A Dynamic Approach for Combining Abstract Argumentation Semantics – Technical Report

Jérémie Dauphin*, Marcos Cramer, and Leendert van der Torre*

Abstract Abstract argumentation semantics provide a direct relation from an argumentation framework to corresponding sets of acceptable arguments, or equivalently to labeling functions. Instead, we study step-wise update relations on argumentation frameworks whose fixpoints represent the labeling functions on the arguments. We make use of this dynamic approach in order to study novel ways of combining abstract argumentation semantics. In particular, we introduce the notion of a *merge* of two argumentation semantics, which is defined in such a way that the merge of the preferred and the grounded semantics is the complete semantics. Finally we consider how to define new semantics using the merge operator, in particular how meaningfully combine features of naive-based and complete-based semantics.

1 Introduction

Following the methodology in non-monotonic logic, logic programming and belief revision, formal argumentation theory defines a diversity of semantics. This diversity has the advantage that a user can select the semantics best fitting her application, but it leads also to various practical challenges. First of all, how to choose among the considerable number of semantics existing in the argumentation literature for a particular application? The behaviour of semantics on examples can already be

---

Jérémie Dauphin
University of Luxembourg, Luxembourg; e-mail: jeremie.dauphin@uni.lu

Marcos Cramer
TU Dresden, Germany; e-mail: marcos.cramer@uni.lu

Leendert van der Torre
University of Luxembourg, Luxembourg; e-mail: leon.vandertorre@uni.lu

* The work of Jérémie Dauphin and Leendert van der Torre was supported by the H2020 Marie Skłodowska-Curie grant number 690974 for the project MIREL.

insightful, and Baroni and Giacomin [3] address the need for more systematic comparison of semantics based on a set of principles. However, what to do when no currently considered semantics is perfect? May there be a better semantics that has not been discovered yet? How to guide the search for new and hopefully better argumentation semantics? In this paper, we propose a new approach: the combination of abstract argumentation semantics.

1. How to combine two abstract semantics to yield a third semantics?
2. How to obtain the complete semantics by combining the preferred and grounded semantics?
3. Can we meaningfully combine features of naive-based and complete-based semantics?

Concerning our first research question, there are various ways in which abstract argumentation semantics can be combined. For example, in multi-sorted argumentation [9, 1, 8], one part of the framework can be evaluated according to for example grounded semantics, whereas another part of the framework is evaluated according to the preferred semantics. Another approach manipulates directly the sets of extensions. For example, the grounded and preferred can be combined by simply returning both the grounded and preferred extensions. Neither of these two approaches is very satisfactory. For multi-sorted argumentation, we need to specify explicitly which semantics must be applied to which part of the framework. For the direct combination method, the approach seems too coarse-grained and the number of ways to combine semantics seems relatively limited.

We therefore introduce a dynamic approach in this paper, which is based on the labeling approach to argumentation semantics, in which the three labels *in*, *out* and *undec* are used. In our dynamic approach, we define step-wise versions of standard semantics based on epistemic labelings, which associate with each argument a nonempty *set* of labels from $\{in, out, undec\}$. Intuitively, the set represents uncertainty about the label. We start with labeling each argument of the framework with the set $\{in, out, undec\}$. This represents that we do not know the labeling yet. Then in each step we refine the labels by removing some of the labels. Finally we end up with a single label for each argument, and thus with a standard labeling. To represent the possibility of multiple extensions, the steps are not deterministic. The steps are represented by an abstract *update relation*, which mathematically is simply a binary relation among epistemic labelings. Note that there are many distinct update relations representing the same standard semantics, and it is this additional expressive power that we will use in our first approach to combining abstract argumentation semantics.

Concerning our second research question, it is well known that the grounded semantics returns the smallest complete extension, and that the preferred semantics returns maximal complete extensions. This suggests that by combining the grounded and preferred semantics, we can again recover all complete extensions. Note that there may be complete extensions that are neither minimal nor maximal, and that it is therefore non-trivial to recover all the complete extensions using the grounded and the preferred semantics, without loosing any complete extensions. Though the derivation of the complete semantics from the grounded and preferred semantics does not serve any practical purpose, it serves to show that our dynamic semantic framework has sufficient expressive power to combine abstract semantics. We therefore pursue this second question.

Note that we do not claim it to be possible to retrieve the full set of complete extensions from preferred and grounded extensions alone, as they do not provide sufficient information, even when represented as labelings. Indeed, some argumentation frameworks have the same preferred and the same grounded labelings, yet differ in their complete labelings. We present two such frameworks in Section 2. Hence, the approaches we propose still take the structure of the framework into account when combining the different semantics.

Concerning the third research question, note that recently naive-based semantics like stage semantics [11] and CF2 semantics [4] have received some attention, for example in the work of Gaggl and Dvořák [7], who define a new semantics (*stage2*) that combines features of stage and CF2 semantics, and in the work of Cramer and Guillaume [5], who performed an empirical study that showed that these naive-based semantics are better predictors of human argument acceptance than complete-based semantics like the grounded and preferred semantics.

For argumentation frameworks without odd cycles, the stage semantics fully agrees with the preferred semantics. One difference between the preferred semantics and the stage semantics is that the stage semantics generally provides a way to select accepted arguments even when odd cycles are around, whereas the preferred semantics tends to mark as *undecided* all arguments that are in an odd cycle or attacked by an odd cycle. One difference between the preferred semantics and the complete semantics is that the complete semantics allows one to locally not make choices for some unattacked even cycles while making choices for other unattacked even cycles, whereas the in the preferred semantics one has to make choices for all unattacked even cycles. This motivates the following research question: Is there a sensible semantics that allows one to locally make choices for some unattacked odd or even cycles while not making choices for other unattacked odd or even cycles.

The layout of this paper is as follows. After providing some preliminaries about argumentation semantics in Section 2, we introduces our dynamic approach based on epistemic labelings and update relations in Section 3. Section 4 addresses the second research question by showing how grounded and preferred semantics can be combined to obtain the complete semantics using an algorithmic approach to updates. As this approach is dependent on the choice of algorithm on which the update relation is based, we proceed in Section 5 to defining the *merge* of two argumentation semantics, a modification of our first approach that is applicable to any pair of semantics independently of any algorithmic considerations. In Subsection 5.1, we motivate the definition of the merge operator by considering how to use it to get the complete semantics from the grounded and preferred semantics without adding any algorithmic information. In Subsection 5.2, we show how the merge operator can be used to give rise to novel argumentation semantics, and, in particular, how it can be used to meaningfully combine features of naive-based and complete-based semantics. We conclude with an overview of further work in Section 6.

2 Preliminaries

An argumentation framework (AF) is a directed graph $\langle A, R\rangle$, where A is called the set of arguments, and R is called the attack relation. In this work, we only consider AFs. Standard argumentation semantics come in two variants. Extension-based semantics associates with each AF a set of extensions (sets of the arguments). Labelling-based semantics attribute to each argument the label *in*, *out* or *undecided*. The two approaches are inter-definable, in the sense that an argument is labeled *in* when it is in the extension, it is labeled *out* when it is not in the extension and there is an argument in the extension attacking it, and it is *undecided* otherwise. Our dynamic approach uses an epistemic labelling, which associates with each argument a nonempty *set* of labels. Intuitively, the set represents uncertainty about the label.

We assume familiarity with 3-labeling semantics of argumentation frameworks as defined in [2]. Note that we will make use of the multi-labeling approach, where a set of labels is assigned to each argument. This set represents the possible labels for a given argument. The standard approach corresponds to the case where arguments are given singleton sets as labels.

Definition 1. Let $F = \langle A, R\rangle$ be an AF. We say that any function L from A to $\{in, out, undec\}$ is a 3-labeling of F.

The 3-labeling approach makes use of the notions of *legal labels*.

