Semantics of Argumentation under Incomplete Information

Ken Satoh [1]*Kazuko Takahashi [2]

[1] National Institute of Informatics / Sokendai
[2] School of Science&Technology, Kwansei Gakuin University

Abstract: We discuss a semantics of argumentation under incomplete information. In this paper, we mean by "incomplete information" that an agent does not know the other agent's knowledge and therefore, the agent cannot predict which arguments are attacked and which counter-arguments are used in order to attack the arguments. In this paper, we provide a more general framework for such argumentation system than previous proposed framework and provide a computational method how to decide acceptability of argument by logic programming if both agents are eager to give all the arguments.

1 Introduction

Argumentation system is a hot topic in legal reasoning and in more general setting such as negotiation in multi-agent systems[Rahwan09]. However, most of the work on argumentation is based on the assumption where complete information about argumentation is provided[Dung95]. It would be appropriate for an application domain where we can see all the arguments and counter-arguments so that we can conclude the most appropriate result based on all the arguments. However, in reality, there would be another type of argumentation where relevant agents only have their own belief and they do not know other agents' belief and so they do not predict how other agents attack their own arguments.

Consider the following slightly modified example taken from[Okuno09][1].

$p0$**:** "You killed the victim."

$c1$**:** "I did not commit murder! There is no evidence!"

$p1$**:** "There is evidence. We found your license near the scene."

$c2$**:** "It's not evidence! I had my license stolen!"

$p2$**:** "It is you who killed the victim. Only you were near the scene at the time of the murder."

$c3$**:** "I didn't go there. I was at facility A at that time."

$p3$**:** "At facility A? Then, it's impossible to have had your license stolen since facility A does not allow a person to enter without a license."

In the above example, $c2$ is not firstly attacked but after the argument of $c3$ is given, $c2$ is attacked by $p3$. Since agent p does not believe that the suspect was at facility A, p could not use the counter-argument $p3$ at first. But after $c3$ is provided, p can attack $c2$ by pointing out the contradiction with $c3$. This phenomena cannot be formalized in argumentation system based on complete information

*Correspondence: National Institute of Informatics/Sokendai
2-1-2 Chiyoda-ku Hitotsubashi, 101-8430, Tokyo
E-mail: ksatoh@nii.ac.jp

[1] In the example, c and p are two parties and numbers attached with c and p express order of arguments.

about arguments and so we need a new framework.

Pioneer works on this direction would be, as far as we know, [Okuno09, Okuno10, Takahashi11] where counter-arguments, which cannot be used at the starting point of argumentation since these counter-arguments are not convinced by the agent itself, are triggered by other agents' arguments. In this paper, we extend this direction to provide more general framework than [Okuno09, Okuno10, Takahashi11]. The difference between their works and this work are as follows:

- We let an agent to give as many counter-arguments against other agent's arguments as they like where as [Okuno09, Okuno10, Takahashi11] allow only one counter-argument against one argument at one turn.
- We do not employ any specific strategy how to make counter-argument whereas [Okuno09, Okuno10, Takahashi11] impose an agent to stick to one line of arguments until no counter-argument is made, then the agent change counter-argument in the other line of arguments.

To formalize the above, we introduce *sources of arguments* which represent usable arguments. This means that even if there are potential counter-arguments against the other agent's arguments, the agent cannot use the argument if the argument in the source. We also introduce *derivation rule of sources* which represent augmentation of arguments which were not initially able to be used, but later become usable based on the other agent's new arguments and its own belief. By these mechanisms, we let agents not know whether potential arguments would be usable in the future since there are incomplete information about the other agents' behavior.

Then, we show a computational method to decide which arguments are accepted by translating argumentation framework into logic programming from the bird's eye view under the assumption that all the possible arguments will eventually be done by both parties.

