A Three-Layer Argumentation Framework

Paulo Maio and Nuno Silva

GECAD – School of Engineering – Polytechnic of Porto
Rua Dr. Bernardino de Almeida 431, 4200-072 Porto, Portugal
{pam,nps}@isep.ipp.pt

Abstract. Argumentation frameworks which are abstract are suitable for the study of independent properties of any specific aspect (e.g. arguments sceptical and credulous admissible) that are relevant for any argumentation context. However, its direct adoption on specific application contexts requires dealing with questions such as the argument structure, the argument categories, the conditions under which an attack/support is established between arguments, etc. This paper presents a generic argumentation framework which comprehends a conceptualization layer to capture the expressivity and semantics of the argumentation data employed in a specific context and simplifies its adoption by applications. The conceptualization layer together with the defined argument structure is exploited to automatically derive the attack and support relationships between arguments.

Keywords: Argumentation Frameworks, Argument Instantiation, Argument Schemes, Bipolar Argumentation, Agents, MAS.

1 Introduction

A crucial problem on BDI agents as described by Wooldridge [1] concerns what should be the agent beliefs and how those beliefs are used (i) to form new intentions, or (ii) to redraw/revise current intentions. On this matter, contributions of the argumentation research field may be exploited internally by BDI agents since argumentation can be used either for reasoning about what to believe (i.e. theoretical reasoning) and/or for deciding what to do (i.e. practical reasoning). Despite existing differences between both, according to [2], from a standpoint of first-personal reflection, a set of considerations for and against a particular conclusion are drawn on both. Yet, agents in multi-agent systems (MAS) may apply argumentation externally during interactions between agents, i.e. agents' dialogues (cf. [3] for details). Within this context, argumentation is seen as an activity where each participant tries to increase (or decrease) the acceptability of a given standpoint for the other participants by presenting arguments. Therefore, argumentation is foreseen as an adequate modeling formalism to reduce the gap between models governing the internal and external agent behavior.

In which concerns to argumentation, there is an abundance of relevant literature in argumentation and argumentation systems. With regards to argumentation modeling formalisms, the abstract argumentation frameworks such as the AF [4], the BAF [5] and the VAF [6] are suitable to represent many different situations without being committed to any domain of application. Due to their abstract nature they are also suitable for the study of independent properties of any specific aspect (e.g. arguments sceptical and credulous admissible) that are relevant for any argumentation context that can be captured and formalized accordingly. On the other hand, this abstract nature represents an expressiveness limitation to the direct adoption of specific application contexts [7, 8]. To overcome this limitation, argumentation systems usually adopt an abstract argumentation framework and extend it in order to get a less abstract formalism, dealing in particular with (i) the construction of arguments and their structure, (ii) the conditions under which argument-relations (i.e. attack and/or support) are established, (iii) categories of arguments, etc. Nevertheless, abstract argumentation frameworks do not provide any machinery facilitating and governing how applications should extend or instantiate the framework. As a result, a significant gap between abstract argumentation frameworks and applications exist.

Regarding arguments acceptability, argumentation systems (e.g. the Prakken version of ASPIC [8]) use the abstract level as an abstraction of the overall system to make logical inferences. That is, systems start with a knowledge base, which is used to instantiate the adopted argumentation framework and then apply a given abstract argumentation semantics such as the ones described in [7] to select the conclusions of the associated sets of arguments. However, as studied in [8] and [9], in light of the arguments' content it is still possible that sets of arguments selected by an abstract argumentation criterion yield to inconsistent conclusions.

This paper proposes a less abstract argumentation framework whose purpose is to reduce existing gaps between abstract argumentation frameworks and applications, namely which concerns with the arguments' instantiation. For that, the proposed framework (i) adopts a general and intuitive argument structure, (ii) includes a conceptual layer for the specification of the semantics of argumentation data applied in a specific domain of application (e.g. e-commerce, legal reasoning and decision making) and (iii) defines a novel conceptual relation between argument-schemes called arguments affectation. In addition, the proposed framework exploits the conceptual information and the defined argument structure to automatically derive the attack and support relationships between arguments. Despite the arguments' acceptability issue is not directly addressed in this paper, applications still profiting from the inherent suitability of abstract argumentation frameworks on the study of independent properties, since information represented according to the proposed argumentation framework is easily transformed (or converted) to BAF [5]. Despite having these new features, the proposed argumentation framework remains general, but less abstract than AF [4], BAF [5] and VAF [6].

The rest of the paper is organized as follows. The next section introduces background concepts about abstract argumentation frameworks. Section 3 presents the proposed argumentation framework. Next, in section 4, an example is provided to illustrate the application of the proposed argumentation framework. Section 5 complements the proposed argumentation framework with a process to automatically derive the attack and support relationships between arguments. Section 6 compares and discusses the proposed framework with the related work. Finally, Section 7 draws conclusions and discusses future work.

2 Abstract Argumentation Frameworks

This section briefly describes the main concepts of the most referenced abstract argumentation frameworks found in the literature: the Argumentation Framework proposed by Dung (AF) [4], the Value Argumentation Framework (VAF) [6] and the Bipolar Argumentation Framework (BAF) [5].

As proposed by Dung [4], the AF core entities are *Argument*, and a binary relation between arguments (R_{att}) as depicted in Fig. 1a. The R_{att} relation is known as the attack relation. An AF can be defined as a tuple $AF = (A, R_{att})$ where A is a set of arguments and R_{att} is a relation on A such that $R_{att} \subseteq A \times A$.

An AF instance may be represented by a directed graph whose nodes are arguments and edges represent the attack relation. For any two arguments, say a_1 and a_2, such that $a_1, a_2 \in A$, one says that a_1 attacks a_2 iif $(a_1, a_2) \in R_{att}$.

