Weakly Complete Semantics Based on Undecidedness Blocking.

Pierpaolo Dondio, Luca Longo

School of Computer Science, Technological University Dublin
pierpaolo.dondio@tudublin.ie, luca.longo@tudublin.ie

Abstract

In this paper we introduce a novel family of semantics called weakly complete semantics. Differently from Dung’s complete semantics, weakly complete semantics employs a mechanism called undecidedness blocking by which the label undecided of an attacking argument is not always propagated to an otherwise accepted attacked argument. The new semantics are conflict-free, non-admissible but employing a weaker notion of admissibility; they allow reinstatement and they retain the majority of properties of complete semantics. We show how both weakly complete and Dung’s complete semantics can be generated by applying different undecidedness blocking strategies, making undecidedness blocking a unifying mechanism underlying argumentation semantics. The semantics are also an example of ambiguity blocking Dunganian semantics and the first semantics to tackle the problem of self-defeating attacking arguments. In the last part of the paper we compare weakly complete semantics with the recent work of Baumann et al. on weakly admissible semantics. Since the two families of semantics do not coincide, a principle-based analysis of the two approaches is provided. The analysis shows how our semantics satisfy a number of principles satisfied by Dung’s complete semantics but not by Baumann et al. semantics, including directionality, abstention, SCC-decomposability and cardinality of extensions, making them a more faithful non-admissible version of Dung’ semantics.

Keywords: Abstract Argumentation Semantics, Admissibility, Ambiguity blocking, Principle-based Analysis
1. Introduction

Abstract argumentation is a framework for non-monotonic reasoning where conclusions are reached by evaluating arguments and their conflict relation. The formalism is centred on the notion of argumentation framework \[1\], a directed graph where nodes represent arguments and links represent an attack relation defined over arguments. One of the main tasks of abstract argumentation is the computation of the acceptability status of the arguments composing the framework. This is performed by the application of an argumentation semantics which identifies the sets of arguments, called extensions, which successfully survive the conflicts encoded in the attack relation. In the labelling approach \[2\] adopted in this paper the effect of an argumentation semantics is to assign to each argument a label \textit{in}, \textit{out} or \textit{undec}. This means that an argument can respectively be accepted, rejected or deemed undecided. The \textit{undec} label represents a situation in which the semantics has no reasons to definitely accept or reject an argument.

This paper extends the studies of abstract argumentation by presenting a new family of abstract semantics, called \textit{weakly complete} semantics. While in Dung’s \textit{complete} semantics the effect of an attack by an undecided argument \(b\) to an otherwise accepted argument \(a\) is always to deem \(a\) undecided - therefore propagating the \textit{undec} label of \(b\) to \(a\) - in \textit{weakly complete} semantics the propagation of the \textit{undec} label could be blocked by the postulates of the semantics, and \(a\) could be accepted. We refer to this mechanism as \textit{undecidedness blocking}. We perform a principle-based analysis of such new semantics and a comparison with the \textit{weakly admissible} semantics recently proposed by Baumann et al. \[3\].

The motivations for these new semantics are two: (1) to model the fundamental reasoning mechanism of \textit{ambiguity blocking} absent in current abstract semantics and (2) to propose semantics able to handle the 25-year old problem of self-defeating attacking arguments, referred in this paper as the problem of \textit{weak attacks}.

Motivation 1: Ambiguity Blocking and Abstract Argumentation

\textit{Ambiguity blocking} semantics are employed by several non-monotonic formalisms, such as defeasible logic. Informally, \textit{ambiguity blocking} is a mechanism by which statements for which there are contradictory evidence about their validity are not used to derive any conclusion and they do not affect the validity of other statements. An ambiguity blocking mechanism is certainly
Figure 1: Motivating examples. Graph A is an example of floating assignment graph. Graph B shows the problem of self-defeating attacking arguments. Graph C is a variation of graph B where the attacking argument is part of odd-length cycle. In Graph D an attacker is added to the three-argument cycle of graph C.

the most appropriate in a legal dispute ([4]): if evidence versus an accused is not definitive or open to multiple interpretations, then evidence is void and the judge rules in favour of the accused. Referring to graph A in figure 1, using ambiguity blocking there is a case for argument b to be accepted, since b is attacked by the two arguments a and c that are conflicting and therefore ambiguous, and therefore a and c is rejected and b accepted. The ambiguity of a and c is not propagated to b and their attacks are de facto neglected. Note how the same mechanism could be used to build a case for b to be accepted in all the graphs of figure 1.

However, the translation of the ambiguity blocking mechanism into abstract argumentation is not straightforward. The notion of ambiguity is formalized in defeasible logic, a rule-based system less abstract than Dung’s framework. In defeasible logic knowledge is represented by rules over literals and inferences are obtained by chaining valid rules. An acyclic superiority relation is defined to model priorities among rules. In DL a literal is ambiguous if there is a valid chain of reasoning for a and a valid one for ¬a and the superiority relation does not solve the conflict. Both ambiguity blocking and ambiguity propagating semantics can be accommodated in DL. In the former ambiguous literals are rejected and their ambiguity is not transferred to other literals, while in the latter the ambiguity of the two literals is propagated to other related literals and rules. Previous researches ([5],[6]) highlighted how Dung’ semantics do not allow for an easy translation of the ambiguity blocking mechanism. Indeed, the notion of ambiguity in Dung’s framework is not even defined. Candidate to represent ambiguous arguments could be unde-
cided arguments or credulously accepted arguments but, as we discuss in a dedicated section of this paper, the concepts do not coincide. [6] showed how grounded semantics can indeed instantiate ambiguity propagating semantics but not ambiguity blocking.

A Dung-like version of ambiguity blocking has been sketched in the context of instantiating defeasible logics semantics and the Carneades argumentation systems into ASPIC+ [5], a structured argumentation framework based on Dung's semantics [7]. Instead of defining a new Dunganian semantics, the authors proposed a solution based on the introduction of additional arguments unidirectionally attacking the conflicting ambiguous arguments to reject both of them and replicate the ambiguity blocking behaviour. In [8] the author noted how this solution requires to introduce a second “attack” relation on arguments with a ripple down effects on the ASPIC+ definitions setting the various statuses of the argument.

Our approach differs substantially from previous attempts. Our aim is to introduce a new abstract semantics where the mechanism of ambiguity blocking is embedded in the postulates of the semantics rather than being mimicked by additional arguments. Our solution rely on connecting the notion of ambiguity blocking with undecidedness blocking. Indeed the concepts of undecidedness blocking and ambiguity blocking are similar but not equivalent and we will show how undecidedness encompasses a larger variety of situations besides ambiguity.

Motivation 2: The weak attacks problem: do arguments have to be defended against all the attacks?

The problem is illustrated in graph B of figure 1. It is the case of a self-defeating argument a attacking argument b. Recently, this example was analysed by Baumann et al. [3] [9], leading to the definition of novel abstract semantics based on a weaker notion of admissibility, referred here as BBU semantics. Dung already stated the problem ([1] p. 351): ‘An interesting topic of research is the problem of self-defeating arguments as illustrated in the following example (figure 1B). The only preferred extension here is empty though one can argue that since A defeats itself, B should be acceptable’.

The observation is extended to situations like the one in graph C in figure 1 where the attacker is not self-defeating but nevertheless it is part of an odd-length cycle of arguments collectively defeating themselves. The problem of self-defeating attacking arguments described so far is indeed strictly related to the problem of the propagation of undecidedness described in motivation
1. The general questions are the following:

Do arguments have to be defended against all attackers? When can we consider an attack weak enough to be neglected (blocked) instead of being considered effective (propagated)?

Those questions are rephrasing the concept of ambiguity propagation vs. ambiguity blocking present in other non-monotonic formalisms. In legal reasoning, the questions are answered by the different standards of proof.

Clearly, both the semantics proposed in this paper and the BBU semantics answer the above questions and they are both non-admissible semantics. A comparison is therefore important and it is detailed in section 6.3. We first claim how our semantics are the first to tackle the above problems. Indeed, the semantics presented in this paper expand our previous works in [10] [11] that chronologically precede BBU semantics by more than one year. In [10] [11] we introduced new semantics "based on a weaker notion of admissibility but retaining the majority of principles of complete semantics" ([11], pg. 4) to answer the above questions. We wondered what attacks are weak enough to be neglected. Our work moved from legal reasoning and standards of proof to the concept of undecidedness as a form of weaker attack, concept that we fully develop in this work. However in [10] some of the semantics presented here are already defined. Those semantics are able to handle correctly all examples of figure 1, therefore including also BBU’s motivating examples. In [12] we also sketched the weakly complete semantics that are developed in this work. Therefore, we claim that our work is indeed independent and prior to BBU semantics. In particular, the problem of weak attacks was first clearly answered by the authors of this paper.

Our answer to 'which arguments are weak enough to be neglected' is nevertheless different from BBU’s one. Both of the solutions remove the admissibility property of Dung’s semantics. However, the semantics do not coincide. Our undecidedness blocking semantics are able to handle all the motivating examples of figure 1, while BBU semantics do not address the problem of graph A, figure 1. The reader could object that graph A is not in the list of BBU motivating examples. However, our semantics allow for an interpretation by which graphs A and B of figure 1 are variations of the same situation: argument b is attacked by a self-conflicting argument in graph B, and it is attacked by two conflicting arguments in graph A. If we take a grounded semantics stance on the two rebutting arguments a and c of graph A, we could make the case that a and c are also self-defeating and b should be accepted, as it happens with ambiguity blocking semantics.
Substantial differences between the two families of semantics are revealed by a principle-based comparison detailed in section 6.3. The analysis shows how our semantics satisfy a number of principles satisfied by Dung’s complete semantics but not by Baumann et al. semantics, including directionality, abstention, SCC-decomposability and cardinality of extensions, making them a more faithful not admissible version of Dung’s semantics.

The paper is organized as follows. Section 2 provides the required background knowledge of abstract argumentation while section 3 is dedicated to the presentation of weakly complete semantics. The presentation of the new semantics is self-contained and it does not refer to ambiguity blocking mechanisms present in other formalisms. In Section 4 we present an algorithm to compute weakly complete semantics while in section 5 we introduce several families of weakly complete semantics. Section 6 presents a principle-based analysis of our semantics and a comparison with recent approaches. Section 7 contains related works to date and section 8 our future works and conclusions.

2. Abstract argumentation semantics

In this section we provide the concepts of abstract argumentation semantics required for the remaining of this paper.

Definition 2.1 An argumentation framework AF is a pair \( \langle Ar, R \rangle \), where \( Ar \) is a non-empty finite set whose elements are called arguments and \( R \subseteq Ar \times Ar \) is a binary relation, called the attack relation. If \( (a, b) \in R \) we say that \( a \) attacks \( b \). An argument is initial if it is not attacked by any arguments, including itself.

An abstract argumentation semantics identifies the sets of arguments that can survive the conflicts encoded by the attack relation \( R \). Dung’s semantics require a group of acceptable arguments to be conflict-free (an argument and its attackers cannot be accepted together) and admissible (the set of acceptable arguments has to defend itself from external attacks).

Definition 2.2 (conflict-free set). A set \( Arg \subseteq Ar \) is conflict-free iff \( \forall a, b \in Arg, (a, b) \notin R \).
Definition 2.3 (admissible set, complete set). A set \( \text{Arg} \subseteq \text{Ar} \) defends an argument \( a \in \text{Ar} \) iff \( \forall b \in \text{Ar} \) such that \( (b, a) \in \mathcal{R} \), \( \exists c \in \text{Arg} \) such that \( (c, b) \in \mathcal{R} \). The set of arguments defended by \( \text{Arg} \) is denoted \( \mathcal{F}(\text{Arg}) \). A conflict-free set \( \text{Arg} \) is admissible if \( \text{Arg} \subseteq \mathcal{F}(\text{Arg}) \) and it is complete if \( \text{Arg} = \mathcal{F}(\text{Arg}) \).

In this paper we follow the labelling approach of [2], where a semantics assigns to each argument a label \text{in}, \text{out} or \text{undec}.

Definition 2.4 (labelling). Let \( \mathcal{AF} = (\text{Ar}, \mathcal{R}) \). A labelling is a total function \( L : \text{Ar} \rightarrow \{\text{in}, \text{out}, \text{undec}\} \). We write \( \text{in}(L) \) for \( \{a \in \text{Ar} | L(a) = \text{in}\} \), \( \text{out}(L) \) for \( \{a \in \text{Ar} | L(a) = \text{out}\} \), and \( \text{undec}(L) \) for \( \{a \in \text{Ar} | L(a) = \text{undec}\} \).

Definition 2.5 (from [2]). Let \( \mathcal{AF} = (\text{Ar}, \mathcal{R}) \). A complete labelling is a labelling such that for every \( a \in \text{Ar} \) holds that:

1. if \( a \) is labelled in then all its attackers are labelled out;
2. if \( a \) is labelled out then it has at least one attacker that is labelled in;
3. if \( a \) is labelled undec then it has at least one attacker labelled undec and it does not have an attacker that is labelled in.

