Theory of Multiple-Valued Defeasible Argumentation and its Applications

Takehisa Takahashi
Graduate School of Science and Technology
Niigata University
8050 2-cho Ikarashi Niigata, 950-2181 JAPAN
takehisa@cs.ie.niigata-u.ac.jp

Hajime Sawamura
Department of Information Engineering
Niigata University
8050 2-cho Ikarashi Niigata, 950-2181 JAPAN
sawamura@ie.niigata-u.ac.jp

Abstract

This paper provides a new departure from the traditional two-valued argumentation frameworks. We address ourselves to formalize an expressive logic of argumentation, called a Logic of Multiple-valued Argumentation (LMA), on top of the very expressive knowledge representation language, called Extended Annotated Logic Programming (EALP), and examine its logical properties in various ways. EALP allows us to represent different kinds of uncertainty such as vagueness and inconsistency (or paraconsistency) in terms of multi-valuedness, and incompleteness with the help of default negation. LMA is the first logic of argumentation in which agents can argue with other contenders, using multiple-valued knowledge base in terms of EALP.

Introduction

Argumentation is a ubiquitous form and way of dialogue in the human society. However, computational studies on argumentation are relatively recent, dating back to 1980s. Since then, argument models have been studied in the various directions, showing that argumentation is a very fruitful research object to be pursued computationally. In comparison to logic, argumentation of dynamic nature is more suitable to describing and processing the dynamic and changing nature of information in a networked distributed information environment.

We have seen many attempts and results on argumentation in the literature of the artificial intelligence (Chesnevar, Maguitman, & Loui 2000). They are basically built on the logic programming languages as knowledge representation since they allow for computationally feasible argumentation frameworks. For example, the extended logic programming (ELP) is employed in the argumentation frameworks (Dung 1993) (Mora, Alferes, & Schroeder 1998) (Prakken & Sartor 1997) (Schweimeier & Schroeder 2002).

However, very few attempts have been made at multiple-valued argument models, in which arguments are built on uncertain information. This paper provides a new departure from those two-valued argumentation frameworks in theoretical rigor. We address ourselves to formalize an expressive logic of argumentation, called a Logic of Multiple-valued Argumentation (LMA), on top of the very expressive knowledge representation language, called Extended Annotated Logic Programming (EALP), and examine its logical properties in various ways. EALP is most expressive in the hierarchy of logic programming depicted in Figure 2 in the sense that it allows to represent different kinds of uncertainty such as vagueness and inconsistency (or paraconsistency) in terms of multi-valuedness, and incompleteness with the help of default negation.

In formalizing logic of argumentation, the most primary concern is the rebuttal relation among arguments since it yields a cause or a momentum of argumentation or dialogue. The rebuttal relation for two-valued argument models is most simple, so that it naturally appears between the contradictory propositions of the form A and $\neg A$. In case of multiple-valued argumentation based on EALP, much complication is to be involved into the rebuttal relation under the different concepts of negation. One of the questions arising from multiple-valuedness is, for example, how a literal with truth-value ρ confronts with a literal with truth-value μ in the involvement with negation. This paper gives a clean and reasonable answer to it, formalizing a logic of argumentation under uncertain information.

The paper is organized as follows. In the following section, we will discuss our motivation, taking up argumentative dialogues that lead us to our logic of argumentation under multiple-valuedness. Then, the underlying language for the logic of multiple-valued argumentation is introduced together with its interpretation. Specifically, different forms of explicit negation and their formal interpretations are described in detail. Next, we introduce an abstract argumentation framework as the preliminaries for the succeeding sections. On the basis of these preliminaries, we describe our main results of this paper: various unique definitions as building blocks for the logic of multiple-valued argumentation (LMA), together with applications. The final section summarizes the paper and discusses some future works.

Motivational examples

The idea of difference of truth-values as a momentum of argumentation or dialogue seems to be very intriguing in itself. However, some logical anomalies immediately arise in realizing the idea. Let us take a look at two examples.

Example 1 *Agent A: This movie was so interesting.* (*represented as* $interesting(movie) : \mu$ *in EALP.*) *Agent B: I don't think so.* (*represented as* $\neg interesting(movie) : \mu$ *in EALP.*)

It seems that they have a different taste on movies. Agent B states an opinion contrary to Agent A, but does not intend to require refusing and taking back Agent A's opinion. In the dialogue, they simply state their own realization on the evaluation of the movie. They are not necessarily in a conflict with each other, and their agreement (if any) would be only that through the dialogue, Agent A and Agent B made it sure that they had a contrary opinion on the matter. Such a negation '$\neg$' is called the epistemological explicit negation (Kifer & Subrahmanian 1992)(Kifer & Lozinskii 1992).

Example 2 *Let us consider the following discussion. Agent C says the accident was caused by Agent D's negligence* (*represented as* $negligence(D)$: ***t*** *in EALP*). *Agent D says he does* *not remember why and how the accident occurred (represented as $negligence(D):\bot$ in EALP), and he can not admit Agent C's assertion (represented as $\sim negligence(D):\boldsymbol{t}$ in EALP).*

Agent D refuses or stands off Agent C's one way idea. Agent D has not only an assertion $negligence(D):\bot$ but also at the same time he is in such a state that he can not accept $negligence(D):\mathbf{t}$ by Agent C. This is different from that he has an assertion $negligence(D):\mathbf{f}$. In this case, it is not between two literals $negligence(D):\mathbf{t}$ and $negligence(D):\bot$ but between $negligence(D):\mathbf{t}$ and $\sim negligence(D):\mathbf{t}$ that two agents equally object to each other. Such a negation '$\sim$' is called the ontological explicit negation.

As can be seen from these argumentative dialogues, in argumentation under multiple-valued knowledge, the rebuttal relation will tend to be complicated among many truth-values.

In order to represent and resolve those complications and anomalies, we will introduce the extended annotated logic programming language with two kinds of explicit negation: Epistemic Explicit Negation '$\neg$' and Ontological Explicit Negation '$\sim$', together with the default negation '$\mathbf{not}$' in the next section. The former is a negation with respect to epistemic states or realization of contenders, and of inclusive nature in the sense that it generally does not raise a conflict among agents concerned, and rather it could provide a way of finding a clue to cooperative actions. The latter is a negation with respect to justification of arguments, and of exclusive nature in the sense that it is usually used to state that other party's opinions can not be accepted.

The terms: epistemic negation and ontological negation, originate from Kifer and Lozinskii (Kifer & Lozinskii 1992). Note, however, that the meaning of our ontological explicit negation is different from their ontological negation, being properly adjusted to argumentation as can be seen in the succeeding sections.

Extended Annotated Logic Programs

Language

Definition 1 (**Annotation and annotated atoms** (Kifer & Subrahmanian 1992)). *We assume a complete lattice $(\mathcal{T}, \leq)$ of truth values, and denote its least and greatest element by $\bot$ and $\top$ respectively. The least upper bound operator is denoted by $\sqcup$. An annotation is either an element of $\mathcal{T}$* (constant annotation), *an annotation variable on $\mathcal{T}$, or an annotation term. Annotation term is defined recursively as follows: an element of $\mathcal{T}$ and annotation variable are annotation terms. In addition, if $x_1, \ldots, x_n$ are annotation terms, then $f(x_1, \ldots, x_n)$ is an annotation term. Here, f is a total continuous function of type $\mathcal{T}^n \rightarrow \mathcal{T}$.*

If A is an atomic formula and μ is an annotation, then $A:\mu$ is an annotated atom. We assume an annotation function $\neg : \mathcal{T} \rightarrow \mathcal{T}$, and define that $\neg(A:\mu) = A:(\neg\mu)$. $\neg A:\mu$ is called the epistemic explicit negation (e-explicit negation) of $A : \mu$.

In this paper, the e-explicit negation $\neg A:\mu$ is embedded into an annotated atom $A:\neg\mu$, and implicitly handled.

Definition 2 (**Annotated literals**). *Let $A : \mu$ be an annotated atom. Then $\sim (A:\mu)$ is the ontological explicit negation (o-explicit negation) of $A : \mu$. An annotated objective literal is either $\sim A:\mu$ or $A : \mu$. The symbol $\sim$ is also used to denote complementary annotated objective literals. Thus $\sim\sim A:\mu = A:\mu$.*

If L is an annotated objective literal, then $\mathbf{not}\ L$ is a default negation of L, and called an annotated default literal. An annotated literal is either of the form $\mathbf{not}\ L$ or L.

For an annotated atom $A : \mu$, we consider an annotation μ as a recognition about A. Intuitively, we read an annotated literal as follows:

- $A:\mu$ $\cdots$ There is a recognition μ about A.
- $\neg A:\mu = A:\neg\mu$ $\cdots$ There is a negative recognition $\neg\mu$ about A.
- $\sim A:\mu$ $\cdots$ There must not be a recognition μ about A.
- $\mathbf{not}\ A:\mu$ $\cdots$ There is no recognition μ about A so far
- $\mathbf{not}\ \sim A:\mu$ $\cdots$There may be a recognition μ about A so far (it is not the case that there must not be a recognition μ about A).