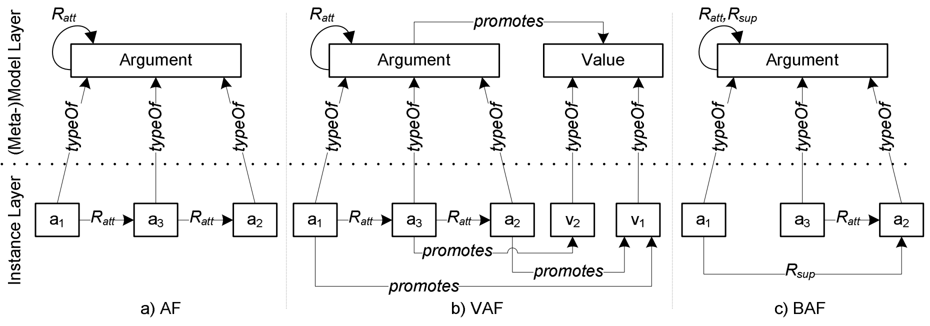

Fig. 1. The main concepts of abstract argumentation frameworks

In Dung's work attacks always succeed (i.e. it defeats the attacked arguments). Yet, one says that an argument y is attacked by a set of arguments S such that $S \subseteq A$ if S contains at least one argument attacking y. Grounded on that, the following notions were defined:

- An argument $a \in A$ is *acceptable* with respect to a set of arguments S, i.e. $acceptable(a, S)$, iif $\forall x: x \in A \wedge (x, a) \in R_{att} \rightarrow \exists y: y \in S \wedge (y, x) \in R_{att}$;
- A set of arguments S if *conflict-free* iif $\nexists x, y: x, y \in S \wedge (x, y) \in R_{att}$;
- A *conflict-free* set of arguments S is admissible iif $\forall x: x \in S \rightarrow acceptable(x, S)$;
- A set of arguments S is a *preferred extension* iif it is maximal (with respect to set inclusion) *admissible* set of A.

A *preferred extension* represents a consistent position within an AF instance, which is defensible against all attacks and cannot be further extended without introducing a conflict. Yet, multiple *preferred extensions* can exist in an AF instance due to the presence of cycles of *even* length in the graph. Given that, one considers that (i) an argument is *sceptical admissible* if it belongs to any *preferred extension* and (ii) an argument is *credulous admissible* if it belongs to at least one *preferred extension.*

While it is reasonable that attacks always succeed when dealing with deductive arguments, in domains where arguments lack this coercive force, arguments provide reasons which may be more or less persuasive and their persuasiveness may vary according to their audience. Accordingly, it is necessary to distinguish between attacks and successful attacks (i.e. defeats) prescribing different strengths to arguments on the basis of the values they promote and/or their motivation in order to accommodate the different interests and preferences of an audience. With that purpose, the VAF [6] extended the AF [4] with (i) the concept of *Value* and (ii) the function *promotes* relating an *Argument* with a single *Value* (depicted in Fig. 1b). Therefore, a VAF can be defined as 4-uple $VAF = (A, R_{att}, V, promotes)$ where A and R_{att} means the same as in the AF, a non-empty set of values V and the function $promotes: A \rightarrow V$ to map elements from A to elements of V. Consequently, an *audience* for a VAF instance corresponds to a binary preference relation $P \subseteq V \times V$ which is transitive, irreflexive and asymmetric. If a pair $(v_1, v_2) \in P$ means that value v_1 is preferred to v_2 in the audience P. An attack between two arguments (i.e. $(a_1, a_2) \in R_{att}$) where a_1 promotes a value v_1 and a_2 promotes a value v_2 succeeds (i.e. a_1 defeats a_2) iif the adopted audience prefers v_1 to v_2 otherwise the attack fails. As a result, previous notions (i.e. *acceptable*, *admissible*, *conflict-free* and *preferred extension*) were redefined accordingly (cf. [6] for details). Notice that for the same audience multiple *preferred extensions* are possible and different audiences may also lead to a unique *preferred extension*. In this way, different agents (each one represented by one audience) can have different perspectives (i.e. *preferred extensions*) over the same arguments.

The AF and the VAF assume that an argument a_1 supports an argument a_2 if a_1 attacks and therefore defeats an argument a_3 that attacks argument a_2. Thus, these frameworks only explicitly represent the negative interaction (i.e. attack), while the positive interaction (i.e. defense/support) of an argument a_1 to another argument a_2 is implicitly represented by the attack of a_1 to a_3. Since support and attack are related notions, this modeling approach adopts a parsimonious strategy, which is neither a complete nor a correct modeling of argumentation [10]. Conversely, the BAF [5] assumes the attack relation is independent of the support relation and both have a diametrically opposed nature and represent repellent forces. As a result, BAF [5] extended the AF [4] with the support relation (R_{sup}) in order to be explicitly represented (depicted in Fig. 1c). Thus, a BAF can be defined as a 3-uple $BAF = (A, R_{att}, R_{sup})$ where A and R_{att} means the same as in the AF and R_{sup} is a binary relation on A such that $R_{sup} \subseteq A \times A$. Given that, for any two arguments, say a_1 and a_2, such that $a_1, a_2 \in A$, one says that a_1 supports a_2 iif $(a_1, a_2) \in R_{sup}$. Consequently, the notions of *acceptable* and *conflict-free* arguments as well as the notion of a *preferred extension* were redefined accordingly (cf. [5] for details).

For all of these frameworks, an argument is anything that may attack/support or be attacked/supported by another argument. The absence of an argument structure and semantics enables the study of independent properties of any specific aspect that are relevant for any argumentation context that can be captured and formalized accordingly. On the other hand, this emphasizes the limited semantics for direct adoption in specific application contexts [7, 8]. Indeed, a given application context requires a less abstract formalism to deal with (i) the construction of arguments and their structure, (ii) the conditions for an argument attack/support another, (iii) categories of arguments, etc.

3 Three-Layer Argumentation Framework

This section presents the proposed argumentation framework, which is denominated as Three-Layer Argumentation Framework (TLAF). First, we give an informal overview of the framework main concepts and their relations. Further, the framework is formally defined.

3.1 Informal Overview

Unlike the abstract argumentation frameworks described, the TLAF features three modeling layers as depicted in Fig. 2 (the line ending with a hollow triangle means specialization/generalization).