Definition 2. Let $F = \langle A, R\rangle$ be an AF, $a \in A$ an argument and L a 3-labeling of F. We say that a is:

- *legally in* with respect to L iff $L(a) = in$ and for all $b \in A$ such that $(b, a) \in R$, $L(b) = out$;
- *legally out* with respect to L iff $L(a) = out$ and for some $b \in A$ such that $(b, a) \in R$, $L(b) = in$;
- *legally undecided* with respect to L iff $L(a) = undec$ and for all $b \in A$ such that $(b, a) \in R$, $L(b) \neq in$ and for at least one such b, $L(b) = undec$.

If all arguments in A are legally labeled with respect to L, then we say that L is a *complete labeling* of F. A complete labeling with a minimal set of *in*-labeled arguments is called a *grounded labeling*. A complete labeling with a maximal set of *in*-labeled arguments is called a *preferred labeling*. A complete labeling without *undec*-labeled arguments is called a *stable labeling*. A complete labeling with a minimal set of *undec*-labeled arguments is called a *semi-stable labeling*.

An *argumentation semantics* is a function that maps an argumentation framework to a set of labelings. The above defined notions give rise to the *complete*, *preferred* and *grounded* argumentation semantics. We call an argumentation semantics σ *complete-based* if all σ-labelings are complete labelings.

We will also refer to the *stage semantics* defined in its extension-based form by Verheij [11]. We adapt it to the labeling-based form by assigning the label *out* to all arguments that are not in the stage extension in Verheij's definition. This labeling-based form of the stage semantics can be defined as follows:

Definition 3. Let $F = \langle A, R \rangle$ be an AF and L a 3-labeling of F. Define L_{in} to be the set $\{a \in A \mid L(a) = in\}$. Define L^{+}_{in} to be the set $\{a \in A \mid \exists b \in L_{in}.(b,a) \in R\}$. We say that L is a *stage labeling* of F if $L_{in} \cup \text{Ł}^{+}_{in}$ is maximal with respect to set inclusion and $L(a) \neq undec$ for all $a \in A$.

We also make use of the notions of *transitive closure* of a relation and *restriction* of a relation to a subset of its domain.

Definition 4. Let *rel* be a relation. We define the *transitive closure* of *rel* to be the smallest set rel^* such that $rel \subseteq rel^*$ and if $(a,b),(b,c) \in rel^*$, then $(a,c) \in rel^*$.

Definition 5. Let A, B be sets, $A' \subseteq A$ and R a relation from A to B. We define the *restriction* of R to A' to be:

$$R \downarrow_{A'} = \begin{cases} \{(a,b) \in R \mid a,b \in A'\} & \text{if } A = B \\ \{(a,b) \in R \mid a \in A'\} & \text{otherwise} \end{cases}$$

The definition of restriction handles separately two cases: if the domain and range of the relation are the same, it then applies the restriction to both of them, for example in the case of the attack relation of an AF. In the case where the domain and range are different sets, it only performs the restriction on the domain set, for example in the case of a labeling function.

In the introduction, we have pointed out that retrieving the set of complete labelings from the preferred and grounded labelings alone is not feasible. We now provide a concrete example of two argumentation frameworks with the same preferred and grounded labelings, but different complete labelings.

Example 1. Consider the two AFs F_1 and F_2 depicted in Fig. 1. Both have $\{(a, undec), (b, undec), (c, undec), (d, undec)\}$ as their grounded labeling, and $\{(a, in), (b, out), (c, in), (d, out)\}$ and $\{(a, out), (b, in), (c, out), (d, undec)\}$ as their preferred labelings. While these are also all the complete labelings for F_1, F_2 also has $\{(a, in), (b, out), (c, undec), (d, undec)\}$ as a complete labeling which is neither preferred nor grounded. Hence, given nothing other than the preferred and grounded labelings of a framework, it is not feasible to always accurately retrieve the set of complete labelings.

Fig. 1 Two AFs with the same preferred and grounded labelings but different complete labelings.

3 Update relations

Standard labeling semantics provide a direct relation between an argumentation framework and a set of labeling functions, which attach to each argument exactly one label. We will now define update relations, which formalize the idea that the final labelings can be determined in a step-wise fashion. For this purpose, we introduce *epistemic labelings*, which associate with each argument a nonempty *set* of labels from $\{in, out, undec\}$. The intuitive idea is that at a certain step in the update process, the set of labels associated with an argument tells us which labels we consider possible for this argument at this step. The steps in an update relation can be interpreted as moves in a dialogue, or as steps in an algorithm, or as learning a framework, or otherwise. Our dynamic semantic framework does not depend on such particular interpretations.

Notice that it makes little sense to separate the labeling function from the underlying framework, as the labeling is meaningless without it. We will hence consider pairs of argumentation framework and labeling functions.

We define $\mathbb{L} = \{in, out, undec\}$ to be the set of possible *labels*.

Definition 6. We define a *labeled argumentation framework* (LAF) to be a pair $(\langle A, R\rangle, Lab)$ where $\langle A, R\rangle$ is a finite argumentation framework and *Lab* a function from A to $\mathscr{P}(\mathbb{L}) \setminus \{\emptyset\}$, called an *epistemic labeling*. Additionally, let $\mathbb{F}$ be the class of all labeled argumentation frameworks.

Observe that a labeling function cannot assign the empty set of labels to an argument, as the set of labels represents the possible final labels for that argument, and thus the empty set would mean that no label can be attached to it, which prevents us from having a final labeling for the framework.

We now introduce the notions of *initial* and *final* labeled frameworks, which correspond to the starting point and endpoint of a labeling process. In an initial LAF, every label is possible for each argument, while in a final LAF, every argument is assigned a singleton set of labels, representing the fact that a unique label has been selected.

Definition 7. Let $F = (\langle A, R\rangle, Lab)$ be a LAF. If for all $a \in A$, $Lab(a) \in \{\{in\}, \{out\}, \{undec\}\}$, we say that F is *final*. If for all $a \in A$, $Lab(a) = \mathbb{L}$, we say that F is *initial*.

Note that there is a one-to-one correspondence between the epistemic labelings *Lab* of the final LAFs $(\langle A, R\rangle, Lab)$ and the 3-labelings of $\langle A, R\rangle$. This one-to-one correspondence can be formally defined as follows:

Definition 8. Let $\langle A, R\rangle$ be an AF and L a 3-labeling of $\langle A, R\rangle$, define the epistemic labeling $T(L)$ by $T(L)(a) := \{L(a)\}$ for all $a \in A$.

In this section with the basic definitions of our approach, we will be careful to make the formal distinction between a 3-labeling L, the corresponding epistemic labeling $T(L)$ and the corresponding final LAF $(\langle A, R\rangle, T(L))$. In order to improve readability, we will not always make this distinction in later section, but instead identify the 3-labeling L with the corresponding epistemic labeling $T(L)$ and the corresponding final LAF $(\langle A,R\rangle,T(L))$. For example, we might speak of an LAF being a complete labeling of a given argumentation framework, even though formally a complete labeling is a 3-labeling.

We now define a precision ordering on the LAFs based on the subset relation between the argument multi-labels, such that the final LAFs are the most precise and the initial LAFs are the least precise. Note however that only LAFs with the same underlying AF are comparable.

Definition 9. Let $F = (\langle A,R\rangle,Lab)$ and $F' = (\langle A',R'\rangle,Lab')$ be two labeled argumentation frameworks. We say that F is *at least as precise* as F' ($F \geq_p F'$), iff $\langle A,R\rangle = \langle A',R'\rangle$, and for all $a \in A$, $\emptyset \subset Lab(a) \subseteq Lab'(a)$. We say that F is *more precise* than F' ($F >_p F'$) iff $F \geq_p F'$ and $F \neq_p F'$.

We will now define the central notion of this paper, namely *update relations*, i.e. relations between LAFs which, starting from an initial LAF, monotonically increase precision, until a fixpoint is reached, at which point the LAF should be final and correspond to a desired output.