Definition 2.6 (complete labelling [2]) Given \( \mathcal{AF} = (\text{Ar}, \mathcal{R}) \), \( L \) is the grounded labelling iff \( L \) is a complete labelling where \( \text{undec}(L) \) is maximal (w.r.t. set inclusion) among all complete labellings of \( \mathcal{AF} \). \( L \) is the preferred labelling iff \( L \) is a complete labelling where \( \text{in}(L) \) is maximal (w.r.t. set inclusion) among all complete labellings of \( \mathcal{AF} \). A labelling \( L \) is stable iff \( \text{undec}(L) = \emptyset \).

Referring to Figure 2, there is only one complete labelling for \( G_1 \), representing also the grounded, preferred and stable labelling, where argument \( a \) is in (no attackers), \( b \) is out and \( c \) is in. Regarding \( G_2 \), grounded semantics assign the undec label to all the arguments. Regarding the preferred semantics, there are two complete labellings that maximise the \( \text{in}(L) \) set: one with \( \text{in}(L_1) = \{b, d\} \), \( \text{out}(L_1) = \{c, a, e\} \), \( \text{undec}(L_1) = \emptyset \) and the other with \( \text{in}(L_2) = \{a\} \), \( \text{out}(L_2) = \{b\} \), \( \text{undec}(L_2) = \{c, d, e\} \).

Besides the labelling approach, one can follow an extension-based approach, where a semantics identifies the sets of accepted argument, called extensions. The labelling and the extension-based approach are equivalent:
given an argumentation framework, for each extension identified by a semantics there is an equivalent set of in-labelled arguments and vice versa (see [2]). An argumentation framework $AF = \langle Ar, R \rangle$ identifies a directed graph. The following are some graph-based definitions needed in this paper.

**Definition 2.7** A (vertex-induced) subgraph of a graph $G = (Ar, R)$ is a graph $G_s = (S, R_s)$, where $S \subseteq Ar$ and $R_s = R \cap (S \times S)$.

A subgraph contains a subset of nodes of the original graph and any link whose endpoints are both in $S$ (note how this subgraph is usually called a vertex-induced subgraph). Given an argumentation framework $AF$, the restriction of an argumentation framework to a set of nodes $S$ is the framework $AF_{\downarrow S}$ corresponding to the vertex-induced subgraph of $AF$ identified by the nodes $S$.

**Definition 2.8** If $G$ is a graph, a strongly connected graph of $G$ is a subgraph of $G$ where, for each pair of nodes $a, b \in G$ there is at least one directed path from $a$ to $b$ and at least one directed path from $b$ to $a$. A strongly connected component (SCC) of $G$ is a maximal (with respect to set inclusion) strongly connected graph of $G$.

A cycle in a graph is a non-empty directed path in which the only repeated vertices are the first and last. Note how a node that is not part of a cycle is a strongly connected component. In this paper we call acyclic argument an argument that is not part of a cycle, and cyclic argument otherwise. Self-attacking arguments are cyclic arguments. We also refer to cyclic strongly connected component to identify a strongly connected component containing at least one cycle.
3. Weakly complete semantics

In this section weakly complete semantics are formally defined. Our aim is to define a semantics whose postulates control the propagation of the undecided label. We have already seen how in all Dung’s complete semantics the label undec is always propagated from the attacker to the attacked argument, unless the former is attacked by an accepted argument as well. Anyhow the attacked argument is never accepted. Weakly complete semantics employ a mechanism to block the propagation of the undecided label, so that an argument can be accepted even if attacked by undecided arguments. The main consequence is the loss of the admissibility property. As in complete semantics, an argument is rejected if and only if it has at least one in-labelled attacker. The definition of weakly complete semantics requires only a small modification of definition 2.5.

**Definition 3.1** Given an argumentation framework $AF = \langle Ar, R \rangle$, a weakly complete labelling is one such that for every $a \in Ar$ it holds that:

1. if $a$ is labelled in then there are no attackers of $a$ labelled in;
2. if $a$ is labelled out then it has at least one attacker that is labelled in;
3. if $a$ is labelled undec then it has at least one attacker labelled undec and it does not have an attacker that is labelled in.

The above definition changes condition 1 of definition 2.5 of Dung’s complete labelling by relaxing it, since now there is the option to accept an argument attacked by undecided arguments. These arguments would be always labelled undec in a complete labelling. In Table 1 we have broken down and named the conditions included in definition 3.1.

| Condition Name   | Description                                           |
|------------------|-------------------------------------------------------|
| 3.1a Admissiblity (ad) | $a$ is in if all its attackers are out                 |
| 3.1b Undecidedness Blocking (ub) | $a$ is in if at least one attacker is undec and all the others are out |
| 3.1c Undecidedness Propagating (up) | $a$ is undec if at least one attacker is undec and all the others are out |
| 3.1d Rejection (re) | $a$ is out if at least one attacker is in               |

Table 1: Conditions for weakly complete semantics

Weakly complete labellings generated without using the undecidedness blocking condition 3.1b are complete, and therefore all complete labellings are
**weakly complete.** Labellings whose at least one label satisfies condition 3.1b instead of condition 3.1c are non-admissible.

### 3.1. Examples of Weakly Complete Labellings

We present some examples of *weakly complete* labellings to familiarize with the new semantics introduced. Figure 3 displays five argumentation frameworks and their *weakly complete* labellings. Note how the first three graphs are also included in the motivating examples of figure 1. In the graph $G_3$ (a cycle of three arguments) the only valid *weakly complete* labelling is the *grounded* one where all the arguments are undecided. In $G_4$ there is an additional labelling besides the *grounded* one, where argument $b$ is accepted by blocking the undecidedness from $a$ to $b$. $G_5$ is an example of *floating assignment* graph. There are four *weakly complete* labellings. Besides the *grounded* labelling (all arguments are labelled undecided) there are two additional *preferred* labellings, in both of which argument $c$ is rejected. There is a new additional *weakly complete* labelling, where $a$ and $b$ are undecided and $c$ is accepted, therefore handling in the expected way our motivating example. $G_6$ will be discussed in details in the next section. $G_7$ shows an example of an argumentation graph consisting of a single strongly connected component. Besides the *grounded* labelling, there is also a non-admissible *weakly complete* one, in which the cycle $a, b, c$ is a source of unresolved conflict and labelled *undec*, but the *undecidedness blocking* condition is applied to the attack from $c$ to $d$ to generate a valid labelling where $d$ is accepted and $e$ rejected.

We already observed that *weakly complete* labellings generated without using the *undecidedness blocking* condition 3.1b are *complete*, while *weakly complete* labelling using condition 3.1b instead of condition 3.1c are non-admissible. In general there are *weakly complete* labelling where a combination of conditions 3.1b and 3.1c is used. These are the labellings where the undecided label is propagated to some parts of the argumentation graph but not to all of them. These labellings represent interesting cases, where an agent might grant *undecidedness blocking* to some arguments, preventing them to be labelled undecided, but not to others. An example is shown in Figure 4. The figure shows three labellings of the same argumentation framework. The first is the *grounded* labelling where the undecided label is always propagated, the other two are *weakly complete* ones. In the labelling on the centre the *ub*-condition 3.1b is applied *earlier* to the attack from $b$ to $d$, while in the labelling on the right the condition is applied to the attack from $d$ to
$e$, but not to the attack from $b$ to $d$, where the undecidedness propagating condition 3.1c is used instead.

4. Computing Weakly Complete Semantics

In this section we propose an algorithm to compute weakly complete semantics, firstly introducing some preliminary notations. Given an argumentation framework $AF = \langle Ar, R \rangle$, we call $\{a\}^- = \{x \in Ar \mid (x, a) \in R\}$ the set of arguments in $AF$ attacking argument $a$ and $\{a\}^+ = \{x \in Ar \mid (a, x) \in R\}$ the set of arguments in $AF$ attacked by argument $a$. We start by observing that, as for complete semantics, a weakly complete labelling $\mathcal{L}$ is fully identified by the set $\text{in}(\mathcal{L})$. 

Figure 3: Examples of weakly complete labellings
Proposition 4.1 Let us consider $AF = \langle Ar, R \rangle$ and a weakly complete labelling $\mathcal{L}$ of $AF$. Then, the sets $\text{out}(\mathcal{L})$ and $\text{undec}(\mathcal{L})$ are uniquely identified by $\text{in}(\mathcal{L})$. It is $\text{out}(\mathcal{L}) = \{ a \in Ar | \exists b \in Ar \land R(b, a) \land \mathcal{L}(b) = \text{in} \}$.

Proof. We label all the arguments of $AF$ with a label unk (=unknown). Then, we assign the label in to the arguments in $\text{in}(\mathcal{L})$ and for the rejection condition 3.1d we necessarily assign the label out to the arguments attacked by at least one argument in $\text{in}(\mathcal{L})$. Since $\text{in}(\mathcal{L})$ is conflict-free, none of the arguments in $\text{in}(\mathcal{L})$ is changed to the label out. The remaining unk arguments cannot be labelled out since they do not have any attacker in $\text{in}(\mathcal{L})$ and they have at least one attacker labelled unk, otherwise they would be labelled in and therefore part of $\text{in}(\mathcal{L})$. Therefore the label undec is the only label that can be assigned to the unk arguments. We have to verify that the resulting labelling is valid. Since unk arguments are attacked by at least one unk argument and arguments labelled out, by changing the unk label to undec, each undecided argument results attacked by at least one undecided argument (previously unk) and by out-labelled arguments, and therefore the undecided labels are valid for the condition 3.1d of weakly complete semantics. Regarding the arguments labelled in, their attackers are either out or undec, and therefore the labels in are valid in a weakly complete labelling for the undecidedness blocking condition 3.1b. □

Under both complete and weakly complete semantics, assigning the label in to an argument $a$ imposes constraints on the label of other connected arguments. It imposes that all the arguments $\{a\}^+$ are labelled out. As a consequence, if an argument $b$ has now all its attackers in $\{a\}^+$ it can be labelled in. In turn, this imposes to all the arguments in $\{b\}^+$ the label out and so forth, stopping when no more labels can necessarily be imposed on arguments or an inconsistency is found when trying to label the same argument.
twice with different labels. The reasoning can be extended to a set of arguments $Gr \subseteq Ar$ instead of a single argument. Assigning the label $\text{in}$ to all the arguments in $Gr$ imposes similar constraints. This forward propagation of the $\text{in}$ and $\text{out}$ labels is formalized in the $\text{in-out-fw}$ algorithm.

**Algorithm 1:** The $\text{in-out-fw}$ algorithm

```
1 Inputs: $AF = (Ar, R)$, $L$ is a labelling of $AF$, $Gr \subseteq \text{undec}(L) \subseteq Ar$;
2 Outputs: an updated labelling $L$ or inconsistency;
3 for $g \in Gr$ do
4     $L(g) \leftarrow \text{in};$
5 end
6 repeat
7     $g \leftarrow \text{pop}(Gr);$ 
8     $L(g) \leftarrow \text{in};$
9     for $x \in \{g\}^+$ do
10        if $L(x) = \text{in}$ then
11            return inconsistency;
12        else
13            $L(x) \leftarrow \text{out};$
14        end
15     $Gr \leftarrow \{x \in Ar \mid L(x) = \text{undec} \land \forall a \in \{x\}^- \ L(a) = \text{out} \} ;$
16 until $Gr = \emptyset$
17 return $L$
```

The inputs of the $\text{in-out-fw}$ algorithm are an argumentation framework $AF$, a set of arguments $Gr \subseteq Ar$ and a labelling $L$ of $AF$. The output is either an inconsistency error or an updated version of $L$. The algorithm first assigns the label $\text{in}$ to all the arguments $a \in Gr$. Then, for each arguments $a$ in $Gr$, the label $\text{out}$ is assigned to the arguments in $\{a\}^+$ and, as a consequence, the arguments those attackers are all labelled $\text{out}$ are added to the set $Gr$ of new arguments that could be promoted to the label $\text{in}$. A new argument in $Gr$ is selected and the labelling process is repeated until $Gr$ is empty or we found an inconsistency. In case of inconsistency all the modifications to $L$ are discarded.

Given a labelling $L$ of $AF$, we refer to the application of the algorithm using the set of argument $Gr$ as $\text{in-out-fw}(Gr, L)$. Each argument in $Gr$ is called a ground of the algorithm.

Since the algorithm uses only the admissibility (3.1a) and rejection (3.1d) conditions, all the labels assigned by the $\text{in-out-fw}$ algorithm (excluding the
one assigned to the ground argument) are valid for both complete and weakly complete semantics. Moreover all the labels assigned by the in-out-fw(Gr, L) algorithm follow necessarily form accepting the ground arguments in Gr and the labels assigned in L. If an inconsistency is found, this means that there is no weakly complete labelling where the arguments in Gr are accepted together with the arguments in in(L); at least one argument in Gr or in in(L) cannot be accepted. On the contrary, if L’ = in-out-fw(Gr, L) ends without an inconsistency, if L was not a complete or weakly complete labelling we are not guaranteed that L’ is complete or weakly complete.