2 Framework for Argumentation under Incomplete Information

Definition 1 *Let Arg_P(Arg_C, respectively) be a set called* an argument set *for P(C, respectively). We write $Arg_P \cup Arg_C$ as Arg.*
An attack relation for P(C, respectively) *$Attack_P$ is a subset of $Arg_P \times Arg_C$($Arg_C \times Arg_P$, respectively). We write $Attack_P \cup Attack_C$ as $Attack$.*
We say P(C, respectively) attacks n' by n if $\langle n, n' \rangle \in Attack_P$($Attack_C$, respectively).
$Source_P$ ($Source_C$, respectively) is a subset of Arg_P (Arg_C, respectively). We write $Source_P \cup Source_C$ as $Source$.
A derivation rule for P(C, respectively) *$Derive_P$ ($Derive_C$, respectively) is a set of the following rule:*

$$n \Leftarrow n_1, ...n_m$$

where $n \in Arg_P$ (Arg_C, respectively) and $n_i \in Arg (1 \leq i \leq m)$. We say n is derived from $n_1, ..., n_m$. We write $Deriv_P \cup Derive_C$ as $Derive$

We call $\langle Arg, Attack, Source, Derive \rangle$ an argumentation framework.

We assume that there is no loop in $Attack_P \cup Attack_C$ to avoid infinite loop of arguments[2].

In the above definition, a derivation rule enables an agent to augment its own source of arguments by adding the conclusion of the derivation rule if condition part is satisfied.

We define an argumentation tree which gives a semantics of acceptance of arguments as follows.

Definition 2 *An* argumentation tree *$Tr = \langle N, E \rangle$ w.r.t. an argumentation framework $\langle Arg, Attack, Source, Derive \rangle$ is an in-tree[3] such that $N \subset Arg$ and $E \subset Attack$ and satisfies the following conditions:*

- *The root of Tr is $p \in Source_P$ called "conclusion".*

[2]We may formalize an argumentation with loop if we follow Dung's stable extension or preferred extension.

[3]An in-tree is an directed tree in which a single node is reachable from every other one (See Fig.1).

- *If* $\langle n, n' \rangle \in E$ *then either of the following holds.*
  - $n \in Source_P$ *and* $n' \in Source_C$ *and* $\langle n, n' \rangle \in Attack_P$.
  - $n \in Source_C$ *and* $n' \in Source_P$ *and* $\langle n, n' \rangle \in Attack_C$.

Let $Tr = \langle N, E \rangle$ *be an argumentation tree.* $n \in N$ *is accepted w.r.t.* Tr *if*

- *there is no edge from* n, *or*
- *there is no* n' *s.t.* $\langle n', n \rangle \in E$ *and* n' *is accepted w.r.t.* Tr.

Now, we can define a game called an *argumentation game* which gives a dialog between two parties. In argumentation game, agents can refer to source of arguments to produce counter-arguments.

Definition 3 *A* move *of an* argumentation game *w.r.t. argumentation tree* $Tr = \langle N, E \rangle$ *and a pair of source sets* $\langle S_P, S_C \rangle$ *is an expansion of* Tr, S_P *and* S_C *defined as follows.*

- P*'s move is a set* $Move_P \subset Attack_P$ *such that for every* n *such that* $\langle n, n' \rangle \in Move_P$, $n \notin N$ *and* $n \in Source_P$ *and* $n' \in N$. *Then, a new set of nodes in a new argumentation tree* N', *a new set of edges in a new argumentation tree* E' *and a new pair of source sets* $\langle S'_P, S'_C \rangle$ *becomes the following.*
  - $N' = N \cup \{n | \langle n, n' \rangle \in Move_P\}$
  - $E' = E \cup Move_P$
  - $S'_P = S_P$
  - $S'_C = S_C \cup \{n | (n \Leftarrow n_1, ..., n_m) \in Derive_C \text{ and } n_i \in N'(1 \leq i \leq m)\}$
- C*'s move is a set* $Move_C \subset Attack_C$ *such that for every* n *such that* $\langle n, n' \rangle \in Move_C$, $n \notin N$ *and* $n \in Source_C$ *and* $n' \in N$. *Then, a new set of nodes in a new argumentation tree* N', *a new set of edges in a new argumentation tree* E' *and a new pair of source sets* $\langle S'_P, S'_C \rangle$ *becomes the following.*
  - $N' = N \cup \{n | \langle n, n' \rangle \in Move_C\}$
  - $E' = E \cup Move_C$
  - $S'_P = S_P \cup \{n | (n \Leftarrow n_1, ..., n_m) \in Derive_P \text{ and } n_i \in N'(1 \leq i \leq m)\}$
  - $S'_C = S_C$

