Argumentation in Multi-agent Systems: Self-stabilizing Defeat Status Computation

Pietro Baroni, Massimiliano Giacomin, and Giovanni Guida
Università di Brescia
Dipartimento di Elettronica per l'Automazione
Via Branze 38, I-25123 Brescia, Italy
email: {baroni, giacomin, guida}@ing.unibs.it

Abstract. In this paper, we consider a multi-agent system where agents perform argumentation activity on the basis of knowledge both stated in their knowledge-bases and acquired from other agents. Starting from a previous proposal, we introduce a distributed approach to argumentation, removing some limiting assumptions, and we show by means of an example that the system can not be guaranteed to reach a state which satisfy a well-founded semantics, and not even to terminate. In order to tackle this problem, we devise two self-stabilizing algorithms for the computation of the defeat status, one specifically tailored to rebutting defeat, and the other able to deal with any form of defeat.

1 Introduction

Argumentation is a framework for defeasible reasoning, where reasoning activity is modeled as the process of constructing arguments for propositions from a given set of premises, by chaining rules of inference, which may represent just provisional reasons for their conclusions. Since different arguments may contradict each other, the core problem is the computation of the *defeat status*, namely determining which arguments emerge undefeated from the conflict and which conclusions should be believed. A key feature of this framework is the capability not only of suppressing a previously derived conclusion if a counterargument emerges, but also of reinstating it in case the counterargument is in turn defeated by further information acquired subsequently.

The role of argumentation as a key form of interaction among autonomous agents has been explicitely recognized in the multi-agent systems literature [14, 8]. For instance, argumentation has been proposed as the basic mechanism by which autonomous agents achieve a sophisticated form of *negotiation* [9, 7], i.e. they try to reach an agreement on some matter through communication. The capability of negotiating is essential when agents are assumed to be fully autonomous. In this case, each agent acquires (generally partial) information about the world, has its own goals and produces plans to achieve them. As a consequence, exchanging information is useful both to exploit knowledge of other agents about the world, possibly dealing with inconsistencies among different agents' views, and to achieve goals in cooperation, by exchanging resources and adjusting conflicting objectives. Argumentation seems to be appropriate for both kinds of tasks, since it offers a flexible and efficient reasoning framework both at the *epistemological level*, i.e. reasoning about what to believe (e.g. in [10, 3]), and at the *practical level*, i.e. reasoning about what to do [12, 9].

An influential approach to argumentation in a multi-agent environment has recently been proposed by Parsons and coauthors [9]. While this proposal is focused on argumentation-based negotiation about goals to be achieved, in this paper we consider, more generally, the process by which different agents exchange beliefs and construct arguments in a context which is not limited to goal negotiation. To this purpose, we remove some assumptions introduced in [9], which may represent limitations in many practical contexts. In particular:

- In [9], the defeat status of arguments is computed according to a semantics based on the *acceptability classes*: if an argument α has a counter-argument β, it is not considered as justified even if additional information invalidates β. Since reinstatement is a desirable feature of non-monotonic reasoning, we adopt here the *grounded semantics*, which includes reinstatement of arguments and is generally adopted in those argumentation systems proposed in the literature which enforce the so-called "single status approach" [13].

- While in [9] only two agents are explicitly considered, here we don't make any assumption about the numbers of involved agents.

- The argumentation model of [9] assumes that agents always exchange complete arguments, i.e. that each piece of information is communicated along with its justification. However, this assumption might be a severe limitation in all those contexts where agents do not always communicate all the reasons supporting a conclusion, either for efficiency purpose (the informed agent might be interested in the truth-value of a proposition only, while trusting the informer agent about the existence of a valid argument), or for privacy purpose (the informer agent might not be willing or allowed to disclose the justifications behind a conclusion).

The aim of this paper is to overcome these limitations by considering a multi-agent system where agents exchange information about the truth-value of propositions, and construct arguments on the basis of knowledge both stated in their knowledge bases (possibly updated by perceptual activities) and acquired from other agents. Moreover, in our model we don't assume any form of centralized control, nor the presence of a specific agent in charge of coordinating the activity of the others, and we don't commit to any kind of synchronism: every agent performs argumentation autonomously, revising its conclusions independently of the other ones and communicating information about its beliefs concerning specific aspects of the world in response to local requests.

2 The problem of stabilization in distributed argumentation

In the distributed model of argumentation outlined in the previous section, an important issue concerns the coordination of the autonomous activities of individual agents, that determine the behavior of the overall system.