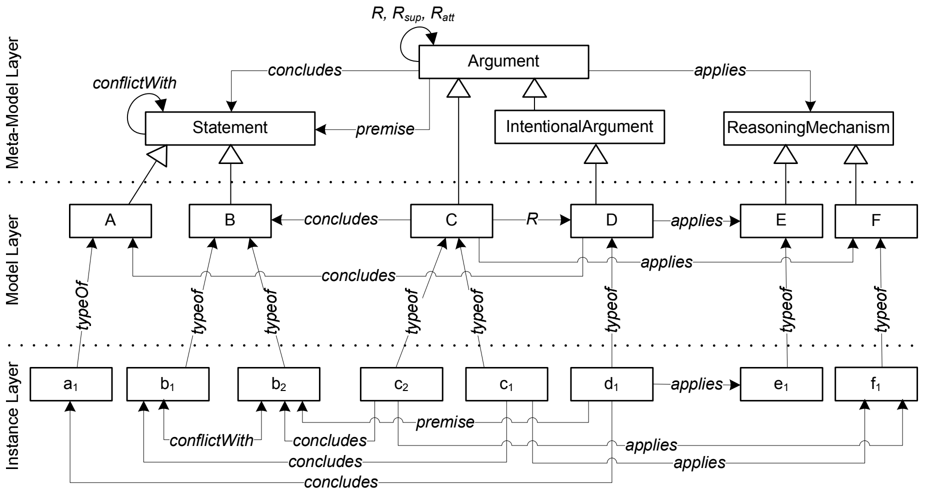

Fig. 2. The three modeling layers of the proposed argumentation framework

Despite existing differences, the TLAF Meta-Model Layer and the TLAF Instance Layer have the same purpose as those of AF [4], BAF [5] and VAF [6] layers with the same name. The TLAF Model Layer intends to capture the semantics of argumentation data (e.g. argument types/schemes) applied in a specific domain of application (e.g. e-commerce, legal reasoning and decision making) and the relations existing between them. In that sense, the model layer is important for the purpose of enabling knowledge sharing and reuse between agents. In this context, a model is a specification used for making model commitments. Practically, a model commitment is an agreement to use a vocabulary in a way that is consistent (but not complete) with respect to the theory specified by a model [11, 12]. Agents then commit to models and models are designed so that the knowledge can be shared among these agents. Accordingly, the content of this layer directly depends on (i) the domain of application to be captured and (ii) the perception one (e.g. a community of agents) has about that domain. Due to this, we adopt the vocabulary of (i) argument (or statement)-instance as an instance of an (ii) argument (or statement)-type defined at the Model Layer. Similarly, we adopt the vocabulary of (i) relation between types, and (ii) relationship between instances.

In TLAF, the meta-model layer defines an argument which is made of three parts: (i) a set of premise-statements, (ii) a conclusion-statement and (iii) an inference from premises to the conclusion enabled by a reasoning mechanism. This argument structure is very intuitive and corresponds to the minimal definition presented by Walton in [13]. For that, the meta-model layer defines the notion of *Argument*, *Statement* and *Reasoning Mechanism*, and a set of relations between these concepts. Following the notion of the BDI model [14, 15], an *IntentionalArgument* is the type of argument whose content corresponds to an intention. Domain data and its meaning are captured by the notion of *Statement*. This mandatorily includes the domain intentions, but also the desires and beliefs. The distinction between arguments and statements allows the application of the same domain data (i.e. statement) in and by different means to arguments. Also the same statement can be concluded by different arguments, and serve as the premise of several arguments. The notion of *Reasoning Mechanism* captures the rules, methods, or processes applied by arguments.

At the model layer, an argument-type (or argument scheme) is characterized by the statement-type it concludes, the applied class of reasoning mechanism (e.g. Deductive, Inductive, Heuristic) and the set of affectation relations (i.e. R) it has. The R relation is a conceptual abstraction of the attack (i.e. R_{att}) and support (i.e. R_{sup}) relationships. The purpose of R is to define at the conceptual level that argument-instances of an argument-type may affect (either positively or negatively) instances of another argument-type. For example, according to the model layer of Fig. 2, $(C, D) \in R$ means instances of argument-type C may attack or may support instances of argument-type D depending on the instances content. On the other hand, if $(X, Y) \notin R$ it means that instances of argument-type X cannot (in any circumstance) attack/support instances of argument-type Y. Yet, the R relation is also used to determine the types of statements that are admissible as premises of an argument-instance. So, an argument-instance of type X can only have as premises statements of type S iif S is concluded by an argument-type Y and Y affects X (i.e. $(Y, X) \in R$). For example, considering again the model layer of Fig. 2, instances of argument-type D can only have as premises statements of type B because D is affected by argument-type C only.

At the instance layer, an argument-instance *applies* a concrete reasoning mechanism to *conclude* a conclusion-statement-instance from a set of *premise*-statement-instances. The relation *conflictWith* is established between two statement-instances only. A statement-instance b_1 is said to be in conflict with another statement-instance b_2 when b_1 states something that implies or suggests that b_2 is not true or do not holds. The *conflictWith* relation is asymmetric (in Fig. 2 b_2 conflicts with b_1 too). In this case, for example, b_1 may represent the statement "Peter is an expert on PCs." and b_2 may represent the statement "Peter is not an expert on PCs". While the R_{att} and R_{sup} relations are established between argument-instances as in BAF [5], these relationships are automatically inferred in TLAF exploiting (i) the argument statements (i.e. conclusion and premises), (ii) the existing conflicts between statement-instances and (iii) based on the R relations defined at the model layer (cf. section 5 for details). It is worth noticing that all instances existing in the instance layer must have an existing type in the model layer and according to the type characterization.

3.2 Formal Definition

The TLAF is formally described as follows.

Definition 1 (TLAF). A TLAF structure is a singleton $TLAF = (E)$, where E is the set of entities of a TLAF.

A TLAF represents a self-contained unit of structured information. Elements in a TLAF are called argumentation entities.

Definition 2 (TLAF Model Layer). A model layer associated with a TLAF is a 6-tuple $ML(TLAF) = (A, IA, S, M, R, \sigma)$ where:

- $A \subseteq E$ is a set of argument-types;
- $IA \subseteq A$ is the sub-set of argument-types whose instances claim corresponds to an *intention* ([14, 15]);
- $S \subseteq E$ is the a set of statement-types;
- $M \subseteq E$ is the set of reasoning mechanisms;
- $R \subseteq A \times A$ establishes a reflexive relation between two argument-types called arguments' affectation. If a pair $(a_1, a_2) \in R$ then argument-instances of type a_1 may affect (positively or negatively) argument-instances of type a_2;
- σ is a function that assigns to every argument-type (i) the concluded statement-type and (ii) the reasoning mechanism applied, such as $\sigma: A \to S \times M$ where:
  - function $concl: A \to S$;
  - function $reason: A \to M$.

Each TLAF has a model layer associated with it. Information captured within the model layer plays an important role by conducting and governing the instantiation process of the framework by an application, namely which concerns the construction and semantics of instances and existing relations between them. In that sense, the model layer can also be used to validate the TLAF Instance Layer.

Notice that argument-types do not define their statement-types used as premises. Instead, these are derived from the R relation established between arguments.