If both agents give $\emptyset$ *in consecutive two moves, then we say that the game is* finished *and we call a final tree after a game is finished* argumentation game tree. *Let* Tr *be an argumentation game tree* $\langle N, E \rangle$. *We say that a node* $n \in N$ *is accepted w.r.t. the argumentation game tree* Tr *if* n *is accepted w.r.t. argumentation tree* Tr.

Note that a move can be $\emptyset$[4]. and a conclusion is decided to be accepted or not using the tree at the final stage.

Example 1 *Consider the example discussed at Introduction. Then,*
$Attack_P = \{\langle p1, c1 \rangle, \langle p2, c1 \rangle, \langle p3, c2 \rangle\}$,
$Source_P = \{p0, p1, p2\}$,
$Derive_P = \emptyset$
$Attack_C = \{\langle c1, p0 \rangle, \langle c2, p1 \rangle, \langle c3, p2 \rangle\}$,
$Source_C = \{c1, c2, c3\}$,
$Derive_C = \{c3 \Leftarrow p3\}$
Note that since initial $Source_P$ *does not include* $p3$ *so we cannot use an attack to* $c2$ *by* $p3$.

1. *Let* $p0$ *be a conclusion. Then* $Tr = \langle \{p0\}, \emptyset \rangle$.
2. C*'s next move has two possibilities, that is, to give either* $\emptyset$ *or* $\{\langle c1, p0 \rangle\}$.
3. *Suppose that* C *gives* $\{\langle c1, p0 \rangle\}$ *Then,* $Tr = \langle \{p0, c1\}, \{\langle c1, p0 \rangle\} \rangle$.
4. P*'s next move has four possibilities, that is, to give either* $\emptyset$ *or* $\{\langle p1, c1 \rangle\}$ *or* $\{\langle p2, c1 \rangle\}$ *or* $\{\langle p1, c1 \rangle, \langle p2, c1 \rangle\}$.
5. *Suppose that* P *gives* $\{\langle p1, c1 \rangle, \langle p2, c1 \rangle\}$. *Then,* $Tr = \langle \{p0, c1, p1, p2\}, \{\langle c1, p0 \rangle, \langle p1, c1 \rangle, \langle p2, c1 \rangle\} \rangle$.
6. C*'s next move has four possibilities, that is, to give either* $\emptyset$ *or* $\{\langle c2, p1 \rangle\}$ *or* $\{\langle c3, p2 \rangle\}$ *or* $\{\langle c2, p1 \rangle, \langle c3, p2 \rangle\}$.

[4]This means that even if there are possible counter-arguments, an agent can be silent.