In order to analyze this problem, let us consider a simple example, which corresponds to an extended version of the "paradox of the liar" taken from [11] and adapted to a distributed context. Let us consider three agents A_1, A_2 and A_3, and let us suppose that A_1 is interested in knowing whether a certain agent, say Smith, is reliable (for instance, because it wants to exploit an information communicated by Smith). Let us suppose that A_1 has in its knowledge base KB_1 the proposition p_1 = "John told me that Smith is unreliable": A_1 can therefore construct the argument $p_1 \Rightarrow p_2$ for the conclusion $p_2 =$ "Smith is unreliable". Now, suppose

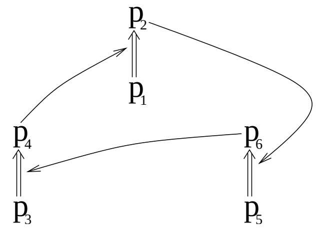

Figure 1: Inference graph of the liar paradox example.

that A_1, in order to verify this argument, asks other agents about the reliability of John, and A_2, whose knowledge base includes p_3 = "Robertson told me that John is unreliable", constructs the argument $p_3 \Rightarrow p_4$ for the conclusion p_4 = "John is unreliable". As soon as A_1 receives information about p_4, it revises its conclusion about Smith, who is then believed reliable. Now, similarly, A_2 acquires information about Robertson from agent A_3, whose knowledge base is supposed to include p_5 = "Smith told me that Robertson is unreliable". As a consequence, on the basis of the argument $p_5 \Rightarrow p_6$ with p_6 = "Robertson is unreliable", A_3 communicates p_6 to A_2, which revises its conclusion about John and communicates in turn this revision to A_1: now the latter has no reason to invalidate its argument about Smith, who is then believed by A_1 to be unreliable. Then, if A_3 asks A_1 information about Smith, it will revise again its belief about Robertson, yielding again a revision of the arguments of A_2 and A_1; but this changes the belief of A_1 about Smith, yielding again a revision of the arguments. It can be easily seen that, in this situation, the system will never reach a stable state, but each agent will continue to revise the truth-value of its conclusion.

The example described above can be modeled as in Figure 1, where double arrows $\Rightarrow$ represent inferential steps while simple arrows $\rightarrow$ denote a relation of attack between arguments. This representation corresponds to the so-called *inference graph* introduced in [10]:

Definition 1. *An inference graph is a triple $\mathcal{IG} = \langle V_I, R_I, R_{DI} \rangle$ where:*

- *V_I are the arguments produced (identified with their conclusions);*
- *R_I are the* inference edges, *i.e. $R_I = \{\langle \sigma_i, \sigma_j \rangle \in V_I \times V_I : \sigma_i \in IMM(\sigma_j)\}$ (where for each $\alpha \in V_I$, $IMM(\alpha)$ is the set of premises from which α has been inferred by means of an inference rule);*
- *R_{DI} are the* defeat edges, *i.e. $R_{DI} = \{\langle \sigma_i, \sigma_j \rangle \in V_I \times V_I :$ concl(σ_i) attacks $\sigma_j\}$, where the relation of 'attack' depends on the semantics of the language at hand.*

Given an argument $\tau \in V_I$, the set of its *subarguments*, denoted as $\mathrm{sub}(\tau)$, is identified with the set of inference-ancestors of τ, i.e. $\mathrm{sub}(\tau) = \tau \cup \bigcup_{\alpha \in IMM(\tau)} \{\mathrm{sub}(\alpha)\}$.

In our model, different agents perform inferential activity and defeat status computation independently of each other. Moreover, as far as the distribution of arguments is concerned, we assume that each process puts forward exactly one argument (therefore it can be identified with a node of the inference graph), and computes its defeat status by acquiring from other processes information about the status assignment of its defeaters. Finally, we claim that the defeat status assignment to the whole set of arguments, which emerges from the autonomous choices of single processes that exploit local information, has to satisfy a well-founded semantics of argumentation, i.e. it must be the one which would be computed by a sound centralized algorithm exploiting global information. However, since we do not assume any kind of synchronism, different processes may start their computation at different times, therefore the initial state of the system cannot be determined in advance. Moreover, processes can be added or removed dynamically, corresponding to the addition of arguments (inferential activity) and the addition or removal of premises (perceptual activity), and the system has to react in such a way that a new correct status assignment is reached in a finite amount of time.

