{L}_{\mathcal{A}}$ whose alphabet is exactly the set $\mathcal{A}$ of arguments. Thus, each argument in $\mathcal{A}$ is a literal of $\mathcal{L}_{\mathcal{A}}$. $\mathcal{L}_{\mathcal{A}}$ contains all the formulas that can be built using the usual logical operators ($\wedge$, $\vee$, $\Rightarrow$, $\neg$, $\Leftrightarrow$) and the constant symbols ($\top$ and $\bot$).

Definition 11 (Constraint, Completion) Let $\mathcal{A}$ be a set of arguments and $\mathcal{L}_{\mathcal{A}}$ its corresponding propositional language.

- $\mathscr{C}$ is a *constraint* on $\mathcal{A}$ iff $\mathscr{C}$ is a formula of $\mathcal{L}_{\mathcal{A}}$.
- The *completion* of a set $\mathcal{B} \subseteq \mathcal{A}$ is $\widehat{\mathcal{B}} = \{a \mid a \in \mathcal{B}\} \cup \{\neg a \mid a \in \mathcal{A} \setminus \mathcal{B}\}$.
- A set $\mathcal{B} \subseteq \mathcal{A}$ *satisfies* $\mathscr{C}$ iff $\widehat{\mathcal{B}}$ is a model of $\mathscr{C}$ ($\widehat{\mathcal{B}} \models \mathscr{C}$).

The completion of a set $\mathcal{B}$ of arguments is a set in which each argument of $\mathcal{A}$ appears either as a positive literal if the argument belongs to $\mathcal{B}$ or as a negative one otherwise. Thus, $|\widehat{\mathcal{B}}| = |\mathcal{A}|$. A constrained argumentation framework (CAF) is defined as follows:

Definition 12 (CAF) A *constrained argumentation framework* (CAF) is a triple $\mathcal{F} = (\mathcal{A}, \mathcal{R}, \mathscr{C})$ where $\mathcal{A}$ is a set of arguments, $\mathcal{R} \subseteq \mathcal{A} \times \mathcal{A}$ is an attack relation and $\mathscr{C}$ is a constraint on the set $\mathcal{A}$.

Dung's semantics are extended to the case of CAFs. The idea is to compute Dung's extensions (under a given semantics), and to keep among those extensions only the ones that satisfy the constraint $\mathscr{C}$.

Definition 13 (C-admissible set) Let $\mathcal{F} = (\mathcal{A}, \mathcal{R}, \mathscr{C})$ be a CAF and $\mathcal{B} \subseteq \mathcal{A}$. The set $\mathcal{B}$ is *$\mathscr{C}$-admissible* in $\mathcal{F}$ iff:

1. $\mathcal{B}$ is admissible,
2. $\mathcal{B}$ satisfies the constraint $\mathscr{C}$.

In [19], it was shown that the empty set is always admissible. However, it is not always $\mathscr{C}$-admissible since the set $\widehat{\varnothing}$ does not always imply $\mathscr{C}$.

Definition 14 (C-preferred, C-stable extension) Let $\mathcal{F} = (\mathcal{A}, \mathcal{R}, \mathscr{C})$ be a CAF and $\mathcal{B} \subseteq \mathcal{A}$.

- $\mathcal{B}$ is a *$\mathscr{C}$-preferred extension* of $\mathcal{F}$ iff $\mathcal{B}$ is maximal for set-inclusion among the $\mathscr{C}$-admissible sets.
- $\mathcal{B}$ is a *$\mathscr{C}$-stable extension* of $\mathcal{F}$ iff $\mathcal{B}$ is a $\mathscr{C}$-preferred extension that attacks all arguments in $\mathcal{A} \backslash \mathcal{B}$.

The following result summarizes the links between the extensions of a CAF $\mathcal{F} = (\mathcal{A}, \mathcal{R}, \mathscr{C})$ and those of its basic version $\mathcal{F}' = (\mathcal{A}, \mathcal{R})$ (i.e. the argumentation framework without the constraint).

