Inductive Negotiation in Answer Set Programming

Chiaki Sakama

Department of Computer and Communication Sciences
Wakayama University, Sakaedani, Wakayama 640-8510, Japan
sakama@sys.wakayama-u.ac.jp

Abstract. This paper provides a logical framework of negotiating agents who have capabilities of evaluating and building proposals. Given a proposal, an agent decides whether it is acceptable or not. If the proposal is unacceptable as it is, the agent seeks conditions to accept it. This attitude is captured as a process of making hypotheses by *induction*. If an agent fails to find a hypothesis, it would concede by giving up some of its current belief. This attitude is characterized using *default reasoning*. We provide a logical framework of such think-act cycle of an agent, and develop a method for computing proposals using *answer set programming*.

1 Introduction

Negotiation is a process of reaching agreement between different agents. In a typical one-to-one negotiation, an agent makes a proposal on his/her request and the opponent agent decides whether it is acceptable or not. If it is unacceptable, the opponent tries to make a counter-proposal. Negotiation proceeds in a series of rounds and each agent makes a proposal at every round until it reaches a (dis)agreement. Our primary interest of this paper is a process of evaluation and construction of proposals. A proposal is acceptable if it does not conflict with the interest of an agent. When a proposal is unacceptable for an agent, he/she seeks conditions to accept it. Those conditions would be found by updating his/her current beliefs: in one way, by introducing new beliefs, and in another way, by giving up some of his/her current belief.

Consider the following dialogue between a buyer B and a seller S (subscripts represent rounds in negotiation).

B_1: "I want an external HDD with 200GB".
S_1: "It costs 120USD".
B_2: "I want to get it at 100USD."
S_2: "We can provide it at the discount price if you pay by cash."
B_3: "I don't want to pay by cash".
S_3: "We can provide an external HDD with 180GB at 100USD".
B_4: "OK, I accept it".

In this dialogue, the buyer does not accept the initial offer S_1 made by the seller. Then, the buyer made a new proposal B_2 for a discount price. In response to this, the seller provides a condition to meet the request (S_2). The buyer does not accept it (B_3), and the seller proposes downgrade of the product (S_3). The buyer accepts it, and negotiation ends.

In the second round, the seller seeks conditions to accept B_2. The process of finding a condition to accept a proposal is logically characterized as follows. Suppose a knowledge base K represented by a first-order theory, and a proposal G represented by a formula. Then, K could accept G under the condition H if the next relation holds:

$$K \cup H \models G\,.$$

Here, H is a set of formulas and bridges the gap between the current belief K of an agent and the request G made by another agent. At this point, there are structural similarities between the problem presented above and the problem of *induction*, a method of machine learning in artificial intelligence. In fact, viewing G as an observed evidence, the problem of finding H is considered a process of building a *hypothesis* to explain G under K. Induction is an ampliative reasoning and extends the original theory to explain observed new phenomena. In negotiation, an agent also extends his/her original belief to accommodate another agent's request. Back to the negotiation dialogue, in response to the proposal S_3, the buyer concedes to accept it. This is done by withdrawing her original request. The process of concession is also formulated as follows. Given a knowledge base K of an agent and a proposal G by another agent, K could conditionally accept G by concession if the next relation holds:

$$(K \setminus J) \cup H \models G\,.$$

Here, J is a part of belief included in K, which could be given up to accept G.

In this paper, we provide a logical framework of negotiating agents who have capabilities of evaluating and building proposals. We first consider an agent who has a knowledge base represented by first-order logic and characterize a process of making proposals using induction. We show that different types of proposals are built in terms of induction. Next, we formulate a process of making a concession in negotiation. We show that concession is done by inference from a default theory. Finally, the proposed method is realized using *answer set programming* [8], a logic programming framework for nonmonotonic reasoning. The rest of this paper is organized as follows. Section 2 characterizes processes of building proposals and making a concession in terms of induction and default inference. Section 3 provides methods for computing proposals in answer set programming. Section 4 discusses related work, and Section 5 concludes the paper. This is an extended version of the paper [17]. Section 3 and a part of Section 4 are entirely new, and all proofs of technical results, which are not in [17], are attached.

2 Negotiation by Induction

2.1 Induction

A *first-order theory* is a set of formulas defined over the first-order language. The definition of the first-order language is the standard one in the literature. A first-order theory T *entails* a formula F (written as $T \models F$) if F is *true* in every model of T. A first-order theory T is *consistent* if it has a model; otherwise, T is *inconsistent*.