Definition 10. We say that $upd : \mathbb{F} \times \mathbb{F}$ is an *update relation* iff:

- for all $F' \in \mathbb{F}$ such that $upd(F,F')$, $F' \geq_p F$;
- if $upd(F,F)$, then F is final.

We now define correspondence between update relations and direct semantics that formalizes the idea that an update relation can be viewed as a step-wise procedure that gives rise to a certain direct semantics. For this we first need an auxiliary definition.

Definition 11. Let *Rel* be a relation on $\mathbb{F}$ and F an LAF. We say that F is *reachable* in *Rel* iff there exists an initial LAF F_i such that there is a path in *Rel* from F_i to F. We say that F is a *reachable fixpoint* in *Rel* iff F is reachable in *Rel* and $(F,F) \in Rel$.

Definition 12. Let *upd* be an update relation and *sem* a semantics. We say that *upd gives rise* to *sem* iff for each 3-labeling L of $\langle A,R\rangle$, $(\langle A,R\rangle,T(L))$ is a reachable fixpoint in *upd* iff L is a *sem* labeling of $\langle A,R\rangle$.

The following theorem, which easily follows from Definition 10, provides a simple way of combining two given update relations to yield a third update relation:

Theorem 1. *If upd_1 and upd_2 are update relations, than $upd_1 \cup upd_2$ is an update relation.*

In Section 4 we will present an example where combining two update relations with a union operation gives us not only the union of the final labelings reachable by either of them, but also additional labelings. This means that the semantics that $upd_1 \cup upd_2$ gives rise is not necessarily induced by the semantics that upd_1 and upd_2 separately give rise to.

We are now interested in the comparison of updates in terms of precision increase per step, i.e. in the granularity of update relations. The idea is that an update relation is more granular than another if it takes more steps to reach its final LAFs. First of all, notice that such a comparison only makes sense for updates which output the same final LAF, i.e. updates which give rise to the same semantics.

Definition 13. Let *upd* be an update relation. We define the *restriction of upd to relevant paths* ($\overline{upd}$) to be the set of pairs in *upd* that are in some *upd*-path from an initial to a final LAF.

Definition 14. Let upd_1 and upd_2 be two update relations. We say that upd_1 is *at least as fined-grained* as upd_2 ($upd_1 \geq_g upd_2$) iff $\overline{upd_1}^* \supseteq \overline{upd_2}$.

We then abstractly define the *most fine-grained* update relation for a given labeling semantics.

Definition 15. Let *sem* be a labeling semantics. We define mfg_{sem} to be the smallest update relation such that for all update relations *upd* that give rise to *sem*, we have $mfg_{sem} \geq_g upd$.

Lemma 1. *For every standard semantics, there exists a unique* mfg_{sem}.

Proof. Define mfg_{sem} as follows: $(F, F') \in mfg_{sem}$ iff either $F = F'$ is a *sem* labeling, or the following three properties are satisfied:

- $F' >_p F$;
- $\nexists F''$ such that $F' >_p F'' >_p F$;
- there exists a final F_f which is a *sem* labeling such that $F_f \geq_p F'$.

By definition, mfg_{sem} includes all possible links in any relevant path from an initial to a final LAF which encompasses a *sem* labeling. Hence, for any update relation *upd* which gives rise to *sem*, $\overline{mfg_{sem}}^* \supseteq \overline{upd}$. Also, mfg_{sem} includes by definition only pairs which are on a relevant path, as the first alternative adds the endpoints of these paths and the third item of the second alternative ensures that the pairs are on a relevant path. The first and second items of the second alternative ensure also that a minimal amount of pairs are added, making mfg_{sem} as small as possible. □

In subsequent sections, we will need the following notion of a sub-framework:

Definition 16. Let $F = (\langle A, R \rangle, Lab)$ be a LAF and $S \subseteq A$. We define the *sub-framework* of F generated by S to be $Sub(F, S) = (\langle S, R\downarrow_S \rangle, Lab\downarrow_S)$.

4 Case analysis: An algorithmic approach for combining preferred and grounded

In this section, we consider update relations which give rise to the preferred and grounded semantics, and which are motivated by algorithms for computing these semantics that have been described by Dauphin and Schulz [6].

The algorithmic update relation for the grounded semantics first identifies the arguments which are only being attacked by arguments which are already labeled $\{out\}$, labels them as $\{in\}$ and any argument they attack as $\{out\}$, and then repeats this process until no arguments can be further labeled, at which point it will label all remaining arguments as $\{undec\}$.

Definition 17. For any labeled argumentation framework $F = (\langle A,R\rangle, Lab)$, we define the set of *unattacked arguments* to be $unattacked(F) = \{a \in A \mid Lab(a) \supsetneq \{in\} \wedge \forall b \in A.((b,a) \in R \rightarrow Lab(b) = \{out\})\}$.

Definition 18. We define $step\_grnd : \mathbb{F} \times \mathbb{F}$ to be the relation such that $((\langle A,R\rangle, Lab),(\langle A,R\rangle, Lab')) \in step\_grnd$ iff one of the following conditions holds:

- $unattacked((\langle A,R\rangle, Lab)) \neq \emptyset$, and $(\langle A,R\rangle, Lab')$ is the least precise LAF that is more precise than $(\langle A,R\rangle, Lab)$ such that for all $a \in unattacked((\langle A,R\rangle, Lab))$, $Lab'(a) = \{in\}$ and for all $c \in A$ such that $(a,c) \in R$ and $out \in Lab(c)$, $Lab'(c) = \{out\}$.
- $unattacked((\langle A,R\rangle, Lab)) = \emptyset$, there is an $a \in A$ such that $Lab(a) \supsetneq \{undec\}$, and $(\langle A,R\rangle, Lab')$ is the least precise LAF that is more precise than $(\langle A,R\rangle, Lab)$ such that for all $a \in A$ such that $Lab(a) \supsetneq \{undec\}$, $Lab'(a) = \{undec\}$.
- $(\langle A,R\rangle, Lab) = (\langle A,R\rangle, Lab')$ is a final LAF.

Note that before labeling arguments out, we ensure that it is a possibility, e.g. by having the condition $out \in Lab(c)$ in the first item of Definition 18. While this requirement will straightforwardly be fulfilled in any reachable LAF, it is required to ensure that the increase in precision is satisfied even for those LAFs that are not reachable from an initial LAF.

The following lemma now easily follows from the above definition:

Lemma 2. *step_grnd is an update relation.*

The following theorem states that *step_grnd* does indeed have the intended property that it gives rise to the grounded labeling:

Theorem 2. *step_grnd gives rise to the grounded semantics.*

Proof sketch. One can easily see that whenever *step_grnd* changes the label of an argument a to $\{in\}, \{out\}$ or $\{undec\}$, argument a is legally labeled $\{in\}, \{out\}$ or $\{undec\}$ respectively. Thus the final labeling reachable in *step_grnd* is a complete labeling. To show that the final labeling reachable in *step_grnd* is the complete labeling that maximizes *undec*, suppose that there is some $A' \subseteq A$ and some complete labeling Lab of $\langle A,R\rangle$ such that for all $a \in A'$, $Lab(a) = undec$. It is now enough to show that *step_grnd* never labels any $a \in A'$ $\{in\}$ or $\{out\}$. Consider for a proof by contradiction the first step where *step_grnd* does label some $a \in A'$ $\{in\}$. Since a is legally labeled *undec* in *Lab*, some $a' \in A'$ must attack a, so by Definitions 17 and 18, a' must already be labeled $\{out\}$ in a previous step, which is a contradiction. □

Let us now examine *step_pref*, a similar update relation which computes the preferred labelings. For this, we first define the notion of minimal non-trivial admissible sets of arguments.