Theorem 7 *[13] Let $\mathcal{F} = (\mathcal{A}, \mathcal{R}, \mathscr{C})$ be a CAF and $\mathcal{F}' = (\mathcal{A}, \mathcal{R})$ be its basic version.*

- *For each $\mathscr{C}$-preferred extension $\mathcal{B}$ of $\mathcal{F}$, there exists a preferred extension $\mathcal{B}'$ of $\mathcal{F}'$ s.t. $\mathcal{B} \subseteq \mathcal{B}'$.*
- *Every $\mathscr{C}$-stable extension of $\mathcal{F}$ is a stable extension of $\mathcal{F}'$. The converse does not hold.*

It is worth noticing that when the constraint of a CAF is a tautology, then the extensions of this CAF coincide with those of its basic version.

Let us now illustrate this notion of CAFs through a simple example.

Example 1 (Cont): Assume an extended version of the argumentation framework $\mathcal{F}_1$ where we would like to accept the two arguments a and c. This is encoded by a constraint $\mathscr{C} : a \wedge c$. It can be checked that the CAF $(\mathcal{A}_1, \mathcal{R}_1, \mathscr{C})$ has one $\mathscr{C}$-stable extension which is $\mathcal{B}_1 = \{a, c\}$. Note that $\mathcal{B}_2 = \{b, d\}$ is a stable extension of $\mathcal{F}_1$ but not a $\mathscr{C}$-stable extension of its constrained version.

4.2 Mappings into CSPs

Let $\mathcal{F} = (\mathcal{A}, \mathcal{R}, \mathscr{C})$ be a given CAF. In order to compute its $\mathscr{C}$-extensions under different semantics, we follow the same line of research as in the previous section. The only difference is that in addition to the constraints defined in Section 3.2, there is an additional constraint which is $\mathscr{C}$.

Let us start with $\mathscr{C}$-stable extensions. They are computed by the stable CSP given in Def. 8 augmented by the constraint $\mathscr{C}$ in its set $\mathcal{C}$.

Definition 15 ($\mathscr{C}$-stable CSP) Let $\mathcal{F} = (\mathcal{A}, \mathcal{R}, \mathscr{C})$ be a constrained argumentation framework. A *$\mathscr{C}-$stable CSP* associated with $\mathcal{F}$ is a tuple $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ where $\mathcal{X} = \mathcal{A}$, $\forall \mathcal{D}_i \in \mathcal{D}$, $\mathcal{D}_i = \{0,1\}$ and $\mathcal{C} = \{(h, H) \mid h = (a, b_1, \ldots, b_n)$ s.t. $(b_i, a) \in \mathcal{R}$ and $H = \lceil a \Leftrightarrow \bigwedge \neg b_i \rceil, a \in \mathcal{A}\} \cup \{(h', H') \mid H' = \lceil \mathscr{C} \rceil, h' = (a_1, \ldots, a_j)$ where $\{a_1, \ldots, a_j\} = \mathtt{Atoms}(\mathscr{C})\}$.

It is easy to check that the number of constraints is exactly the number of arguments of the CAF plus one (the constraint of the CAF).

Property 5 Let $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ be the $\mathscr{C}-$stable CSP instance associated with the CAF $\mathcal{F} = (\mathcal{A}, \mathcal{R}, \mathscr{C})$. It holds that $|\mathcal{C}| = |\mathcal{A}| + 1$.

Proof This follows immediately from the previous definition.

We show next that the solutions of a $\mathscr{C}$-stable CSP return all the $\mathscr{C}$-stable extensions of the corresponding CAF.

Theorem 8 *Let $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ be a $\mathscr{C}$-stable CSP associated with a CAF $\mathcal{F} = (\mathcal{A}, \mathcal{R}, \mathscr{C})$. The tuple $(v(x_1), \ldots, v(x_n))$ is a solution of the CSP iff the set $\{x_j, \ldots, x_k\}$ such that $v(x_i) = 1$ is a $\mathscr{C}$-stable extension of $\mathcal{F}$.*

Proof Let $\mathcal{T} = (\mathcal{X}, \mathcal{D}, \mathcal{C})$ be a $\mathscr{C}$-stable CSP associated with a CAF $(\mathcal{A}, \mathcal{R}, \mathscr{C})$. The $\mathscr{C}$-stable extensions of the CAF $(\mathcal{A}, \mathcal{R}, \mathscr{C})$ are the stable extensions of $(\mathcal{A}, \mathcal{R})$ that verify the constraint $\mathscr{C}$. Besides, let $\mathcal{T}' = (\mathcal{X}, \mathcal{D}, \mathcal{C}')$ be the stable CSP associated with a AF $(\mathcal{A}, \mathcal{R})$. From Theorem 2, there is a bijection between the solutions of the CSP $\mathcal{T}'$ and the stable extensions of $(\mathcal{A}, \mathcal{R})$. Moreover, from Definition 15, the solutions of $\mathcal{T}$ verify all the constraints in $\mathcal{C}'$ and the additional constraint $\mathscr{C}$. Thus, there is a bijection between the solutions of the $\mathscr{C}$-stable CSP and the $\mathscr{C}$-stable extensions of $(\mathcal{A}, \mathcal{R}, \mathscr{C})$.