In the following section we introduce algorithms for defeat status computation that can guarantee that a distributed argumentation system is *self-stabilizing*, i.e. a network of processes which, when started from an arbitrary (and possibly illegal) initial state, always returns to a legal state in a finite number of steps [4].

3 Self-stabilizing algorithms

In [2, 1] we have shown the feasibility of distributed argumentation by considering a different kind of representation, called *defeat graph*, where nodes denote arguments and inference-edges are not explicitely represented. In particular, we have devised two self-stabilizing algorithms for defeat status computation, one specifically tailored to *rebutting defeat* [10] (i.e. where the attack arises only from contradicting conclusions) and the other able to deal also with *undercutting defeat* [10] (i.e. where arguments attack the applicability condition of a defeasible rule). In this paper, we extend our previous results to the inference graph representation: this allows us to model the relationship between an argument and the subarguments it is derived from, and in particular argumentation activity performed by an agent exploiting information acquired from other agents.

In order to introduce the semantics of the underlying argumentation framework, we have to define the relation of *defeat* between arguments (identified with nodes of the inference graph).

Definition 2. *Given an inference graph $\mathcal{IG} = \langle V_I, R_I, R_{DI} \rangle$, an argument $\sigma \in V_I$ defeats another argument $\tau \in V_I$ iff there is a subargument $\underline{\tau}$ of τ such that σ attacks $\underline{\tau}$, i.e. $\langle \sigma, \underline{\tau} \rangle \in R_{DI}$.*

We adopt a *single status approach* to the argumentation semantics [13], i.e. the defeat status of the arguments is defined in such a way that there is always exactly one possible way to assign them a status. In a distributed environment, this seems to be more viable than a multiple status approach, since it might be difficult for several asynchronous processes to consider different global status assignments and subsequently to evaluate the justification of their arguments on that basis. More specifically, we adopt the *grounded semantics* introduced in [5], which partitions the nodes (i.e. the arguments) into three classes:

- undefeated (UNDEF), namely justified;
- defeated (DEF), namely not justified;
- provisionally defeated (PROV), denoting a controversial situation, as in the case in which there are two equally believable counterarguments, so that neither of them should be justified (in the liar paradox example presented above, all the three arguments should be provisionally defeated).

We define the grounded semantics inductively following Pollock's approach [10], by introducing the notion of level:

Definition 3. *Given an inference graph $\mathcal{IG} = \langle V_I, R_I, R_{DI} \rangle$,*

- *All arguments of V_I are* in *at level* 0.
- *An argument of V_I is* in *at level $n+1$ iff it is not defeated by any argument* in *at level n.*

Definition 4. *Given an inference graph $\mathcal{IG} = \langle V_I, R_I, R_{DI} \rangle$, the defeat status of the arguments of V_I is defined as follows:*

- *An argument is* undefeated *iff there is a level m such that for every $n \geq m$, the argument is* in *at level n.*
- *An argument is* defeated *iff there is a level m such that for every $n \geq m$, the argument is* out *at level n.*
- *An argument is* provisionally defeated *iff there is no level m such that the argument is* in *at all higher levels or* out *at all higher levels.*

3.1 The case of rebutting defeat

In this subsection, we describe a distributed self-stabilizing algorithm which can handle only the case of rebutting defeat.

We assume that each argument α has a strength value, denoted as strength(α), which represents the conclusive force of the argument and is computed on the basis of strength values associated to the rules of inference and premises used in its construction. In this respect, we do not commit to any particular criterion for the computation of the strength of arguments, we only assume that any argument cannot be strictly stronger that any of its subarguments:

$$\forall \underline{\alpha} \in \text{sub}(\alpha),\ \text{strength}(\underline{\alpha}) \geq \text{strength}(\alpha) \tag{1}$$

This condition, corresponding to one of the three axioms on strength introduced in [15], is sufficiently general to ensure that no interesting distribution of strength is excluded beforehand. The strength of the arguments plays a role in the determination of the attack relation between arguments, namely in an inference graph $\mathcal{IG} = \langle V_I, R_I, R_{DI} \rangle$ we have that

$$R_{DI} = \{\langle \sigma_i, \sigma_j \rangle \in V_I \times V_I : \text{concl}(\sigma_i) = \neg\text{concl}(\sigma_j) \wedge \text{strength}(\sigma_i) \geq \text{strength}(\sigma_j)\} \tag{2}$$

Conditions 1 and 2 entail specific topological properties of the inference graph that can be exploited in the definition of the algorithm.