Induction in first-order logic is defined as follows. Given a *background knowledge base* K as a consistent first-order theory and a formula G as an *observation*, *induction* produces a set H of formulas as a *hypothesis* satisfying the condition:

$$K \cup H \models G \tag{1}$$

where $K \cup H$ is consistent. When H satisfies the above condition, we say that a hypothesis H *covers* (or *explains*) G with respect to K. This type of induction is used in the context of *inductive logic programming* [11].

Example 2.1. Suppose the knowledge base K and the observation G:[1]

$$K: \ swan(a) \wedge swan(b),$$
$$G: \ white(a) \wedge white(b).$$

Then,

$$H: \ \forall x \ (swan(x) \rightarrow white(x))$$

covers G with respect to K.

2.2 Building Proposal

We consider an agent who has a knowledge base K represented by a consistent first-order theory.

Definition 2.1. (proposal) A *proposal* G is a formula. In particular, G is called a *critique* if $G = accept$ or $G = reject$ where $accept$ and $reject$ are the reserved propositions.

A critique is a response as to whether or not a given proposal is accepted. It is decided by evaluating a proposal in a knowledge base of an agent.

Definition 2.2. (acceptability) Given a knowledge base K and a proposal G,

- G is *accepted* in K if $K \models G$.
- G is *acceptable* in K if $K \cup \{G\}$ is consistent.
- G is *unacceptable* (or *rejected*) in K if $K \cup \{G\}$ is inconsistent.

[1] Throughout the paper, we shall omit braces { } in examples to represent the sets K and H of formulas, but the meaning is clear from the context.

If a proposal G made by an agent Ag_1 is accepted/rejected by another agent Ag_2, Ag_2 returns the critique *accept*/*reject* to Ag_1. On the other hand, if a proposal is acceptable, an agent seeks conditions to accept it.

Definition 2.3. (conditional acceptance) Given a knowledge base K and a proposal G, G is *conditionally accepted* (with H) in K if

$$K \cup H \models G \tag{2}$$

holds for a set H of formulas such that $K \cup H$ is consistent. A set H of formulas is called an *accepting set of conditions* (with respect to K and G). In particular, H is called a *minimal* accepting set of conditions if H is a minimal set (under set inclusion) satisfying (2).

By the definition, it is easily seen that G is conditionally accepted in K if and only if it is acceptable in K. The notion of acceptance in Definition 2.2 is a special case of conditional acceptance with $H = \emptyset$. By Definition 2.3, we can see that the problem of finding a condition H for accepting a proposal G is identical to the problem of finding inductive hypothesis in (1). That is, by viewing a proposal G as an observation, an accepting set H of conditions is considered a hypothesis which covers G with respect to a background knowledge base K. This correspondence is not only in the definition of formulas, but also in the ground of their usage. In induction, when an agent observes a new evidence that cannot be explained in its current knowledge base, the agent induces a hypothesis which well accounts for the evidence and updates the knowledge base if necessary. In negotiation, on the other hand, an agent also observes a new proposal that is not entailed by its current knowledge base. Then, the agent constructs a hypothesis which well accounts for the proposal. Among accepting sets of conditions, we are interested in minimal accepting sets of conditions which represent minimal requirements for accepting a proposal. For this reason, we hereafter consider minimal accepting sets of conditions unless stated otherwise.

There are different types of accepting sets of conditions satisfying the relation (2). We provide some typical types of proposals in negotiation based on this definition. Suppose that an agent Ag_1 makes a proposal G and another agent Ag_2 who has a knowledge base K builds a counter-proposal in response to G.

Consent : When $H = \{G\}$, it holds that $K \cup H \models G$. In this case, Ag_2 accepts a proposal G if it is acceptable. Then, Ag_2 returns the critique $G' = accept$ to Ag_1.

Constraint : When $H = \{ G \wedge C \}$, it holds that $K \cup H \models G$. In this case, Ag_2 accepts a proposal G with a constraint C. Then, Ag_2 returns the counter-proposal $G \wedge C$ to Ag_1. For example, given $G = go(restaurant)$, $C = on(Saturday)$ represents a constraint for accepting G.

Generalization : When $H = \{ G' \}$ such that $G'\theta = G$ for some substitution θ, it holds that $K \cup H \models G$. In this case, Ag_2 returns the counter-proposal G' which is more general than G. For example, given $G = show\_product(TV, b)$ with some specific brand-name b, $G' = show\_product(TV, x)$ with a variable x represents TV of any brand.