Example 1 (Cont): The $\mathscr{C}$-stable CSP associated with the CAF extending $\mathcal{F}_1$ with the constraint $\mathscr{C} : a \wedge c$ is $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ s.t. $\mathcal{X} = \{a, b, c, d\}$, $\mathcal{D} = \{\{0,1\}, \{0,1\}, \{0,1\}, \{0,1\}\}$, and $\mathcal{C} = \{((a, c)\ \lceil a \wedge c \rceil), ((a, d), \lceil a \Leftrightarrow \neg d \rceil)$, ((b,a), $\lceil b \Leftrightarrow \neg a \rceil$), ((c,b), $\lceil c \Leftrightarrow \neg b \rceil$), ((d,c), $\lceil d \Leftrightarrow \neg c \rceil\}$). This CSP has one solution which is $(1, 0, 1, 0)$. It returns the $\mathscr{C}$-stable extension $\{a, c\}$ of the CAF.

A CSP which computes the $\mathscr{C}$-admissible sets of a CAF is grounded on the admissible CSP introduced in Definition 9.

Definition 16 ($\mathscr{C}$-admissible CSP) Let $\mathcal{F} = (\mathcal{A}, \mathcal{R}, \mathscr{C})$ be a constrained argumentation framework. A *$\mathscr{C}$-admissible CSP* associated with $\mathcal{F}$ is a tuple $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ where $\mathcal{X} = \mathcal{A}$, for each $a_i \in \mathcal{X}$, $\mathcal{D}_i = \{0,1\}$ and $\mathcal{C} = \mathcal{C}_1 \cup \mathcal{C}_2 \cup \mathcal{C}_3$ where

- $\mathcal{C}_1 = \{(h, H) \mid h = (a, b_1, \ldots, b_m), H = \left\lceil a \Rightarrow \bigwedge_{(b_i,a)\in\mathcal{R}} \neg b_i \right\rceil \mid a \in \mathcal{A}\}$
- $\mathcal{C}_2 = \{(h, H) \mid h = (a, c_1, \ldots, c_n), H = \left\lceil a \Rightarrow \bigwedge_{b:(b,a)\in\mathcal{R}} (\bigvee_{(c_i,b)\in\mathcal{R}} c_i) \right\rceil \mid a \in \mathcal{A}\}$
- $\mathcal{C}_3 = \{(h', H') \mid H' = \lceil \mathscr{C} \rceil, h' = (a_1, \ldots, a_j)$ where $\{a_1, \ldots, a_j\} = \texttt{Atoms}(\mathscr{C})\}$

Property 6 Let $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ be the $\mathscr{C}$-admissible CSP instance associated with the CAF $\mathcal{F} = (\mathcal{A}, \mathcal{R}, \mathscr{C})$. It holds that $|\mathcal{C}| = 2 * |\mathcal{A}| + 1$.

Proof This follows immediately from the previous definition.

We show that the solutions of this CSP are $\mathscr{C}$-admissible extensions of the corresponding CAF.

Theorem 9 *Let $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ be a $\mathscr{C}$-admissible CSP associated with a CAF $\mathcal{F} = (\mathcal{A}, \mathcal{R}, \mathscr{C})$. The tuple $(v(x_1), \ldots, v(x_n))$ is a solution of the CSP iff the set $\{x_j, \ldots, x_k\}$ s.t. $v(x_i) = 1$ is a $\mathscr{C}$-admissible set of the CAF $\mathcal{F}$.*