Definition 3 (TLAF Instance Layer). An instance layer associated with a TLAF is a 6-tuple $IP(TLAF) = (I, instA, instS, instM, \Sigma, sconflict)$ where:

- $I \subseteq E$, is a set of instances;
- function $instA: A \to 2^I$ relates an argument-type with a set of instances. Consequently, the set of all argument instances AI is defined according to equation 1 (see below). Furthermore, we define the inverse function as $instA^{-}: AI \to A$;
- function $instS: S \to 2^I$ relates a statement-type with a set of instances. Consequently, the set of all statement instances SI is defined according to equation 1. Furthermore, we define the inverse function as $instS^{-}: SI \to S$;

- function $instM: M \rightarrow 2^{I}$ relates a reasoning mechanism with a set of instances. Consequently, the set of all reasoning mechanism instances MI is defined according to equation 1. Furthermore, we define the inverse function as $instM^{-}: MI \rightarrow M$;
- function $\Sigma: AI \rightarrow SI \times MI \times 2^{SI}$, defines for every argument-instance (i) the statement-instance concluded, (ii) the reasoning mechanism instance used to infer the conclusion and (iii) the set of statement-instances used as premises, where:
  - function $iconcl: AI \rightarrow SI$, defines the statement-instance that plays the role of conclusion on an argument-instance. Indeed, an argument-instance has only one statement-instance as conclusion while a statement-instance is concluded by at least one argument-instance;
  - function $ireason: AI \rightarrow MI$, defines the reasoning mechanism instance that is used by an argument-instance.
  - function $ipremise: AI \rightarrow 2^{SI}$, defines the statement-instances used as premises on an argument-instance. Moreover, statement-instances used as premises are also concluded by other arguments;
- function $sconflict: SI \rightarrow 2^{SI}$, defines the statement-instances that are in conflict with a statement-instance.

$$AI = \bigcup_{\forall x: x \in A} instA(x), \qquad SI = \bigcup_{\forall x: x \in S} instS(x), \qquad MI = \bigcup_{\forall x: x \in M} instM(x) \qquad (1)$$

As the reader might have noticed, the instance layer definition is concerned with the generation of argument-instances, statement-instances and their inter-relationships (Σ and $sconflict$). Despite the fact that this is a domain dependent process, it profits from the subjacent TLAF model, namely due to the rules complementing the $iconcl$, $ipremise$ (see next definition) and $sconflict$ (see section 5), that have the ability to conduct and simplify the process.

Definition 4 (TLAF Interpretation). An interpretation of a TLAF is a structure $\mathfrak{I} = (\Delta^{\mathfrak{I}}, A^{\mathfrak{I}}, S^{\mathfrak{I}}, M^{\mathfrak{I}}, I^{\mathfrak{I}})$ where:

- $\Delta^{\mathfrak{I}}$ is the domain set;
- $A^{\mathfrak{I}}: A \rightarrow 2^{\Delta^{\mathfrak{I}}}$ is an argument interpretation function that maps each argument-type to a subset of the domain set;
- $S^{\mathfrak{I}}: S \rightarrow 2^{\Delta^{\mathfrak{I}}}$ is a statement interpretation function that maps each statement-type to a subset of the domain set;
- $M^{\mathfrak{I}}: M \rightarrow 2^{\Delta^{\mathfrak{I}}}$ is a reasoning mechanism interpretation function that maps each reasoning mechanism to a subset of the domain set;
- $I^{\mathfrak{I}}: I \rightarrow \Delta^{\mathfrak{I}}$ is an instance interpretation function that maps each instance to a single element in the domain set;

An interpretation is a model of TLAF if it satisfies the following properties:

- $\forall a, i: a \in A \wedge i \in instA(a) \Rightarrow I^{\mathfrak{I}}(i) \in A^{\mathfrak{I}}(a)$;
- $\forall s, i: s \in S \wedge i \in instS(s) \Rightarrow I^{\mathfrak{I}}(i) \in S^{\mathfrak{I}}(s)$;
- $\forall m, i: m \in M \wedge i \in instM(m) \Rightarrow I^{\mathfrak{I}}(i) \in M^{\mathfrak{I}}(m)$;

— $\forall a: a \in IA \Rightarrow A^{\mathfrak{T}}(a)\ are\ intentions$
— $\forall a, i: a \in A \wedge i \in instA(a) \Rightarrow I^{\mathfrak{T}}(iconcl(i)) \in S^{\mathfrak{T}}(concl(a)) \wedge$
$I^{\mathfrak{T}}(ireason(i)) \in M^{\mathfrak{T}}(reason(a))$;
— $\forall a, i, p: a \in A \wedge i \in instA(a) \wedge p \in ipremise(i) \Rightarrow$
$\exists x, y: I^{\mathfrak{T}}(y) \in A^{\mathfrak{T}}(x) \wedge p = iconcl(y) \wedge (x, a) \in R$
— $\forall a, s: a \in A \wedge s \in S \Rightarrow A^{\mathfrak{T}}(a) \cap S^{\mathfrak{T}}(s) = \emptyset$;
— $\forall a, m: a \in A \wedge m \in M \Rightarrow A^{\mathfrak{T}}(a) \cap M^{\mathfrak{T}}(m) = \emptyset$;
— $\forall s, m: s \in S \wedge m \in M \Rightarrow S^{\mathfrak{T}}(s) \cap M^{\mathfrak{T}}(m) = \emptyset$.

Definition 5 (Argument Properties). An argument-type $a \in A$ and all its argument-instances (i.e. $\forall a_i: a_i \in AI \wedge a_i \in instA(a)$) are said to be:

— *intentional* if $a \in IA$;
— *non-intentional* if $a \notin IA$;
— *defeasible* if $\exists x: x \in A \wedge x \neq a \wedge (x, a) \in R$;
— *indefeasible* if $\forall x: x \in A \wedge x \neq a \wedge (x, a) \notin R$.

Arguments may be used with two purposes: (i) to represent and communicate *intentions* (i.e. intentional arguments) and (ii) to provide considerations (i.e. *beliefs*, *desires*) for and against those *intentions* (i.e. non-intentional arguments). Thus, an intentional argument may be affected by several non-intentional arguments. Additionally, to capture dependency between *intentions*, intentional arguments may be also affected (directly or indirectly) by other intentional arguments. A defeasible argument is affected by other (sub-) arguments (i.e. the ones concluding its premises or the ones undermining those premises) while an indefeasible argument can only be affected by its negation since it cannot have premises. Given that, in a TLAF Model Layer, intentional arguments should be always defeasible. On the contrary, non-intentional arguments can be both defeasible and indefeasible.

4 A Walk-through Example

This section provides an example whose purpose is to show the application of TLAF. For that, we decide on a common and simple scenario such as buying digital cameras. First, for the scenario in hands a possible TLAF model is introduced and discussed. Next, a short and somewhat contrived dialogue is used to demonstrate how the TLAF model guides the instantiation process of TLAF.