In this section we show how any weakly complete labelling can be obtained by repeatedly applying the in-out-fw algorithm. Therefore, we start by providing some properties of the labellings generated by the in-out-fw algorithm needed to prove the main results of this section. Given AF = ⟨Ar, R⟩, we call Lund the labelling function assigning to each argument in Ar the label undec. The following is a corollary of Proposition 4.1:

Corollary 4.1 Given AF = ⟨Ar, R⟩ and a weakly complete labelling W of AF, it is W = in-out-fw(in(W), Lund).

Proof. The algorithm in-out-fw(in(W), Lund) assigns the label in to all the arguments in in(W) and out to all the arguments attacked by an argument in in(W). Since W is conflict-free, none of the arguments in in(W) are labelled out. Then the algorithm stops since the remaining undecided arguments are at least attacked by an undecided argument. Indeed, if an undecided argument a is attacked only by out-labelled arguments, a should necessarily be in in(W), and therefore W is not a valid weakly complete labelling since if a is undec or out, the admissibility property would be violated and W would not be a weakly complete labelling either. By Proposition 4.1, the labelling found is W. □

The following straightforward property is useful in the remaining discussion.

Proposition 4.2 Each application of the in-out-fw algorithm does not change the labels of arguments previously labelled in or out.

Proof. The in-out-fw algorithm cannot change a label of an argument previously labelled in, since that would generate an inconsistency. Neither it cannot change the label of an argument a previously labelled out, since that
would require to change the label in of one or more attackers of a. □

An interesting property of the in-out-fw algorithm is that it can be applied to a set of arguments Gr or sequentially to each argument in Gr and the final labelling is the same, since the algorithm generates the same set of constrains on the labels of the arguments in AF. The order of the arguments in Gr is also irrelevant. Indeed, each consistent application of the in-out-fw algorithm does not change the previously labelled arguments, and arguments requiring constraints from more than one ground arguments in order to be labelled are labelled only when the last of the required ground argument’s in-out-fw is executed, independently from the order of execution. For example, an argument defended by the set of arguments \{a,b\} can be labelled in only when both in-out-fw(a) and in-out-fw(b) are executed. We formally express this property of the in-out-fw algorithm:

**Property 4.1** Given a framework \(AF = \langle Ar, R \rangle\) and its labellings \(L_{ab} = \text{in-out-fw}\{a, b\}, L_{und}\), \(L_a = \text{in-out-fw}\{a\}, L_{und}\), \(L_b = \text{in-out-fw}\{b\}, L_{und}\). If \(L_{ab} \neq L_{und}\) then \(L_{ab} = L_a = L_b\).

The above proposition is true only if in-out-fw(\{a, b\}, \(L_{und}\)) ends without an inconsistency (\(L_{ab} \neq L_{und}\)). For instance if we consider the framework \(G_5\) and the set \{a, c\}, it is in-out-fw(\{a, c\}, \(L_{und}\)) = \(L_{und}\) but if we apply in-out-fw frist to a and then to c we obtain the labelling \{\{a\}, \emptyset, \{b, c\}\} and if we apply in-out-fw first to c and then to a we obtain the labelling \{\{c\}, \emptyset, \{a, b\}\}.

The following proposition shows that if \(L = \text{in-out-fw}(A, L_{und})\) does not generate an inconsistency, then all the arguments accepted in \(L\) are also accepted in any weakly complete labelling accepting the set of arguments \(A\).

**Proposition 4.3** Given a framework \(AF = \langle Ar, R \rangle\), let us consider the labelling of \(AF\) \(L = \text{in-out-fw}(A, L_{und})\). If \(W\) is a weakly complete labelling with \(A \subseteq \text{in}(W)\), then \(\text{in}(L) \subseteq \text{in}(W)\).

**Proof.** According to property 4.1, we can build the labelling \(W\) sequentially starting from \(L_{und}\) and using all the arguments in \(\text{in}(W)\). Starting from \(L_{und}\), we applying the in-out-fw algorithm to all the arguments in \(A\), obtaining \(L\). We then apply the in-out-fw algorithm to all the arguments in \(\text{in}(W) \setminus A\) to obtain \(W\). None of the applications of the in-out-fw can generate an incon-
sistency (otherwise the ground argument generating the inconsistency would not be in in(W), a contradiction) and since no application of the \texttt{in-out-fw} algorithm can change the labels in and out previously assigned (Proposition 4.2), it is in(L) \subseteq in(W). $\square$

We can prove the following proposition useful in the remaining discussion:

**Proposition 4.4** Given $AF = \langle Ar, R \rangle$, let us consider its labelling $L_{gr} = \texttt{in-out-fw}(Gr, L_{und})$. We consider $g \in Ar \setminus Gr$. If $\texttt{in-out-fw}(\{g\}, L_{gr})$ generates an inconsistency, the inconsistency is generated when trying to label out the argument $g$ or an argument in the set of ground Gr.

*Proof.* We observe that in $L_{gr}$ the arguments previously used as ground (set Gr) that did not generate an inconsistency are labelled in and they might have some attacking arguments labelled undec. Differently, the labels of the arguments not in Gr follows the admissibility or rejection condition: if they are labelled out they have at least one attacker labelled in and if they are labelled in they have all their attackers labelled out.

We prove that an inconsistency cannot be generated when the \texttt{in-out-fw} algorithm labels an argument $a \notin Gr \cup g$. When the ground $g$ is promoted from undec to in, arguments in $\{g\}^+$ are changed to the label out. An inconsistency is generated only if an argument $a$ in $\{g\}^+$ had $L_{gr}(a) = \text{in}$. However, if an argument $a$ in $\{g\}^+$ is not in $Gr \cup g$, an inconsistency cannot be generated since $L_{gr}(a) \neq \text{in}$; since $a \in \{g\}^+$ and therefore $a$ is attacked by $g$ that was undecided in $L_{gr}$. Argument $a$ was either labelled out because of the effect of an in-labelled attacker distinct from $g$ or labelled undec otherwise. Therefore no inconsistency can be generated when trying to label arguments not in $Gr \cup g$.

The effect of labelling out the arguments in $\{g\}^+$ could be to promote some arguments in $\{g\}^{++} = \bigcup \{h\}^+, h \in \{g\}^+$ to the label in. Those arguments were labelled in or undec in $L_{gr}$ but not out, since arguments in $\{g\}^{++}$ are attacked by arguments in $\{g\}^+$ that were not labelled in in $L_{gr}$ as shown above.

Changing the label of an argument $a \in \{g\}^{++}$ to in generates the same constraints we discussed when the label in was assigned to the ground argument $g$, concluding that if at each step the attacked arguments $\{a\}^+$ are not in $Gr \cup g$, no inconsistency can be generated. We also observe that only the labels of arguments undecided in $L_{gr}$ are changed by the effect of \texttt{in-out-fw}. 

Therefore, the only way to generate an inconsistency is to have an argument \( x \) with \( \mathcal{L}_{gr}(x) = \text{in} \) that had at least one undecided attacker in \( \mathcal{L}_{gr} \) that is changed to the label \( \text{in} \) by \( \text{in-out-fw}(g, \mathcal{L}_{gr}) \). Since \( \mathcal{L}_{gr}(x) = \text{in} \) and \( x \) had at least an undecided attacking argument in \( \mathcal{L}_{gr} \), \( x \) must be in \( \text{Gr} \cup g \) and therefore the thesis is proven. □

Given \( AF = \langle Ar, R \rangle \), we call \( I(AF) = \{ a \in Ar | \# b \in Ar \wedge R(b, a) \} \) the set of initial arguments of \( AF \). We consider the labelling \( \mathcal{L}_I \) of \( AF \) obtained by \( \text{in-out-fw}(I(AF), \mathcal{L}_{und}) \).

According to Proposition 4.4, \( \text{in-out-fw}(I(AF), \mathcal{L}_{und}) \) cannot generate an inconsistency, since the ground arguments are in \( I(AF) \) and therefore they cannot be labelled twice by the \( \text{in-out-fw} \) algorithm. Therefore \( I(AF) \subseteq \text{in}(\mathcal{L}_I) \). We prove that \( \mathcal{L}_I \) is the grounded labelling of \( AF \).

**Proposition 4.5** Let us consider \( AF = \langle Ar, R \rangle \) and its grounded labelling \( \mathcal{G}_{AF} \). Then \( \mathcal{L}_I = \mathcal{G}_{AF} \).

*Proof.* We observe that the labelling \( \mathcal{L}_I \) is a Dung’s complete labelling. All the initial arguments are labelled \( \text{in} \) and all the other labels are assigned by the \( \text{in-out-fw} \) algorithm and therefore they all satisfy the constrains of definition 2.5. In the following we exploit the fact that the grounded semantics is the complete semantics minimizing the set of \( \text{in} \)-labelled arguments.

1) We prove that \( \text{in}(\mathcal{L}_I) \subseteq \text{in}(\mathcal{G}_{AF}) \). Since \( \mathcal{L}_I = \text{in-out-fw}(I(AF), \mathcal{L}_{und}) \) and \( I(AF) \subseteq \text{in}(\mathcal{G}_{AF}) \), then for Proposition 4.4 it is \( \text{in}(\mathcal{L}_I) \subseteq \text{in}(\mathcal{G}_{AF}) \).

2) We prove that \( \text{in}(\mathcal{G}_{AF}) \subseteq \text{in}(\mathcal{L}_I) \) By contradiction, if \( \text{in}(\mathcal{G}_{AF}) \not\subseteq \text{in}(\mathcal{L}_I) \), then \( \mathcal{G}_{AF} \) is not the complete labelling minimizing the set of \( \text{in} \)-labelled argument w.r.t. set inclusion and therefore \( \mathcal{G}_{AF} \) is not the grounded labelling of \( AF \), a contradiction. □

We also prove the following theorem, showing that every weakly complete extension is a super-set of the grounded semantics.

**Theorem 4.1** Let us consider \( AF = \langle Ar, R \rangle \). For each weakly complete labelling \( \mathcal{W} \) of \( AF \), it holds that \( \text{in}(\mathcal{G}_{AF}) \subseteq \text{in}(\mathcal{W}) \), where \( \mathcal{G}_{AF} \) is the grounded labelling of \( AF \).
Proof. We notice that every \textit{weakly complete} labelling $\mathcal{W}$ accepts all the initial arguments of the framework, and therefore $I(AF) \subseteq in(\mathcal{W})$. Since $G_{AF} = \mathcal{L}_I = in-out-fw(I(AF), \mathcal{L}_{und})$ and $I(AF) \subseteq in(\mathcal{W})$, then for Proposition 4.4 it is $in(G_{AF}) \subseteq in(\mathcal{W})$. □

The following proposition is the mechanism to compute any \textit{weakly complete} labelling.

\textbf{Proposition 4.6} Given an argumentation framework $AF = \langle Ar, R \rangle$ and a \textit{weakly complete} labelling $\mathcal{W}_{AF}$, if argument $g \in Ar$ is labelled $\text{undec}$ in $\mathcal{W}_{AF}$ and the $\text{in-out-fw}\{g\}, \mathcal{W}_{AF}$ algorithm ends without generating an inconsistency, than the resulting labelling $\mathcal{W}_{g}^{AF}$ is a valid \textit{weakly complete} labelling.

Proof. By Proposition 4.2, the labelling $\mathcal{W}_{g}^{AF}$ is such that $in(\mathcal{W}_{AF}) \subseteq in(\mathcal{W}_{g}^{AF})$. The new labels $in$ or $out$ in $\mathcal{W}_{AF}$ that were $\text{undec}$ in $\mathcal{W}_{AF}$ are a consequence of the $\text{in-out-fw}\{g\}, \mathcal{W}_{AF}$ algorithm. Therefore the new labels are a necessary consequence of accepting the labelling in $\mathcal{W}_{AF}$ (valid by hypothesis) and accepting $g$. Therefore we just need to prove that the label in assigned to the \textit{ground} argument $g$ in $\mathcal{W}_{g}^{AF}$ is valid in a \textit{weakly complete} labelling. Since $g$ was labelled $\text{undec}$ in $\mathcal{W}_{AF}$, at least one of its attackers was $\text{undec}$ and none of them $in$. The $\text{in-out-fw}\{g\}, \mathcal{W}_{AF}$ algorithm might change some or all the undecided attackers of $g$ to $out$, but not to $in$, otherwise an inconsistency would be found violating our hypothesis. Therefore after the application of $\text{in-out-fw}\{g\}, \mathcal{W}_{AF}$ all the attackers of $g$ are either labelled $out$ or $\text{undec}$ and therefore for the \textit{ub-condition} 3.1b the label in assigned to $g$ is valid in a \textit{weakly complete} labelling. □

According to propositions 4.6 and 4.2, every consistent application of the $\text{in-out-fw}$ algorithm finds a new valid \textit{weakly complete} labelling and the set of $in$-labelled arguments of each new labelling strictly contains the previous one, since the new application of the $\text{in-out-fw}\{g\}$ does not change the previously assigned labels (Proposition 4.2) and it adds (at least) $g$ to the set of $in$-labelled arguments.