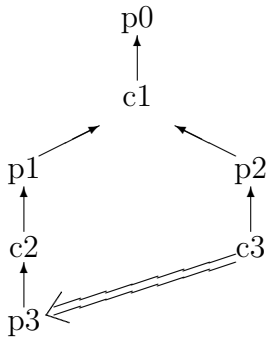

Figure 1: Representation of Arguments and Derive Relation for Example 1

7. *Suppose that C gives* $\{\langle c2, p1\rangle, \langle c3, p2\rangle\}$. *Then,*
$Tr = \langle\{p0, c1, p1, p2, c2, c3\},$
$\{\langle c1, p0\rangle, \langle p1, c1\rangle, \langle p2, c1\rangle, \langle c2, p1\rangle,$
$\langle c3, p2\rangle\}\rangle$.
Then, since $(c3 \Leftarrow p3) \in Derive_C$, $Source_P$ *becomes* $\{p0, p1, p2, p3\}$.
8. *P's next move has only two possibilities, that is, to give* $\{\langle p3, c2\rangle\}$ *or* $\emptyset$ *since* $p3$ *is now in* $Source_P = \{p0, p1, p2, p3\}$ *and* $\langle p3, c2\rangle$ *becomes usable.*
9. *Suppose that P gives* $\{\langle p3, c2\rangle\}$. *Then,*
$Tr = \langle\{p0, c1, p1, p2, c2, c3, p3\},$
$\{\langle c1, p0\rangle, \langle p1, c1\rangle, \langle p2, c1\rangle, \langle c2, p1\rangle,$
$\langle c3, p2\rangle, \langle p3, c2\rangle\}\rangle$.
10. *There is no move from both sides so we are done.*
11. *Then,* $p3$ *is accepted and so* $c2$ *is not accepted. Then* $p1$ *is accepted and* $c1$ *is not accepted. Finally* $p0$ *is accepted.*

In this example, $p3$ is a key to rebut $c2$ and $p3$ was not in initial source but is invoked after $c3$ is made. This invocation is made by a derivation rule $p3 \Leftarrow c3$ (See Fig.1).

There are many ways to develop an argumentation game tree, but we can show that it converges into one argumentation tree if both parties eventually give all possible arguments. We call this strategy *eager*, so we can say that an argumentation game tree will converge into one if both agents are eager. On the other hand, we can define a *lazy* agent which gives only necessary counter-arguments. In other words, a lazy agent will choose one counter-argument among possible counter-argument against the other agent's argument and it will choose another counter-argument only if the chosen counter-argument is rebutted by the other agent's counter-counter-argument. Then, in this lazy agent's case some of derivation rules might not be invoked so that an effective counter-argument might not be produced. We show such an example as follows.

Example 2 *Consider the following case where we add* $\langle c4, p2\rangle$ *to* $Attack_C$ *of the previous example. Then,*
$Attack_P = \{\langle p1, c1\rangle, \langle p2, c1\rangle, \langle p3, c2\rangle\},$
$Source_P = \{p0, p1, p2\},$
$Derive_P = \emptyset$
$Attack_C = \{\langle c1, p0\rangle, \langle c2, p1\rangle, \langle c3, p2\rangle,$
$\langle c4, p2\rangle\},$
$Source_C = \{c1, c2, c3, c4\},$
$Derive_C = \{c3 \Leftarrow p3\}$
We show an example when an agent does not give full arguments but hides an argument.

1. *Let* $p0$ *be a conclusion. Then* $Tr = \langle\{p0\}, \emptyset\rangle$.
2. *C's next move has two possibilities, that is, to give either* $\emptyset$ *or* $\{\langle c1, p0\rangle\}$.
3. *Suppose that C gives* $\{\langle c1, p0\rangle\}$ *Then,* $Tr = \langle\{p0, c1\}, \{\langle c1, p0\rangle\}\rangle$.
4. *P's next move has four possibilities, that is, to give either* $\emptyset$ *or* $\{\langle p1, c1\rangle\}$ *or* $\{\langle p2, c1\rangle\}$ *or* $\{\langle p1, c1\rangle, \langle p2, c1\rangle\}$.
5. *Suppose that P gives* $\{\langle p1, c1\rangle, \langle p2, c1\rangle\}$. *Then,*
$Tr = \langle\{p0, c1, p1, p2\},$
$\{\langle c1, p0\rangle, \langle p1, c1\rangle, \langle p2, c1\rangle\}\rangle$.
6. *C's next move has eight possibilities, that is, to give a subset of* $\{\langle c2, p1\rangle, \langle c3, p2\rangle, \langle c4, p2\rangle\}$.
7. *Suppose that C gives* $\{\langle c2, p1\rangle, \langle c4, p2\rangle\}$. *Then,*
$Tr = \langle\{p0, c1, p1, p2, c2, c4\},$
$\{\langle c1, p0\rangle, \langle p1, c1\rangle, \langle p2, c1\rangle, \langle c2, p1\rangle,$
$\langle c4, p2\rangle\}\rangle$.
Note that since C did not choose $\langle c3, p2\rangle$, *we cannot make* $p3$ *usable.*
8. *Only P's next move is to give* $\emptyset$.