4.1 A TLAF Model

Consider the partial TLAF model layer graphically depicted in Fig. 3[1], where the rectangles denote non-intentional argument types, the rectangles with rounded corners denote intentional argument-types and the oriented arrows denote an R-relation between two argument types.

[1] Instead of a formal definition, we present a partial graphical view of the model layer because we consider it to be more informative to the reader.

The intention of buying a camera is captured by the argument-type *BuyCamera* which is affected by considerations about (i) the *Requirement* to buy a camera, (ii) the general trend of received *Reviews*, (iii) the general perspective about the cameras' *Features* and (iv) the *PriceRelation* (i.e. expensive vs. cheap). The *PriceRelation* grounds on considerations about the *CurrentPrice* and the *PastPrice*. The *Requirement* is affected by two types of considerations: (i) *HobbyReq* (i.e. a hobby requirement) or (ii) a *JobReq* (i.e. job requirement). *Reviews* are affected by each individual opinion (i) of friends (*FriendsReview*) and (ii) of experts (*ExpertReview*). The latter requires that the reviewer is considered an expert (*PersonExpert*). The *Features* are affected by considerations about the *Zoom* which is made based on the *DigitalZoom* and *OpticalZoom*. Additionally, for the sake of brevity, consider that each of these arguments concludes a statement-type with a similar name (e.g. argument *OpticalZoom* concludes *OpticalZoomStmt*) and applies a heuristic or presumptive reasoning mechanism. Notice that the provided conceptualization do not intends to be neither complete nor the most accurate approach for the scenario in hands.

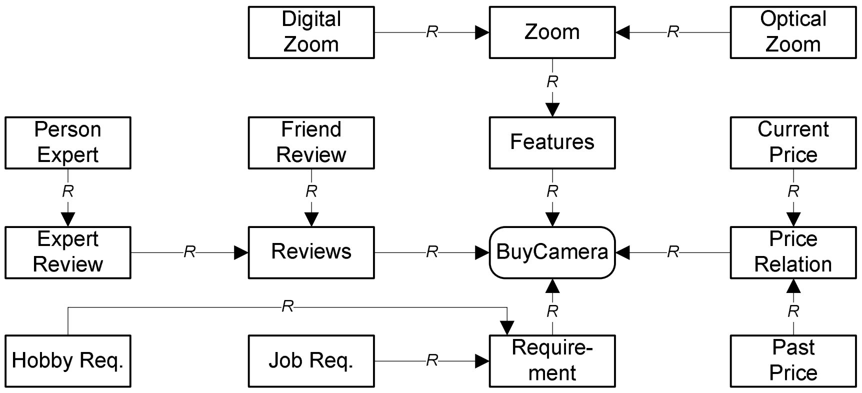

Fig. 3. A partial view over a TLAF model layer for buying cameras

This TLAF model has several indefeasible argument-types (e.g. *PersonExpert*, *CurrentPrice*, *PastPrice*) and several defeasible argument-types (e.g. *Reviews*, *Requirement*, *PriceRelation*). Regarding the former ones, agents are only able to agree or disagree with the conclusions of those argument-instances. For example, an agent can agree or disagree with other agent on the fact that someone is expert on digital cameras but it cannot argue about the information behind such position (i.e. belief). On the contrary, agents are able to argue about the information behind the conclusions of defeasible arguments. For example, an agent that does not agree about the general trend of reviews about a given digital camera presented by another agent is able to present a set of individual reviews (provided by friends and/or experts) supporting its position, which may lead the other agent to change its initial position.

Since a TLAF model captures the perception, the understanding and the rationality that someone (e.g. an agent or a community of agents) has on a given moment about a domain of application, it may evolve over time. For example, this model may evolve in order to allow agents to argue about the fact of someone to be or not to be an expert.

The information used for that purpose (e.g. the person's skills) should be conceptual analyzed and captured on the TLAF model. The resulting statement and argument types must be connected with the already existing argument types through R-relations.

4.2 Instantiating a TLAF Model

Consider the following dialogue takes place between husband (H) and wife (W). In the light of previous TLAF model, the relevant statements (i.e. domain data) uttered by both are marked as st_i (with $i > 0$).

H. I am looking forward to buy camera X (st_1).
W. Why? We don't need it (st_2).
H. That is not true (st_3). I need a camera to perform the task that Sam assigned to me (st_4). Besides that, the camera received several good reviews on a website (st_5).
W. Susan and Mary bought that camera and they told me that they regret their option (st_6 and st_7).
H. Oh, come on Honey. Peter Noble is an expert on the matter (st_8) and he says great things about the camera (st_9).
W. How much it costs? Is it expensive?
H. No! Currently, there is a great opportunity in the city mall (st_{10}). It only costs 100€ (st_{11}). Last week, the price was 150€ (st_{12}).
W. That camera is a discontinued product.
H. I don't care about that.
W. I am reading in this magazine that it lacks some minimal features (st_{13}) such as zoom (st_{14}).
H. Nonsense! Camera X has a digital zoom of "80x" (st_{15});
W. Yeah! But, the optical zoom is only of "4x" (st_{16}).

It is worth noting that: (i) the information stating camera X is a discontinued product did not give raise any statement because it was not envisioned in the TLAF model layer being used; and (ii) despite Susan and Mary have the same opinion, two statements (i.e. st_6 and st_7) were identified such that each statement corresponds to the opinion of a single person (i.e. Susan and Mary respectively), which is consistent with the semantics of the underlying TLAF model.

Even though this is a short dialogue, it already may be difficult to keep track of all information used and how it is inter-related in the form of argument-instances. As the result of an instantiation process, consider the arguments, the statements and the relationships between arguments and statements presented in Table 1.

To make evident how the information captured in a TLAF model can be exploit during the instantiation process, let us roughly describe the one adopted here. It consists of three distinct and complementary steps. First step, each statement identified in the dialogue gives raise to one argument-instance concluding that statement. Second step, because the premises of argument-instances are not always explicit in the dialogue, the instantiation process infers the premises through the information captured in the model layer. Thus, it sets as premise-statements of an argument-instance t_1 of type t the statement-instances concluded by argument-instances whose types affect t and that show to support the idea concluded by t_1. For example, st_5 is set as premise of argument a_1 of type *BuyCamera* because st_5 is concluded by a_5 of type $Reviews$ and $(Reviews, BuyCamera) \in R$ and the idea underlying st_5 somehow contributes for the idea expressed by the conclusion of a_1 which is st_1. Third, conflicts between statement-instances are established based on two conditions:

- two statement instances are in mutual conflict if both statement-instances are of the same type but they express contradictory ideas (e.g. st_2 and st_3); or
- a statement-instance s_1 is in conflict with a statement-instance s_2 if both are concluded by two distinct argument-types (say t_1 and t_2 respectively) and t_1 affects t_2 (i.e. $(t_1, t_2) \in R$) and the idea expressed by s_1 suggest that s_2 is not true or do not holds (e.g. st_2 and st_1).