Definition 19. Let $F = (\langle A,R\rangle, Lab)$ be a labeled argumentation framework. We define $min\_adm(F) \subseteq \mathscr{P}(A)$ to be the set of all minimal subsets S of A that satisfy the following conditions:

- $S \neq \emptyset$;
- for all $a \in S$, $Lab(a) \supsetneq \{in\}$;
- for all $a,b \in S$, $(a,b) \notin R$;
- for all $a \in S$ and $b \in A$ such that $Lab(b) \neq \{out\}$ and $(b,a) \in R$, there exists $a' \in S$ such that $(a',b) \in R$.

So the function $min\_adm(F)$ returns all minimal non-empty admissible sets of arguments whose label could still be changed to $\{in\}$. The update relation *step_pref* proceeds with a process similar to the one in the *step_grnd* update, iteratively labeling $\{in\}$ all arguments with all attackers $\{out\}$, and then labeling all arguments attacked by those as $\{out\}$. The difference lies in the case where $unattacked(F)$ is empty, where the preferred update relation looks for minimal non-trivial admissible sets, label them $\{in\}$ and arguments they attack $\{out\}$.

Definition 20. We define $step\_pref : \mathbb{F} \times \mathbb{F}$ to be the relation such that $((\langle A,R\rangle, Lab),(\langle A,R\rangle, Lab')) \in step\_pref$ iff one of the following conditions holds:

- $unattacked((\langle A,R\rangle, Lab)) \neq \emptyset$, and $(\langle A,R\rangle, Lab')$ is the least precise LAF that is more precise than $(\langle A,R\rangle, Lab)$ such that for all $a \in unattacked((\langle A,R\rangle, Lab))$, $Lab'(a) = \{in\}$ and for all $c \in A$ such that $(a,c) \in R$ and $out \in Lab(c)$, $Lab'(c) = \{out\}$.
- $unattacked((\langle A,R\rangle, Lab)) = \emptyset$, and for some $S \in min\_adm(F)$, $(\langle A,R\rangle, Lab')$ is the least precise LAF that is more precise than $(\langle A,R\rangle, Lab)$ such that for all $a \in S$, $Lab'(a) = \{in\}$ and for all $c \in A$ such that $(a,c) \in R$ and $out \in Lab(c)$, $Lab'(c) = \{out\}$.
- $unattacked((\langle A,R\rangle, Lab)) = min\_adm(F) = \emptyset$, and there is an $a \in A$ such that $Lab(a) \supsetneq \{undec\}$, and $(\langle A,R\rangle, Lab')$ is the least precise LAF that is more precise than $(\langle A,R\rangle, Lab)$ such that for all $a \in A$ such that $Lab(a) \supsetneq \{undec\}$, $Lab'(a) = \{undec\}$.
- $(\langle A,R\rangle, Lab) = (\langle A,R\rangle, Lab')$ is a final LAF.

The following lemma now easily follows from the above definition:

Lemma 3. *step_pref is an update relation.*

The following theorem, which can be proved in a similar way as Theorem 2, states that *step_pref* has the intended property that it gives rise to the preferred labeling:

Theorem 3. *step_pref gives rise to the preferred semantics.*

We now find the interesting result that combining these two update relations with a union operation gives us not only the union of the final labelings reachable by either of them, but also the complete labelings which are neither grounded nor preferred:

Theorem 4. *$step\_grnd \cup step\_pref$ gives rise to the complete semantics.*

Proof sketch. One can easily see that any final labeling reachable in $step\_grnd \cup step\_pref$ is a complete labeling, as the two update relations preserve the legality of argument labels. So we only prove that each complete labeling *Lab* is reachable in $step\_grnd \cup step\_pref$.

Let $F = (\langle A,R\rangle, Lab)$ be an initial LAF and Lab_c the complete labeling we want to reach. First, apply either *step_grnd* or *step_pref* until we reach F' where the set of unattacked arguments is empty. At this point, the set S of *in* arguments is the grounded extension, and thus these arguments must also be *in* in Lab_c, since the grounded extension is the intersection of all complete extensions. Let S' be the set of arguments which are *in* in Lab_c but not $\{in\}$ in F'. $S \cup S'$ forms an admissible set, since it is a complete extension. Hence, there is a minimal, non-empty subset of S', S'_1, such that $S \cup S'_1$ is admissible. There is an edge in the relation *step_pref* which labels the arguments in S'_1 as *in* and any argument they attack as *out*, according to Def. 20 second item. The rest of the arguments in S' are labeled *in* via Def. 20, either with the first item, or again with the second item as above. Once we have reached the LAF where all *in* arguments from Lab_c are $\{in\}$ and any argument they attack $\{out\}$, we can make a step with *step_grnd* following Def. 18, second item, to label all remaining arguments as $\{undec\}$. We have then reached the fixpoint $F_f = (\langle A,R\rangle, T(Lab_c))$, as desired. □

Example 2. Let us examine the initial LAF $F = (\langle A,R\rangle, Lab)$ where $A = \{a,b,c,d\}$, $R = \{(a,b),(b,a),(b,c),(c,d),(d,c)\}$. Since $unattacked(F) = \emptyset$, *step_grnd* will send F to the fixpoint where all arguments are labeled $\{undec\}$. This is depicted in Fig. 2.

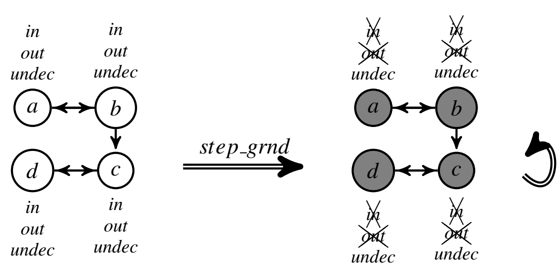


Fig. 2 Example path from the initial LAF F to the corresponding final LAF in *step_grnd*.

Let us now consider the same LAF F under the *step_pref* update relation this time. Again, $unattacked(F) = \emptyset$, but $min\_adm(F) = \{\{a\},\{b\},\{d\}\}$. The relation hence branches out in three paths. Let us focus the path with $\{a\}$. So the relation *step_pref* sends F to the LAF F_{pref1} where a is $\{in\}$ and b is $\{out\}$, as depicted in Fig. 3. $unattacked(F_{pref1}) = \emptyset$, but $min\_adm(F_{pref1}) = \{\{c\},\{d\}\}$, which gives us once again two possible directions in which to branch out. We will examine the one which selects $\{c\}$. This then gives us the final fixpoint $F_{pref2} = (\langle A,R\rangle, Lab_{pref2})$, where $Lab_{pref2}(a) = Lab_{pref2}(c) = \{in\}$ and $Lab_{pref1}(b) = Lab_{pref1}(d) = \{out\}$.

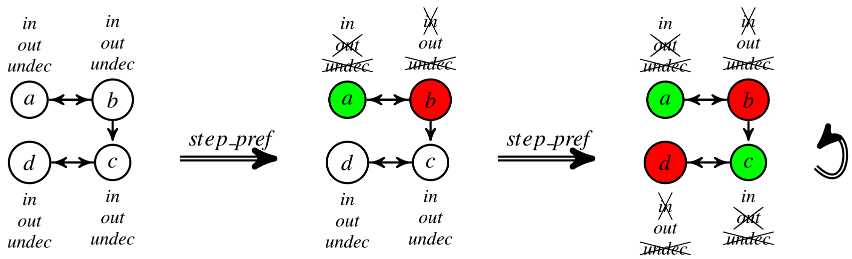


Fig. 3 Example path from the initial LAF F to one of the corresponding final LAFs in *step_pref*.

We now consider the union of both relations. We can first send F to F_{pref1} using the same step from *step_pref* as above. However this time we can apply *step_grnd* to F_{pref1}, and since $unattacked(F_{pref1} = \emptyset$, the remaining arguments c and d are assigned the $\{undec\}$ label, sending F_{pref1} to the fixpoint F_{comp}, where a is $\{in\}$, b is $\{out\}$ and c,d are $\{undec\}$. Notice that F_{comp} corresponds to a complete labeling of F which is neither preferred nor grounded. This situation is depicted in Fig. 4.