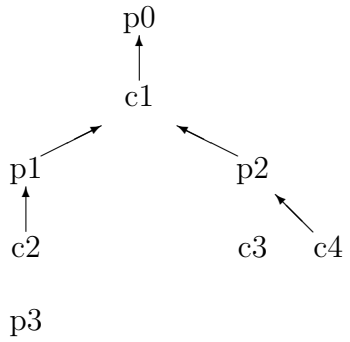

Figure 2: Representation of Arguments for Example 2

9. *C hides the another counter-argument* $\{c3, p2\}$ *and gives* $\emptyset$*. Then, there is no move from both sides so we are done.*

10. *Then,* $c2$ *and* $c4$ *are accepted. Then either* $p1$ *nor* $p2$ *is accepted and* $c1$ *is accepted. Finally* $p0$ *is not accepted.*

In this example, an agent c does not use $c3$ to rebut $p2$ but use $c4$ therefore, $p3$ is not invoked and $p0$ is not accepted. An agent c does not need to make more argument using $c3$ since the current counter-arguments are enough to win the example (See Fig. 2).

3 Computing Acceptance in Argumentation Framework

In this section, we assume that agents are both eager. Then we can translate an argumentation framework into a logic program in order to compute acceptability of a given argument from the bird's eye view. There is a proposal of computing Dung's argumentation semantics by translating the Dung's framework into a logic program and corresponding answer set of the program with acceptability[Osori05]. We extend their work by adding an extra condition reasoning about "sources". In order to do so, we introduce new predicate "announced(A)" meaning that an argument A is actually used for building an argumentation game tree. If an argument can be attacked by satisfying the condition that there is an attack relation for the argument and counter-argument is in the source, then counter-argument becomes *announced* to the other agent so that the agent can use other sources of arguments.

Definition 4 *Let* $\langle Arg, Attack, Source, Derive\rangle$ *be an argumentation framework. For* $A \in Arg$*, we define* $Counter_A = \{B | \langle B, A\rangle \in Attacks\}$*. For each argument A, we define the translation of argument A to rules of logic programming as follows:*

$$accepted(A) \leftarrow \bigwedge_{B \in Counter_A} not\ (source(B) \wedge accepted(B)).$$

Note that if $Counter_A$ *is empty then the above rule becomes* $accepted(A)$*.*
For every $B \in Counter_A$[5]*,*

$$announced(B) \leftarrow announced(A) \wedge source(B).$$

We also add the following rules for $(A \Leftarrow A_1, ..., A_m) \in Derive_C$*:*

$$source(A) \leftarrow \bigwedge_{i=1}^{m} body_C(A_i).$$

where $body(A_i)$ *is defined as follows:*

$$body_C(A_i) = \begin{cases} source(A_i) & \text{if } A_i \in Arg_C \\ announced(A_i) & \text{if } A_i \in Arg_P \end{cases}$$

Similarly, we add the following rules for $(A \leftarrow A_1, ..., A_m) \in Derive_P$*:*

$$source(A) \leftarrow \bigwedge_{i=1}^{m} body_P(A_i).$$

where $body_P(A_i)$ *is defined as follows:*

$$body_P(A_i) = \begin{cases} source(A_i) & \text{if } A_i \in Arg_P \\ announced(A_i) & \text{if } A_i \in Arg_C \end{cases}$$

We also add the following for an argument A in the initials source sets:

$$source(A).$$

[5]If the parent node is announced and the current node is in the source, then the current node will be announced. This rule expresses the eager strategy of argumentation.