Let us introduce two preliminary definitions. Given a node α of an inference graph $\mathcal{IG} = \langle V_I, R_I, R_{DI} \rangle$, the set of its *direct-defeaters* is denoted as

$$\text{d-parents}(\alpha) = \{\beta \mid (\beta, \alpha) \in R_{DI}\}$$

Given an inference graph $\mathcal{IG}$ the *strongly connected components* of $\mathcal{IG}$ are the equivalence classes of vertices under the relation of path-equivalence, where two nodes are path-equivalent if it is possible to reach each other by means of a path made up of inference and/or defeat edges. By definition a node is path-equivalent to itself, therefore any node $\alpha \in V_I$ belongs to a strongly connected component, denoted as SCC(α).

The following lemma follows (proof is omitted due to space limitation):

- If $\exists \beta \in IMM(\alpha) : \mathrm{s}[\beta] = \mathrm{DEF}$ then $\mathrm{s}[\alpha] := \mathrm{DEF}$
- If $\exists \beta \in IMM(\alpha) : \mathrm{s}[\beta] = \mathrm{PROV}$ and $\forall \gamma \in IMM(\alpha)\ \mathrm{s}[\gamma] \neq \mathrm{DEF}$ then

$$\mathrm{s}[\alpha] := \begin{cases} \mathrm{DEF} & \text{if } \exists \beta \in \text{superiors}(\alpha) : \mathrm{s}[\beta] = \mathrm{UNDEF} \\ \mathrm{PROV} & \text{otherwise} \end{cases}$$

- If $IMM(\alpha) = \emptyset$ or $\forall \beta \in IMM(\alpha)\ \mathrm{s}[\beta] = \mathrm{UNDEF}$ then
  - If $\exists \beta \in \text{superiors}(\alpha) : \mathrm{s}[\beta] = \mathrm{UNDEF}$ then $\mathrm{s}[\alpha] := \mathrm{DEF}$
  - If $\exists \beta \in \text{superiors}(\alpha) : \mathrm{s}[\beta] = \mathrm{PROV}$ and $\forall \gamma \in \text{superiors}(\alpha)\ \mathrm{s}[\gamma] \neq \mathrm{UNDEF}$ then $\mathrm{s}[\alpha] := \mathrm{PROV}$
  - If $\forall \gamma \in \text{superiors}(\alpha)\ \mathrm{s}[\gamma] = \mathrm{DEF}$ or $\text{superiors}(\alpha) = \emptyset$ then

$$\mathrm{s}[\alpha] := \begin{cases} \mathrm{UNDEF} & \text{if } \forall \gamma \in \text{contenders}(\alpha)\ \mathrm{s}[\gamma] = \mathrm{DEF} \\ \mathrm{PROV} & \text{otherwise} \end{cases}$$

Figure 2: The self-stabilizing algorithm to deal with rebutting defeat.

Lemma 1. *Given an inference graph $\mathcal{IG} = \langle V_I, R_I, R_{DI} \rangle$, and given $\alpha, \beta \in V_I$ such that $\alpha \in SCC(\beta)$, if $\langle \alpha, \beta \rangle \in R_{DI}$ then also $\langle \beta, \alpha \rangle \in R_{DI}$.*

The basis of the algorithm is as follows. Given a node α, its direct defeaters can be partitioned in two sets, namely superiors(α), whose elements have a strength strictly greater than α, and contenders(α), whose elements have the same strength as α and therefore are attacked by α in turn. Starting from an arbitrary initial state, each node α continuously monitors the assignments of its defeaters and of its sub-arguments, and every time α detects a change, it updates its defeat status according to the rules shown in Figure 2.

The correctness proof of this algorithm is organized in two parts. First, we prove the so-called partial correctness, i.e. that if the algorithm terminates then the resulting global defeat status assignment is correct.