Proposition 4.6 also solves the credulous acceptance problem for \textit{weakly complete} semantics. Given an argument $a$, the problem is whether there is at least one valid \textit{weakly complete} labelling where $a$ is accepted. In order to check this, the \textit{grounded} labelling $G_{AF}$ can be computed first. If the
argument is accepted or rejected by grounded semantics, the credulously acceptance of \( a \) is decided, since all weakly complete labellings include \( \mathcal{G}_{AF} \). For the arguments left undecided in \( \mathcal{G}_{AF} \), the credulous acceptance can be checked by computing the \( \text{in-out-fw}(\{a\}, \mathcal{G}_{AF}) \) and by verifying if the algorithm returns an inconsistency. If not, by Proposition 4.6 we have found a valid weakly complete labelling where \( a \) is accepted.

**Theorem 4.2.** Given \( AF = \langle Ar, R \rangle \) and its grounded labelling \( \mathcal{G}_{AF} \), let us consider an argument \( a \) so that \( \mathcal{G}_{AF}(a) = \text{undec} \). Then \( a \) is credulously accepted by weakly complete semantics iff \( \text{in-out-fw}(\{a\}, \mathcal{G}_{AF}) \) does not generate an inconsistency.

**Proof.**

\( \leftarrow \) If the \( \text{in-out-fw}(\{a\}, \mathcal{G}_{AF}) \) algorithm ends without inconsistency then for Proposition 4.6 there is a valid weakly complete labelling containing argument \( a \) and therefore \( a \) is credulously accepted.

\( \rightarrow \) By contradiction, we prove that if \( \text{in-out-fw}(\{a\}, \mathcal{G}_{AF}) \) generates an inconsistency, then \( a \) is not credulously accepted. If \( \text{in-out-fw}(\{a\}, \mathcal{G}_{AF}) \) generates an inconsistency, this means that assigning the label \( \text{in} \) to argument \( a \) necessarily generates a contradiction with an argument labelled \( \text{in} \) by \( \mathcal{G}_{AF} \). Therefore in any labelling \( \mathcal{L}_a \) where \( a \) is accepted at least one argument in \( \text{in}(\mathcal{G}_{AF}) \) has to be rejected and therefore \( \text{in}(\mathcal{G}_{AF}) \not\subseteq \text{in}(\mathcal{L}_a) \). However, since for all weakly complete labellings \( W \) it is \( \text{in}(\mathcal{G}_{AF}) \subseteq \text{in}(W) \), then \( \mathcal{L}_a \) is not a weakly complete labelling and \( a \) is not credulously accepted. \( \square \)

The following Proposition provides the computational class of the credulous acceptance decision problem of weakly complete semantics.

**Proposition 4.5.** The computational class of the credulous acceptance problem of weakly complete semantics is \( \text{PTIME} \).

**Proof.** If argument \( a \) is labelled \( \text{in} \) or \( \text{out} \) by the grounded semantics, the decision problem is solved in polynomial time since this is the computational class of the grounded semantics. If \( a \) is left undecided by grounded semantics, then it is sufficient to compute \( \text{in-out-fw}(\{a\}, \mathcal{G}_{AF}) \) that performs a graph visit of all the nodes reachable from \( a \), and this visit has a complexity of \( \text{PTIME} \) as well. \( \square \)
We stress the difference with the credulous acceptability problem of complete semantics, that is $NP$-complete. Under complete semantics, the constraint of admissibility requires an accepted argument $a$ to have all its attackers labelled out. An application of the $in-out-fw(\{a\})$ algorithm might not guarantee it, and testing the credulous acceptance in general would require to apply $in-out-fw$ to other arguments that are supposed to defend $a$, potentially generating a combinatorial number of applications of the algorithm. On the contrary, under weakly complete semantics we do not need to check if all the attackers of $a$ are out, since it is sufficient that $in-out-fw(\{a\})$ does not change one of the attackers of $a$ to in. The credulous acceptance of $a$ is therefore responsibility of argument $a$ only, it does not depend on other arguments that might be required to defend $a$.

We now describe our algorithm to compute weakly complete semantics, consisting in the repeated application of Proposition 4.6 on different sequences of arguments. The ground-based algorithm for weakly complete labelling is described below.
Algorithm 2: The ground-based algorithm

1. **Input:** $AF = \langle Ar, R \rangle$
2. **Outputs:** $W_I = \text{set of weakly complete labellings of } AF$

3. $L \leftarrow \text{Grounded}(AF)$
4. $W_I \leftarrow W_I \cup L$
5. $I \leftarrow \{a \in Ar \mid \nexists x \in Ar : R(x, a) \in R \}$
6. $V \leftarrow \emptyset$
7. `compute($W_I, I, V$)

8. **function compute($L, Gr, V$)**
   9. $\text{Candidates} \leftarrow \{x \in Ar \mid L(x) = \text{undec} \land x \notin V \}$
   10. **for** $g \in \text{Candidates}$ **do**
       11. $L' \leftarrow \text{in-out-fw}(\{g\}, L)$
       12. $V \leftarrow V \cup \{g\}$
       13. **if** $L' \neq \text{inconsistency}$ **then**
           14. $W_I \leftarrow W_I \cup L'$
           15. $Gr \leftarrow Gr \cup \{g\}$
           16. `compute($L', Gr, V$)
       17. **end**
   18. **end**

   The algorithm returns $W_I$, the set of all the weakly complete labellings of an argumentation framework. For each labelling $L$, the variable $Gr$ represents the sequence of arguments used as ground in the successive calls of the in-out-fw algorithm used to generate $L$.

First, the algorithm computes grounded semantics and add the initial arguments of the framework to the set of grounds $Gr$. Then, it calls a recursive function `compute`. The function tries to apply the in-out-fw algorithm to each undecided argument $g$. If the in-out-fw($\{g\}$) algorithm does not generate inconsistency, a new labelling is found, $g$ is added to the list of grounds $Gr$ for the new labelling and the algorithm recursively calls itself on the labelling just identified. If an inconsistency is generated, the recursive branch is terminated. The algorithm terminates when there are no more undecided arguments to try. The variable $V$ is used to store the list of nodes already used as ground in each recursive step. We describe the functioning of the ground-based algorithm with the following example.
Example 4.1 Let us consider figure 5, that shows the application of the ground-based algorithm to the graph $G_6$. Table 2 shows the five labellings found. Note how the set of initial arguments for the graph $G_6$ is empty.

| Labelling | Grounds | Comment         |
|-----------|---------|-----------------|
| $L_1$     | $\emptyset$ | Grounded Labelling |
| $L_2$     | $\{a\}$ | Complete        |
| $L_3$     | $\{b\}$ | Complete        |
| $L_4$     | $\{d\}$ | Non complete labelling |
| $L_5$     | $\{b,d\}$ | Same as $L_3$   |

Table 2: Weakly complete labellings for graph $G_6$.

After finding the grounded labelling ($L_1$) with all the arguments undecided, the algorithm selects as a ground the argument $a$ and in-out-fw($\{a\}, L_1$) algorithm finds the labelling $L_2$. All the attempts to extend $L_2$ fail, since selecting as ground argument $c$, $d$ or $e$ lead to an inconsistency generated by the cycle of three arguments. This ends this recursive branch of the execution. The algorithm now selects $b$ as a ground and it finds $L_3$, a stable labelling with no undecided arguments that therefore ends this branch of the computation. Any attempt to select as ground $c$ or $e$ fails due to the inconsistency of the odd-length cycle. The selection of $d$ as a ground generated the labelling $L_4$, that is weakly complete but not complete. Attempts to extend $L_4$ using $a$ as ground fails, since $b$ becomes out, $c$ in and it conflicts with $d$. Also the choice of $c$ fails for the same conflicts. Choosing $b$ as a ground generates another valid labelling $L_5$. This is the same labelling as $L_3$, showing that the ground $b$ was enough to identify this labelling.

The following theorem proves that the algorithm is correct.

**Theorem 4.3** Given $AF = \langle Ar, R \rangle$, the ground-based algorithm returns all and only the weakly complete labellings of $AF$.

**Proof.** $\rightarrow$ Any labelling found by the algorithm is weakly complete. This is proved using Proposition 4.6, since each labelling found by the algorithm is generated by consistent applications of the in-out-fw algorithm starting from a valid weakly complete labelling, using the grounded labelling as the
starting labelling.

← Let us consider a valid weakly complete labelling $W_{AF}$. By proposition 4.1 we saw that $W_{AF} = \text{in-out-fw}(W_{AF}, L_{und})$.

We show that the labelling identified by $\text{in-out-fw}(W_{AF}, L_{und})$ is included in the output of the ground-based. Since $\text{in-out-fw}(W_{AF}, L_{und})$ does not generate an inconsistency since $W_{AF}$ is a valid weakly complete labelling, we can also compute it by applying the $\text{in-out-fw}$ sequentially starting from $L_{und}$. We first use as grounds all the initial arguments, obtaining the grounded labelling, and then all the arguments that are in $\text{in}(W_{AF}) \setminus \text{in}(G_{AF})$ obtaining $W_{AF}$. Note how some of the arguments in $\text{in}(W_{AF}) \setminus \text{in}(G_{AF})$ might not be used for the $\text{in-out-fw}$ algorithm since they could have been labelled in by the effect of other ground arguments used before them. In this case they are skipped since their constrains have been already generated by previous arguments. Note how these arguments cannot be labelled out by previous application of $\text{in-out-fw}$, otherwise $W_{AF} \not\models \text{in-out-fw}(\text{in}(W_{AF}), L_{und})$, a
contradiction. □

5. Families of weakly complete semantics

In this section several families of weakly complete semantics are defined. The first three semantics are defined in a similar way in which families of complete semantics are defined, i.e. by identifying subset of semantics whose set of in- or undec-labelled arguments satisfy maximality or minimality conditions. The last two semantics are called undecidedness blocking semantics.

We have seen how several weakly complete labellings can be generated by deciding when and how often to use the ub-condition 3.1b instead of the undecidedness propagating (up-condition) 3.1c. In the undecidedness blocking semantics the use of the ub-condition 3.1b is constrained by the postulates of the semantics. These semantics mimic the behaviour of ambiguity blocking semantics of defeasible logic, where ambiguity is always blocked and never propagated. The first semantics, called undecidedness blocking preferred semantics, is the semantics where the ub-condition is used as much as possible and as earlier as possible. The second, called undecidedness blocking grounded semantics, is a single-status non-admissible variant of grounded semantics retaining the majority of properties of its Dung’s counterpart.

5.1. Weakly preferred semantics

Definition 5.1 Given an argumentation framework $AF = \langle Ar, R \rangle$, the weakly preferred labelling of $AF$ is the weakly complete labelling where the set of in-labelled arguments is maximal w.r.t. set inclusion.

We have seen how the set of in-labelled arguments increases strictly monotonically each time the in-out-fw algorithm is applied without inconsistency. Since each application of the in-out-fw(\{a\}) corresponds to an application of the ub-condition 3.1b on argument $a$, a weakly preferred labelling is the labelling where the ub-condition is used as much as possible: no more undecided arguments can be promoted to the label in without causing an inconsistent application of the in-out-fw algorithm. In terms of the ground-based algorithm, the weakly preferred labellings are the ones identified by the terminal nodes of each recursive branch. In the graph $G_6$ of figure 5, the weakly preferred labellings are $L_2$ and $L_3 = L_5$, that are the terminal nodes of the recursion.
5.2. Weakly Grounded semantics

**Definition 5.2** Given an argumentation framework $AF = \langle Ar, R \rangle$, the weakly grounded labelling of $AF$ is the weakly complete labelling where the set of $\text{undec}$-labelled arguments is maximal w.r.t. set inclusion.

Since every weakly complete labelling $W$ is found by applying the $\text{in-out-fw}$ algorithm to Dung’s grounded labelling and, and since at each subsequent consistent application of the $\text{in-out-fw}$ algorithm the set $\text{in}(W)$ is strictly increased and therefore $\text{undec}(W)$ is reduced, the weakly grounded semantics is Dung’ grounded semantics. In terms of the ground-based algorithm, the grounded semantics is obtained by using as ground arguments the initial arguments of the framework (Proposition 4.4).

5.3. Weakly Stable Semantics

**Definition 5.3** Given an argumentation framework $AF = \langle Ar, R \rangle$, the weakly stable labelling of $AF$ is the weakly complete labelling where the set of $\text{undec}$-labelled arguments is empty.

The weakly stable labelling coincides with Dung’s stable labelling. Note how a stable labelling is always a weakly preferred labelling: since no undecidedness is present in the labelling the $\text{ub-condition}$ cannot be used further.