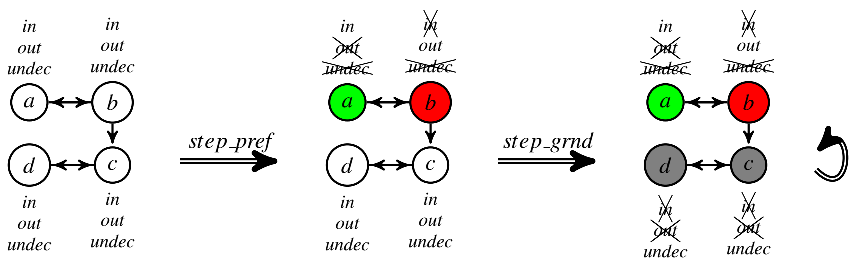


Fig. 4 Example path from the initial LAF F to one of the corresponding final LAFs in $step\_grnd \cup step\_pref$ which neither update can reach by itself.

5 Merging semantics through the most fined-grained update relation

In the previous section, we have shown that we can obtain the complete semantics by taking the union of two algorithmically motivated update relations giving rise to the grounded and the preferred semantics respectively. The success of this approach was dependent on the details of the algorithmic update relations that we defined, so it cannot be generlized to combine arbitrary semantics. In this section, we want to generalize our methodology to make it applicable to the combination of arbitrary semantics. For this purpose, we will examine a way to combine any two standard semantics via their most fine-grained update and a combination operation we call merging.

5.1 Merging preferred and grounded

If we were to attempt to combine mfg_{pref} and mfg_{grnd} by simply taking their union, as we have done in the algorithmic approach, it follows from their definition that we would simply obtain as reachable fixpoints the labelings which are either preferred or grounded. The main issue is that mfg_{pref} and mfg_{grnd} are not applicable to LAFs which do not agree with some final LAF of that semantics.

For an example of this issue, we consider again the same LAF F as in Example 2. Suppose we want to reach the same complete labeling that we reached in Figure 4, i.e. the one in which a is $\{in\}$, b is $\{out\}$, and c and d are $\{undec\}$. We could start by doing those six steps of mfg_{grnd} that are compatible with the complete labeling that we want to reach, as depicted in Figure 5, yielding the intermediate LAF F'. Now we would like to apply mfg_{pref} to F' in order to delete the *undec*-labels from a and b. However, mfg_{pref} cannot be applied at all to F', as F' is not compatible with any preferred labeling of F.

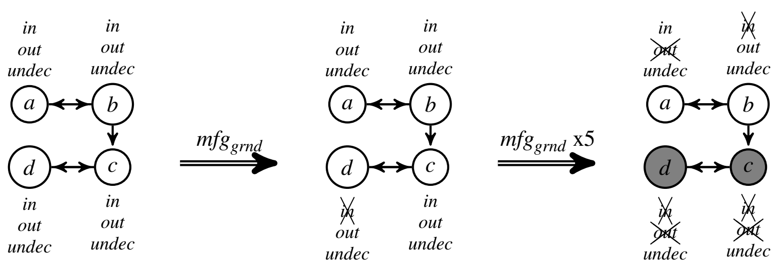


Fig. 5 Example path from the initial LAF F to an intermediate LAF F' in mfg_{grnd}.

So instead of just taking the union of mfg_{pref} and mfg_{grnd}, we will define a more complicated operation called the *merge* of two update relations, which we denote by

$upd_1 \uplus upd_2$. The idea is that once neither mfg_{pref} nor mfg_{grnd} allow us to get closer to a desired complete labeling, we will focus on a particular sub-framework and draw analogies with another framework which also contains that sub-framework. The details of this approach are somewhat complicated, so let us first sketch the approach by seeing how it can be applied to the example that we just looked at.

The idea is that we focus on the set $S = \{a, b\}$ of arguments, as we want to remove labels from a and b. In order to work with mfg_{pref} on the sub-framework $Sub(F', S)$ induced by S, we consider an alternative framework F_2 that also has $Sub(F', S)$ as a sub-framework, but to which mfg_{pref} can be applied. A suitable choice of F_2 is depicted on the left in Figure 6. Now we apply mfg_{pref} twice to F_2 as depicted in Figure 6, removing the labels from a and b that we wanted to remove. If certain conditions are satisfied, we may import the changes we have made to F_2 back to F, as depicted in Figure 7.

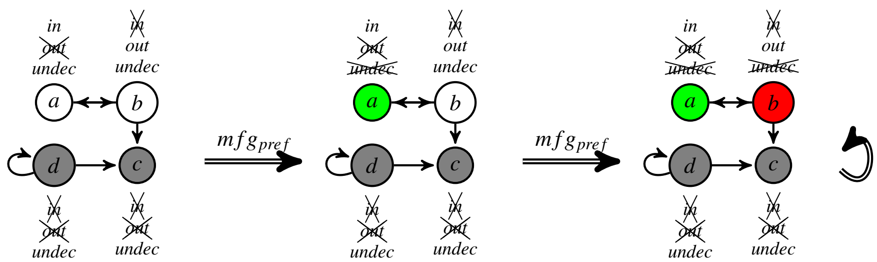


Fig. 6 Example path on a parallel F_2 framework with $S = \{a, b\}$ and $I = \{c\}$, where mfg_{pref} is applicable.

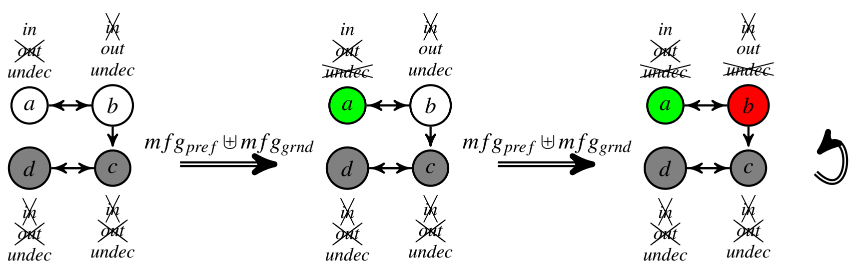


Fig. 7 Importing the steps made in Fig. 6 into F' allows us to reach a complete labeling which is neither grounded nor preferred.

Now what are the conditions that need to be satisfied in order to allow for this import of changes from one framework to another? In order to describe these conditions, we need to split the original framework into three parts, based on sets of arguments:

- S, the arguments we will focus on;

- I, called the *interface*, which is a set of arguments which already have a maximally precise label (i.e. a singleton) and which separate the set S from the rest of the framework;
- $A \setminus (S \cup I)$, the rest of the framework, on which the two frameworks may differ.

The basic idea is that in order to import some change that an update relation mfg_{sem} can make on F_2 to the LAF F, we have to choose F_2 in such a way that in both F and F_2, the interface I separates S from the rest of the framework. Furthermore, we have to choose F_2 in such a way that mfg_{sem} can actually be applied to F_2, which is only possible if the maximally precise labels of the arguments in I are possible labels for these arguments in F_2 under the semantics *sem*.