Table 1. Instances of arguments and statements constructed and their relationships

Argument		Premise Statements	Conclusion-Statement	
ID	**Type**		**Statement**	**conflictWith**
a_1	BuyCamera	st_3, st_5, st_{10}	st_1	
a_2	Requirement		st_2	st_1, st_3
a_3	Requirement	st_4	st_3	st_2
a_4	JobReq		st_4	
a_5	Reviews	st_9	st_5	
a_6	FriendReview		st_6	st_5
a_7	FriendReview		st_7	st_5
a_8	PersonExpert		st_8	
a_9	ExpertReview	st_8	st_9	
a_{10}	PriceRelation	st_{11}, st_{12}	st_{10}	
a_{11}	CurrentPrice		st_{11}	
a_{12}	PastPrice		st_{12}	
a_{13}	Features	st_{14}	st_{13}	st_1
a_{14}	Zoom	st_{16}	st_{14}	
a_{15}	DigitalZoom		st_{15}	st_{14}
a_{16}	OpticalZoom		st_{16}	

It is envisaged that each scenario of application may require an instantiation process able to deal with its own particularities. However, it is our conviction that most of those processes may take advantage of the R-relations in a very similar way to the described process.

Once the instantiation process ends, support and attack relationships between argument-instances are inferred automatically. This is the subject of next section.

5 Deriving Arguments Relationships

According to the formal definitions introduced in section 3.2, the R_{att} and R_{sup} relationships between argument-instances of an $IP(TLAF)$ are not explicitly defined. Instead, these relationships are derived based on two distinct kinds of information:

- extensional information (existing at the instance layer):
  - the premises and conclusions of the argument-instances;
  - the conflicts between statement-instances, and;

– conceptual information (existing at the model layer), namely the R relations defined between argument-types.

5.1 Deriving Support Relationships

A support relationship between two argument-instances (say x and y) is established (i.e. $(x, y) \in R_{sup}$) when the argument-type of x (say a) affects the argument-type of y (say b), i.e. $(a, b) \in R$, and either (i) the conclusion of x is a premise of y or (ii) both argument-instances have the same conclusion. The following rules (graphically depicted in Fig. 4) capture the conditions required to establish support relationships between argument-instances:

R1. $\forall a, b, x, y: a, b \in A \wedge (a, b) \in R \wedge x \in instA(a) \wedge y \in instA(b) \wedge x \neq y \wedge iconcl(x) \in ipremise(y) \Rightarrow (x, y) \in R_{sup}$ (Fig. 4a);

R2. $\forall a, b, x, y: a, b \in A \wedge (a, b) \in R \wedge x \in instA(a) \wedge y \in instA(b) \wedge x \neq y \wedge iconcl(x) = iconcl(y) \Rightarrow (x, y) \in R_{sup}$ (Fig. 4b).

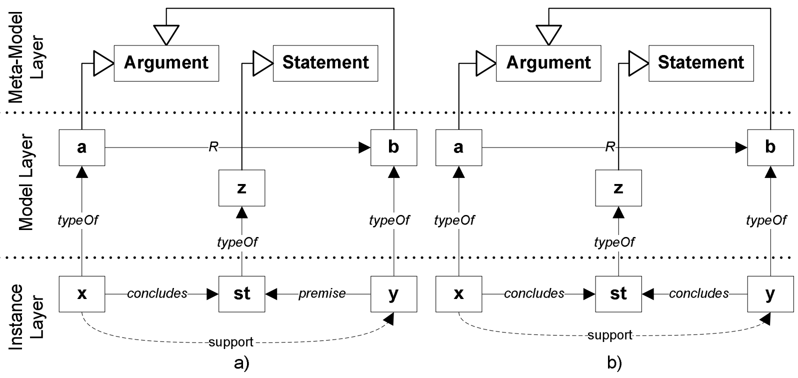

Fig. 4. Conditions to derive a support relationship between two argument-instances

Notice that two argument-instances might achieve the same conclusion starting from a different set of premises and/or reasoning mechanisms. In those circumstances, a support relation between argument-instances exists if there is a R relation between both (depicted in Fig. 4b). For a mutual support, two R relationships are required: one from a to b (i.e. $(a, b) \in R$) and another one from b to a (i.e. $(b, a) \in R$).

5.2 Deriving Attack Relationships

An attack relationship between two argument-instances (say x and y) is established (i.e. $(x, y) \in R_{att}$) when the argument-type of x (say a) affects the argument-type of y (say b), i.e. $(a, b) \in R$, and either (i) the conclusion of x is in conflict with any premise of y or (ii) the conclusion of x is in conflict with the conclusion of y. The following rules (graphically depicted in Fig. 5) capture the conditions required to establish attack relationships between argument-instances:

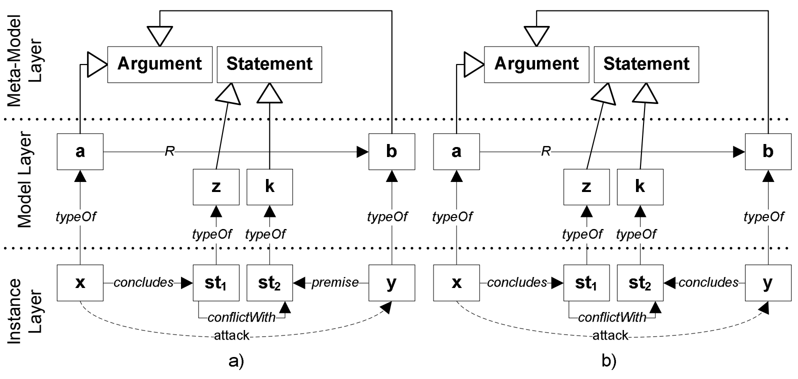

Fig. 5. Conditions to derive an attack relationship between two argument-instances

R3. $\forall a,b,x,y,s\colon a,b \in A \wedge (a,b) \in R \wedge x \in instA(a) \wedge y \in instA(b) \wedge x \neq y \wedge s \in ipremise(y) \wedge s \in sconflict(iconcl(x)) \Rightarrow (x,y) \in R_{att}$ (Fig. 5a);

R4. $\forall a,b,x,y\colon a,b \in A \wedge (a,b) \in R \wedge x \in instA(a) \wedge y \in instA(b) \wedge x \neq y \wedge iconcl(y) \in sconflict(iconcl(x)) \Rightarrow (x,y) \in R_{att}$ (Fig. 5b).