Proposition 1. *Given an inference graph, a defeat status assignment which satisfies the rules of Figure 2 is unique and satisfies Definition 4.*

Proof. **(Sketch.)** In [2, 6], we have proved that the same result holds for a defeat status assignment of the so-called *defeat graph*, provided that a certain set of conditions, called supercoherence conditions, are satisfied. The defeat graph has the same nodes as the inference graph, but it has just one kind of edges, denoting the relation of defeat between arguments. Now, it can be proved that if the considered status assignment satisfies the rules of Figure 2, then it also satisfies the supercoherence conditions with reference to the defeat graph. As a consequence, taking into account the result concerning the defeat graph, the thesis follows immediately. □

Then, taking into account Lemma 1, we prove termination of the algorithm (in the following, proofs will be omitted due to space limitation). Given a strongly connected component S, we denote as parents$^*(S)$ the set parents$^*(S) = \{\alpha \mid \alpha \notin S \wedge \exists \beta \in S : \langle \alpha, \beta \rangle \in (R_I \cup R_{DI})\}$.

Proposition 2. *Given an inference graph $\mathcal{IG}$ and one of its strongly connected components S, if $\forall \gamma \in$ parents$^*(S)$ γ is stable (i.e. it does not change its state any more) then all the nodes of S become stable in a finite amount of time.*

The correctness of the algorithm easily follows from the above propositions. In fact, considering the strongly connected components of an inference graph as single nodes, the obtained graph is acyclic. As a consequence, termination of the algorithm easily follows by inductive application of Proposition 2. Moreover, in the termination state, where each process does not update its state any more, all the conditions of Figure 2 are satisfied, therefore by Proposition 1 the global status assignment is the one prescribed by the grounded semantics.

3.2 The General Case

In the approach proposed in the previous subsection, defeaters attack other arguments by denying their (possibly intermediate) conclusions (rebutting defeaters). However, there is another class of defeaters, namely those which prevent the acceptance of other arguments by attacking the connection between premises and conclusion of a defeasible rule used by them (undercutting defeaters). For instance, in the example presented in Section 2 arguments attack the applicability condition of rules of inference of the kind 'X tells Y ⇒ Y', without denying their conclusions. In fact, if a witness asserts Y we have a defeasible reason to believe Y. If the witness turns out to be unreliable the reason to believe Y is invalidated, even if we have not a reason supporting $\neg Y$.

Undercutting defeaters invalidate the constraints on the topology of the inference graph discussed above, and this prevents the algorithm presented in Section 3.1 to work: the right status assignment is not always enforced, and the system is not even stable, i.e. there are initial states and particular process scheduling decisions, that prevent computation to terminate. It is easy to verify that this happens in the example considered in Section 2.

In order to enforce a correct behavior even in case of generic inference graphs, an additional variable $\mathrm{d}\left[\alpha\right]$ is introduced for every node α, whose domain is the set of natural numbers. The variable $\mathrm{d}\left[\alpha\right]$ is meaningful only when $\mathrm{status}\left[\alpha\right]$ is either DEF or UNDEF. The state of α is therefore identified by variables $\mathrm{status}\left[\alpha\right] \in \{\mathrm{DEF}, \mathrm{UNDEF}, \mathrm{PROV}\}$ and $\mathrm{d}\left[\alpha\right]$, and for the sake of brevity will be indicated as $\mathrm{s}\left[\alpha\right] = \mathrm{DEF}\left(x\right) \mid \mathrm{UNDEF}\left(x\right) \mid \mathrm{PROV}$, where x is the value of $\mathrm{d}\left[\alpha\right]$. The algorithm is presented in Figure 3, where $N \geq 0$ is a constant and the following notations are used:

$$\mathit{DIMM}(\alpha) = \{\beta \in \mathit{IMM}(\alpha) \mid \mathrm{status}\left[\beta\right] = \mathrm{DEF}\}$$
$$\mathit{UIMM}(\alpha) = \{\beta \in \mathit{IMM}(\alpha) \mid \mathrm{status}\left[\beta\right] = \mathrm{UNDEF}\}$$
$$\mathit{DDIR}(\alpha) = \{\beta \in \text{d-parents}(\alpha) \mid \mathrm{status}\left[\beta\right] = \mathrm{DEF}\}$$
$$\mathit{UDIR}(\alpha) = \{\beta \in \text{d-parents}(\alpha) \mid \mathrm{status}\left[\beta\right] = \mathrm{UNDEF}\}$$
$$\mathit{DISTU}(\alpha) = \bigcup_{\beta \in \mathit{UIMM}(\alpha)} \{\mathrm{d}\left[\beta\right]\} \ \cup \ \bigcup_{\beta \in \mathit{DDIR}(\alpha)} \{\mathrm{d}\left[\beta\right] + 1\}$$
$$\mathit{DISTD}(\alpha) = \bigcup_{\beta \in \mathit{DIMM}(\alpha)} \{\mathrm{d}\left[\beta\right]\} \ \cup \ \bigcup_{\beta \in \mathit{UDIR}(\alpha)} \{\mathrm{d}\left[\beta\right] + 1\}$$
$$\mathit{EXUNDEF}(\alpha) = \begin{cases} \mathrm{TRUE} & \text{if } \mathit{DIMM}(\alpha) \neq \emptyset \text{ or } \mathit{UDIR}(\alpha) \neq \emptyset \\ \mathrm{FALSE} & \text{otherwise} \end{cases}$$
$$\mathit{EXPROV}(\alpha) = \begin{cases} \mathrm{TRUE} & \text{if } \exists \beta \in (\mathit{IMM}(\alpha) \cup \text{d-parents}(\alpha)) : \mathrm{status}\left[\beta\right] = \mathrm{PROV} \\ \mathrm{FALSE} & \text{otherwise} \end{cases}$$