Proof Let $\mathcal{T} = (\mathcal{X}, \mathcal{D}, \mathcal{C})$ be a $\mathscr{C}$-admissible CSP associated with a CAF $(\mathcal{A}, \mathcal{R}, \mathscr{C})$. The $\mathscr{C}$-admissible extensions of the CAF $(\mathcal{A}, \mathcal{R}, \mathscr{C})$ are the admissible extensions of $(\mathcal{A}, \mathcal{R})$ that verify the constraint $\mathscr{C}$. Besides, let $\mathcal{T}' = (\mathcal{X}, \mathcal{D}, \mathcal{C}')$ be the admissible CSP associated with a AF $(\mathcal{A}, \mathcal{R})$. From Theorem 3, there is a bijection between the solutions of the CSP $\mathcal{T}'$ and the admissible extensions of $(\mathcal{A}, \mathcal{R})$. Moreover, from Definition 16, the solutions of $\mathcal{T}$ verify all the constraints in $\mathcal{C}'$ and the additional constraint $\mathscr{C}$. Thus, there is a bijection between the solutions of the $\mathscr{C}$-admissible CSP and the $\mathscr{C}$-admissible extensions of $(\mathcal{A}, \mathcal{R}, \mathscr{C})$.

What about the $\mathscr{C}$-preferred extensions of a constrained argumentation framework? Recall that $\mathscr{C}$-preferred extensions are maximal (for set inclusion) $\mathscr{C}$-admissible sets. Thus, the following result follows from the previous one.

Theorem 10 *Let $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ be a $\mathscr{C}$-admissible CSP associated with a CAF $\mathcal{F} = (\mathcal{A}, \mathcal{R}, \mathscr{C})$. Each maximal (for set inclusion) set $\{x_j, \ldots, x_k\}$, s.t. $v(x_i) = 1$ and $(v(x_1), \ldots, v(x_n))$ is a solution of the CSP, is a $\mathscr{C}$-preferred extension of $\mathcal{F}$.*

Proof This follows from the definition of a $\mathscr{C}$-preferred extension and Theorem 9.

5 Preference-based frameworks

Is is well acknowledged in argumentation literature that arguments may not have the same strength. For instance, arguments built from certain information are stronger than arguments built from uncertain information. Consequently, in [2] Dung's framework was extended in such a way to take into account the strengths of arguments when evaluating them. The idea is to consider in addition to the attack relation, another binary relation $\succeq$ which represents preferences between arguments. This relation can be instantiated in different ways. Writing $a \succeq b$ means that a is at least as good as b. Let $\succ$ be the strict relation associated with $\succeq$. It is defined as follows: $a \succ b$ iff $a \succeq b$ and not $b \succeq a$. In Dung's framework, an attack always succeeds (if the attacked argument is not defended). In preference-based frameworks, an attack may fail if the attacked argument is stronger than its attacker.

Definition 17 (PAF) A *preference-based argumentation framework* (PAF) is a tuple $\mathcal{F} = (\mathcal{A}, \mathcal{R}, \succeq)$ where $\mathcal{A}$ is a set of arguments, $\mathcal{R} \subseteq \mathcal{A} \times \mathcal{A}$ is an attack relation and $\succeq$ is (partial or total) preorder on $\mathcal{A}$ ($\succeq \subseteq \mathcal{A} \times \mathcal{A}$).
The extensions of $\mathcal{F}$ (under any semantics) are those of the AF $(\mathcal{A}, \texttt{Def})$ where $(a, b) \in \texttt{Def}$ iff $(a, b) \in \mathcal{R}$ and not$(b \succ a)$.

It is clear that each preference-based argumentation framework has a corresponding basic framework (in the sense of Definition 4). This latter is used for computing the extensions of the PAF under various semantics.

Property 7 Let $\mathcal{F} = (\mathcal{A}, \mathcal{R}, \succeq)$ be a PAF and $\mathcal{T} = (\mathcal{A}, \texttt{Def})$ its corresponding AF. The extensions of $\mathcal{F}$ under semantics x are exactly the extensions of $\mathcal{T}$ under the same semantics.

Proof This follows from the definition.

It is thus obvious that the extensions of a PAF $\mathcal{F} = (\mathcal{A}, \mathcal{R}, \succeq)$ under semantics x are computed by the x-CSP that is associated with the basic framework $\mathcal{T} = (\mathcal{A}, \texttt{Def})$. For instance, the stable extensions of $\mathcal{F}$ are computed by the stable-CSP associated with $\mathcal{T}$, the preferred extensions of $\mathcal{F}$ are computed by the preferred-CSP associated with $\mathcal{T}$, and so on.