We also add the following for the conclusion A_0 which is the root of the argumentation game tree:

$$announced(A_0).$$

Example 3 *Consider the setting of Example 1. The translated logic program becomes as follows:*

$accepted(c1) \leftarrow$
$\quad not\ (source(p1) \wedge accepted(p1)) \wedge$
$\quad not\ (source(p2) \wedge accepted(p2)).$
$accepted(c2) \leftarrow$
$\quad not\ (source(p3) \wedge accepted(p3)).$
$accepted(p0) \leftarrow$
$\quad not\ (source(c1) \wedge accepted(c1)).$
$accepted(p1) \leftarrow$
$\quad not\ (source(c2) \wedge accepted(c2)).$
$accepted(p2) \leftarrow$
$\quad not\ (source(c3) \wedge accepted(c3)).$
$accepted(c3).$
$accepted(p3).$
$announced(p1)$
$\quad \leftarrow announced(c1) \wedge source(p1).$
$announced(p2)$
$\quad \leftarrow announced(c1) \wedge source(p2).$
$announced(p3)$
$\quad \leftarrow announced(c2) \wedge source(p3).$
$announced(c1)$
$\quad \leftarrow announced(p0) \wedge source(c1).$
$announced(c2)$
$\quad \leftarrow announced(p1) \wedge source(c2).$
$announced(c3)$
$\quad \leftarrow announced(p2) \wedge source(c3).$
$source(p3) \leftarrow announced(c3).$
$source(p0).\quad source(p1).\quad source(p2).$
$source(c1).\quad source(c2).\quad source(c3).$
$announced(p0).$

Then, we can show that $accepted(p0)$ is derived from the above program.

Theorem 1 *Let $\langle Arg, Attack, Source, Derive\rangle$ be an argumentation framework and A_0 be a conclusion and Tr be a final argumentation game tree w.r.t. the framework for the eager strategy and Pr be a translated logic program from the framework. Then, A_0 is accepted if and only if $Pr \models accepted(A_0)$*

4 Conclusion

The contributions of the paper are as follows.

- We give more general framework of arguments under incomplete information.
- We give a computational method of how to decide the acceptability of the arguments using a translation from an argumentation framework to a logic program under the assumption that every possible arguments are made.

As a future research, we should give a computational method of acceptability for a lazy agent. The method must reflect multiple extensions of arguments related with choices of arguments.

References

[Dung95] Dung, P. M., "On the Acceptability of Arguments and its Fundamental Role in Nonmonotonic Reasoning, Logic Programming and N-Person Games", Artificial Intelligence, Vol. 77, pp.321 – 357 (1995).

[Osori05] Osori, M., Zepeda, C, Nieves, J. C., Corte's, U., "Inferring Acceptable Arguments with Answer Set Programming", `http://www.lsi.upc.edu/~jcnieves/JCNieves-Publications/Conference/ENC05.pdf`, Proc. of ENC'05, pp.198 – 205 (2005).

[Okuno09] Okuno, K., Takahashi, K., "Argumentation System with Changes of an Agent's Knowledge Base", Proc. of IJCAI 2009, pp.226–232 (2009).

[Okuno10] Okuno, K., Takahashi, K., "Argumentation System Allowing Suspend/Resume of an Argumentation Line", Proc. of ArgMAS2010, pp.145–162 (2010).

[Rahwan09] I.Rahwan, and G.Simari (eds.), "Argumentation in Artificial Intelligence", Springer (2009).

[Takahashi11] Takahashi, K., Nambu, Y., "A Semantics for Dynamic Argumentation Frameworks", Proc. of ArgMAS2011 (2011)