- If $EXUNDEF(\alpha)$ then

$$\mathrm{s}\left[\alpha\right] := \begin{cases} \mathrm{DEF}\left(\min DISTD(\alpha)\right) & \text{if } \min DISTD(\alpha) \leq N \\ \mathrm{PROV} & \text{otherwise} \end{cases}$$

- If $\neg EXUNDEF(\alpha)$ and $EXPROV(\alpha)$ then $\mathrm{s}\left[\alpha\right]$:= PROV
- If $\neg EXUNDEF(\alpha)$ and $\neg EXPROV(\alpha)$ then

$$\mathrm{s}\left[\alpha\right] := \begin{cases} \mathrm{UNDEF}\left(\max DISTU(\alpha)\right) & \text{if } \max DISTU(\alpha) \leq N \\ \mathrm{PROV} & \text{otherwise} \end{cases}$$

Figure 3: The general self-stabilizing algorithm.

The underlying idea is the following. If an argument α is defeated in the right status, according to Definition 4 there is a level k at which it becomes stably out, i.e. α is *in* at level $k-1$ and *out* at all levels $m \geq k$. In this case, it can be proved that α has at least a defeater β, denoted as determinant node for α, which becomes stably in at level $k-1$: in a sense, β is the cause for α being defeated. In case an argument α is undefeated, by Definition 4 either it has no defeaters, or it becomes stably in at a level $k > 0$. In the latter case, it can be proved that all the defeaters of α become stably out at lower than k levels, and there is a defeater β which becomes stably out at level $k-1$: again, β is the determinant node for α. Basically, the algorithm works as follows: each node α updates its defeat status according to simple conditions entailed by Definition 4, and it updates $\mathrm{d}\left[\alpha\right]$ to the proper value $\mathrm{d}\left[\beta\right]+1$, where β is the node which α recognizes as its determinant node. The constant N represents the maximum level at which a generic node can become stably in or stably out, and plays a role when a node has to be provisionally defeated in the termination state: basically, if $\mathrm{d}\left[\beta\right]+1 > N$, then α updates its state to PROV.

Also in this case, the correctness proof is organized in two parts.

Proposition 3. *Let $\mathcal{IG}$ be an inference graph $\mathcal{IG} = \langle V_I, R_I, R_{DI}\rangle$, and let s be a status assignment $s : V_I \rightarrow \{DEF\,(x) \mid UNDEF\,(x) \mid PROV\}$ satisfying the rules of Figure 3. For each $\alpha \in V_I$, if α becomes stably in (out) at a level $m \leq N$ then status$\left[\alpha\right]$ is UNDEF (DEF), otherwise status$\left[\alpha\right] = $ PROV.*

Proof. **(Sketch.)** The proof can be obtained in a similar way as the one of Proposition 1, taking into account that an analogous result has been proved in [1] for a certain set of conditions holding for the defeat graph representation. □

This proposition yields easily the partial correctness of the algorithm provided the constant N is sufficiently high:

Proposition 4. *Let $\mathcal{IG} = \langle V_I, R_I, R_{DI}\rangle$ be an inference graph, and let MAXLEVEL $(\mathcal{IG})$ be the maximum level at which an argument of V_I becomes stably in or stably out. If $N \geq$ MAXLEVEL $(\mathcal{IG})$, then in the termination state of the algorithm the defeat status assignment is right according to Definition 4.*