Put it differently, $\sim A:\mu$ reads "a recognition μ about A is never acknowledged", and $\mathbf{not}\ \sim A:\mu$ reads "a recognition μ about A is acknowledged".

Definition 3 (**Extended Annotated Logic Programs**). *An extended annotated logic program (EALP) is a set of annotated rules of the form:*

$$H \leftarrow L_1\ \&\ \ldots\ \&\ L_n.$$

where H is an annotated objective literal, and L_i $(1 \leq i \leq n)$ are annotated literals in which the annotation is either a constant annotation or an annotation variable.

For simplicity, we assume that a rule with annotation variables or objective variables represents every ground instance of it. In this assumption, since every annotated term in the heads of rules is substituted for elements of $\mathcal{T}$, we restrict ourselves to constant annotations till the end of this paper.

The head of a rule is called a *conclusion* of a rule. Annotated objective literals and annotated default literals in the body of the rule are called *antecedents* of the rule and *assumptions* of the rule respectively. We identify a distributed EALP with an *agent*, and treat a set of EALPs as a *multi-agent system* as in (Mora, Alferes, & Schroeder 1998).

Example 3 *We introduce the complete lattices of truth values in EALP. $\mathcal{FOUR} = (\{\bot, \boldsymbol{t}, \boldsymbol{f}, \top\}, \leq)$, $\forall x, y \in \{\bot, \boldsymbol{t}, \boldsymbol{f}, \top\}$ $x \leq y \Leftrightarrow x = y \vee x = \bot \vee y = \top$ is a well-known complete lattice (depicted in the left of Figure 1). It turns out to play an important role in argumentation under inconsistent information. The closed interval $\Re[0, 1]$ of real numbers is useful for argumentation under uncertain information.*

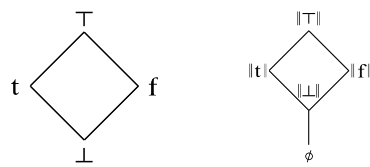


Figure 1: $\mathcal{FOUR}$ and $\mathcal{I}(\mathcal{FOUR})$

Example 4 *The following knowledge base expresses a particular agent's stance on car accidents. Rules are represented in EALP on $\mathcal{FOUR}$.*

$$negligence(A):\bot \leftarrow \mathbf{not}\ (hit(A's\_car, B's\_car):\boldsymbol{t})$$
$$\sim negligence(A):\boldsymbol{t} \leftarrow \mathbf{not}\ (hit(A's\_car, B's\_car):\boldsymbol{t})$$
$$negligence(A):\boldsymbol{f} \leftarrow hit(A's\_car, B's\_car):\boldsymbol{f}$$
$$negligence(A):\boldsymbol{t} \leftarrow hit(A's\_car, B's\_car):\boldsymbol{t}\ \&\ \sim hit(A's\_car, B's\_car):\boldsymbol{f}$$

The first rule says that agent A does not know whether the accident was caused by his negligence if there is no evidence to show *that A's car hit B's car. The second rule says that A can not admit his negligence if there is no evidence to show that A's car hit B's car. The third rule is simple. Finally, the fourth rule says that agent A acknowledges his negligence in the accident if he hit agent B's car and he can not overturn that he hit agent B's car. If the information about $hit(A's\_car, B's\_car)$ were inconsistent (i. e. $hit(A's\_car, B's\_car) : \top$), agent A could assert $negligence(A) :$ **f**, and could not assert $negligence(A) :$ **t** since $hit(A's\_car, B's\_car) : \top$ prevents the fourth rule from being applied. This knowledge base characterizes an agent who does not want to accept his negligence as far as he can avoid it. Thus the introduction of o-explicit negation allows agents to incorporate their intentions into knowledge and belief.*

Thus the introduction of o-explicit negation brings expressive power to describing intention of agents in more detail. We depict the relationship between EALP and other logic programming frameworks in Figure 2.

Remark 1 *EALP with no o-explicit negation coincides with NALP* (*Normal Annotated Logic Programs*) (T. Takahashi & Sawamura 2003). *If default negation is not included then it coincides with ALP* (*Annotated Logic Programs*) (Kifer & Subrahmanian 1992). *Under a single truth value $\mathcal{T} = \{\boldsymbol{t}\}$, EALP, NALP and ALP coincide with ELP* (*Extended Logic Programs*)*, NLP* (*Normal Logic Programs*) *and LP* (*Logic Programs*) *respectively. Thus, EALP is a language of larger class than ELP and ALP.*

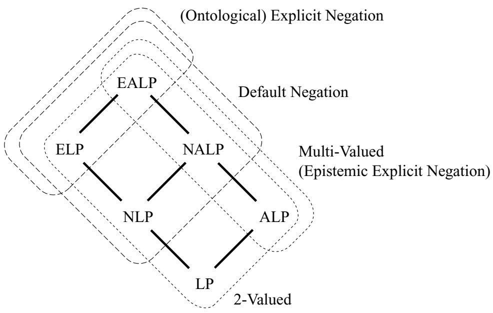


Figure 2: Hierarchy of extensions of logic programming according to different kinds of negation

Interpretation and satisfaction

In this section, we define interpretation and satisfaction for EALP. We view the interpretation as an epistemic state of an agent.

Definition 4 (**Extended annotated Herbrand base**). *The set of all annotated literals constructed from an EALP P on a complete lattice $\mathcal{T}$ of truth values is called the extended annotated Herbrand base $H_P^{\mathcal{T}}$.*

Example 5 *Suppose the EALP $P = \{p(a) : \boldsymbol{t} \leftarrow \mathbf{not}\ p(b) : \bot\}$ on the complete lattice $\mathcal{T} = (\{\boldsymbol{t}, \bot\}, \{\boldsymbol{t} \leq \bot\})$ of truth values. Then,*

$$
\begin{array}{lllll}
H_P^{\mathcal{T}} = \{ & & & & \\
p(a):\boldsymbol{t}, & p(a):\bot, & \sim p(a):\boldsymbol{t}, & \sim p(a):\bot & \\
\mathbf{not}\ p(a):\boldsymbol{t}, & \mathbf{not}\ p(a):\bot, & \mathbf{not}\ \sim p(a):\boldsymbol{t}, & \mathbf{not}\ \sim p(a):\bot & \\
p(b):\boldsymbol{t}, & p(b):\bot, & \sim p(b):\boldsymbol{t}, & \sim p(b):\bot & \\
\mathbf{not}\ p(b):\boldsymbol{t}, & \mathbf{not}\ p(b):\bot, & \mathbf{not}\ \sim p(b):\boldsymbol{t}, & \mathbf{not}\ \sim p(b):\bot & \}.
\end{array}
$$

Definition 5 (**Interpretation**). *Let $\mathcal{T}$ be a complete lattice of truth values, and P be an EALP. Then, the interpretation on P is the subset $I \subseteq H_P^{\mathcal{T}}$ of the extended annotated Herbrand base $H_P^{\mathcal{T}}$ of P such that for any annotated atom A,*

1. *If $A : \mu \in I$ and $\rho \leq \mu$, then $A : \rho \in I$;*
2. *If $A : \mu \in I$ and $A : \rho \in I$, then $A : (\mu \sqcup \rho) \in I$;*
3. *If $\sim A : \mu \in I$ and $\rho \geq \mu$, then $\sim A : \rho \in I$.*

The conditions 1 and 2 of Definition 5 are based on the definition of the ideal of truth values which is used for the interpretation of GAP (Kifer & Subrahmanian 1992).

We define three notions of inconsistencies corresponding to three concepts of negation in EALP.

Definition 6 (**Inconsistency**). *Let I be an interpretation. Then,*

1. *$A : \mu \in I$ and $\neg A : \mu \in I \Leftrightarrow I$ is epistemologically inconsistent (e-inconsistent).*
2. *$A : \mu \in I$ and $\sim A : \mu \in I \Leftrightarrow I$ is ontologically inconsistent (o-inconsistent).*
3. *$A : \mu \in I$ and $\mathbf{not}\ A : \mu \in I$, or $\sim A : \mu \in I$ and $\mathbf{not}\ \sim A : \mu \in I \Leftrightarrow I$ is inconsistent in default (d-inconsistent).*

When an interpretation I is o-inconsistent or d-inconsistent, we simply say I is *inconsistent*. We do not see the e-inconsistency as a problematic inconsistency since by the condition 2 of Definition 5, $A : \mu \in I$ and $\neg A : \mu = A : \neg\mu \in I$ imply $A : (\mu \sqcup \neg\mu) \in I$ and we think $A : \mu$ and $\neg A : \mu$ are an acceptable differentia.