Subsumption : When H is a concept which subsumes G and K contains subsumption knowledge between H and G, it holds that $K \cup H \models G$. In this case, Ag_2 returns a counter-proposal H to Ag_1. For example, let $G = go(bookstore)$ and K contains $go(shopping\text{-}mall) \rightarrow go(bookstore)$, then $go(shopping\text{-}mall)$ becomes a counter-proposal.

Implication : When $H = \{ F \rightarrow G \}$ and $K \cup H \models G$, F represents a condition to accept G. In this case, Ag_2 returns the counter-proposal F to Ag_1. For example, let $G = want(chocolate)$ and K contains $want(biscuit)$, then $H = \{ want(biscuit) \rightarrow want(chocolate) \}$ represents exchange of sweets and $want(biscuit)$ becomes a counter-proposal.

In the above, *Consent* characterizes very generous attitude of an agent. *Constraint* and *Generalization* are considered special cases of *Implication* as both $G \wedge C \rightarrow G$ and $G' \rightarrow G'\theta$ hold. *Subsumption* is also a special case of *Implication* such that K contains a dependence relation between F and G. In case of subsumption, *abduction* [7] is used for the purpose instead of induction. Abduction is also hypothetical reasoning satisfying the relation (1). In contrast to induction which constructs a rule $F \rightarrow G$ from K and G, abduction extracts a fact F from G and a rule $F \rightarrow G$ which is derived from K.

2.3 Concession

An agent rejects a proposal if it is unacceptable. On the other hand, an agent can take an action of *concession* if he/she wants to reach an agreement in negotiation. To characterize agents who may concede in negotiation, we suppose agents who have two different types of knowledge: the one is strong belief and the other is weak belief. Strong belief is persistent belief or strong desire that cannot be abandoned. By contrast, weak belief can be given up depending on situation. Formally, a first-order theory K is divided into two disjoint sets:

$$K = \Sigma \cup \Gamma$$

where Σ represents *strong belief* and Γ represents *weak belief*. We assume that an agent gives up weak belief but not strong one when he/she makes a concession.

Definition 2.4. (acceptable by concession) Let K be a knowledge base such that $K = \Sigma \cup \Gamma$ as above. Then, a proposal G is *acceptable by concession* in K if there is a set J of formulas such that $J \subseteq \Gamma$ and $(K \setminus J) \cup \{G\}$ is consistent.

Definition 2.5. (conditional acceptance by concession) Let K be a knowledge base such that $K = \Sigma \cup \Gamma$. Then, a proposal G is *conditionally accepted by concession* (with H) in K if

$$(K \setminus J) \cup H \models G \tag{3}$$

holds for some sets H and J of formulas such that $J \subseteq \Gamma$ and $(K \setminus J) \cup H$ is consistent. A set J of formulas is called an *accepting set of concessions* (with respect to K and G). In particular, J is called a *minimal* accepting set of concessions if J is a minimal set (under set inclusion) satisfying (3).

Proposition 2.1 *A proposal G is conditionally accepted by concession in K iff G is acceptable by concession in K.*

Proof. If G is acceptable by concession in K, $(K \setminus J) \cup \{G\}$ is consistent for some $J \subseteq \Gamma$. As $(K \setminus J) \cup \{G\} \models G$, G is conditionally accepted by concession in K. Conversely, if G is conditionally accepted by concession in K, $(K \setminus J) \cup H \models G$ holds for some $J \subseteq \Gamma$ and H such that $(K \setminus J) \cup H$ is consistent. Then, any model M of $(K \setminus J) \cup H$ satisfies G, so M becomes a model of $(K \setminus J) \cup G$. Hence, $(K \setminus J) \cup G$ is consistent and G is acceptable by concession. □

Comparing Definition 2.5 with Definition 2.3, concession may give up (a part of) the current belief of an agent for accepting proposals. In particular, the relation (3) reduces to (2) when $J = \emptyset$. We assume that an agent wants to give up his/her current belief as little as possible, so we hereafter consider minimal accepting sets of concessions as well as minimal accepting sets of conditions.