According to the rule/scenario depicted in Fig. 5b, one cannot say that argument y also attacks argument x because the conflict relation between statements is asymmetric. However, that would happen iif statement st_2 is also in conflict with statement st_1 (i.e. $st_1 \in sconflict(st_2)$) and a R relationship between b and a (i.e. $(b,a) \in R$) exists too.

5.3 Exploiting the Derivation Process

The application process used to identify and establish conflicts between statement-instances may exploit the knowledge embedded in rules R3 and R4 to reduce and drive the search/combination space between statements. Indeed, it is worth establishing a conflict relationship between two statement-instances (say st_1 and st_2) iif their statement-types (say z and k respectively) satisfy at least one of the following conditions:

- There is an argument-type (say a) concluding z that affects any other argument-type (say b), i.e. $(a,b) \in R$, where statement-instances of type k can be used as premises of argument-instances of type b;
- There is an argument-type (say a) concluding z that affects any other argument-type (say b), i.e. $(a,b) \in R$, where k is concluded by b.

Notice that, these conditions can be verified using the information captured at the model layer only. On the other hand, if a conflict relationship is established between two statement-instances and none of these conditions apply then it has no impact on derived attack relationships between arguments.

Regarding the argument-instances of the example introduced in section 4, these four rules would derive the support and attack relationships graphically depicted in Fig. 6.

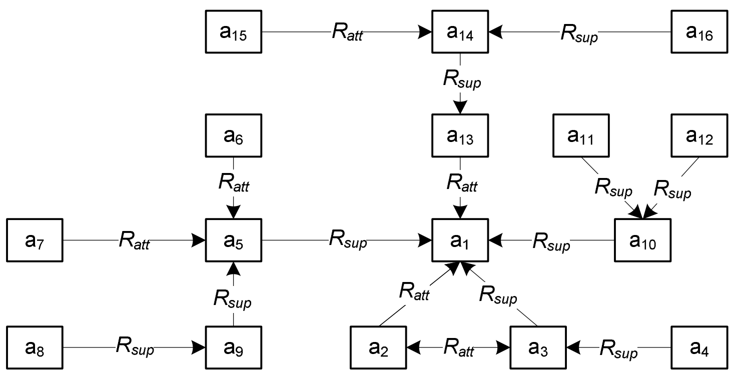

Fig. 6. Derived support and attack relationships between argument-instances of the example

6 Related Work

In this paper the advantages of having a conceptual model layer and the consequent adoption of a structured argumentation are exploited to reduce the existing gap between the most referenced abstract argumentation frameworks and its adoption by applications, namely which concerns to the instantiation process. Regarding the conceptual model only, to the best of our knowledge the most similar work existing in literature is the Description Logic formalizations of the Argument Interchange Format (AIF) [16] proposed by Iyad Rahwan in [17, 18]. In common to the AIF-based work, the TLAF has mainly two aspects:

- the adopted argument structure suggested by Walton [13]; and
- the possibility of the TLAF model layer being represented by means of an OWL ontology as the reader may confirm on [19].

However, although both works adopt the same argument structure they diverge on their purpose and consequently on the modeling approach taken. While the main purpose of the AIF-based work is to take advantage of the powerful reasoning capabilities of OWL to automatically classify argument types (or argument schemes) and argument instances, the TLAF purpose is to show the advantages that applications have with respect to the argument instantiation process by adopting an argumentation framework which comprehends a model layer to specify the types of arguments used and how they affect each other. Consequently, the modeling approach taken by both works diverge on several issues too. The most evident is that TLAF explicitly distinguishes between argument-types and the reasoning mechanisms, while in the AIF-based work the reasoning mechanisms are implicit in the name of the argument-scheme. However, the most relevant difference concerns the way premises of argument-types are defined. In the AIF-based work each argument-type defines explicitly the set of statement-types it has as premises. On the contrary, in the TLAF the set of admissible statement-types that an argument-type has as premises is inferred through the R-relations established between argument-types. This lets you constrain that an argument-type only accepts a given statement-type as a premise when it is concluded by a specific reasoning mechanism. Moreover, similarly to the Carneades framework [20], in TLAF an argument has zero or more statements as premise.

On the contrary, in the AIF-based work an argument has at least one statement as premise. Another difference between the AIF-based work and TLAF is the fact that in the former an argument-instance can be classified into several types (one or more) while in the latter an argument-instance is classified into one type only, which must be the most specific/representative one of that instance. While the multi-classification of argument-instances is useful for several tasks (e.g. querying of arguments), it raises acceptability problems that are not completely understood yet.

In the general abstract framework for rule-based argumentation described by Prakken [8] arguments apply either a *strict* or a *defeasible* rule over a set of axioms (i.e. premises) to conclude another axiom, such that axioms are defined in a logical language. In TLAF, these two kinds of rules may correspond to two kinds of reasoning mechanisms and the concrete rules may correspond to instances of those reasoning mechanisms. However, TLAF does not constraint rules to be classified only in two types. Yet, the three types of attack relationship between argument-instances: (i) rebutting, (ii) undercutting and (iii) undermining described in [8] are captured by the TLAF rules to derive such relationship. Prakken work also describes arguments as trees of inference rules such that an argument contains other sub-arguments concluding intermediate conclusions and so on. TLAF comprehends such trees of arguments at the model layer (through the R-relation) and also at the instance layer such that the root of the trees are intentional arguments. Still, since TLAF allows capturing mutual dependency between two intentional arguments, one can think on arguments as graphs rather than trees. Contrary to the Carneades framework [20], where it is assumed that argument graphs contain no cycles, the argument graphs of TLAF may contain cycles since no restriction exists at the model layer level.

Finally, as claimed by Prakken, other relevant work on structured argumentation, such as DefLog [21], is a special case of its general framework [8]. In that sense, no additional discussion with such work is provided.

7 Conclusions and Future Work

This paper describes the Three-Layer Argumentation Framework (TLAF) that reduces the existing gap between the most referenced abstract argumentation frameworks and its adoption by applications. The main novelty of the proposed argumentation framework relies on its conceptualization layer (i.e. model layer), namely the R relation. This layer captures the structure and semantics of the argumentation data employed in a specific context constraining and conducting the modeling process of the argumentation specific scenario. Even though, for the same scenario very different modeling approaches are possible.