Let I be an interpretation such that $\sim A : \mu \in I$. By the condition 1 of Definition 5, for any ρ such that $\rho \geq \mu$, if $A : \rho \in I$ then I is o-inconsistent. In other words, $\sim A : \mu$ rejects all recognitions ρ such that $\rho \geq \mu$ about A. This is the underlying reason for adopting the condition 3 of Definition 5.

The coherence principle is an important property for the semantics of ELP, and is required to properly interpret ELP (Alferes & Pereira 1996). We extend the coherence principle for multi-valuedness as follows.

Definition 7 (**Coherence Principle**). *For an annotated objective literal L and an interpretation I, I is said to be coherent, when I satisfies the following condition: if $\sim L \in I$, then $\mathbf{not}\ L \in I$.*

The coherence principle requires that for some atom A, if there must not be a recognition μ explicitly ($\sim A : \mu$), there is no recognition μ so far ($\mathbf{not}\ A : \mu$), and if there is a recognition μ explicitly ($A : \mu$), there may be a recognition μ for now ($\mathbf{not}\ \sim A : \mu$). Then we define satisfaction.

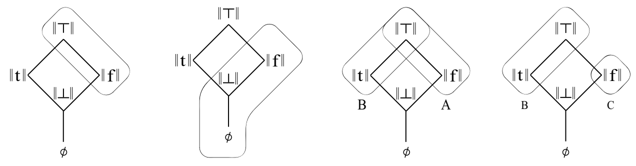
Definition 8 (**Satisfaction**). *Let I be an interpretation. For any annotated objective literal H and annotated literal L and L_i, we define the satisfaction relation denoted by '$\models$' as follows.*

- $I \models L \Leftrightarrow L \in I$
- $I \models L_1 \,\&\, \cdots \,\&\, L_n \Leftrightarrow I \models L_1, \ \ldots, \ I \models L_n$
- $I \models H \leftarrow L_1 \,\&\, \cdots \,\&\, L_n \Leftrightarrow I \models H$ *or* $I \not\models L_1 \,\&\, \cdots \,\&\, L_n$

Explicit negation and interpretation

Here we discuss the formal difference between o-explicit negation and e-explicit negation, using the notion of interpretation. We consider an interpretation that assigns an ideal of a complete lattice of truth values to an annotated atom. The ideals-based interpretation was first introduced in (Kifer & Subrahmanian 1992). The ideals constructed from $\mathcal{FOUR}$, denoted by $\mathcal{I}(\mathcal{FOUR})$, form a complete lattice under set inclusion (see the right part of Figure 1, where $\|\mu\| = \{\rho \in \mathcal{T} \mid \rho \leq \mu\}$).

Let us consider the negligence of car accidents, referring to Example 4, and suppose an agent A asserts $\neg negligence(A) : \mathbf{t} = negligence(A) : \mathbf{f}$ (meaning that A does not have a negligence). The interpretation which can satisfy this assertion is the one that assigns an ideal containing **f** to $negligence(A)$, and has the range shown in Figure 3 (1). The ideal $\|\top\|$ of this range means that agent A can accept e-consistency when $negligence(A)$: **t** (A has a negligence) is asserted by someone. Then let us consider an agent A who asserts $\sim negligence(A)$: **t** (an assertion which says that the negligence is on the side of A is never acknowledged). If the interpretation is to be o-consistent, it has to assign an ideal not containing **t** (i. e., ϕ, $\|\bot\|$ or $\|\mathbf{f}\|$) to $negligence(A)$, and has the range shown in Figure 3 (2). These difference between $\sim fault(A)$: **t** and $\neg fault(A)$: **t**$(= fault(A)$: **f**) turn out to play a significant role in our formalization of multiple-valued argumentation. Then, the argumentation procedure, such as either respecting or rejecting other agents' assertions, is to be represented through these kinds of negations. If both $\neg negligence(A)$: **t** and $\sim negligence(A)$: **t** are asserted at the same time, the ideal which should be assigned to $negligence(A)$ comes to have the overlapped range of two ranges (i. e., $\|\mathbf{f}\|$).



(1) $\neg fault(A)$: **t** (2) $\sim fault(A)$: **t** (3) Agent A & B (4) Agent B & C

Figure 3: The range of the interpretation

Next, let us consider argumentation about movies. Suppose three agents A, B and C assert the following respectively: $interesting(movie\_\alpha)$: **f**, $interesting(movie\_\alpha)$: **t**, and $interesting(movie\_\alpha)$: **f** & $\sim interesting(movie\_\alpha)$: **t**.

Agent B says movie α is interesting, and Agent A says movie α is not interesting. The agents A and B just assert their own opinions, and do not intend to reject the other opinion. As shown in Figure 3 (3), the range of interpretation which satisfies both assertions is the overlapped portion of two regions (including $interesting(movie\_\alpha) : \top$). It means that two agents can acknowledge each other's opinion. In contrast, agent C not only says the movie α is not interesting, but also says C can not acknowledge B's opinion. That is, Agents C takes an exclusive attitude in the argument or dialogue. The range of interpretation which satisfies both assertions B and C has no overlapping portion (i. e., they are in an o-inconsistent state) as shown in Figure 3 (4)

Abstract argumentation framework

We introduce an abstract argumentation framework as preliminaries for the succeeding sections. We use the same definitions as (Dung 1993)(Prakken & Sartor 1997)(Schweimeier & Schroeder 2002) though the knowledge representation language is different. The abstract set of arguments and attack relation on arguments are concretized below.

Acceptable and Justified arguments

We define the argumentation semantics as the least fixpoint of the function which collects all acceptable arguments.

Definition 9 (**Attack relation** (Dung 1993)). *Let $Args$ be a set of abstract arguments. An attack relation x on $Args$ is a binary relation on $Args$, i. e., $x \subseteq Args \times Args$.*

Definition 10 (x/y-**acceptable and justified argument** (Dung 1993)). *Let x and y be attack relations on $Args$. Suppose $Arg_1 \in Args$ and $S \subseteq Args$. Then Arg_1 is x/y-acceptable wrt. S if for every $Arg_2 \in Args$ such that $(Arg_2, Arg_1) \in x$ there exists $Arg_3 \in S$ such that $(Arg_3, Arg_2) \in y$.*

The function $F_{Args,x/y}$ mapping from $\mathcal{P}(Args)$ to $\mathcal{P}(Args)$ is defined by $F_{Args,x/y}(S) = \{Arg \in Args \mid Arg$ is x/y-acceptable wrt. $S\}$. We denote a least fixpoint of $F_{Args,x/y}$ by $J_{Args,x/y}$. An argument Arg is x/y-justified if $Arg \in J_{x/y}$; an argument is x/y-overruled if it is attacked by a x/y-justified argument; and an argument is x/y-defensible if it is neither x/y-justified nor x/y-overruled.

We write simply $F_{x/y}$ and $J_{x/y}$ for $F_{Args,x/y}$ and $J_{Args,x/y}$ when $Args$ is obvious. Since $F_{x/y}$ is monotonic, it has a least fixpoint, and can be constructed by the iterative method (Dung 1993).

When argumentation is treated as one of the methods of a consensus-attainment or a collision-avoidance, the justified arguments can not conflict each other. We define the conflict-freeness for a set of justified arguments in an abstract argumentation framework as follows.

Definition 11 (**Conflict-free** (Dung 1993)). *Let $Args$ be an abstract argument set, and x be an attack relation on $Args$. $S \subseteq Args$ is conflict-free wrt. $x \Leftrightarrow S$ does not contain arguments $Arg1$ and $Arg2$ such that $(Arg1, Arg2) \in x$.*

In ELP, Schweimeier and Schroeder studied a condition in which a set of justified arguments is conflict-free wrt. x (Schweimeier & Schroeder 2002). This result applies not only for an argumentation on ELP but also for abstract argumentation frameworks.

Theorem 1 *Let x and y be attack relations on $Args$. If $x \supseteq y$ then $J_{Args,x/y}$ is conflict-free wrt. x.*

Proof Similar to (Schweimeier & Schroeder 2002).

Dialectical proof theory

Justified arguments can be dialectically determined from a set of arguments by the dialectical proof theory. We give the sound and complete dialectical proof theory for the abstract argumentation semantics $J_{Args,x/y}$.