5.4. The undecidedness-blocking semantics

Undecidedness blocking semantics ($\text{ub}$-semantics) are weakly complete semantics where the undecidedness blocking condition is used as much as possible and as earlier as possible. As it happens in the weakly preferred semantics, saying that the $\text{ub-condition}$ is used as much as possible means that it cannot be used further. However, this does not guarantee that the undecidedness is blocked as earlier as possible. Figure 6 shows two weakly complete labelling for a graph (a third valid labelling exists, and it is the grounded labelling). The source of undecidedness is the self-attacking argument $a$. Indeed, in the labelling on the left undecidedness is blocked at argument $b$, earlier than in the labelling on the right, where it is blocked at argument $c$. However, both of them are weakly preferred labelling, since both of them maximise the set of in-labelled arguments.

The concept of as earlier as possible has to consider the topological order of the arguments of the argumentation framework. In $\text{ub}$-semantics, a sufficient condition to block undecidedness is the following: if argument $a$
has an attacker $b$, and there is no directed path back from $a$ to $b$, then in all the labellings where $b$ is undecided the undecidedness blocking condition 3.1b is always applied to $a$ and the attack from $b$ is de facto neglected. In absence of other $\text{in}$-labelled attackers, $a$ is therefore labelled $\text{in}$. This leaves open the problem of how to block undecidedness as earlier as possible inside a cyclic strongly connected component where no topological order is defined. Different strategies lead to different variants of $\text{ub}$-semantics.

The importance of the topological order makes convenient to model $\text{ub}$-semantics using an instance of the SCC-recursive schema [13], that we quickly describe. We consider the graph $G_{\text{SCC}}$ composed by the strongly connected components of the original graph $G$. $G_{\text{SCC}}$ is a direct acyclic graph for which it is possible to define a topological order. Using the SCC-recursive schema, the extensions of a semantics are computed following the topological order of the $G_{\text{SCC}}$ graph. Initial strongly connected components are labelled using a base function $B$. A non-initial strongly connected component $S$ is labelled considering external attacks from argument belonging to SCCs preceding $S$ in the topological order of the graph $G_{\text{SCC}}$ and therefore already labelled by the semantics. In the general SCC-recursive scheme, one has to consider attacks from arguments in the extension (labelled $\text{in}$) and provisionally defeated arguments (labelled $\text{undec}$). In $\text{ub}$-semantics, in order to implement the undecidedness blocking constrain described above, when labelling a non-initial strongly connected component $S$ we neglect attacks to arguments in $S$ from undecided arguments external to $S$, and only attacks from $\text{in}$-labelled arguments are considered. Undecidedness results blocked rather than propagated into $S$. The $\text{ub}$-labelling function corresponding to $\text{ub}$-semantics can be formalized as follows.

**Definition 5.4** Let us consider the argumentation framework $AF = \langle \text{Ar}, \mathcal{R} \rangle$, and a labelling function $\mathcal{L}$. The $\text{ub}$-labelling is identified by the function $\mathcal{L}_{\text{ub}}$ defined as follows:
• if $|SCCS_{AF}| = 1$, $L_{ub} = \mathcal{L}$

• otherwise, $\forall S \in SCCS_{AF}$, it is

$$L_{ub} = \begin{cases} L_{ub}, & \forall x \in S \setminus S^+_{in} \\ \text{out}, & \forall x \in S^+_{in} \end{cases}$$

where $SCCS_{AF}$ is the set of all the strongly connected components of $AF$ and $S^+_{in}$ is the set of arguments in $S$ externally attacked by an $\text{in}$-labelled argument: $S^+_{in} = \{a \in S \mid \exists b \notin S : R(b, a) \land L_{ub}(b) = \text{in} \}$.

We require the base function $\mathcal{L}$ to return labellings that are weakly complete. Indeed, if $\mathcal{L}$ returns weakly complete labellings then $L_{ub}$ returns weakly complete labellings as well. This follows from the fact that the $L_{ub}$ labelling is generated by applying the weakly complete labelling function $\mathcal{L}$ on strongly connected components (or restrictions of such components) of $AF$, while the undecidedness blocking mechanism that neglects attacks from undecided arguments to a non-initial strongly connected component is justified by the $\text{ub-condition } 3.1b$ of weakly complete labellings.

The following Proposition shows how the undecidedness is not propagated outside the strongly connected component where it was generated, and acyclic arguments are labelled either $\text{in}$ or $\text{out}$.

**Proposition 5.1.** Let us consider the argumentation framework $AF = \langle Ar, R \rangle$. If an argument $a \in Ar$ is acyclic, then $L_{ub}(a) \neq \text{undec}$.

**Proof.** Since $a$ is not part of a cycle, it is a strongly connected component of the original graph. Therefore, according to the definition of $\text{ub-semantics}$, the labelling of $a$ is either the application of the base function on $a$ alone - and therefore $a$ is labelled $\text{in}$ since the base function is weakly complete - or it is labelled $\text{out}$ if the argument has one attacker labelled $\text{in}$. $\square$

### 5.4.1. Undecidedness Blocking Grounded Semantics

**Definition 5.5.** The undecidedness-blocking grounded semantics $G_{ub}$ is the $\text{ub-semantics}$ using as base function the grounded semantics $G$.

The use of the grounded semantics as base function implies that the arguments part of a cyclic strongly connected component $S$ with no external
attacks from in-labelled arguments (i.e. $S^+_m = \emptyset$) are all labelled undecl, as it happens with grounded semantics. However, the use of the SCC-recursive scheme of ub-semantics guarantees that the acyclic arguments are labelled in or out by ub-grounded semantics (Proposition 5.1), and therefore the undecided label is not propagated (but blocked) outside the cycle where it was generated. Unattacked conflicting arguments are unresolved conflicts generating undecidedness, but this undecidedness is not propagated to other arguments not part of the conflict. Undecidedness is therefore blocked as earlier as possible (in the topological sense) outside cycles. An example of ub-grounded semantics is shown in figure 6. Only the weakly complete labelling on the left is a ub-grounded labelling. The source of undecidedness is the self-attacking argument $a$. Indeed, in the labelling on the left undecidedness is blocked earlier than in the labelling on the right.

**Proposition 5.2** The ub-grounded labelling always exists and it is unique.

*Proof.* Building the ub-grounded labelling equates to a series of application of the grounded semantics on strongly connected components or restriction of strongly connected components of the argumentation framework. Therefore the thesis is proven by relying on the fact that grounded semantics always exists and it is unique □.

5.5. Undecidedness Blocking Preferred Semantics

The second ub-semantics aims to use the ub-condition not only as earlier as possible but also as much as possible. In the ub-grounded semantics, undecidedness is only blocked for arguments outside cyclic strongly connected components, but it is not blocked inside the components. However, a single strongly connected component can have multiple weakly complete labellings in which undecidedness is blocked, as shown in figure 7.

While the notion of as much as possible has been already captured by weakly preferred labellings, it is more challenging to decide when a labelling uses the ub-condition earlier than another. Referring to figure 7, which one is earlier? In the definition of ub-semantics the concept of earlier has been mapped to the notion of topological order, that is nevertheless not applicable inside a single strongly connected component where no topological order defined.

We might decide that there is no point in considering an ordering among elements of the same strongly connected component and use the weakly pre-
ferred labelling as base function, since it guarantees that the *ub-condition* is used as much as possible. This solution is not completely satisfactory. Referring to figure 7, the two labellings are both *weakly preferred* labellings, but intuitively undecidedness is blocked earlier in the graph on the right (where undecidedness is blocked at argument *b*) than in the graph on the left (where undecidedness is blocked at *c*). Note how *b* is also closer to *a*, the source of the unresolved conflict generating the undecided label.

How can we capture the above intuition? Since the effect of the *ub-condition* is to assign the label **in** to otherwise undecided arguments, we might hypothesize that it is a question of selecting the labelling minimizing the set of undecided arguments. However, this does not capture the property we want, since both the labellings of figure 7 minimize the set of undecided arguments. What we need to define is an ordering among arguments of a strongly connected component based not on the topology of the graph but rather on the relation among their labels. The ordering is formalized as follows:

**Figure 7:** Two *weakly complete* labellings. The undecidedness blocking condition is applied to argument *c* on the left and *b* on the right.

**Definition 5.6 (semantic precedence).** Given $AF = (Ar; \mathcal{R})$, we define $\mathcal{P} \subset Ar \times Ar$, a binary relation over arguments so that $(a, b) \in \mathcal{P}$ iff $a$ and $b$ are part of the same strongly connected component, $a$ is at least credulously accepted by weakly complete semantics and for all the weakly complete labellings $\mathcal{L}$ of $AF$ it holds that $(\mathcal{L}(a) = \text{in}) \rightarrow (\mathcal{L}(b) \neq \text{undec})$. We define *semantic precedence* the binary relation $\succsim \subset Ar \times Ar$ so that $a \succsim b$ iff $(a, b) \in \mathcal{P} \land (b, a) \notin \mathcal{P}$.

The *semantic precedence* relation $\succsim$ captures the idea that the acceptance of an argument affects the status of another, but not vice versa. Refer-
ring to the graph in figure 7, it is \( b \succ s c, c \succ s d \) and \( c \succ s e \). The idea is that in order to apply the \textit{ub-condition as earlier as possible}, we should consider the relation \( \succ s \). The \textit{ub-condition} should be applied to argument \( a \) before \( b \) if \( a \succ s b \).

In order to formalize this idea, we consider the set \( I_p \) of all the arguments for which there is no argument preceding them, that is \( I_p = \{ a \in Ar | \not\exists b \in Ar \text{ so that } b \succ s a \} \). We now consider the \textit{weakly preferred} labelling \( L \) that maximises the set of arguments accepted in \( I_p \). This labelling uses the \textit{ub-condition} as much as possible (since it is a \textit{weakly preferred} labelling) but it also considers the semantic precedence and it first accepts arguments for which there are no arguments semantically preceding them. A \textit{ub-preferred} semantics is the \textit{ub-semantics} using as base function \( L \). Formally:

\textbf{Definition 5.7} Let us consider an argumentation framework \( AF = \langle \text{Ar}, \mathcal{R} \rangle \) and the \textit{weakly preferred} labelling function \( W_{pr} \). We define the labelling function \( L \) as the function returning all the labellings for which \( \text{in}(W_{pr}) \cap I_p \) is maximal w.r.t. set inclusion. The \textit{ub-preferred} semantics \( L_{ubpr} \) is the \textit{ub-semantics} where the base function is \( L \).

Referring to figure 7, only the labelling on the right is a \textit{preferred ub-labelling}, since \( I_p = \{ b \} \). Referring to figure 6, only the graph on the left is a preferred ub-labellings, since \( I_p = \{ b \} \) as well. In graph \( G_5 \) of figure 3, \( I_p = \{ a, b \} \) and therefore only the two Dung’s \textit{complete preferred} labellings are \textit{ub-preferred} labellings.

\textbf{6. Discussion and Properties}

In this section we provide a principle-based analysis of \textit{weakly complete} semantics w.r.t. a set of properties commonly referred in literature and collected from [14] and [15]. Table 3 summarizes the results. The first four columns refers to Dung’s \textit{complete}, \textit{grounded}, \textit{preferred} and \textit{stable} semantics, and they serve as a comparison. We remind how \textit{weakly stable} and \textit{weakly grounded} semantics coincide with Dung’s \textit{stable} and \textit{grounded} semantics. The following four columns refer respectively to \textit{weakly complete}, \textit{weakly preferred}, \textit{undecidedness blocking grounded} and \textit{undecidedness blocking preferred} semantics. The last three columns refer to the \textit{weakly admissible} version of \textit{complete}, \textit{grounded} and \textit{preferred} semantics proposed by Baumann et al.
Table 3: Properties of weakly complete, ub-semantics

| Property          | $co$ | $gr$ | $pr$ | $st$ | $wc$ | $wp$ | $ub_{gr}$ | $ub_{pr}$ | $co_{blu}$ | $gr_{blu}$ | $pr_{blu}$ |
|-------------------|------|------|------|------|------|------|-----------|-----------|------------|------------|------------|
| Conflict-free     | Yes  | Yes  | Yes  | Yes  | Yes  | Yes  | Yes       | Yes       | Yes        | Yes        | Yes        |
| Naiveity          | No   | No   | No   | No   | No   | No   | No        | No        | No         | No         | No         |
| Admissible        | Yes  | Yes  | Yes  | No   | No   | No   | No        | No        | No         | No         | No         |
| Reinstatement     | Yes  | Yes  | Yes  | Yes  | Yes  | Yes  | Yes       | Yes       | Yes        | Yes        | Yes        |
| Rejection         | Yes  | Yes  | Yes  | Yes  | Yes  | Yes  | Yes       | Yes       | Yes        | Yes        | Yes        |
| Directionality    | Yes  | Yes  | Yes  | No   | Yes  | Yes  | Yes       | Yes       | No         | No         | ?          |
| Abstention        | Yes  | Yes  | No   | No   | Yes  | Yes  | No        | No        | No         | No         | No         |
| Cardinality       | $\geq 1$ | $1$ | $\geq 0$ | $\geq 1$ | $\geq 1$ | $1$ | $\geq 1$ | $\geq 1$ | $\geq 1$ | $\geq 1$ |
| I-maximality      | No   | Yes  | Yes  | No   | Yes  | Yes  | Yes       | Yes       | No         | Yes        | Yes        |
| Cycle-Homo.       | No   | Yes  | No   | No   | No   | Yes  | No        | No        | No         | No         | No         |

[3]. For such semantics we report the results of the principle-based analysis conducted in [16].