We are now ready to present the formal definition of the merge $upd_1 \uplus upd_2$:

Definition 21. Let upd_1 and upd_2 be two update relations. We define the *merge* of upd_1 and upd_2 ($upd_1 \uplus upd_2$) as the smallest relation such that:

1. $upd_1 \uplus upd_2 \supseteq upd_1 \cup upd_2$;
2. For $F = (\langle A,R\rangle, Lab)$ and $F' = (\langle A,R\rangle, Lab')$, $(F,F') \in upd_1 \uplus upd_2$ if there exist disjoint sets $S,I \subseteq A$ and two LAFs $F_2 = (\langle A_2,R_2\rangle, Lab_2)$ and $F_2' = (\langle A_2,R_2\rangle, Lab_2')$ such that the following conditions are satisfied:

   a. $(F_2,F_2') \in upd_1 \cup upd_2$;
   b. $Sub(F_2, S\cup I) = Sub(F, S\cup I)$;
   c. $\forall s \in S, \forall a \in A \setminus (I \cup S)$, $(s,a),(a,s) \notin R, R_2$;
   d. $\forall a \in I$, $Lab(a) = Lab_2(a)$ is a singleton;
   e. $Lab' \downarrow_S \neq Lab \downarrow_S$;
   f. $Lab' \downarrow_{A\setminus S} = Lab \downarrow_{A\setminus S}$;
   g. $Sub(F, A\setminus S)$ is reachable by $upd_1 \uplus upd_2$;
   h. $Lab' \downarrow_S = Lab_2' \downarrow_S$.

3. if F is final and reachable by $upd_1 \uplus upd_2$, then $(F,F) \in upd_1 \uplus upd_2$.

Given the complexity of this definition, let us explain it a bit more: Item 1 expresses the fact that we can still preform any step which is available in either one of the base updates. However, as we have seen previously, this is not enough in order to obtain meaningful combinations of most fine-grained updates, which is why we have item 2. Given a labeled argumentation framework F, additional changes are potentially possible if we can identify two disjoint sets of arguments S and I, where S is the set of arguments we are interested in and where the update will be occurring and I is a fully-labeled interface between S and the rest of the framework, meaning that no argument in S attacks nor is attacked by an argument in $A \setminus (S \cup I)$. Once such sets have been identified, we observe other labeled argumentation frameworks F_2 which also contain $S \cup I$ with the same structure and epistemic labels but can differ in structure and labels in the rest of the framework. If an update with $upd_1 \cup upd_2$ is possible in such a framework, we then allow this change to be imported into F to produce F'. In more details, sub-item a specifies that there must be a $upd_1 \cup upd_2$ step which relates F_2 to F_2'. Sub-item b ensures that the parallel framework F_2 agrees with F on the structure and epistemic labels of $S \cup I$. Sub-item c guarantees that there are no connections between S and $A \setminus (S \cup I)$ in neither F nor F_2. Sub-item d ensures that I is fully labeled, which is required in order to ensure the well-behavior of the merge operation. The idea is that once this interface has been fully labeled by one of the two updates, if we can modify $A \setminus (S \cup I)$ in order to make sense of these labels for the second update, then we can also perform steps from this second update inside S, and then by perhaps modifying $A \setminus (S \cup I)$ again we can switch back to using the first update again and so on. Sub-item e ensures that change happens inside S, while sub-item f ensures that no change is made outside of S, so that change happens in S and exclusively there. Sub-item g provides an additional restriction on the partitioning to ensure that for an argument i in the interface I which has a justification $a \in S$ for its label which hasn't been assigned yet, we do not introduce a new justification in $A \setminus (S \cup I)$ for i's label and hence allow for a different label to be assigned to a, leaving i with no justification for its label in F'. Sub-item h simply specifies that the change made in the parallel LAF be imported into the original one to produce F', and combined with the sub-item e entails that a change within S is necessary between F_2 and F_2'. Finally, with item 3 we ensure that reachable final frameworks are also fixpoints, which is needed since the second item of the definition does not produce any fixpoints, as it requires some change to happen in the LAF with the first sub-item.

In definition 21, we have defined the merge between two arbitrary update relations. In this paper, we always apply this merge operation to two maximally fine-grained update relations and focus on the semantics that the resulting update relation gives rise to. In this way, the notion of a merge between two update relations gives rise to the following notion of a merge between two argumentation semantics:

Definition 22. Given two argumentation semantics sem_1 and sem_2, we define $sem_1 \uplus sem_2$ to be the semantics that $mfg_{sem_1} \uplus mfg_{sem_2}$ gives rise to.

We originally motivated the definition of the merge operation with the goal to combine the grounded and preferred semantics to yield the complete semantics. The following theorem shows that this is indeed the case for the merge operation as we have defined it.

Theorem 5. *preferred* $\uplus$ *grounded* $=$ *complete.*

Proof. By Definition 22, we need to show that $mfg_{pref} \uplus mfg_{grnd}$ gives rise to the complete semantics. So we need to prove that every complete labeling is a reachable fixpoint in $mfg_{pref} \uplus mfg_{grnd}$ and that every labeling that is a reachable fixpoint in $mfg_{pref} \uplus mfg_{grnd}$ is a complete labeling. We start by proving that every complete labeling is a reachable fixpoint in $mfg_{pref} \uplus mfg_{grnd}$:

Let $AF = \langle A, R \rangle$ be an argumentation framework, and let L be the complete labeling of AF we want to reach. We want to show that $F_f = (\langle A, R \rangle, Lab_f)$ is a reachable fixpoint, where $Lab_f = T(L)$. Let $C = \{a \in A \mid L(a) = in\}$, $I = \{b \in A \mid L(a) = out\}$ and $S = A \setminus (C \cup I)$. Consider the LAF $F_i = (\langle A, R \rangle, Lab_i)$, where for all $a \in C$, $Lab_i(a) := \{in\}$, for all $b \in I$, $Lab_i(b) := \{out\}$, and for all $c \in S$, $Lab_i(c) := \{in, out, undec\}$. Since L is a complete labeling, C is admissible, so there exists a preferred labeling where all arguments in C are *in*. Thus F_i is reachable with mfg_{pref}.

We now want to apply item 2 of Definition 21 to F_i multiple times in order to remove all the *in* and *out* labels from the arguments in S. For this purpose, we choose a "new" argument z, i.e. an argument $z \notin A$, and consider the LAF $F_2 = (\langle A_2, R_2\rangle, Lab_2)$ where $A_2 = (A \setminus C) \cup \{z\}$, $R_2 = R\downarrow_{A_2} \cup \{(z,a) \mid a \in I\}$ and $Lab_2 = Lab\downarrow_{A_2} \cup \{(z,\{in\})\}$. Consider the final LAF $F_{2f} = (\langle A_2, R_2\rangle, Lab_{2f})$, where for all $a \in A_2 \setminus S$, $Lab_{2f}(a) = Lab_2(a)$ and for all $a \in S$, $Lab_{2f}(a) = \{undec\}$.

We want to show that F_{2f} is grounded. For this purpose, we first establish that F_{2f} is complete, i.e. that all labels in F_{2f} are legal labels: z is unattacked and is therefore legally labeled *in* in F_{2f}. All arguments in I are attacked by z, so they are legally labeled *out* in F_{2f}. Furthermore, since C does not defend any arguments it does not contain, every argument in S is attacked by at least one other argument in S. Additionally, the only *in* argument, z, does not attack any arguments in S. Thus the arguments in S are legally labeled *undec* in F_{2f}. Therefore, F_{2f} is a complete LAF, and since the only *in* argument, z, has to be labeled *in*, it is also grounded.

Therefore F_{2f} is reachable in mfg_{grnd} from F_2. So by multiple applications of $mfg_{pref} \uplus mfg_{grnd}$, using item 2 of Def 21, one can reach F_f from F_i. Since F_f is final, F_f is a fixpoint, and thus F_f is a reachable fixpoint.

So far, we have shown that every complete labeling is a reachable fixpoint in $mfg_{pref} \uplus mfg_{grnd}$. Now we still need to show that every labeling that is a reachable fixpoint in $mfg_{pref} \uplus mfg_{grnd}$ is a complete labeling:

Let $F = (\langle A, R\rangle, Lab)$ be a reachable LAF in $mfg_{pref} \uplus mfg_{grnd}$. We show by induction on $|A|$ that there exists a final complete LAF which is at least as precise as F.