Definition 12 (x/y-**dialogue** (Prakken & Sartor 1997)). *An x/y-dialogue is a finite nonempty sequence of moves $move_i = (Player_i, Arg_i), (i \geq 1)$ such that*

1. *$Player_i = P$ (Proponent) iff i is odd; and $Player_i = O$ (Opponent) $\Leftrightarrow$ i is even.*
2. *If $Player_i = Player_j = P$ $(i \neq j)$ then $Arg_i \neq Arg_j$.*
3. *If $Player_i = P$ $(i \geq 3)$ then $(Arg_i, Arg_{i-1}) \in y$; and if $Player_i = O$ $(i \geq 2)$ then $(Arg_i, Arg_{i-1}) \in x$.*

Definition 13 (x/y-**dialogue tree** (Prakken & Sartor 1997)). *An x/y-dialogue tree is a tree of moves such that every branch is an x/y-dialogue, and for all moves $move_i = (P, Arg_i)$, the children of $move_i$ are all those moves $(O, Arg_{i+1,j})$ $(j \geq 1)$ such that $(Arg_{i+1,j}, Arg_i) \in x$.*

Definition 14 (**Provably** x/y-**justified** (Prakken & Sartor 1997)). *An x/y-dialogue D is a winning x/y-dialogue $\Leftrightarrow$ the termination of D is a move of proponent. An x/y-dialogue tree T is a winning x/y-dialogue tree $\Leftrightarrow$ every branch of T is a winning x/y-dialogue. An argument Arg is a provably x/y-justified argument $\Leftrightarrow$ there exists a winning x/y-dialogue tree with Arg as its root.*

Theorem 2 *Let $Args$ be an abstract argument set. Then $Arg \in Args$ is provably x/y-justified $\Leftrightarrow$ Arg is x/y-justified*[1].

[1]Refer to (Takahashi & Sawamura 2004) for the proofs omitted in this paper.

Multiple-valued argumentation for EALP

In this section, we define the notion of arguments in EALP and associated attack relationship, and formalize the semantics of multiple-valued argumentation and its dialectical proof theory by concretizing abstract argumentation framework. We further describe the properties of the multiple-valued argumentation using interpretation induced from justified arguments.

Annotated arguments

Kifer and Subrahmanian introduced the notion of *reductants* for complete proof theory of GAP (Kifer & Subrahmanian 1992). In our multiple-valued argumentation, reductants are needed to allow agents to build arguments in EALP. We further introduce *minimal reductants* in order to exclude redundant and irrelevant arguments.

Definition 15 (**Reductant and Minimal reductant**). *Suppose P is an EALP, and C_i $(1 \leq i \leq k)$ are annotated rules in P of the form:*

$$A\!:\!\rho_i \leftarrow L_1^i \,\&\, \ldots \,\&\, L_{n_i}^i$$

in which A is an atom. Let $\rho = \sqcup\{\rho_1, \ldots, \rho_k\}$. Then the following annotated rule is a reductant of P.

$$A\!:\!\rho \leftarrow L_1^1 \,\&\, \ldots \,\&\, L_{n_1}^1 \,\&\, \ldots \,\&\, L_1^k \,\&\, \ldots \,\&\, L_{n_k}^k.$$

A reductant is called a minimal reductant when there does not exist non-empty proper subset $S \subset \{\rho_1, \ldots, \rho_k\}$ such that $\rho = \sqcup S$.

Definition 16 (**Annotated arguments**). *Let P be an EALP. An annotated argument in P is a finite sequence $Arg = [r_1, \ldots, r_n]$ of rules in P such that for every i $(1 \leq i \leq n)$,*

1. *r_i is either a rule in P or a minimal reductant in P.*
2. *For every annotated atom $A\!:\!\mu$ in the body of r_i, there exists a r_k $(n \geq k > i)$ such that $A\!:\!\rho$ $(\rho \geq \mu)$ is head of r_k.*
3. *For every o-explicit negation $\sim A:\mu$ in the body of r_i, there exists a r_k $(n \geq k > i)$ such that $\sim A\!:\!\rho$ $(\rho \leq \mu)$ is head of r_k.*
4. *There exists no proper subsequence of $[r_1, \ldots, r_n]$ which meets from the first to the third conditions, and includes r_1.*

A subargument of Arg is a subsequence of Arg which is an argument. The conclusions of rules in Arg are called conclusions of Arg, and the assumptions of rules in Arg are called assumptions of Arg. We write $concl(Arg)$ for the set of conclusions and $assm(Arg)$ for the set of assumptions of Arg. We denote the set of all arguments in P by $Args_P$, and define the set of all arguments in a set of EALPs $MAS = \{KB_1, \ldots, KB_n\}$ by $Args_{MAS} = Args_{KB_1} \cup \cdots \cup Args_{KB_n}$.

Example 6 *Let P be the following EALP.*

$$\begin{array}{llll} P = \{ & p\!:\!\boldsymbol{t} \leftarrow q\!:\!\boldsymbol{t} \,\&\, \mathbf{not}\ r\!:\!\boldsymbol{t}, & p\!:\!\boldsymbol{f} \leftarrow q\!:\!\boldsymbol{f}, & \\ & \sim p\!:\!\boldsymbol{t} \leftarrow r\!:\!\boldsymbol{t} \,\&\, \sim q\!:\!\boldsymbol{t}, & q\!:\!\top \leftarrow, & \\ & \sim q\!:\!\bot \leftarrow, & r\!:\!\boldsymbol{t} \leftarrow & \} \end{array}$$

Then a set of all arguments constructed from P is

$$\begin{array}{ll} Args_P = \{ & [p\!:\!\boldsymbol{t} \leftarrow q\!:\!\boldsymbol{t} \,\&\, \mathbf{not}\ r\!:\!\boldsymbol{t},\ q\!:\!\top \leftarrow] \\ & [p\!:\!\boldsymbol{f} \leftarrow q\!:\!\boldsymbol{f},\ q\!:\!\top \leftarrow] \\ & [p\!:\!\top \leftarrow q\!:\!\boldsymbol{t} \,\&\, q\!:\!\boldsymbol{f} \,\&\, \mathbf{not}\ r\!:\!\boldsymbol{t},\ q\!:\!\top \leftarrow] \\ & [\sim p\!:\!\boldsymbol{t} \leftarrow \sim q\!:\!\boldsymbol{t} \,\&\, r\!:\!\boldsymbol{t},\ \sim q\!:\!\bot \leftarrow,\ r\!:\!\boldsymbol{t} \leftarrow] \\ & [q\!:\!\top \leftarrow],\ [\sim q\!:\!\bot \leftarrow],\ [r\!:\!\boldsymbol{t} \leftarrow] \qquad \}. \end{array}$$

$Arg_3 = [p:\top \leftarrow q:\boldsymbol{t} \,\&\, q:\boldsymbol{f} \,\&\, \mathbf{not}\ r:\boldsymbol{t},\ q:\top \leftarrow]$ in $Args_P$ is an argument in which the first rule is the minimal reductant constructed from the first and second rules of P. By Definition 16 (2), it contains the fourth rule of P which is annotated by $\top$ greater than $\boldsymbol{t}$ or $\boldsymbol{f}$ in the end of the argument, as the ground for the antecedents $q\!:\!\boldsymbol{t}$ and $q\!:\!\boldsymbol{f}$. The set of conclusions and the set of assumptions are $concl(Arg_3) = \{p\!:\!\top, q\!:\!\top\}$ and $assm(Arg_3) = \{\mathbf{not}\ r\!:\!\boldsymbol{t}\}$ respectively. The ground for the assumption $\mathbf{not}\ r\!:\!\boldsymbol{t}$ is not required.

$Arg_4 = [\sim p:\boldsymbol{t} \leftarrow\sim q:\boldsymbol{t} \,\&\, r:\boldsymbol{t},\ \sim q:\bot \leftarrow,\ r:\boldsymbol{t} \leftarrow]$ is an argument containing the third rule of P at the head. By Definition 16 (3), it has the fifth rule $\sim q\!:\!\bot \leftarrow$ of P which is annotated by $\bot$ less than $\boldsymbol{t}$ as the ground for the antecedent $\sim q\!:\!\boldsymbol{t}$.

A non-minimal reductant usually results in having a longer antecedent than a minimal reductant. If an argument is made by non-minimal reductants, the argument may be redundant in its content. Furthermore, a non-minimal reductant occasionally tends to be irrelevant rule (refer to (T. Takahashi & Sawamura 2003) for details).

We stated that the epistemic states of agents are represented by interpretations. Here, we define the notion of satisfaction for arguments, and view the epistemic state of an agent who has a set of arguments as interpretation which satisfy those arguments.

Definition 17 (**Satisfaction for arguments**). *Let P be an EALP, I be an interpretation on P, and Arg be an argument in P. Then, $I \models Arg \ \Leftrightarrow\ \forall H \in concl(Arg)\ I \models H$ and $\forall \mathbf{not}\ L \in assm(Arg)\ I \models \mathbf{not}\ L$. In addition, letting $Args$ be a set of arguments in P, $I \models Args \ \Leftrightarrow\ \forall Arg \in Args\ \ I \models Arg$.*

Corollary 1 *Let Arg be an argument, and I be an interpretation such that $I \models Arg$. Then, for all rules $r \in Arg$, $I \models r$.*

Proof It follows from Definition 8, 16 and 17.