Example 2.2. ([12]) Suppose that an agent Ag_1 has the knowledge base K:

$$
\begin{aligned}
f_1 &: \mathit{have}(\mathit{mirror}) \wedge \mathit{have}(\mathit{nail}) \rightarrow \mathit{hang}(\mathit{mirror}),\\
f_2 &: \mathit{have}(\mathit{mirror}) \wedge \mathit{have}(\mathit{screw}) \rightarrow \mathit{hang}(\mathit{mirror}),\\
f_3 &: \mathit{give}(\mathit{nail}) \rightarrow \neg \mathit{have}(\mathit{nail}),\\
f_4 &: \mathit{have}(\mathit{screw}) \rightarrow \mathit{give}(\mathit{nail}),\\
f_5 &: \forall x\ \mathit{get}(x) \rightarrow \mathit{have}(x),\\
f_6 &: \mathit{have}(\mathit{mirror}),\\
f_7 &: \mathit{have}(\mathit{nail}),
\end{aligned}
$$

where the strong belief Σ consists of f_1–f_6 and the weak belief Γ consists of f_7. The meaning of each formula is: f_1 and f_2 represent conditions to hang a mirror. If Ag_1 gives a nail, he/she does no longer have the nail (f_3). If Ag_1 has a screw, he/she can give a nail (f_4). If one gets an object, one has the object (f_5). Ag_1 has both a mirror (f_6) and a nail (f_7). Suppose that Ag_1 has the intention of hanging a mirror. Consider that another agent Ag_2 makes the request

$$G : \mathit{give}(\mathit{nail}).$$

This proposal is unacceptable in K because $K \cup \{G\}$ is inconsistent. The agent Ag_1 may reject G with this reason, but he/she could look for conditions for concession. Ag_1 finds the solution

$$J : \mathit{have}(\mathit{nail})$$

and

$$H : \mathit{get}(\mathit{screw})$$

where $(K \setminus J) \cup H$ is consistent and satisfies the relation $(K \setminus J) \cup H \models G$. Then, Ag_1 offers a counter-proposal H to Ag_2.

Our next question is how to distinguish different types of belief in both syntactic and semantic ways. For this purpose, we use *default logic* [13] for representing a knowledge base. Default logic distinguishes two types of knowledge as first-order formulas and default rules. Formally, a *default theory* is defined as a pair $\Delta = (D, W)$ where D is a set of default rules and W is a set of first-order formulas. A default rule (or simply *default*) is of the form:

$$\frac{\alpha : \beta_1, \ldots, \beta_n}{\gamma}$$

where $\alpha, \beta_1, \ldots, \beta_n$ and γ are quantifier-free formulas and respectively called the *prerequisite*, the *justifications* and the *consequent*. A default is *ground* if it contains no variable. Any default with variables represents the set of its ground instances over the language of Δ. As defaults and first-order formulas are syntactically distinguishable, we often put a default theory $\Delta = W \cup D$ for convenience. A set S of formulas is *deductively closed* if $S = Th(S)$ where Th is the deductive closure operator as usual. A set E of formulas is an *extension* of (D, W) if it coincides with a minimal deductively closed set E' of formulas satisfying the conditions: (i) $W \subseteq E'$, and (ii) for any ground default $\alpha : \beta_1, \ldots, \beta_n / \gamma$ from D, $\alpha \in E'$ and $\neg\beta_i \notin E$ $(i = 1, \ldots, n)$ imply $\gamma \in E'$. An extension E is *consistent* if E is a consistent set of formulas. A default theory may have none, one or multiple extensions in general.

To represent weak belief of an agent, we use default rules of the form:

$$\frac{: \gamma}{\gamma}. \tag{4}$$

This type of rule is called *super-normal* and a *super-normal default theory* is a default theory in which every default has the form (4). The rule (4) is read as "if it is consistent to assume γ, then believe γ". We represent weak belief of an agent by super-normal defaults in D, and distinguish them from strong belief represented by first-order formulas in W.

Definition 2.6. (default representation) Let K be a first-order theory such that $K = \Sigma \cup \Gamma$. Then, a *default representation* of K is defined as a super-normal default theory $\Delta_K = (D, W)$ such that $D = \{ \frac{:\gamma}{\gamma} \mid \gamma \in \Gamma \}$ and $W = \Sigma$.

Concession is characterized in a default theory as follows.

Theorem 2.2. *Let K be a first-order theory such that $K = \Sigma \cup \Gamma$.*

(i) A proposal G is acceptable by concession in K iff $\Delta_K \cup \{G\}$ has a consistent extension.