Induction hypothesis 1: Assume that for every LAF $F' = (\langle A', R'\rangle, Lab')$ such that $|A'| < |A|$ and F' is reachable in $mfg_{pref} \uplus mfg_{grnd}$, there exists a final complete LAF which is at least as precise as F'.

We now use a second induction on the steps required to reach F.

Base case: F is initial. Since there always exists a complete labeling for any framework, there exists a final complete LAF more precise than F.

Inductive step: F is not initial, but is reached in $mfg_{pref} \uplus mfg_{grnd}$ through an LAF $F^* \neq F$ for which the required property holds. In other words, we have the following induction hypothesis for F^*:

Induction hypothesis 2: Assume that for $F^* = (\langle A, R\rangle, Lab^*)$ such that $F^* \neq F$ and $(F^*, F) \in mfg_{pref} \uplus mfg_{grnd}$, there exists a final complete LAF $F^*_f = (\langle A, R\rangle, Lab^*_f)$ such that $F^* \leq_p F^*_f$.

We distinguish three cases:

1. $(F^*, F) \in mfg_{pref}$. Then, by the definition of mfg_{pref}, there exists a final LAF which represents a preferred labeling of $\langle A, R\rangle$ and is at least as precise as F. Since preferred labelings are also complete, we are done.
2. $(F^*, F) \in mfg_{grnd}$. Similarly to the case above, it follows from the definition of mfg_{grnd} that there exists a complete final LAF which is at least as precise as F.

3. $(F^*,F) \notin mfg_{pref} \cup mfg_{grnd}$. Since $F^* \neq F$, the item 2 of Definition 21 must be satisfied. In other words, there exist disjoint sets $S,I \subseteq A$ and two LAFs F_2 and F_2' that satisfy the conditions a to h from item 2 of Definition 21. By condition a, $(F_2,F_2') \in mfg_{pref} \cup mfg_{grnd}$, so by the same reasoning as in cases 1 and 2 above, we can conclude that there exists a final LAF $F_{f2} = (\langle A_2,R_2\rangle, Lab_{f2})$ which is complete and at least as precise as F_2'. Also, by condition g, $F_s = Sub(F^*, A \setminus S)$ is reachable by $mfg_{pref} \uplus mfg_{grnd}$, and by condition e, $S \neq \emptyset$, i.e. $|A \setminus S| < |A|$. So by induction hypothesis 1, there exists a final complete LAF $F_{sf} = (\langle A \setminus S, R\downarrow_{A \setminus S}\rangle, Lab_{sf})$ such that $F_{sf} \geq_p F_s$.
   We now construct the final LAF $F_f = (\langle A,R\rangle, Lab_f)$ as follows: For all $a \in A \setminus S$, $Lab_f(a) := Lab_{sf}(a)$, and for all $a \in S$, $Lab_f(a) = Lab_{f2}(a)$. From the construction of F_f and conditions f and h of Definition 21, it follows that F_f is more precise than F. To complete the proof, we now still need to show that F_f is a complete labeling, i.e. that all arguments in A are legally labeled in F_f.
   According to the definition of legal labeling (Definition 2), a labeling being legal depends only on the label of the arguments it is directly attacking or attacked by. According to condition c of Definition 21, the only arguments which are attacking or attacked by arguments in S are in $S \cup I$. The arguments in S are legally labeled in F_{2f}, and thus they are also legally labeled in F_f, since $Sub(F_{2f}, S \cup I) = Sub(F_f, S \cup I)$. Similarly, since the arguments in $A \setminus (S \cup I)$ are legally labeled in F_{sf}, they are also legally labeled in F_f. Now take an arbitrary $a \in I$. We distinguish three cases:

   a. $Lab_f(a) = \{out\}$: Then, since F_{sf} is complete, there exists an argument $b \in A \setminus S$ such that $(b,a) \in R$ and $Lab_f(b) = \{in\}$. So a is legally *out* in F_f.
   b. $Lab_f(a) = \{in\}$: Then, since F_{sf} is complete, for all $b \in A \setminus S$ such that $(b,a) \in R$, $Lab_f(b) = \{out\}$. Also, since F_{2f} is complete, for all $b \in S$ such that $(b,a) \in R$, $Lab_f(b) = \{out\}$. Hence, a is legally *in* in F_f.
   c. $Lab_f(a) = \{undec\}$: Then, since since F_{sf} is complete, for all $b \in A \setminus S$ such that $(b,a) \in R$, $Lab_f(b) \neq \{in\}$, and for at least one such b, $Lab_f(b) = \{undec\}$. Also, since F_{2f} is complete, for all $b \in S$ such that $(b,a) \in R$, $Lab_f(b) \neq \{in\}$. Hence, a is legally *undec* in F_f.

   So all arguments in F_f are legally labeled and thus F_f is complete. Hence, there exists a final complete LAF which is at least as precise as F.

Therefore, for all reachable LAFs, there exists a complete LAF which is at least as precise. Since $mfg_{pref} \uplus mfg_{grnd}$ is an update relation, every reachable fixpoint is final, and thus every reachable fixpoint is complete. □

5.2 Defining new semantics via merging

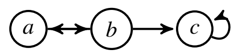
The merge operation defined in Definition 22 can be used to combine two arbitrary argumentation semantics to yield another argumentation semantics. So far, we have shown that merging grounded and preferred semantics yields the complete semantics. In this section, we show how applying this merge operation to other pairs of semantics gives rise to completely new argumentation semantics.

First, notice that the second part of the proof of Theorem 5 only makes use of the fact that the labelings reached at the different stages are complete, but not of any other properties particular to preferred or grounded. Hence, the merge of any two complete-based semantics is a complete-based semantics itself, i.e. all the labelings it returns are also complete.

Theorem 6. *Let sem_1 and sem_2 be two complete-based argumentation semantics. Then $sem_1 \uplus sem_2$ is also a complete-based semantics.*

For example, by merging stable and grounded, we obtain labelings which are complete. However, in this case, we do not recover all complete labelings as we did when merging grounded and preferred. Let us examine this short example to see why.

Example 3. Consider the following AF, which we call F:



Using $mfg_{stab} \uplus mfg_{grnd}$, one can reach the labelings $\{(a,out),(b,in),(c,out)\}$ and $\{(a,undec),(b,undec),(c,undec)\}$. However, suppose we wish to reach the complete labeling $Lab = \{(a,in),(b,out),(c,undec)\}$. Since there is no stable labeling, we cannot make any steps via mfg_{stab} from the initial LAF. Also, attempting to find a similar framework from which one could import changes, will not work at this point where the LAF is initial, because the interface I would have to be empty, which only works for disconnected AFs.

Hence, one can only make steps in mfg_{grnd} in order to reduce c's epistemic labeling to $\{undec\}$, A's to $\{in,undec\}$ and B's to $\{out,undec\}$. This, however, is as close as one can get to Lab using $mfg_{stab} \uplus mfg_{grnd}$. Any F_2 satisfying the conditions of item 2 of Definition 21 must have $I = \{c\}$. In this case, the set S on which we want to make changes would have to be $\{a,b\}$. But then $I \cup S$ includes all arguments, so that F_2 would have to be identical to F, so that we cannot use item 2 of Definition 21 to make any change that we cannot already make with item 1 of Definition 21.

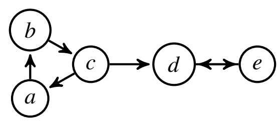
Therefore, Lab is unreachable with $mfg_{stab} \uplus mfg_{grnd}$.

An interesting note to make is that all labelings reachable by $mfg_{stab} \uplus mfg_{grnd}$ are complete, according to Theorem 6, and hence this combination provides a novel complete-based semantics which returns more labelings than both the stable semantics and the grounded semantics.

Similarly, the merge of the semi-stable and grounded semantics returns a novel complete-based semantics. One can check this by replacing stable by semi-stable in the situation described in Example 3: The desired complete labeling is still unreachable.