Generally, in ordinary logic programs, interpretations which satisfy every rule are taken into account, and called models. In the argumentation, however, it becomes necesary to consider that the satisfaction for arguments (i. e., the satisfaction for all conclusions and all assumptions) should be taken into account rather than the satisfaction for rules only.

Example 7 *Let I_1 and I_2 be interpretations such that $I_1 \not\models \mathbf{not}\ q\!:\!\top$, $I_2 \models p\!:\!\boldsymbol{t} \,\&\, \mathbf{not}\ q\!:\!\top$, and $Arg = [p\!:\!\boldsymbol{t} \Leftarrow \mathbf{not}\ q\!:\!\top]$ be an argument. Then I_1 satisfies only one rule in Arg ($I_1 \models p:\boldsymbol{t} \Leftarrow \mathbf{not}\ q\!:\!\top$), but $I_1 \not\models Arg$ because it does not satisfy an assumption $\mathbf{not}\ q\!:\!\top$ in Arg. In contrast, because I_2 satisfies all conclusions and all assumptions in Arg, $I_2 \models Arg$. An argument Arg asserts its conclusion based on its assumption $\mathbf{not}\ q:\top$, and so I_1 is not suitable as the epistemic state of an agent with Arg.*

Attack relation

The semantics of the argumentation depends on what sort of attack relation is considered to deal with conflicts among arguments. It would be reasonable to think that conflicts among arguments occur when the interpretation satisfying a set of arguments is inconsistent.

We define the "rebut" as an attack relation associated with o-inconsistency and the "undercut" as an attack relation associated with d-inconsistency.

Definition 18 (**Rebut**). *Arg_1 rebuts Arg_2 ⇔ there exists $A:\mu_1 \in concl(Arg_1)$ and $\sim A:\mu_2 \in concl(Arg_2)$ such that $\mu_1 \geq \mu_2$, or exists $\sim A:\mu_1 \in concl(Arg_1)$ and $A:\mu_2 \in concl(Arg_2)$ such that $\mu_1 \leq \mu_2$.*

Definition 19 (**Undercut**). *Arg_1 undercuts Arg_2 ⇔ there exists $A:\mu_1 \in concl(Arg_1)$ and $\mathbf{not}\ A:\mu_2 \in assm(Arg_2)$ such that $\mu_1 \geq \mu_2$, or exists $\sim A:\mu_1 \in concl(Arg_1)$ and $\mathbf{not}\ \sim A:\mu_2 \in assm(Arg_2)$ such that $\mu_1 \leq \mu_2$.*

Proposition 1 *For some Arg_1 and Arg_2 in $Args$, if Arg_1 rebuts Arg_2, I such that $I \models Args$ is o-inconsistent. And if Arg_1 undercuts Arg_2, I such that $I \models Args$ is d-inconsistent.*

Proof It follows from Definition 5, 8, 17, 18 and 19.

The rebuttal relation is a symmetrical attack relation. And the undercut relation is not always one-directional (for example, $[p : \mathbf{t} \leftarrow \mathbf{not}\ \sim q : \mathbf{t}]$ and $[\sim q : \mathbf{t} \leftarrow \mathbf{not}\ p : \mathbf{t}]$). The one-directional undercut relation is called the "strictly undercut".

Definition 20 (**Strictly undercut**). *Arg_1 strictly undercuts Arg_2 $\Leftrightarrow$ Arg_1 undercuts Arg_2 and Arg_2 does not undercut Arg_1.*

We also define the combined attack relation associated with o-inconsistency and d-inconsistency.

Definition 21 (**Attack**). *Arg_1 attacks Arg_2 $\Leftrightarrow$ Arg_1 rebuts or undercuts Arg_2.*

Under the idea that a conclusion can not hold when its assumption is negated, the undercut may be stronger than rebuttal. Taking this into consideration, we define another combined attack relation called "defeat" (in ELP, it is defined in (Prakken & Sartor 1997)).

Definition 22 (**Defeat**). *Arg_1 defeats Arg_2 $\Leftrightarrow$ Arg_1 undercuts Arg_2, or Arg_1 rebuts Arg_2 and Arg_2 does not undercut Arg_1.*

Proposition 2 *For some Arg_1 and Arg_2 in $Args$, if Arg_1 attacks or defeats Arg_2, I such that $I \models Args$ is o-inconsistent or d-inconsistent.*

Proof It follows from Definition 21, 22, and Proposition 1.

Argumentation semantics for EALP

To concretize an abstract argumentation framework, we need a concrete argument set and two concrete attack relations. We already defined the notion of argument set $Args_P$ for an EALP P and $Args_{MAS}$ for a set of EALPs MAS. In what follows, we simply write an argument set by $Args$ when we do not tell $Args_P$ and $Args_{MAS}$ apart. Now we have to select two attack relations. If the argumentation is treated as the methods of a consensus-attainment or a collision-avoidance, justified arguments should be conflict-free under those attack relations.

By Theorem 1, it is sufficient to adopt $J_{Args,a/y}$ (a = attack, $a \supseteq y$) in order to avoid the rebuttal and the undercut among justified arguments (i. e., $J_{Args,a/y}$ will be conflict-free wrt. attack). But, actually, we can show the conflict-free wrt. the defeat relationship implies the conflict-free wrt. the attack, and $J_{Args,d/y}$ (d = defeat, $d \supseteq y$) is also the conflict-free wrt. the attack.

Theorem 3 *If a set of arguments S is conflict-free wrt. the defeat relationship, then S is conflict-free wrt. the attack.*

We think $J_{Args,d/y}$ is closer to human intuition than $J_{Args,a/y}$ as can be seen in (Prakken & Sartor 1997) for the case of ELP. If we adopt y relation such that $d \supseteq y$, $J_{Args,d/y}$ will be conflict-free wrt. the defeat. The corresponding ones in the subsection are the defeat, the undercut and the strictly undercut. For the set of justified arguments associated with these attack relations, the following relationship holds.

Proposition 3 $J_{Args,d/su} = J_{Args,d/u} = J_{Args,d/d}$.

Thus, the set of justified arguments is always the same even if we adopt any attack relation. And so we adopt the strictly undercut because it is smallest or simplest in these attack relations, and straightforward .

Let P be an EALP, and MAS be a set of EALPs. Then we define the semantics of argumentation on P by $J_{Args_P,d/su}$ (J_P for short), and define the semantics of argumentation on MAS by $J_{Args_{MAS},d/su}$ (J_{MAS} for short). When we do not need to tell apart P and MAS, we simply denote $Args_P$ and $Args_{MAS}$ by $Args$, and denote J_P and J_{MAS} by J. J_P and J_{MAS} can be dialectically determined by the dialectical proof theory.

Example 8 *We describe an example of argumentation about the pros and cons of the death penalty of murderers. Suppose a complete lattice $\mathcal{T} = \Re[0,1]^2$ where $(\mu_1, \rho_1) \leq (\mu_2, \rho_2) \Leftrightarrow \mu_1 \leq \mu_2$ and $\rho_1 \leq \rho_2$, and in $(\mu, \rho) \in \mathcal{T}$, μ and ρ represent the degrees of an affirmation and a negation respectively. This truth value can represent e-inconsistent or unknown state and degree of truth together. Let $MAS = \{KB_1, KB_2, KB_3\}$ be a set of EALPs, and each KB_i be a knowledge base such that:*

$KB_1 = \{$
$\sim agree(death) : (0.0, 0.8) \leftarrow$
$\quad hate(family, murderer) : (0.8, 0.0)\ \&$
$\quad desire(family, death) : (0.7, 0.0),$
$hate(family, murderer) : (1.0, 0.0) \leftarrow$
$\quad \sim allow(family, murderer) : (0.5, 0.0),$
$desire(family, death) : (0.7, 0.0) \leftarrow,$
$\sim desire(family, death) : (0.0, 0.8) \leftarrow,$
$\sim allow(family, murderer) : (0.5, 0.0) \leftarrow\ \},$

$KB_2 = \{$
$agree(death) : (0.2, 0.4) \leftarrow atone(death, guilt) : (0.0, 0.5),$
$atone(death, guilt) : (0.2, 0.8) \leftarrow$
$\quad \mathbf{not}\ remorse(dead) : (1.0, 0.0),$
$agree(death) : (0.0, 1.0) \leftarrow$
$\quad \mathbf{not}\ \sim allow(family, murderer) : (0.8, 0.0)\ \},$

$KB_3 = \{$
$agree(death) : (0.0, 0.6) \leftarrow desire(family, death) : (0.0, 0.6),$
$desire(family, death) : (0.0, 1.0) \leftarrow$
$\quad \mathbf{not}\ assuage(death, family) : (0.7, 0.0),$
$agree(death) : (0.6, 0.0) \leftarrow$
$\quad hate(family, murderer) : (0.6, 0.0),$
$hate(family, murderer) : (0.9, 0.2) \leftarrow\ \},$

Figure 4 shows every possible argument and every possible attack relation among them. The justified arguments are $J_{KBs} = \{Arg_{11}, Arg_{12}, Arg_{13}, Arg_{14}, Arg_{21}, Arg_{22}, Arg_{31}, Arg_{32}\}$. Comprehensively, the justified arguments would be able to read as follows: "It is sure that a bereaved family hates a murderer (Arg_{31}), and so agreement to the death penalty is partly possible (Arg_{32}). In addition, because a bereaved family can not accept allowing a murderer (Arg_{11}), and desires the death penalty (Arg_{13}), a complete opposition of the death penalty is never acknowledged (Arg_{14}). However, because there is no evidence to show that a dead man is in remorse for the crime, and the death can not atone a guilt (Arg_{21}), an opposition of the death penalty is acknowledged to some degree at the same time (Arg_{22})."