(ii) A proposal G is conditionally accepted by concession with H in K iff $\Delta_K \cup H$ has a consistent extension E such that $G \in E$.

Proof. (i) Let $\Delta_K = (D, W)$ be a default representation of K. When G is acceptable by concession in K, $(K \setminus J) \cup \{G\}$ is consistent for a minimal set $J \subseteq \Gamma$. As $(K \setminus J) \cup \{G\} = W \cup (\Gamma \setminus J) \cup \{G\}$, put $E = Th(W \cup (\Gamma \setminus J) \cup \{G\})$. If E

is not a consistent extension of $\Delta_K \cup \{G\}$, there is a minimal deductively closed consistent set $E' = Th(W \cup \Gamma' \cup \{G\})$ such that $\Gamma' \subseteq \Gamma$. In case of $E' \subset E$, $\Gamma' \subset (\Gamma \setminus J)$. Put $\Xi = (\Gamma \setminus J) \setminus \Gamma'$. Then, $E = Th(W \cup \Gamma' \cup \Xi \cup \{G\})$ becomes inconsistent. This contradicts the assumption that E is consistent. In case of $E \subset E'$, $(\Gamma \setminus J) \subset \Gamma'$. Put $\Gamma' = \Gamma \setminus J'$ for some $J' \subset J$. Then, $(K \setminus J') \cup \{G\}$ becomes consistent. This contradicts the assumption that J is a minimal set which makes $(K \setminus J) \cup \{G\}$ consistent. Hence, E becomes a consistent extension of $\Delta_K \cup \{G\}$. Conversely, if E is an extension of $\Delta_K \cup \{G\}$, E is a minimal deductively closed set $E = Th(W \cup \Gamma'' \cup \{G\})$ where $\Gamma'' = \{\, \gamma \mid \frac{:\gamma}{\gamma} \in D \text{ and } \neg\gamma \notin E \,\}$. Then, there is a (minimal) set $J \subseteq \Gamma$ such that $\Gamma'' = \Gamma \setminus J$. For this J, it holds that $(K \setminus J) \cup \{G\}$ is consistent. Hence, G is acceptable by concession.

(ii) If $\Delta_K \cup \{G\}$ has a consistent extension, by putting $H = \{G\}$, $\Delta_K \cup H$ has a consistent extension E such that $G \in E$. Conversely, if for a set H of formulas $\Delta_K \cup H$ has a consistent extension E such that $G \in E$, E is also a consistent extension of $\Delta_K \cup H \cup \{G\}$ ([13, Theorem 2.6]). In this case, $\Delta_K \cup \{G\}$ has a consistent extension. (If $\Delta_K \cup \{G\}$ is inconsistent, $\Delta_K \cup H \cup \{G\}$ is inconsistent [13].) Thus, $\Delta_K \cup \{G\}$ has a consistent extension iff for a set H of formulas, $\Delta_K \cup H$ has a consistent extension E such that $G \in E$. Then, the result holds by Proposition 2.1. □

Theorem 2.2 represents that conditional acceptance by concession is characterized in terms of default inference of G from $\Delta_K \cup H$.

Example 2.3. (cont. Example 2.2) The knowledge base is represented by the default theory $\Delta_K = (D, W)$ where

$$
\begin{aligned}
W:\ & have(mirror) \wedge have(nail) \rightarrow hang(mirror),\\
& have(mirror) \wedge have(screw) \rightarrow hang(mirror),\\
& give(nail) \rightarrow \neg have(nail),\\
& have(screw) \rightarrow give(nail),\\
& \forall x\ get(x) \rightarrow have(x),\\
& have(mirror),\\
D:\ & \frac{:\, have(nail)}{have(nail)}.
\end{aligned}
$$

Given the request $G = give(nail)$, G is included in a default extension of $\Delta_K \cup \{\, get(screw) \,\}$.