6.1. Principle-based Analysis

All weakly complete semantics are non-admissible, since they can accept arguments that are not defended from the attacks of undec-labelled arguments. These semantics could be seen as employing a different form of admissibility since they still require an argument to be defended from the attacks of in-labelled arguments but, under some conditions, not from the attacks of undec-labelled arguments.

Weakly complete semantics are conflict-free, since if an argument $a$ is accepted, the arguments attacked by $a$ are not (rejection condition 3.1b). Therefore weakly complete semantics also satisfy rejection, since arguments attacked by at least one in-labelled argument are always labelled out and explicitly rejected.

All weakly complete semantics satisfy the reinstatement property since, if an argument has all its attackers labelled out, it is necessarily labelled in. However, an additional form of reinstatement is possible in weakly complete semantics that is not possible in Dung's complete semantics. Indeed, in ub-grounded semantics an argument $a$ defeated by $b$ is fully reinstated even by an argument $c$ rebutting $b$, since the attack of $c$ creates a cycle with $b$ and therefore there is a weakly complete semantics where $c$ and $b$ are labelled undecided and the undecided label is blocked using the ub-condition 3.1d and $a$ is accepted.

The property of directionality is defined as follows (from [17]):

31
Definition 6.1. Let us consider a framework $AF = \langle Ar, R \rangle$ and $\mathcal{L}_\sigma(AF)$, the set of all the labelings of $AF$ according to semantics $\sigma$. We consider $US_{AF}$, the set of initial strongly connected components of $AF$. The semantics $\sigma$ satisfies directionality iff $\forall U \in US_{AF}, in(\mathcal{L}_\sigma(AF_U)) = in(\mathcal{L}_\sigma(AF_{\cap U}))$, where $in(\mathcal{L}_\sigma(AF_U)) = \{in(\mathcal{L})|\mathcal{L} \in \mathcal{L}_\sigma(AF_U)\}$ and $in(\mathcal{L}_\sigma(AF_{\cap U})) = \{in(\mathcal{L}) \cap U|\mathcal{L} \in \mathcal{L}_\sigma(AF)\}$.

The idea is that the justification state of an argument $a$ is affected only by the justification state of the defeaters of $a$ (which in turn are affected by their defeaters and so on).

Proposition 6.1. Weakly complete semantics satisfies directionality.

Proof. We need to prove that $\forall U \in US_{AF}, in(\mathcal{L}_\sigma(AF_U)) = in(\mathcal{L}_\sigma(AF_{\cap U}))$.

1) $in(\mathcal{L}_\sigma(AF_U)) \subseteq in(\mathcal{L}_\sigma(AF_{\cap U}))$.

We prove that $\forall L_U \in \mathcal{L}_\sigma(AF_U)$, $\exists L_{\cap U} \in \mathcal{L}_\sigma(AF_{\cap U})$ so that $in(L_U) \subseteq in(L_{\cap U})$. Each labeling $L_U \in \mathcal{L}_\sigma(AF_U)$ is generated by applying the in-out-fw algorithm to a set of ground arguments $G = \{g_1, \ldots, g_n\} \subseteq U$ (Theorem 4.3). We consider the labeling $L \in \mathcal{L}_\sigma(AF)$ generated using as grounds the initial arguments of the framework and the arguments in $G$. We observe that none of the applications of the in-out-fw applied to the sequence of arguments $G$ produces an inconsistency. Indeed, according to Proposition 4.2, in-out-fw($\{g\}$) ($g \in G$) can generate an inconsistency only with another ground argument or itself. The constraints generated by in-out-fw($\{g\}$) cannot conflict with any other ground argument in $G$ (since $L_U$ is a valid weakly admissible labelling), and neither they can conflict with a ground arguments external to $U$, since $\mathcal{L}_\sigma(AF)$ does not have any other non-initial ground arguments outside $U$. Therefore $in(L_U) \subseteq in(L)$ and therefore $in(L_U) \subseteq in(L) \cap U = in(L_{\cap U})$.

2) $in(\mathcal{L}_\sigma(AF_U)) \supseteq in(\mathcal{L}_\sigma(AF_{\cap U}))$.

We prove that $\forall L_{\cap U} \in \mathcal{L}_\sigma(AF_{\cap U})$, $\exists L_U \in \mathcal{L}_\sigma(AF_U)$ so that $in(L_U) \supseteq in(L_{\cap U})$. This is proved by observing that the constraints of the in-out-fw($\{g\}$) with $g \notin U$ do not affect the labels of arguments in $U$, since $U$ is initial in $AF$. Therefore, for each labelling $L \in \mathcal{L}_\sigma(AF)$ the set $in(L) \cap U$ is included in the set of in-labelled arguments of at least one labelling $L_U \in \mathcal{L}_\sigma(AF_U)$ and therefore $in(\mathcal{L}_\sigma(AF_U)) \supseteq in(\mathcal{L}_\sigma(AF_{\cap U})). \square$

All the ub-semantics satisfies directionality, since, as proven in Proposi-
tion 55 of [17], they have been defined using an SCC-recursive schema and they have at least one valid labelling. The weakly preferred labelling does not satisfy directionality. As a counter-example, we consider the floating assignment graph $G_5$ of figure 3 and the initial strongly connected component $\{a, c\}$. If $\mathcal{L}(AF)$ is the set of the three weakly preferred labellings of $AF$, it is $in(\mathcal{L}(AF)) \cap \{a, c\} = \{\{a\}, \{c\}, \emptyset\}$ and $in(\mathcal{L}(AF_{\downarrow \{a,c\}})) = \{\{a\}, \{c\}\}$.

The property abstention states that, if an argument $a$ is labelled out in at least one valid labelling and labelled in in at least another, then there must be a valid labelling where $a$ is labelled undec. Among Dung’ semantics, only complete semantics satisfies it. It can be proved that weakly complete semantics satisfy abstention.

Proposition 6.2. Weakly complete semantics satisfies abstention.

Proof. Let us consider an argumentation framework $AF = \langle Ar, \mathcal{R} \rangle$ and $a \in Ar$. If an argument $a$ is in in one weakly complete labelling and out in another one, $a$ is necessarily labelled undec by the grounded labelling $\mathcal{G}_{AF}$. This is because the set of in-labelled arguments of any weakly complete labelling contains $in(\mathcal{G}_{AF})$ and, since grounded semantics is unique, all the arguments labelled out or in by grounded semantics retain their labels in all the weakly complete labellings. Therefore if there is a weakly complete labelling where $a$ is labelled in and another where $a$ is labelled out, then there is at least one third valid labelling, the grounded labelling, where $a$ is labelled undec and the property is proven. □

Weakly preferred labelling does not satisfy abstention, a counter-example is a graph with two rebutting arguments. The same counter-argument is valid for ub-preferred semantics, while the single-status ub-grounded semantics satisfies it, as Dung’s grounded semantics does.

Regarding the cardinality of each semantics, multiple weakly complete labellings of the same graph could exist and at least one (the grounded labelling) always exists. The same is for weakly preferred semantics and ub-preferred semantics. Ub-grounded semantics has exactly one labelling (Proposition 5.2).

A semantics satisfies I-maximality if no extension is a strict subset of another. It is indeed satisfied by single status semantics like the ub-grounded semantics. It is not satisfied by weakly complete semantics, as it is evident from the ground-based algorithm. Indeed, given a valid weakly complete labelling, it could be possible to extend the set of in-labelled arguments by
The maximality condition of the set of in-labelled arguments of weakly preferred labellings implies that the semantics satisfies I-maximality. Indeed, if the in-set of a labelling $L_1$ includes the in-set of $L_2$, than $L_2$ is not a weakly preferred labelling. The same reasoning applies to ub-preferred semantics since by definition ub-preferred labellings are all weakly preferred labellings.

The property of cycle-homogeneity is introduced in this paper. A semantics satisfies this property iff all the arguments in an argumentation framework composed by a single cycle are labelled in the same way in all the valid labellings of the semantics. Formally:

**Definition 6.2.** Let us consider an argumentation framework $AF = \langle Ar, R \rangle$ composed by a single cycle of arguments, a semantics $\sigma$ and $L_\sigma(AF)$ the set of labellings of $AF$ generated by $\sigma$. The semantics $\sigma$ satisfies the principle of cycle-homogeneity iff $\forall a \in Ar, \exists L_1, L_2 \in L_\sigma(AF)$ so that $L_1(a) \neq L_2(a)$.

It is a well-known behaviour of multi-status Dung’ semantics to label unattacked cycles of arguments in a different way depending on the length of the cycle. An odd-length cycle has an empty complete extension, while an even-length cycle has multiple valid labellings. This peculiar way of assigning the acceptability status to odd-length cycles has been indicated as “puzzling” by Pollock [18]. Among Dung’ semantics, the property of cycle-homogeneity is satisfied only by grounded semantics, where all the arguments are deemed undecided. This behaviour is retained in the ub-grounded semantics.

| Semantics     | Undecidedness Blocking Constraints                                                                 |
|---------------|------------------------------------------------------------------------------------------------------|
| Grounded w.a. | - No undecidedness blocking.                                                                        |
| Preferred w.a.| - Undecidedness blocked as much as possible.                                                        |
|               | - weakly complete labellings with maximal in-set                                                    |
|               | - Terminal nodes of the ground-based recursion tree                                                  |
| UB-Grounded   | - Undecidedness blocked as earlier as possible only outside cyclic strongly connected components     |
| UB-Preferred  | - Undecidedness blocked as earlier as possible and as much as possible                              |

Table 4: Conditions for weakly complete semantics

a new consistent application of the in-out-fw algorithm.
6.2. Generating Argumentation Semantics using Undecidedness Blocking

The ground-based algorithm presented in section 4 shows how any weakly complete labelling (therefore including any complete labelling) can be generated by repeatedly applying the in-out-fw algorithm. Labellings are build in a sequential way by multiple applications of the in-out-fw algorithm, and each application of the algorithm is a tentative to reduce the set of undecided arguments by blocking undecidedness at the argument used as ground for the in-out-fw algorithm.

The ground-based algorithm unifies the computation of all weakly complete semantics, therefore including Dung’s complete semantics. As such, it provides a way of interpreting the different semantics as a gradual effort to block undecidedness.

On the one end of the spectrum there is the grounded semantics, where undecidedness is not blocked, and on the other side stable semantics, where undecidedness is not present.

Every weakly complete labelling is a superset of the grounded semantics, that represents a necessary baseline for each acceptability strategy. Grounded semantics accepts only initial arguments and all the arguments defended directly or indirectly by initial arguments. The in-out-fw algorithm is applied only to all the initial arguments of the framework, accepted to satisfy the admissibility condition 3.1a. The undecidedness blocking condition 3.1d is not applied to any non-initial undecided arguments, making the grounded semantics the only truly undecidedness propagating semantics, since the ub-condition is not used in generating the labelling.

All the other labellings are obtained by repeatedly apply the ub-condition on some undecided arguments and propagating the necessary constraints. They are therefore all employing some undecidedness blocking mechanisms, where the set of undecided arguments is somehow reduced by tentatively accepting some of them. However, (1) the arguments on which the ub-condition can be used, (2) when and how often it is used and (3) the conditions to accept as valid the resulting labelling define the different weakly complete semantics. Table 4 show the undecidedness blocking strategies of the families of weakly complete semantics presented in this paper.

In weakly complete semantics any argument \( a \) can be selected as ground (thus blocking undecidedness at \( a \)) as long as in-out-fw(\{a\}) does not generate an inconsistency. The ub-grounded semantics blocks undecidedness as earlier as possible outside a cyclic strongly connected component and it lets undecidedness propagate inside a cyclic strongly connected component.
The \textit{ub-preferred} semantics and the \textit{weakly preferred} semantics use the undecidedness blocking condition respectively \textit{as much as possible} and \textit{as earlier and as much as possible}.

The ground-based algorithm suggests that the generation of Dung’s \textit{complete} semantics can also be revisited as a special case of \textit{undecidedness blocking} semantics with the extra condition of admissibility. The admissibility condition requires the accepted arguments to be able to collectively defend themselves by labelling \texttt{out} their attackers. This means that in Dung’s \textit{complete} semantics not only undecidedness is blocked by using the \textit{ub-condition}, but it is also \textit{resolved}. The conflicts between arguments that created the undecided situation are resolved by promoting some of them to the label \texttt{in}, so that none of these arguments is attacked by an undecided argument anymore in the resulting labelling. Since the undecidedness is solved, in the resulting labelling there is no trace that it was blocked.