In Example 8, assume $\neg(\mu, \rho) = (1.0 - \mu, 1.0 - \rho)$, and replace every o-explicit negation '$\sim$' in each KB_i by e-explicit negation '$\neg$'. Then, Arg_{15} will not rebut Arg_{33}, and both conclusions hold (a bereaved family desires and does not desire the death penalty). The degree of opposition of the death penalty will increase to 0.6 (Arg_{34}). Thus agents can incorporate what they intend to do in argumentation into their knowledge base by selecting appropriate explicit negation according to a topic for argumentation.

Example 9 *Let us consider an argumentation about the monthly schedule management. Here we use an unconventional complete lattice of truth values which is the power set $\mathcal{P}(\{1, \ldots, 31\})$ of the set of the monthly dates, with the order by the set-inclusion relation. Then an annotated atom $work(a) : \{5, 6\}$ reads "Agent a works on the 5th and the 6th". It asserts that the proposition $work(a)$ is true only in a certain time interval. $\sim work(a) : \{5, 6\}$ reads "Agent a does not work on the 5th and the 6th". We define the epistemic explicit negation so as to be $\neg\mu = \{1, \ldots, 31\} - \mu$, and thus $\neg work(a) : \{5, 6\}$ reads "Agent a works on the dates*

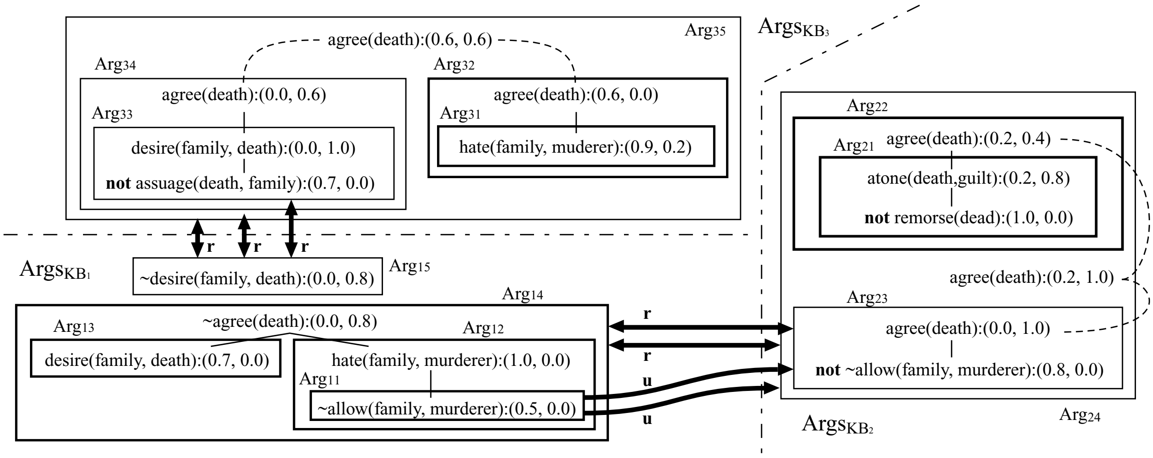


Figure 4: Relation among arguments in Example 8, where '**u**' stands for undercut, and '**r**' stands for rebuttal. The broken lines stand for the operation of the reductant, and arguments framed in a thick line are justified arguments.

except the 5th and the 6th". The difference and significance between the ontological and epistemic explicit negations is obvious. Under this complete lattice of truth values, we consider $MAS = \{KB_m, KB_a, KB_b, KB_o\}$, where KB of each agent is, in EALP,

$$
\begin{aligned}
&KB_m = \{\\
&\quad finish(project):\{6\} \leftarrow\\
&\qquad work(a):\{3,4,5\}\,\&\,arrive(component):\{5\},\\
&\quad work(a):\{3,4,5\} \leftarrow,\\
&\quad arrive(component):\{5\} \leftarrow,\\
&\quad pay(upcharge):\{8\} \leftarrow\ \},
\end{aligned}
$$

$$
\begin{aligned}
&KB_a = \{\\
&\quad \sim work(a):\{5\} \leftarrow \mathbf{not}\ work(b):\{5\}\,\&\,holiday:\{5\},\\
&\quad \sim work(a):\{12\} \leftarrow\\
&\qquad \mathbf{not}\ work(b):\{12\}\,\&\,holiday:\{12\},\\
&\quad holiday:\{5,6,12,13\} \leftarrow\ \},
\end{aligned}
$$

$$
\begin{aligned}
&KB_b = \{\\
&\quad \neg work(b):\{12,19,26\} \leftarrow,\\
&\quad holiday:\{5,6,12,13\} \leftarrow\ \},
\end{aligned}
$$

$$
\begin{aligned}
&KB_o = \{\\
&\quad \sim arrive(component):\{5\} \leftarrow \mathbf{not}\ pay(upcharge):\phi\ \}.
\end{aligned}
$$

Where KB_m, KB_a, KB_b and KB_o stand for knowledge bases of a manager agent m, employee agents a, b and a subcontractor agent o respectively. Agent m's argument which has the conclusion $finish(project):\{6\}$ (the project will finish on the 6th) is justified by the dialectical proof theory as shown in the figure 5.

In the winning dialogue tree, initially Agent m says "if a component will arrive on the 5th, and Agent a works on the 3th, the 4th and the 5th, then the project will finish on the 6th". Then Agent o defeats it as follows "I will be not able to bring a component on the 5th if the additional charge is not paid". But Agent m strictly undercuts o's argument by saying "I will pay it to you on the 8th". For the first argument of Agent m, Agent a also defeats by saying "the 5th is a holiday, and if the coworker b does not work, I do not want to work on the 5th". However Agent b strictly undercuts it by saying "I will work on days except the 12th, 19th and the 26th".

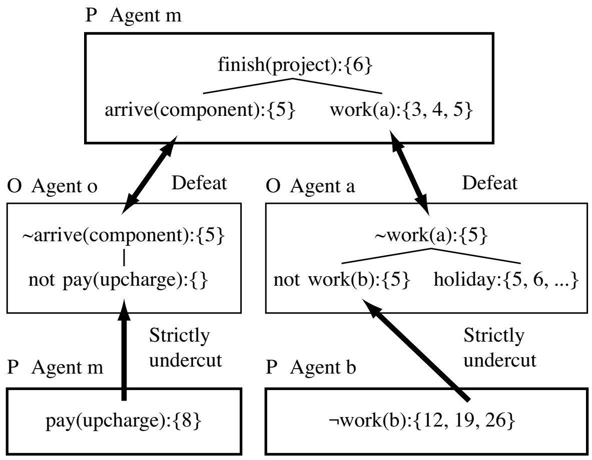


Figure 5: The winning dialogue tree in Example 9

Consequently, the first argument of Agent m is justified.

These idiosyncratic uses of complete lattices of different type make our points: (i) the variety of the expressiveness of EGAP, and (ii) the versatility of the multiple-valued argumentation.

Properties of argumentation for EALP

In this section, we examine logical properties of argumentation semantics for EALP. For this purpose, the examination by interpretation is better than by the set of arguments since we view the interpretation satisfying arguments as the epistemic state of an agent. In addition, even if the justified arguments are conflict-free, the interpretation satisfying them is not always consistent.

We begin with the mapping from the justified arguments to an interpretation, and define the semantics of EALP by its interpretation. This amounts to specifying only one epistemic state of an agent (or agent group) from the justified arguments, and to looking at whether each annotated literal is justified (i. e., agreed) in an argumentation.