As motivated in the introduction, we are also interested in the following research question related to combining features of naive-based and complete-based semantics: Is there a sensible semantics that allows one to locally make choices for some unattacked odd or even cycles while not making choices for other unattacked odd or even cycles. Given our methodology for merging semantics, an obvious candidate for such a semantics is *stage* $\uplus$ *grounded*, i.e. the semantics resulting from merging the stage semantics with the grounded semantics. By considering its application to an example, we show that this semantics does indeed have this feature.

Example 4. Consider the following AF, which we call F':



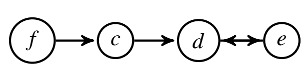
The stage labelings of F' are

$$\begin{aligned}
Lab_1 &= \{(a,in),(b,out),(c,out),(d,in),(e,out)\},\\
Lab_2 &= \{(a,in),(b,out),(c,out),(d,out),(e,in)\},\\
Lab_3 &= \{(a,out),(b,in),(c,out),(d,in),(e,out)\},\\
Lab_4 &= \{(a,out),(b,in),(c,out),(d,out),(e,in)\},\\
Lab_5 &= \{(a,out),(b,out),(c,in),(d,out),(e,in)\}.
\end{aligned}$$

Its grounded labeling is $Lab_6 = \{(a,undec),(b,undec),(c,undec),(d,undec),(e,undec)\}$. Additionally to these six labelings, it has three further *stage* $\uplus$ *grounded*-labelings:

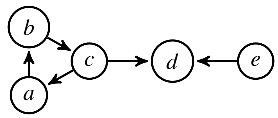
$$\begin{aligned}
Lab_7 &= \{(a,in),(b,out),(c,out),(d,undec),(e,undec)\},\\
Lab_8 &= \{(a,out),(b,in),(c,out),(d,undec),(e,undec)\},\\
Lab_9 &= \{(a,undec),(b,undec),(c,undec),(d,out),(e,in)\}.
\end{aligned}$$

Lab_7 can be reached using $mfg_{stage} \uplus mfg_{grnd}$ by first applying mfg_{stage} several times to reduce the epistemic labels on a, b and c to $\{in\}$, $\{out\}$ and $\{out\}$ respectively and then applying item 2 of Definition 21 with the interface $I := \{c\}$, the set $S := \{d,e\}$ and the following parallel framework F_2^7:



We can then apply mfg_{grnd} multiple times to this parallel framework to reduce the epistemic labels of d and e to $\{undec\}$ and import these changes to the labeling on the main frameowrk F' using item 2 of Definition 21. Lab_8 can be reached using $mfg_{stage} \uplus mfg_{grnd}$ in a similar way using the same parallel framework.

Lab_9 can be reached using $mfg_{stage} \uplus mfg_{grnd}$ by first applying mfg_{stage} several times to reduce the epistemic labels on d and e to $\{out\}$ and $\{in\}$ respectively and then applying item 2 of Definition 21 with the interface $I := \{c\}$, the set $S := \{a,b,c\}$ and the following parallel framework F_2^9:



We can then apply mfg_{grnd} multiple times to this parallel framework to reduce the epistemic labels of a, b and c to $\{undec\}$ and import these changes to the labeling on the main frameowrk F' using item 2 of Definition 21.

The stage semantics forces us to make a choice on the odd cycle $\{a,b,c\}$, and unless we choose to accept the argument c that attacks the even cycle, we are also forced to make a choice on the even cycle $\{d,e\}$. In the grounded semantics, there are no choices and all arguments become undecided. In *stage* $\uplus$ *grounded*, we can combine these features of stage and groudned: We can for example choose a from the odd cycle, but stay undecided about the arguments in the even cycle – this possible choice is formalized by Lab_7.

So *stage* $\uplus$ *grounded* allows one to locally make choices for some unattacked odd or even cycles while not making choices for other unattacked odd or even cycles. It thus provides a positive answer to our third research question from the introduction.

6 Conclusion and future work

In this paper we introduce a dynamic approach to combine two argumentation semantics to yield a third one. In particular, we provide a formal environment for the analysis of step-wise relations between labeled framework with an increase in the label precision, whose reachable fixpoints correspond to some standard direct semantics. We define and discuss two approaches to combining two given update relations to yield a third update relation, an approach based on algorithmically motivated update relations and an approach based on *merging* maximally fine-grained update relations. For both approaches, we examine how to obtain update relations for the complete labeling by combining update relations for the preferred and grounded labelings. Furthermore, we have defined novel semantics using the merge approach, including a semantics that meaningfully combines features of naive-based and complete-based semantics.

Our paper gives rise to various topics for further research. Concerning the combination of argumentation semantics, many questions remain. Further new semantics can bed efined using our approach, and properties of the newly defined semantics can be studied systematically using the principle-based approach [3, 10].

Though we introduced our update relations to combine argumentation semantics, we believe that this dynamic semantics framework can be used for other applications as well. Most importantly, one of the main challenges in formal argumentation is the gap between graph based semantics and dialogue theory. Our more dynamic semantics framework may be used to decrease or even close the gap. In particular, in dialogue each statement may increase the knowledge and thus the set of arguments of participants. This is also related to the formalization of learning in the context of formal argumentation. Moreover, an important approach in argumentation semantics is the SCC recursive scheme. This scheme can be represented naturally using update relations. Various algorithms have been proposed for argumentation semantics, and these algorithmic approaches may be expressed naturally using update relations. Finally, the principle based analysis of argumentation semantics can be extended to the more fine grained update relations.

References

1. Ryuta Arisaka, Ken Satoh, and Leendert van der Torre. Anything you say may be used against you in a court of law. In *Artificial Intelligence and the Complexity of Legal Systems (AICOL)*. Springer, 2018.
2. Pietro Baroni, Martin Caminada, and Massimiliano Giacomin. An introduction to argumentation semantics. *The Knowledge Engineering Review*, 26(4):365–410, 2011.
3. Pietro Baroni and Massimiliano Giacomin. On principle-based evaluation of extension-based argumentation semantics. *Artificial Intelligence*, 171(10-15):675–700, 2007.
4. Pietro Baroni, Massimiliano Giacomin, and Giovanni Guida. Scc-recursiveness: a general schema for argumentation semantics. *Artificial Intelligence*, 168(1-2):162–210, 2005.
5. Marcos Cramer and Mathieu Guillaume. Two approaches to dialectical argumentation: Admissible sets and argumentation stages. In *Proceedings of the International Conference on Computational Models of Argument (COMMA)*, 2018.
6. Jeremie Dauphin and Claudia Schulz. Arg Teach – A Learning Tool for Argumentation Theory. In *Tools with Artificial Intelligence (ICTAI), 2014 IEEE 26th International Conference on*, pages 776–783. IEEE, 2014.
7. Sarah Alice Gaggl and Wolfgang Dvořák. Stage semantics and the SCC-recursive schema for argumentation semantics. *Journal of Logic and Computation*, 26(4):1149–1202, August 2016.
8. Massimiliano Giacomin. Handling Heterogeneous Disagreements Through Abstract Argumentation. In *International Conference on Principles and Practice of Multi-Agent Systems*, pages 3–11. Springer, 2017.
9. Tjitze Rienstra, Alan Perotti, Serena Villata, Dov M Gabbay, and Leendert van der Torre. Multi-sorted argumentation. In *International Workshop on Theorie and Applications of Formal Argumentation*, pages 215–231. Springer, 2011.
10. Leendert van der Torre and Srdjan Vesic. The principle-based approach to abstract argumentation semantics. In Pietro Baroni, Dov Gabbay, Massimiliano Giacomin, and Leendert van der Torre, editors, *Handbook of Formal Argumentation*. College Publications, 2018.
11. Bart Verheij. Two approaches to dialectical argumentation: Admissible sets and argumentation stages. In *Proceedings of the biannual International Conference on Formal and Applied Practical Reasoning (FAPR) workshop*, pages 357–368. Universiteit, 1996.