6 Related work

There are very few attempts in the literature for modeling argumentation frameworks as a CSP. To the best of our knowledge, the only work on the topic is the one done in [7]. In this work, the authors studied the problem of encoding *weighted* argumentation frameworks by semirings. In a weighted framework, attacks do not necessarily have the same weights. Thus, a weight (i.e. a value between 0 and 1) is associated with each attack between two arguments. When all the attacks have weight 1, the corresponding framework collapses with Dung's abstract framework recalled in Section 3. The authors focused only on two semantics: stable and complete. In our paper, we proposed an alternative approach for computing the extensions of a basic argumentation framework ([19]) under *all* Dung's semantics (i.e; even under admissible, preferred and grounded semantics). Moreover, we studied the same semantics for two other types of argumentation frameworks: constrained argumentation frameworks proposed in [13] and preference-based frameworks introduced in [2]. Finally, our approach for defining CSPs is simpler and more natural than that followed in [7]. Indeed, in [7], the authors used soft CSP whereas in our paper we used simple CSP.

Another interesting piece of work which is closer to our is that presented in [6]. Indeed, the authors encoded Dung's semantics as a satisfiability problem (SAT). Besides, it was shown in [10] that SAT is a particular case of CSPs. Moreover, a mapping from SAT to CSP was given in the same paper. In our paper, we took advantage of that mapping and we presented different CSPs which encode Dung's semantics not only for Dung's framework, but also for constrained frameworks and preference-based ones.

Finally, a system called ASPARTIX implemented Dung's semantics using answer set programming (ASP) [24]. ASPARTIX relies on a fixed disjunctive datalog program which takes an argumentation framework as input, and uses the answer-set solver DLV for computing its extensions under a given semantics. This work is complementary to our especially since the ASP community established some links between ASP and CSP.

7 Conclusion

In this paper, we expressed the problem of computing the extensions of an argumentation framework under a given semantics as a CSP. We investigated three types of frameworks: Dung's argumentation framework [19], its constrained version proposed in [13], and its extension with preferences [2]. For each of these frameworks, we proposed different CSPs which compute the extensions under various semantics, namely admissible, preferred, stable, complete and grounded. Such mappings are of great importance since they allow the use of the *efficient solvers* that were developed by CSP community. Thus, the efficiency of our CSPs depend on that of the solver that is chosen to solve them. The free-CSP has the form of 2-satisfiability (or 2SAT) problem. 2SAT consists of determining whether a collection of Boolean variables with constraints on pairs of variables can be assigned values satisfying all the constraints. This is particularly the case of free-CSP. It is acknowledged in SAT community that a 2SAT problem has a polynomial time solution [27]. In [25], polynomial time algorithms enumerating all the solutions are provided. It is worth mentioning that in the particular case of grounded semantics, there is an additional test of minimality that is required after computing the solutions of the corresponding CSP. This increases thus the complexity of computing the grounded extension of an argumentation framework. Consequently, this particular extension should be computed using existing algorithms in argumentation literature [1] and not by a CSP.

There are a number of ways to extend this work. One future direction consists of proposing the CSPs that return other semantics like semi-stable [9] and ideal [20]. Another idea consists of encoding weighted argumentation frameworks [22] as CSPs. In a weighted framework, attacks may not have the same importance. Such framework can be encoded by valued CSP in which constraints are associated with weights. Finally, we should study the kind of solvers that are more appropriate with our CSPs, and test their efficiency on real benchmarks.

References

1. *Argumentation in Artificial Intelligence.* I. Rahwan and G. Simari (eds.), Springer, 2009.
2. L. Amgoud and C. Cayrol. A reasoning model based on the production of acceptable arguments. *Annals of Mathematics and Artificial Intelligence*, 34:197–216, 2002.
3. L. Amgoud, C. Devred, and M. Lagasquie. Generating possible intentions with constrained argumentation systems. *International Journal of Approximate Reasoning*, 52(9):1363–1391, 2011.
4. T. J. M. Bench-Capon. Persuasion in practical argument using value-based argumentation frameworks. *Journal of Logic and Computation*, 13(3):429–448, 2003.
5. J. Bentahar and Z. Maamar. Complexity results for argumentation-based agent communication. In *IEEE International Conference on Innovations in Information Technology*, pages 506–510, 200.
6. P. Besnard and S. Doutre. Checking the acceptability of a set of arguments. In *NMR*, pages 59–64, 2004.