Definition 23 (**Interpretation by argumentation**). *Suppose a set of arguments $Args$ and a set of justified arguments J on $Args$. The* *interpretation by justified arguments $\mathcal{A}'_{Args}(J)$ is the least interpretation wrt. the inclusive relation as follows: for every annotated atom $A\!:\!\mu$,*

1. $\forall Arg \in J \;\; A\!:\!\mu \in concl(Arg) \;\Rightarrow\; \mathcal{A}'_{Args}(J) \models A\!:\!\mu$;
2. $\forall Arg \in J \sim A\!:\!\mu \in concl(Arg) \;\Rightarrow\; \mathcal{A}'_{Args}(J) \models \sim A\!:\!\mu$;
3. $\{\forall S = \{Arg_1, \ldots, Arg_n\} \subseteq Args \;\;\; \forall \rho_1, \ldots, \rho_n$ *such that* $\sqcup\{\rho_1, \ldots, \rho_n\} \geq \mu \;\;\; \exists Arg_i \in S \;\;\; A : \rho_i \notin concl(Arg_i)$ *or* Arg_i *is overruled* $\} \;\Rightarrow\; \mathcal{A}'_{Args}(J) \models \mathbf{not}\; A\!:\!\mu$;
4. $\{\forall Arg \in Args \;\; \forall \rho \leq \mu \;\sim A\!:\!\rho \notin concl(Arg)$ *or* Arg *is overruled* $\} \;\Rightarrow\; \mathcal{A}'_{Args}(J) \models \mathbf{not} \;\sim A\!:\!\mu$.

Let P be an EALP, MAS be a set of EALPs. The interpretation by argumentation on P ($\mathcal{A}(P)$) and the interpretation by argumentation on MAS ($\mathcal{A}(MAS)$) are defined as follows: $\mathcal{A}(P) = \mathcal{A}'_{Args_P}(J_P)$ and $\mathcal{A}(MAS) = \mathcal{A}'_{Args_{MAS}}(J_{MAS})$.

The semantics of EALP P is defined by the interpretation by argumentation $\mathcal{A}(P)$, and the semantics of a set of EALPs MAS is defined by the interpretation by argumentation $\mathcal{A}(MAS)$.

We focus on four properties, o-consistency, d-consistency, coherence principle and a satisfaction of justified arguments. Initially we state d-consistency.

Theorem 4 *Let MAS be a set of EALPs. Then $\mathcal{A}(MAS)$ is consistent wrt. the default negation.*

Since $\mathcal{A}(MAS)$ coincides with $\mathcal{A}(P)$ in the case of $MAS = \{P\}$, $\mathcal{A}(P)$ is also consistent wrt. the default negation by Theorem 4.

FromTheorem 4, it follows that the Definition 23 has been defined to guarantee d-consistency. In contrast, o-consistency is not generally satisfied as shown in the following example.

Example 10 *Assume a set of EALPs $MAS = \{\{p : \boldsymbol{t} \leftarrow\}, \{p : \boldsymbol{f} \leftarrow\}, \{\sim p : \top \leftarrow\}\}$. Since $Args_{MAS} = \{[p : \boldsymbol{t} \leftarrow], [p : \boldsymbol{f} \leftarrow], [\sim p : \top \leftarrow]\}$, and the defeat relation does not occur on arguments in $Args_{MAS}$, $J_{MAS} = Args_{MAS}$. By Definition 23 and 5, $\mathcal{A}(MAS) \models p : \top \,\&\, \sim p : \top$. Thus $\mathcal{A}(MAS)$ is o-inconsistent.*

In the above example, the problem lies in having dealt with the attack relation between each argument in the argumentation framework one by one. The argument $[\sim p : \top \leftarrow]$ do not have a conflict individually with $[p : \mathbf{t} \leftarrow]$ and $[p : \mathbf{f} \leftarrow]$, but the argument $[\sim p : \top \leftarrow]$ conflicts with the group of them. This is a problem proper to the multiple-valued argumentation that does not appear in the two-valued argumentation. To universally resolve this problem, we would need to modify the fundamental structure of argumentation framework, taking into consideration new attack relations among any groups of arguments. Later we will give another solution to it.

If the knowledge base is a single EALP $P = \{p\!:\!\mathbf{t} \leftarrow,\; p\!:\!\mathbf{f} \leftarrow,\; \sim p : \top \leftarrow\}$, we can have an argument $[p : \top \leftarrow]$ by making reductant from $p\!:\!\mathbf{t} \leftarrow$ and $p\!:\!\mathbf{f} \leftarrow$. An argument $[p\!:\!\top \leftarrow]$ is identified with the combination of $[p\!:\!\mathbf{t} \leftarrow]$ and $[p\!:\!\mathbf{f} \leftarrow]$, and conflicts with an argument $[\sim p : \top \leftarrow]$. The interpretation by argumentation on the single EALP can be o-consistenct by reductants.

Theorem 5 *Let P be an EALP. Then $\mathcal{A}(P)$ is ontologically consistent.*

Next, let us examine the satisfaction of justified arguments. We think the interpretation by argumentation has to satisfy every justified argument to view it as an epistemic state of an agent. However it is not generally satisfied as shown in the following example.

Example 11 *Assume a set of EALPs $MAS = \{\{p : \boldsymbol{t} \leftarrow\}, \{p : \boldsymbol{f} \leftarrow\}, \{q : \boldsymbol{t} \leftarrow \mathbf{not}\; p : \top\}\}$. Since $Args_{MAS} = \{[p : \boldsymbol{t} \leftarrow], [p : \boldsymbol{f} \leftarrow], [q : \boldsymbol{t} \leftarrow \mathbf{not}\; p : \top]\}$, and the defeat relation does not occur on arguments in $Args_{MAS}$, $J_{MAS} = Args_{MAS}$. Since there exist non-overruled arguments which have conclusions $p : \boldsymbol{t}$ or $p\!:\!\boldsymbol{f}$, and $\top = \boldsymbol{t} \sqcup \boldsymbol{f}$, by Definition 23, $\mathcal{A}(MAS) \not\models \mathbf{not}\; p\!:\!\top$. Then $\mathcal{A}(MAS) \not\models J_{MAS}$ since $\mathbf{not}\; p\!:\!\top$ is an assumption of an argument $[q\!:\!\boldsymbol{t} \leftarrow \mathbf{not}\; p\!:\!\top]$.*

Though there is no defeat relation in $Args_{MAS}$ since an argumentation framework does not deal with the attack relations among groups of arguments as we have already stated, it seems that the combination of $[p : \mathbf{t} \leftarrow]$ and $[p : \mathbf{f} \leftarrow]$ undercuts an argument $[q\!:\!\mathbf{t} \leftarrow \mathbf{not}\; p\!:\!\top]$. Hence a justified argument $[q\!:\!\mathbf{t} \leftarrow \mathbf{not}\; p\!:\!\top]$ is not satisfied by the interpretation by argumentation.

This problem does not occur in a single EALP by making reductants as discussed in the previous restoration of o-consistency. If the knowledge base is a single EALP, an argument $[q\!:\!\mathbf{t} \leftarrow \mathbf{not}\; p\!:\!\top]$ will be undercut by $[p\!:\!\top \leftarrow]$ which is constructed from $p\!:\!\mathbf{t} \leftarrow$ and $p\!:\!\mathbf{f} \leftarrow$.

Theorem 6 *Let P be an EALP. Then $\mathcal{A}(P) \models J_P$.*

Finally, we examine the coherence principle. The coherence principle is not generally satisfied as shown in the following example.

Example 12 *Suppose EALP $P = \{p\!:\!\boldsymbol{t} \leftarrow,\; p\!:\!\boldsymbol{f} \leftarrow,\; \sim p\!:\!\top \leftarrow\}$. Then $Args_P = \{[p\!:\!\boldsymbol{t} \leftarrow], [p\!:\!\boldsymbol{f} \leftarrow], [p\!:\!\top \leftarrow], [\sim p\!:\!\top \leftarrow]\}$, and arguments $[p\!:\!\top \leftarrow]$ and $[\sim p\!:\!\top \leftarrow]$ rebut each other. Arguments $[p : \boldsymbol{t} \leftarrow]$ and $[p : \boldsymbol{f} \leftarrow]$ which are not defeated are justified. And, $[p : \top \leftarrow]$ and $[\sim p : \top \leftarrow]$ are defensible. By Definition 23, $\mathcal{A}(P) = \{p : \bot,\; p : \boldsymbol{t},\; p : \boldsymbol{f},\; p : \top,\; \mathbf{not} \;\sim p : \bot,\; \mathbf{not} \;\sim p : \boldsymbol{t},\; \mathbf{not} \;\sim p\!:\!\boldsymbol{f}\}$. Since $\mathbf{not} \;\sim p\!:\!\top$ is not in $\mathcal{A}(P)$ though $p\!:\!\top$ is in $\mathcal{A}(P)$, $\mathcal{A}(P)$ does not satisfy the coherence principle.*

Similarly, since $\mathcal{A}(MAS)$ coincides with $\mathcal{A}(P)$ if $MAS = \{P\}$, for a set of EALPs MAS, $\mathcal{A}(MAS)$ does not always satisfy the coherence principle.