The diagram in Figure 8 represents the relations among Dung’s \textit{complete} semantics and the \textit{weakly complete} semantics introduced in this paper.

![Diagram of semantics hierarchy](image)

\hspace{1cm}\textbf{Figure 8: Hierarchy of weakly complete semantics}

\subsection{6.3. Weakly Complete versus Baumann-Brewka-Ulbricht semantics}

Baumann et al. \cite{Baumann} proposed a solution to the problem of self-defeating attackers, defining a family of semantics (here called \textit{BBU} semantics) based on a weaker notion of Dung’s admissibility. While the definition 3.1 of our
weakly complete semantics requires a small modification of Dung’s complete labelling, the definition of BBU semantics require some preliminary concepts. Given an argumentation framework \( AF = \langle Ar, R \rangle \) and a set \( E \in Ar \), we define the set \( AF^E \), called the \( E \)-reduct of \( AF \) as its restriction \( AF_{E^*} \), where \( E^* = A \setminus (E \cup E^+) \), that is the restriction containing all the arguments except the arguments in \( E \) and the ones attacked by \( E \).

A weakly admissible set \( S \) is a set that is required to defend itself only from attacks of arguments that are weakly admissible in \( AF^S \). Formally:

**Definition 6.3** Given the framework \( AF = \langle Ar, R \rangle \), the set of weakly admissible sets of \( AF \) is denoted \( ad^w(AF) \) and defined by \( E \in ad^w(AF) \) iff \( E \) is conflict-free and for every attacker \( y \) of \( E \) we have \( y \in \bigcup ad^w(AF^E) \).

Based on the notion of weakly admissible sets, the authors defined the concept of weak defence, so that every conflict-free set is weakly admissible if and only if it weakly defends itself.

**Definition 6.4** Given the framework \( AF = \langle Ar, R \rangle \), a set \( E \subseteq Ar \) weakly defends a set \( X \subseteq Ar \) whenever, for every attacker \( y \) of \( X \), either \( E \) attacks \( y \), or \( y \not\in \bigcup ad^w(AF^E) \), \( y \not\in E \) and \( X \subseteq X' \in ad^w(AF^E) \).

The BBU complete, preferred and grounded semantics are defined as follows:

**Definition 6.5** Given \( AF = \langle Ar, R \rangle \) and \( E \subseteq Ar \), we say that \( E \) is:

- a BBU complete extension of \( AF \) \((E \in co_{bbu}(AF)) \) iff \( E \in ad^w(AF) \) and for every \( X \) such that \( E \subseteq X \) that is weakly defended by \( E \), we have \( X \subseteq E \).

- a BBU preferred extension of \( AF \) \((E \in pr_{bbu}(AF)) \) iff \( E \) is maximal w.r.t. set inclusion in \( ad^w(F) \).

- a BBU preferred extension of \( AF \) \((E \in gr_{bbu}(AF)) \) iff \( E \) is minimal w.r.t. set inclusion in \( co^w(F) \).

We start our comparison by observing how our semantics and the BBU semantics do not coincide. Let’s consider the floating assignment example in figure 1A. According to the definition of weakly admissible set, the set \( \{a\} \) is weakly admissible \((F^a = \emptyset) \) and so is \( \{b\} \). This implies that \( c \) is not in any weakly admissible set since \( a \) and \( b \) are. The BBU complete extensions are
\{{a}, \{b\}, \emptyset\}, coinciding with Dung’s semantics. Regarding graphs B and C in figure 1, b is accepted since a self-attacking argument (graph 1B) or an odd-length cycle not externally attacked (1C) are not weakly admissible sets.

Our weakly complete semantics generate the same labelling for graph B and C of figure 1, while in graph 1A it adds the additional labelling where c is accepted and a and b are undecided. In graph 1A, our semantics allow for the interpretation where the attacks to c are from two arguments that are conflicting and therefore not strong enough to defeat c. This is certainly the case if we take a grounded semantics stance on the two rebutting arguments a and c. Indeed graphs A, B and C of figure 1 could be interpreted as instances of the same situation where argument b is receiving attacks from conflicting arguments. The only difference is that in graph A argument b is attacked by an even-length cycle. Indeed, the preferred semantics interpretation of graph A is a valid (and it is indeed included in our weakly complete labelling), but not the only reasonable one. We believe the extra labelling added by weakly complete semantics shows how such semantics are able to transfer the sceptical stance proper of grounded semantics to the non-admissible case, while the non-admissible BBU version of grounded semantics does not.

Figure 9: A graph with two BBU weakly grounded extensions \{a_1, b_1\}, \{a_2, b_2\}. From [3]

The principle-based analysis of the two families of semantics reveals some important differences. The results for BBU semantics are taken from [16]. We report our main observations:

- **Grounded semantics.** The BBU version of grounded semantics strongly deviates from Dung’s counterpart. There could be multiple BBU grounded extensions of a framework, therefore losing the single-status property, and it is possible that none of those extensions is Dung’s grounded semantics. For instance, the graph of figure 9 has two grounded BBU extension: \{a_1, b_1\}, \{a_2, b_2\}, while Dung’s grounded semantics is
empty. On the contrary, both our weakly grounded and ub-grounded semantics are single-status, they always exist and they can be computed in polynomial computational time. Our semantics retain the scepticism of Dung’s grounded semantics in the way cycles are treated, while the BBU version of grounded semantics loses much of the sceptical stance of its Dung’s counterparts.

- **Directionality.** BBU complete and grounded semantics do not satisfy directionality and no result has been proved for preferred BBU semantics. On the contrary, weakly complete and ub-semantics do satisfy directionality.

- **Abstention.** None of the BBU semantics satisfy abstention. Abstention is an interesting property satisfied by Dung’s complete semantics. If an argument is labelled out in one valid labelling and in in another, the argument is disputed, and a semantics satisfying abstention also provides at least a third labelling where such an argument is undec. Our weakly complete semantics satisfy abstention. Our grounded and ub-grounded trivially satisfy it, while the preferred does not. Therefore, they indeed behave like Dung’s counterparts.

- **SCC-Decomposition.** Negative results are proved for BBU complete and grounded semantics, and no proof is provided for preferred semantics. Our ub-preferred and ub-grounded semantics are SCC-recursive, and therefore they are guaranteed to satisfy SCC-Decomposition. According to proposition 55 in [17], weakly preferred semantics is not SCC-decomposable since it has a cardinality of extensions greater than zero and it does not satisfy directionality.

- **Set inclusion.** BBU weakly complete semantics do not include Dung’s complete semantics, while Dung’s complete labellings are a subset of our weakly complete semantics labellings.

In general, the above principle-based analysis shows how our semantics satisfy all the principles satisfied by BBU semantics, but they also satisfy other principles like directionality, SCC-decomposition and abstention that are satisfied by Dung’s counterparts. Our semantics results indeed closer to the principles underlying Dung’s complete semantics; weakly complete and ub-grounded semantics exactly satisfy the same principles of Dung’s counterparts. Based on this, we believe that weakly complete semantics represent a
more faithful extension of Dung’ semantics to the non-admissible case and a more faithful solution to the problem of self-defeating attacking arguments.

6.4. Ambiguity Blocking and Undecidedness Blocking

The introduction of weakly complete semantics was motivated by the idea of introducing ambiguity blocking mechanisms in abstract argumentation. So far our semantics have been presented as undecidedness blocking semantics in a self-contained way, without references to ambiguity blocking mechanisms present in other non-monotonic formalisms. Nevertheless, undecidedness blocking is indeed inspired by the mechanism of ambiguity blocking and the willingness to solve situations like the floating assignment example in the way ambiguity blocking semantics do, that is accepting the attacked argument on the basis that the two attackers are conflicting.

In this section we explore the relation between ambiguity blocking and undecidedness blocking. The relation is a loose one; the two concepts do not coincide but nevertheless they share strong conceptual similarities. We claim how undecidedness is a broader concept able to model ambiguity but not limited to it.

We first need to specify what we mean by ambiguity. Here we refer to the concept of ambiguity as defined in defeasible logic. Defeasible logic [19] is a rule-based non-monotonic formalism. Rules can be strict \( (A \rightarrow B) \), representing monotonic rules where \( B \) follows always if \( A \) is true or defeasible rule \( (A \Rightarrow B) \), meaning that from \( A \) defeasibly follows \( B \). Defeasible rules represent default positions that are assumed valid unless contrary evidence is provided. Rules with empty body represent facts. \( \rightarrow B \) is an indisputable fact, \( \Rightarrow B \) is a defeasible fact. An acyclic superiority relation is used to define priorities among rules, that is, where one rule may override the conclusion of another rule. Conclusions can be classified as definite (also called strict) or defeasible. A conclusion can be therefore provable (or not provable) strictly or defeasibly. A rule is applicable if its antecedents are defeasible provable. A rules is not applicable if it has been proved that at least one antecedent of the rule is not defeasible provable. Strict conclusions are obtained by forward chaining of strict rules, while a defeasible conclusion \( A \) can be derived if there is an applicable strict or defeasible rule with conclusion \( A \), and either (1) \( \neg A \) is not definitely provable and each rules concluding \( \neg A \) has been proved to be not applicable, or (2) every applicable rule for \( \neg A \) is weaker (according to the superiority relation) than an applicable strict or defeasible rule for \( A \).
A defeasible theory is a set of rules and a superiority relation. A literal $a$ is ambiguous if there is a valid chain of reasoning concluding $a$ and another concluding $\neg a$ and the superiority relation cannot resolve such conflict. The ambiguity blocking mechanism of defeasible logic semantics is that both $a$ and $\neg a$ are refuted and their ambiguity is not propagated to other conflicting literals, as shown in the following example.

Example 6.1. Let us consider the following defeasible theory:

$$\mathcal{D} = \{ \Rightarrow a, \Rightarrow \neg a, \neg a \Rightarrow b, \Rightarrow \neg b \}$$

(1)

In the ambiguity blocking defeasible logics semantics only $\neg b$ can be defeasibly proved. Indeed, since literal $a$ and $\neg a$ are ambiguous (we have the two conflicting facts $\Rightarrow a$ and $\Rightarrow \neg a$), they are both refuted and therefore the rule $\neg a \Rightarrow b$ is not applicable, and $\neg b$ is proved. Using an ambiguity propagating semantics none of the literals could be proved and both $a$ and $b$ are ambiguous.

Example 6.2. The following theory models a conflict similar to the floating assignment situation:

$$\mathcal{D} = \{ \Rightarrow e, \Rightarrow \neg e, e \Rightarrow g, \neg e \Rightarrow g \}$$

(2)

where the literal $g$ means guilty and $e$ and $\neg e$ are the conflicting evidence used to accuse the defendant. Using ambiguity blocking $g$ cannot be proved since $e$ is ambiguous.

While ambiguity is well defined in defeasible logic (DL), in Dung’s abstract argumentation the concept of ambiguous argument is not even defined. In [6] Governatori et al. referred to Dung’s grounded semantics as an ambiguity propagating semantics, but their study is in the context of defeasible logic, not abstract argumentation, and a definition of ambiguous argument in Dung’s framework is outside the scope of their work. Our intuition was to model ambiguity with undecidedness. What is propagated by grounded semantics is indeed the undecided status of arguments.

Ambiguity blocking and undecidedness blocking have some strong similarities. First of all, ambiguous literals and undecided arguments both signal unresolved conflicts. In the first case the superiority relation does not help to resolve the conflict, while in the second the postulates of the semantics do not provide a definitive reason to accept or reject the conflicting arguments. Second, both ambiguity and undecidedness are blocked by rejecting unresolved
conflicts. The ambiguity blocking behaviour is to reject the ambiguous literals and all the rules containing ambiguous literals in their premises. Literals that were in conflict with such rejected literals become provable. Undecidedness blocking semantics do indeed mimic this behaviour. The mechanism blocks attacks from undecided arguments so that attacked arguments become now acceptable.

There are also notable differences. Conflicts in $DL$ are represented by complementary literals, defining a symmetric conflict relation. The superiority relation of defeasible logic is by definition acyclic, meaning that the clash of complementary literals is the only form of “cyclic” conflict in $DL$. Two symmetrical conflicting arguments are most likely to be modelled in abstract argumentation by a set of two rebutting arguments, but it is not guaranteed that those arguments will generate undecidedness. This indeed depends on the semantics used: undecidedness is generated if we are using grounded semantics, but not if we are using preferred semantics. Linking ambiguity with undecidedness is therefore semantics-dependant, since what is undecided depends on the stance of the semantics employed, hence the different families of undecidedness-blocking semantics presented in this paper.