Termination is proved inductively as follows. Let us say that, at a given instant of time, the computation is *stable at level k* iff all nodes α such that $\mathrm{d}\left[\alpha\right] = k$ will not make moves any more, and no node β will update $\mathrm{d}\left[\beta\right]$ to k. It is possible to prove the following proposition:

Proposition 5. *If the computation is stable at level k, with $0 \leq k < N$, then the computation becomes stable at level $k+1$ in a finite amount of time.*

As it can be easily verified, each node α cannot make any move that update d $[\alpha]$ to a value strictly greater than N, therefore termination of the algorithm can be proved by applying inductively the above proposition. Moreover, it can be proved that MAXLEVEL $(\mathcal{IG}) \leq n-1$, where n is the number of nodes of the inference graph, therefore the algorithm turns out to be correct provided that $N \geq n-1$.

4 Conclusions

In this paper, we have presented two self-stabilizing algorithms for defeat status computation, one specifically tailored to rebutting defeat, and the other able to deal with any form of defeat. The aim of this proposal is to extend to inference graphs the results on self-stabilizing algorithms for defeat status computation previously applied to defeat graphs. This enables the application of these algorithms in a multi-agent context, allowing to overcome some limitations present in previous approaches to multi-agent argumentation.

To compare the proposed algorithms, we note that the more general algorithm requires a guess on N, which represents the maximum length of the chains of the defeat relationships that can be handled by the algorithm: it can be proved that all the arguments having more than N arguments preceding them in their defeat chain are assigned the status of PROV, while all other ones get the correct assignments.

Among future research directions we mention the experimentation of the proposed model in a multi-agent architecture and the study of an integrated approach to self-stabilizing argumentation systems that combines the advantages of both the proposed algorithms.

References

[1] P. Baroni and M. Giacomin. Argumentation through a distributed self-stabilizing approach. *Journal of Experimental and Theoretical Artificial Intelligence*. To appear.

[2] P. Baroni and M. Giacomin. A distributed self-stabilizing algorithm for argumentation. In *Proceedings of the 2001 International Parallel and Distributed Processing Symposium (IPDPS2001)*, San Francisco, CA, 2001. IEEE Press.

[3] P. Baroni, M. Giacomin, and G. Guida. Extending abstract argumentation systems theory. *Artificial Intelligence*, 120(2):251–270, 2000.

[4] S. Dolev. *Self-Stabilization*. MIT Press, Cambridge, MA, 2000.

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

[6] M. Giacomin. Self-stabilizing distributed algorithms for defeat status computation in argumentation. In V. Marik, O. Stepankova, H. Krautwurmova, and M. Luck, editors, *Multi-Agent Systems and Application II, Selected Revised Papers: 9th ECCAI-ACAI/EASSS 2001, AEMAS 2001, HoloMAS 2001 (LNAI 2322)*, pages 136–145. Springer-Verlag, Prague, Czech Republic, 2002.

[7] N. R. Jennings, S. Parsons, P. Noriega, and C. Sierra. On argumentation-based negotiation. In *Proceedings of the International Workshop on Multi-Agent Systems*, Boston, USA, 1998.

[8] S. Kraus, K. Sycara, and A. Evenchik. Reaching agreements through argumentation: A logical model and implementation. *Artificial Intelligence*, 104(1–2):1–69, 1998.

[9] S. Parsons, C. Sierra, and N. Jennings. Agents that reason and negotiate by arguing. *Journal of Logic and Computation*, 8(3):261–292, 1998.

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

[11] J. L. Pollock. Justification and defeat. *Artificial Intelligence*, 67:377–407, 1994.

[12] J. L. Pollock. The logical foundations of goal-regression planning in autonomous agents. *Artificial Intelligence*, 106(2):267–334, 1998.

[13] H. Prakken and G. A. W. Vreeswijk. Logics for defeasible argumentation. In D. M. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic, Second Edition*. Kluwer Academic Publishers, Dordrecht, 2001.

[14] C. Sierra, N. R. Jennings, P. Noriega, and S. Parsons. A framework for argumentation-based negotiation. In M. P. Singh, A. Rao, and M. J. Wooldridge, editors, *Proceedings of the 4th International Workshop on Agent Theories, Architectures, and Languages (ATAL-97)*, volume 1365 of *LNAI*, pages 177–192, Berlin, 1998. Springer-Verlag.

[15] G. A. W. Vreeswijk. Abstract argumentation systems. *Artificial Intelligence*, 90(1–2):225–279, 1997.