Despite being generic, TLAF is mainly targeted to be adopted by autonomous agents. In relation to that, the TLAF adopts and follows some terminology from the BDI model, namely by distinguishing between intentional arguments and non-intentional arguments. Based on the conceptual relations captured by the framework and the defined argument structure, a clear and minimal set of conditions was established for an argument-instance to attack/support another one. Given that, the support and attack relations between argument-instances are automatically derived according to the subjacent TLAF model. Despite the fact that the argument-instances generation process, and the Σ and $sconflict$ functions are fully domain dependent, their definition profits from the established TLAF model.

While not directly addressed in this paper, the TLAF has the following advantages: (i) when generating statements it constrains the scope in which it is valuable to establish a conflict relationship between statements (i.e. $sconflict$), and therefore simplifies the automation of the process that discovers or instantiates the $sconflict$ relation, by reducing and driving the search/combination space between statements; (ii) when generating arguments upon existing statements, it constraints the type of conclusion and premises, and the reasoning mechanism associated with an argument-instance, therefore simplifying the automation of the process that instantiates arguments, that establishes the premises and conclusion relationships with statements and establishes the $R_{att}$and R_{sup} relationships between arguments.

Besides the new features provided by TLAF, it is generic enough to be adopted by different domain applications. Moreover, a TLAF instance can be easily represented in a more abstract formalism such BAF [5], where the AI set corresponds to the set of arguments of BAF and the derived argument-instances relationships, i.e. R_{sup} and R_{att}, correspond to the BAF binary relations with the same name respectively. This is especially relevant because TLAF does not impose any particular argument evaluation process. Therefore, one can use this feature to apply an argument evaluation process such as the ones proposed in [10, 22-24]. However, because none of these processes is able to take advantage of the TLAF Model Layer we are working to propose one as well. For that, we need to take into consideration the argumentation abstract semantics described in literature as well the rationality postulates introduced by Caminada and Amgoud [9] and Prakken [8].

The authors consider that no experiences would be relevant for the evaluation of the proposed framework, as its application depends on the modeling approaches of the domain, and less of the framework. This suggests the need for further development of methods and methodologies for argument modeling.

In order to simplify the modeling process and profit from experience, for example, in the software engineering and ontology development fields, the authors envisage the need to provide modularity and extensibility modeling features to TLAF. These new features potentially promote TLAF in the scope of heterogeneous, ill-specified, emergent multi-agent systems as it provides the mechanisms to model private argumentation models in respect (specializing) to other argumentation models, thus inheriting a common model.

Acknowledgments. This work is partially supported by the Portuguese projects: COALESCE (PTDC/EIA/74417/2006) of MCTES-FCT and World Search (QREN11495) of FEDER. The authors would like to acknowledge Jorge Santos, Maria João Viamonte, Jorge Coelho and Besik Dundua for their useful counsels and Owen Gilson for his revisions of the document.

References

1. Wooldridge, M.: Reasoning about rational agents. The MIT press, Cambridge (2000)
2. Moran, R.: Authority and Estrangement: An Essay on Self-Knowledge. Princeton University Press (2001)
3. Walton, D.N., Krabbe, E.C.W.: Commitment in dialogue. Suny Press (1995)

4. Dung, P.M.: On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. Artificial Intelligence 77, 321–357 (1995)
5. Cayrol, C., Lagasquie-Schiex, M.C.: On the Acceptability of Arguments in Bipolar Argumentation Frameworks. In: Godo, L. (ed.) ECSQARU 2005. LNCS (LNAI), vol. 3571, pp. 378–389. Springer, Heidelberg (2005)
6. Bench-Capon, T.J.M.: Persuasion in Practical Argument Using Value-based Argumentation Frameworks. J. Logic Computation 13, 429–448 (2003)
7. Baroni, P., Giacomin, M.: Semantics of Abstract Argument Systems. In: Argumentation in Artificial Intelligence, pp. 25–44 (2009)
8. Prakken, H.: An abstract framework for argumentation with structured arguments. Argument & Computation 1, 93 (2010)
9. Caminada, M., Amgoud, L.: On the evaluation of argumentation formalisms. Artificial Intelligence 171, 286–310 (2007)
10. Cayrol, C., Lagasquie-Schiex, M.C.: Gradual Valuation for Bipolar Argumentation Frameworks. In: Godo, L. (ed.) ECSQARU 2005. LNCS (LNAI), vol. 3571, pp. 366–377. Springer, Heidelberg (2005)
11. Gruber, T.R.: A translation approach to portable ontology specifications. Journal of Knowledge Acquisition 5, 199–220 (1993)
12. Gruber, T.: What is an Ontology?, `http://www-ksl.stanford.edu/kst/what-is-an-ontology.html`
13. Walton, D.N.: Fundamentals of critical argumentation. Cambridge Univ. Pr. (2006)
14. Bratman, M.: Intention, Plans and Practical Reason. Harvard University Press, Cambridge (1987)
15. Wooldridge, M.: An Introduction to MultiAgent Systems. Wiley (2009)
16. Chesñevar, C., McGinnis, J., Modgil, S., Rahwan, I., Reed, C., Simari, G., South, M., Vreeswijk, G., Willmott, S.: Towards an Argument Interchange Format. The Knowledge Engineering Review 21, 293–316 (2006)
17. Rahwan, I., Banihashemi, B.: Arguments in OWL: A Progress Report. In: Proceeding of the 2008 Conference on Computational Models of Argument: Proceedings of COMMA 2008, pp. 297–310. IOS Press, Amsterdam (2008)
18. Rahwan, I., Banihashemi, B., Reed, C., Walton, D., Abdallah, S.: Representing and classifying arguments on the semantic web. The Knowledge Engineering Review (2011)
19. Maio, P., Silva, N.: TLAF Meta-Model Layer as an Ontology, `http://www.dei.isep.ipp.pt/~pmaio/TLAF/Ontology/TLAF_Ontology.owl`
20. Gordon, T.F., Prakken, H., Walton, D.: The Carneades model of argument and burden of proof. Artif. Intell. 171, 875–896 (2007)
21. Verheij, B.: DefLog: on the Logical Interpretation of Prima Facie Justified Assumptions. Journal of Logic and Computation 13, 319–346 (2003)
22. Amgoud, L., Cayrol, C., Lagasquie-Schiex, M.C., Livet, P.: On bipolarity in argumentation frameworks. Int. J. Intell. Syst. 23, 1062–1093 (2008)
23. Karacapilidis, N., Papadias, D.: Computer Supported Argumentation And Collaborative Decision Making: The Hermes System. Information Systems 26, 259–277 (2001)
24. Verheij, B.: On the existence and multiplicity of extensions in dialectical argumentation, cs/0207067 (2002)