Moreover, undecidedness can arise in situations where there is no ambiguity. Paradoxical situations like a cycle of three arguments with unidirectional attacks also generate an undecided situation in abstract argumentation, but they are not possible in defeasible logic since the superiority relation is required to be acyclic. Therefore undecidedness is a much general concept signalling unresolved conflicts, including dilemma or paradoxical situations that might not have anything to do with ambiguity. Indeed, if we consider the meaning of the English word ambiguous (that is "a situation open to multiple interpretations"), is clearly not the same as being undecided. For instance, in a paradox there is no ambiguity since we do not have multiple potentially valid interpretations but rather all the interpretations are contradictory. These observations suggest how ambiguity is a special case of the more general concept of undecidedness. Therefore, undecidedness blocking semantics could be used to generalize ambiguity blocking semantics to the case of cyclic conflict relations among arguments and to model less sceptical multi-status ambiguity blocking semantics.

A future research direction is the formal investigation of the relation between defeasible logic ambiguity blocking and our undecidedness blocking semantics. Our hypothesis is that the $ub$-grounded semantics can be used to instantiate ambiguity blocking defeasible logic semantics in the same way
grounded semantics was used to instantiate ambiguity propagation defeasible logic semantics in [8]. Since DL is a rule-based structured formalism while Dung’s framework is abstract, a formal analysis of the possibility of using our semantics to instantiate ambiguity blocking would require to use structured argumentation frameworks such as ASPIC+.

7. Related Works

One of the main reason to introduce weakly complete semantics was to translate ambiguity blocking into abstract argumentation. The work in [6] represents our main reference regarding the link between ambiguity blocking defeasible logic and abstract argumentation. We want to stress the difference between the two works: while in [6] authors translate Dung’ semantics into ambiguity propagating defeasible logic, in this paper we went the opposite direction and we translated the notion of ambiguity blocking into abstract argumentation.

This paper extends our preliminary works [11, 10], where we proposed an abstract semantics to model the principle of beyond reasonable doubt. The resulting semantics was a first attempt to define an undecidedness blocking semantics. The semantics proposed is a subset of weakly complete semantics, the ub-grounded semantics is defined in [10] but its links to defeasible logics and ambiguity blocking semantics are not studied. A Dung-like version of ambiguity blocking has been investigated in the context of the instantiation of the Carneades argumentation system [20] into the Dung-based structured argumentation system ASPIC+ [7]. For instance, in [5] a translation mechanism is proposed to model the Carneades argumentation systems into an ASPIC+ argumentation system. The relevance to our work lies in the fact that the Carneades argumentation system is ambiguity blocking while ASPIC+, by relying on Dung’ semantics, is ambiguity propagating. Therefore a translation from Carneades to ASPIC+ has to deal with the problem of modelling ambiguity blocking in a Dung-like system. The authors state how this was the “the main difficulty” of the translation process. Instead of introducing a new Dunganian semantics, the authors solved the problem by introducing additional argument nodes, allowing for an explicit representation of applicability and acceptability of rules. In this translation, a couple undercutter defeaters (unidirectional attacks) is added for each contradictory literals to refute both of them and replicate the ambiguity-blocking behaviour. The authors do not propose a new ambiguity-blocking Dunganian semantics, but in
the context of a structured argumentation system they mimic the ambiguity blocking behaviour using Dung’s semantics and additional nodes.

This approach is implicitly questioned in [8], where the relation between Defeasible logic and ASPIC+ is investigated. The authors show how ASPIC+ using grounded semantics is equivalent to the ambiguity propagation version of defeasible logic semantics, and they consider how the ambiguity blocking DL semantics can be instantiated in ASPIC+. The authors conclude how such a translation could result problematic; the DL with ambiguity blocking would require to introduce a second “attack” relation on arguments with a ripple down effects on the ASPIC+ definitions setting the various statuses of the argument. Therefore, an ambiguity blocking abstract semantics has not been developed, and in the context of structured argumentation there are works proposing non-trivial translations requiring the addition of new nodes and meta-concepts.

Similar conclusions have been reached by works on standard of proof and legal reasoning applied to abstract argumentation, since legal reasoning is often ambiguity blocking.

In particular, the standard of proof beyond reasonable doubt is responsible for the ambiguity blocking mechanism of legal reasoning, since evidence that are not beyond reasonable doubt are deemed not sufficiently credible and therefore blocked. Standards of proof have been extensively study in argumentation theory [21], but only few studies are relevant to abstract argumentation. In the context of structured argumentation, we mention the work of Prakken [22] on modelling standards of proof, and the modification of the Carneades framework [20] to accommodate various standards of proof. Regarding abstract argumentation, the most explicit study about standards of proof is [23]. Here, the authors consider how each Dung’s semantics has a different level of cautiously that is mapped to a corresponding legal standard of proof. Only initial arguments are beyond doubt, but they consider the sceptically preferred justification a beyond reasonable doubt position. In the floating assignment example (shown in Figure 1), the authors recognize the two attackers as doubtful, but they consider the sceptically preferred rejection of the attacked argument beyond reasonable doubt. It could be noticed that this position is failing to acknowledge that, if each of the attackers are considered doubtful, their effect cannot be (at last in all the situations) beyond doubt. Brewka et al. [24] also criticise [23] since they doubt the fact that various Dung’s semantics can capture the intuitive meaning of legal standards of proof (detailed discussion in here [21]). In the case of beyond
reasonable doubt, we agree with Brewka, complete Dung’s semantics are not adequate to model this principle.

Prakken has analysed the floating assignment and its link to standards of proof in his work [25], where he responds to objections advanced by Horty in [26]. Prakken underlines that, in many problematic situations including the floating assignment, there could be hidden assumptions about the specific problem which, if made explicit, are nothing but extra information that defeat the defeasible inference. In the case of the floating assignment, Prakken agrees that if beyond reasonable doubt is our standard of proof - like in a criminal case where there are two conflicting testimonies - we should not conclude that the accused is guilty.

In his presentation of semi-stable semantics, Caminada [27] also provides another example of the logical assumptions that could be hidden beyond the treatment of the floating assignment. In particular, he observes how the preferred semantics solution is based on the assumption that we know with certainty that one of the two attacking arguments is valid, since in this case we do not need to know which one is valid in order to safely discard the third argument. However, the above observations do not mean that argumentation semantics are somehow invalid. In the case of conflicting testimonies, as already showed by Pollock [28], the situation could be correctly modelled by making some hidden assumptions explicit and adding extra arguments to model such assumptions. In the conflicting testimonies, the fact that two witnesses contradict each other is a reason to add an argument undercutting the credibility of both. Instead of adding arguments interacting with existing arguments, in this work we have tackled the problem by embedding assumptions directly in a novel abstract argumentation semantics.

8. Conclusions and Future Works

In this paper we have explored a new family of semantics called weakly complete semantics. Unlike Dung’ semantics, in weakly complete semantics the undecided label can be blocked by the postulates of the semantics rather than being propagated from an undecided attacking argument to an otherwise accepted attacked argument.

The new semantics are conflict-free, non-admissible (in Dung’s sense), but employing a more relaxed defence-based notion of admissibility; they allow reinstatement, they generate extensions that are super sets of grounded semantics and they retain the majority of properties of complete semantics.
The semantics can also provide a solution to the 25-year old problem of self-defeating attackers. We have studied the properties of these new semantics and we have provided an algorithm to compute them. The algorithm suggests an interpretation of the various *weakly complete* and *complete* semantics as different strategies to reduce undecidedness. We have identified families of *weakly complete* semantics satisfying specific constraints on the set of acceptable arguments and employing different *undecidedness blocking* strategies. Regarding computational complexity, the main result is that the credulous acceptance problem for *weakly complete* semantics can be solved in polynomial time.

We have performed a principle-based comparison between our semantics and the *weakly admissible* semantics recently proposed by Baumann et al. The analysis shows how our semantics satisfy a number of principles satisfied by Dung’s complete semantics but not by Baumann et al. semantics, including *directionality, abstention, SCC-decomposability* and *cardinality* of extensions, making them a more faithful translation of Dung’ semantics to the non-admissible case.

Future research directions include the optimization of the computational algorithm proposed and the formal investigation of how *ub-grounded* semantics can be used in Dung-based structured frameworks like *ASPIC+* to instantiate ambiguity blocking defeasible logic semantics. It also includes the comparison of the newly proposed weakly complete semantics empirically [29] and its comparison against other semantics in order to evaluate its inferential capacity.

References

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

[2] M. W. Caminada, D. M. Gabbay, A logical account of formal argumentation, Studia Logica 93 (2009) 109–145.

[3] R. Baumann, G. Brewka, M. Ulbricht, Revisiting the foundations of abstract argumentation–semantics based on weak admissibility and weak defense, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 34, 2020, pp. 2742–2749.

[4] R. Calegari, G. Contissa, F. Lagioia, A. Omicini, G. Sartor, Defeasible
systems in legal reasoning: A comparative assessment., in: JURIX, 2019, pp. 169–174.

[5] B. van Gijzel, H. Prakken, Relating carneades with abstract argumentation via the aspic+ framework for structured argumentation, Argument & Computation 3 (2012) 21–47.

[6] G. Governatori, M. J. Maher, G. Antoniou, D. Billington, Argumentation semantics for defeasible logic, Journal of Logic and Computation 14 (2004) 675–702.

[7] S. Modgil, H. Prakken, The aspic+ framework for structured argumentation: a tutorial, Argument & Computation 5 (2014) 31–62.

[8] H.-P. Lam, G. Governatori, R. Riveret, On aspic+ and defeasible logic., in: COMMA, 2016, pp. 359–370.

[9] R. Baumann, G. Brewka, M. Ulbricht, Comparing weak admissibility semantics to their dung-style counterparts–reduct, modularization, and strong equivalence in abstract argumentation, in: Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning, volume 17, 2020, pp. 79–88.

[10] P. Dondio, L. Longo, Beyond reasonable doubt: A proposal for undecidedness blocking in abstract argumentation, Intelligenza Artificiale 13 (2019) 123–135.

[11] P. Dondio, L. Longo, A proposal to embed the in dubio pro reo principle into abstract argumentation semantics based on topological ordering and undecidedness propagation., in: AF³@AI* IA, 2018, pp. 42–56.

[12] P. Dondio, Weakly-admissible semantics and the problem of ambiguity blocking in abstract argumentation, in: Technical Report, Technological University Dublin, 2019, pp. 1–8.

[13] P. Baroni, M. Giacomin, G. Guida, Scc-recursiveness: a general schema for argumentation semantics, Artificial Intelligence 168 (2005) 162–210.

[14] P. Baroni, M. Caminada, M. Giacomin, An introduction to argumentation semantics, The Knowledge Engineering Review 26 (2011) 365–410.

[15] L. van der Torre, S. Vesic, The principle-based approach to abstract argumentation semantics, IfCoLog Journal of Logics and Their Applications (2017).

[16] L. V. D. T. Jeremie Dauphin, Tjitze Rienstra, A principle-based analysis of weakly admissible semantics, Computational Models of Argument: Proceedings of COMMA 2020 326 (2020) 167.
[17] P. Baroni, M. Giacomin, On principle-based evaluation of extension-based argumentation semantics, Artificial Intelligence 171 (2007) 675–700.
[18] J. L. Pollock, Defeasible reasoning with variable degrees of justification, Artificial intelligence 133 (2001) 233–282.
[19] D. Nute, Defeasible logic, in: International Conference on Applications of Prolog, Springer, 2001, pp. 151–169.
[20] T. F. Gordon, D. Walton, The carneades argumentation, in: Computational models of argument: Proceedings of COMMA 2006, volume 144, IOS Press, 2006, p. 195.
[21] T. F. Gordon, D. Walton, Proof burdens and standards, in: Argumentation in artificial intelligence, Springer, 2009, pp. 239–258.
[22] H. Prakken, G. Sartor, On modelling burdens and standards of proof in structured argumentation., in: JURIX, Citeseer, 2011, pp. 83–92.
[23] K. Atkinson, T. Bench-Capon, Argumentation and standards of proof, in: Proceedings of the 11th international conference on Artificial intelligence and law, 2007, pp. 107–116.
[24] G. Brewka, T. F. Gordon, Carneades and abstract dialectical frameworks: A reconstruction, in: Proceedings of the 2010 conference on Computational Models of Argument: Proceedings of COMMA 2010, 2010, pp. 3–12.
[25] H. Prakken, Intuitions and the modelling of defeasible reasoning: some case studies, arXiv preprint cs/0207031 (2002).
[26] J. F. Horty, Argument construction and reinstatement in logics for defeasible reasoning, Artificial intelligence and Law 9 (2001) 1–28.
[27] M. W. Caminada, W. A. Carnielli, P. E. Dunne, Semi-stable semantics, Journal of Logic and Computation 22 (2012) 1207–1254.
[28] J. L. Pollock, Cognitive carpentry, a blueprint for how to build a person, Mit Press, 1995.
[29] L. Rizzo, L. Majnaric, P. Dondio, L. Longo, An investigation of argumentation theory for the prediction of survival in elderly using biomarkers, in: IFIP International Conference on Artificial Intelligence Applications and Innovations, Springer, 2018, pp. 385–397.