3 Computing Proposals in Answer Set Programming

3.1 Answer Set Programming

Answer set programming (ASP) [8] represents incomplete knowledge in a logic program and realizes nonmonotonic default reasoning. In ASP a logic program is

given by an *extended disjunctive program* (EDP). An EDP (or simply a *program*) is a set of rules of the form:

$$L_1 ; \cdots ; L_l \leftarrow L_{l+1}, \ldots, L_m, not\, L_{m+1}, \ldots, not\, L_n \tag{5}$$

($n \geq m \geq l \geq 0$) where each L_i is a positive/negative literal, i.e., A or $\neg A$ for an atom A, and *not* is *negation as failure* (NAF). $not\, L$ is called an *NAF-literal*. The above rule is read "some of $L_1, \ldots, L_l$ is believed if all $L_{l+1}, \ldots, L_m$ are believed and all $L_{m+1}, \ldots, L_n$ are disbelieved". The left-hand side of the arrow is the *head*, and the right-hand side is the *body*. For each rule r of the form (5), $head(r)$, $body^+(r)$ and $body^-(r)$ denote the sets of literals $\{L_1, \ldots, L_l\}$, $\{L_{l+1}, \ldots, L_m\}$, and $\{L_{m+1}, \ldots, L_n\}$, respectively. Also, $not\_body^-(r)$ denotes the set of NAF-literals $\{not\, L_{m+1}, \ldots, not\, L_n\}$. A rule r is often written as $head(r) \leftarrow body^+(r), not\_body^-(r)$ or $head(r) \leftarrow body(r)$ where $body(r) = body^+(r) \cup not\_body^-(r)$. A rule r is *disjunctive* if $head(r)$ contains more than one literal. A rule r is a *constraint* if $head(r) = \emptyset$; and r is a *fact* if $body(r) = \emptyset$. A program is *NAF-free* if no rule contains NAF-literals. A program, rule, or literal is *ground* if it contains no variable. A program P with variables is a shorthand of its *ground instantiation* $Ground(P)$, the set of ground rules obtained from P by substituting variables in P by elements of its Herbrand universe in every possible way.

The semantics of an EDP is defined by the *answer set semantics* [5]. Let Lit be the set of all ground literals in the language of a program. Suppose a program P and a set S of ground literals. Then, the *reduct* P^S is the program which contains the ground rule $head(r) \leftarrow body^+(r)$ iff there is a rule r in $Ground(P)$ such that $body^-(r) \cap S = \emptyset$. Given an NAF-free EDP P, let S be a set of ground literals which is (i) *closed* under P, i.e., for every ground rule r in $Ground(P)$, $body(r) \subseteq S$ implies $head(r) \cap S \neq \emptyset$; and (ii) *logically closed*, i.e., it is either consistent or equal to Lit. An *answer set* of an NAF-free program P is a minimal set S satisfying both (i) and (ii). Given an EDP P and a set S of ground literals, S is an *answer set* of P if S is an answer set of P^S. A program has none, one, or multiple answer sets in general. The set of all answer sets of P is written as $AS(P)$. An answer set is *consistent* if it is not Lit. A program P is *consistent* if it has a consistent answer set; otherwise, P is *inconsistent*.

A literal L is a consequence of *cautious inference* in a program P if L is included in every answer set of P. On the other hand, a literal L is a consequence of *brave inference* in a program P if L is included in some answer set of P.[2] We write $P \models_b L$ if a literal L is a consequence of brave inference in P.

Example 3.1. The program:

$$tea ; coffee \leftarrow,$$
$$milk \leftarrow tea, not\ lemon,$$
$$lemon \leftarrow tea, not\ milk,$$
$$milk \leftarrow coffee,$$

[2] Cautious/brave inference is also called *skeptical/credulous* inference.

has the three answer sets: $\{ tea, milk \}$, $\{ tea, lemon \}$, and $\{ coffee, milk \}$, which represent possible options for drink.

3.2 Computing Proposals

As presented in Definition 2.6, weak beliefs are represented by super-normal default rules. In an EDP, a super-normal default rule

$$\frac{: L}{L}$$

with a literal L is represented as a rule

$$L \leftarrow not\ \neg L\,.$$

An EDP can represent other forms of default knowledge in terms of rules with NAF. By contrast, persistent belief is represented by rules without NAF. Thus, both weak and strong beliefs are uniformly represented in an EDP.

Based on Theorem 2.2, we rephrase the notion of conditional acceptance by concession in the context of ASP as follows.

Definition 3.1. (conditional acceptance by concession) Let P be a program representing a knowledge base, and G a ground literal representing a proposal. Then, G is *acceptable by concession* in P iff $P \cup \{G\}$ is consistent. And G is *conditionally accepted by concession* in P iff $P \cup H \models_b G$ holds for some set H of rules such that $P \cup H$ is consistent.