The EALP in the above example has the same rules as in Example 10. Although an argument $[p : \top \leftarrow]$ is identified with the combination of $[p\!:\!\mathbf{t} \leftarrow]$ and $[p\!:\!\mathbf{f} \leftarrow]$, note that they are not treated equally. Since $[p\!:\!\mathbf{t} \leftarrow]$ and $[p\!:\!\mathbf{f} \leftarrow]$ are justified though $[p\!:\!\top \leftarrow]$ is defeated, The coherence principle is not satisfied.

The interpretation not satisfying the coherence principle shown in Example 12 is unlikely to human intuition. Therefore, we turn to giving a subclass of EALP satisfying the coherence principle by restricting annotations in negations.

Definition 24 (**Well-Behaved EALP**). *Let P be an EALP on a complete lattice $\mathcal{T}$ of truth values, $\sim A : \mu$ be a conclusion of a rule in P, and $S \subseteq \mathcal{T}$ be a finite subset such that $S \neq \phi$. Then, if P satisfies the following condition, P is called well-behaved EALP. $\mu \leq \sqcup S \Rightarrow \exists \rho \in S \;\; \mu \leq \rho$.*

In addition, if an assumption $\mathbf{not}\; A : \mu$ of a rule in P also satisfies the above condition, P is called strictly well-behaved EALP.

Example 13 *Suppose the complete lattice $\mathcal{T} = (\{\bot, \boldsymbol{t}_1, \boldsymbol{t}_2, \boldsymbol{f}, \top\}, \leq), \forall x, y \in \{\bot, \boldsymbol{t}_1, \boldsymbol{t}_2, \boldsymbol{f}, \top\} \;\; x \leq y \;\Leftrightarrow\; x = y \,\vee\, x = \bot \,\vee\, y = \top \,\vee\, (x = \boldsymbol{t}_2 \,\wedge\, y = \boldsymbol{t}_1)$ and the EALP $P = \{\sim p : \boldsymbol{t}_1 \leftarrow\}$ on the $\mathcal{T}$. Then P is not well-behaved since $\boldsymbol{t}_1 \leq \sqcup\{\boldsymbol{t}_2,\; f\} = \top$, and there exists no $\rho \in \{\boldsymbol{t}_2, \boldsymbol{f}\}$ such that $\boldsymbol{t}_1 \leq \rho$. If the annotation in P is either $\bot$, $\boldsymbol{f}$, or $\boldsymbol{t}_2$, P is a well-behaved EALP.*

The well-behaved EALP actually yields not only o-consistency but also the coherence principle. The strictly well-behaved EALP also yields the satisfaction of justified arguments. Under the well-behaved EALP, we anew examine the properties of the interpretation by argumentation.

Theorem 7 *Let MAS be a set of well-behaved EALPs. Then, $\mathcal{A}(MAS)$ satisfies the coherence principle ($\mathcal{A}(MAS) \models \sim L \Rightarrow \mathcal{A}(MAS) \models \mathbf{not}\; L$).*

Table 1: Properties of the interpretation by argumentation

Interpretation by argumentation	$\mathcal{A}(P)$	$\mathcal{A}(P')$	$\mathcal{A}(MAS)$	$\mathcal{A}(MAS')$	$\mathcal{A}(MAS'')$
Ontological consistency	◯	◯	×	◯	◯
Consistency in default	◯	◯	◯	◯	◯
Coherence principle	×	◯	×	◯	◯
Satisfaction of justified arguments	◯	◯	×	×	◯

Each knowledge base is an EALP P, a well-behaved EALP P', a set of EALPs MAS, a set of well-behaved EALPs MAS' and a set of strictly well-behaved EALPs MAS''.

Since $\mathcal{A}(MAS)$ coincides with $\mathcal{A}(P)$ in the case of $MAS = \{P\}$, $\mathcal{A}(P)$ also satisfies the coherence principle by Theorem 7.

Theorem 8 *Let MAS be a set of well-behaved EALPs. Then, $\mathcal{A}(MAS)$ is ontologically consistent.*

Finally, we examine the satisfaction of justified arguments. MAS in Example 11 is clearly well-behaved since it does not include o-explicit negation. Hence, $\mathcal{A}(MAS) \models J_{MAS}$ is not generally satisfied even though MAS is a set of well-behaved EALPs. In order to generally satisfy $\mathcal{A}(MAS) \models J_{MAS}$, we need a set of strictly well-behaved EALPs

Theorem 9 *Let MAS be a set of strictly well-behaved EALPs. Then, $\mathcal{A}(MAS) \models J_{MAS}$.*

Table 1 summarizes the results developed so far in this paper. Thus, if we adopt the well-behaved EALP for argumentation on a single EALP (a single agent), and the strictly well-behaved EALPs for argumentation on a multi-agent environment, the consistency, coherence principle and satisfaction of justified arguments are satisfied. This is our principal result.

Conclusion and future work

We presented an attempt to a logic of multiple-valued argumentation (LMA) in which agents can argue with other contenders, using multi-valued knowledge base in the extended annotated logic programming (EALP), and scrutinized its logical properties closely.

EALP was proposed as an expressive knowledge representation language with three kinds of negation, with which agents can make versatile argumentative dialogues since they can signify a momentum or driving force of argumentation. The more negations, the more momentum for argumentation.

LMA was proposed as a logic of argumentation that has multi-valued argumentation semantics and dialectical proof theory. We proved the soundness and completeness for it. Furthermore, we gave the semantics to EALP via the multi-valued argumentation semantics, and examined the properties of the inconsistency, coherence and satisfaction of justified arguments in the semantics. Then, we pointed out a proper problem caused by the introduction of multi-valuedness into argumentation, but discovered a subclass of EALP, a (strictly) well-behaved EALP that satisfies those three properties, resulting in a solution to the problem.

There are some works that can be related to our approach to multiple-valued argumentation (e. g., (Pollock 1987), (Amgoud & Cayrol 2002)). They introduce multi-valuedness with simple order or a preference relation over propositions. There also are other types of logic programming different from ALP and EALP, and hence different argumentation models proposed (e. g., abductive or defeasible logic programming (Garcia & Simari 2004)). We have to omit discussions on the comparison due to the limited space.

In the future, we are going to further develop LMA in the following directions: introduction of an attack relation among argument groups, WFSX$_P$-like(Alferes & Pereira 1996) semantics for EALP, an efficient method for computing justified arguments, for example, in case of the infinite set of truth-values on $\Re[0, 1]$, introduction of dialectics to LMA, and use and application of LMA as an apparatus of cooperation, compromise, consensus-attainment under multiple-valuedness and the epistemic explicit negation of inclusive nature.

Acknowledgments

The authors would like to thank the anonymous reviewers who contributed to increase the quality of this paper.

References

Alferes, J. J., and Pereira, L. M. 1996. *Reasoning with Logic Programming*, volume 1111. Springer-Verlag.

Amgoud, L., and Cayrol, C. 2002. A reasoning model based on the production of acceptable arguments. *Annals of Mathematics and Artificial Intelligence* 34(1–3):197–215.

Chesnevar, C. I.; Maguitman, G.; and Loui, R. P. 2000. Logical models of argument. *ACM Computing Surveys* 32:337–383.

Dung, P. M. 1993. An argumentation semantics for logic programming with explicit negation. In *Proc. of 10th Int. Conference on Logic Programming*, 616–630.

Garcia, A. J., and Simari, G. R. 2004. Defeasible logic programming: An argumentative approach. *Theory and Practice of Logic Programming* 4(1–2):95–138.

Kifer, M., and Lozinskii, E. L. 1992. A logic for reasoning with inconsistency. *J. of Automated Reasoning* 9:179–215.

Kifer, M., and Subrahmanian, V. S. 1992. Theory of generalized annotated logic programming and its applications. *J. of Logic Programming* 12:335–397.

Mora, I.; Alferes, J. J.; and Schroeder, M. 1998. Argumentation and cooperation for distributed extended logic programs. In *Working notes of the Workshop on Non-monotonic Reasoning*.

Pollock, J. 1987. Defeasible reasoning. *Cognitive Science* 11(4):481–518.

Prakken, H., and Sartor, G. 1997. Argument-based extended logic programming with defeasible priorities. *J. of Applied Non-Classical Logics* 7(1):25–75.

Schweimeier, R., and Schroeder, M. 2002. Well-founded argumentation semantic for extended logic programming. In *Proc. of Int. Workshop on Non-monotonic Reasoning*.

T. Takahashi, Y. U., and Sawamura, H. 2003. Formal argumentation frameworks for the extended generalized annotated logic programs. In *Proc. of KES'2003, LNAI*, volume 2773, 95–138. IEEE Press.

Takahashi, T., and Sawamura, H. 2004. A logic of multiple-valued argumentation. Technical report, Niigata University, http://www.cs.ie.niigata-u.ac.jp/˜takehisa/.