7. S. Bistarelli and F. Santini. A common computational framework for semiring-based argumentation systems. In *ECAI*, pages 131–136, 2010.
8. L. Bordeaux, Y. Hamadi, and L. Zhang. Propositional satisfiability and constraint programming: A comparative survey. *ACM Computing Surveys*, 38(4):1–54, 2006.
9. M. Caminada. Semi-stable semantics. In *Proceedings of the 1st International Conference on Computational Models of Argument (COMMA'06)*, pages 121–130, 2006.
10. T. Castell and H. Fargier. Propositional satisfaction problems and clausal csps. In *ECAI*, pages 214–218, 1998.
11. C. Cayrol, S. Doutre, and J. Mengin. On decision problems related to the preferred semantics for argumentation frameworks. *Journal of Logic and Computation*, 13(3):377–403, 2003.
12. M. Cooper. An optimal k-consistency algorithm. *Artificial Intelligence*, pages 89–95, 1989.
13. S. Coste-Marquis, C. Devred, and P. Marquis. Constrained argumentation frameworks. In *KR*, pages 112–122, 2006.
14. N. Creignou. The class of problems that are linearly equivalent to satisfiability or a uniform method for proving np-completeness. *Theor. Comput. Sci.*, 145(1&2):111–145, 1995.
15. R. Dechter, I. Meiri, and J. Pearl. Temporal constraint networks. In *KR*, pages 83–93, 1989.
16. C. Devred, S. Doutre, C. Lefèvre, and P. Nicolas. Dialectical proofs for constrained argumentation. In *COMMA*, pages 159–170, 2010.
17. Y. Dimopoulos, B. Nebel, and F. Toni. Preferred arguments are harder to compute than stable extensions. In *IJCAI'99*, pages 36–43, 1999.
18. Y. Dimopoulos, B. Nebel, and F. Toni. Finding admissible and preferred arguments can be very hard. In *KR'00*, pages 53–61, 2000.
19. P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence Journal*, 77:321–357, 1995.
20. P.M. Dung, P. Mancarella, and F. Toni. Computing ideal skeptical argumentation. *Artificial Intelligence Journal*, 171:642–674, 2007.
21. P. Dunne. Computational properties of argument systems satisfying graph-theoretic constraints. *Artificial Intelligence Journal*, 171 (10-15):701–729, 2007.
22. P. Dunne, A. Hunter, P. McBurney, S. Parsons, and M. Wooldridge. Inconsistency tolerance in weighted argument systems. In *AAMAS*, pages 851–858, 2009.
23. P. Dunne and M. Wooldridge. Complexity of abstract argumentation. *Chapter 5 of 'Argumentation in Artificial Intelligence' (Ed: I. Rahwan and G. Simari)*, pages 85–104, 2009.
24. U. Egly, S. Gaggl, and S. Woltran. ASPARTIX: Implementing argumentation frameworks using answer-set programming. In *International Conference on Logic Programming*, pages 734–738, 2008.
25. T. Feder. Network flow and 2-satisfiability. *Algorithmica*, 11(3):291–319, 1994.
26. E. Freuder. Temporal constraint networks. In *IJCAI*, pages 278–283, 1989.
27. M. Krom. The decision problem for a class of first-order formulas in which all disjunctions are binary. *eitschrift fr Mathematische Logik und Grundlagen der Mathematik*, 13:15–20, 1967.
28. V. Kumar. Depth-first search. *Encyclopaedia of Artificial Intelligence*, 2:1004–1005, 1987.
29. M. Mbarki, J. Bentahar, and B. Moulin. Specification and complexity of strategic-based reasoning using argumentation. *Argumentation in Multi-Agent Systems. Lecture Notes in Artificial Intelligence*, 4766:142–160, 2006.
30. B. Nadel. Some applications of the constraint satisfaction problem. *Technical report, Wayne state university*, 1990.
31. J-C. Régin. Generalized arc consistency for global cardinality constraint. In *Proceedings of the National Conference on Artificial Intelligence*, pages 209–215, 1996.
32. F. Rossi, P. van Beek, and T. Walsh. Handbook of constraint programming (foundations of artificial intelligence). *Elsevier Science Inc. New York, NY, USA*, 2006.
33. E. Tsang. *Foundations of Constraint Satisfaction.* Academic Press. ISBN 0-12-701610-4, 1993.