Suppose a program P and a proposal G. When $P \not\models_b G$, consider the set $\mathbf{R}$ of rules such that

$$\begin{aligned} r \in \mathbf{R} \quad \text{iff} \quad & head(r) = \{G\}\,,\ body(r) \neq \emptyset, \\ & \emptyset \subseteq body^+(r) \subseteq S\,, \text{ and} \\ & body^-(r) \subseteq Lit \setminus S \end{aligned}$$

where $S \in AS(P)$ and there is no $S' \in AS(P)$ such that $(S' \setminus T) \cup (T \setminus S') \subset (S \setminus T) \cup (T \setminus S)$ for an answer set T of $P \cup \{G\}$.

By Definition 3.1, $P \cup \{G\}$ has a consistent answer set if G is acceptable by concession. The condition $(S' \setminus T) \cup (T \setminus S') \subset (S \setminus T) \cup (T \setminus S)$ for no $S' \in AS(P)$ presents that an answer set S of P which is closest to an answer set of $P \cup \{G\}$ is selected. The set $\mathbf{R}$ provides a hypothesis space for solutions.

Theorem 3.1. *Let P be a program and G a proposal such that $P \not\models_b G$. For any non-empty subset $H \subseteq \mathbf{R}$ such that $P \cup H$ is consistent, $P \cup H \models_b G$ holds.*

Proof. First, $(P \cup H)^S = P^S \cup H^S$ holds for any answer set S of P. For any non-empty subset $H \subseteq \mathbf{R}$, $body^-(r) \cap S = \emptyset$ for any $r \in H$. Then, there is a rule $G \leftarrow body^+(r)$ in H^S such that $\emptyset \subseteq body^+(r) \subseteq S$. Since $P \cup H$ is consistent, a consistent set $S \cup \{G\}$ becomes a minimal closed set of $P^S \cup H^S$. Then, $S \cup \{G\}$ becomes a consistent answer set of $P \cup H$. Hence, the result holds. □

Theorem 3.1 shows a method for computing a hypothesis H which is used for conditional acceptance by concession. Generally speaking, however, there are many candidates of hypotheses satisfying the condition. We remark some methods of eliminating hypotheses which are useless in the process of negotiation. First, we eliminate hypotheses by syntactic restriction. Suppose that two rules r_1 and r_2 satisfy the relation $P \cup \{r_1\} \models_b G$ and $P \cup \{r_2\} \models_b G$. In this case, if $body(r_1) \subset body(r_2)$ holds, r_2 is eliminated. This is because r_1 provides a condition simpler than r_2 for accepting G. A rule r in $\mathbf{R}$ is said *minimal* if there is no rule r' in $\mathbf{R}$ such that $\emptyset \subset body(r') \subset body(r)$. We can eliminate every non-minimal rule from $\mathbf{R}$. Moreover, for any $H_1 \subseteq \mathbf{R}$ and $H_2 \subseteq \mathbf{R}$, if both H_1 and H_2 satisfy the condition of Theorem 3.1 and the relation $H_1 \subseteq H_2$ holds, the minimal set H_1 of hypotheses is preferred to non-minimal H_2. Note that we are interested in any rule $r \in \mathbf{R}$ such that $body(r) \neq \emptyset$. If $body(r) = \emptyset$, r becomes a fact $G \leftarrow$ and $P \cup \{G \leftarrow\} \models_b G$ immediately holds. This corresponds to "Consent" in Section 2.2 and is a trivial solution. So in what follows we consider rules in $\mathbf{R}$ having non-empty bodies.

Second, we eliminate hypotheses by semantic restriction. Agents in negotiation are involved in matters of mutual interests. For example, in negotiation between a buyer and a seller, they are interested in buying or selling some designated product. In this case, hypotheses are related to conditions for buying-selling the objective product. They include price, service, quality of products, and so on. Some belief with respect to one's taste for foods is irrelevant to the buying-selling activities (except for the case of buying-selling foods). Let C be a set of literals and NAF-literals specifying specific *context* of negotiation. Then, any rule r in $\mathbf{R}$ is eliminated if $body(r) \not\subseteq C$. In other words, we are interested in getting any rule having conditions in the specific context. We assume that negotiating agents have their contexts in advance. This assumption appears natural because no agent negotiates without any interest. Different agents could have different contexts depending on their interests. These syntactic and semantic restrictions help to reduce candidate hypotheses in $\mathbf{R}$. Such restrictions on a hypothesis space are called *induction bias* [11]. After applying these restrictions to $\mathbf{R}$, there could still exist multiple choices of proposals. In this case, we assume that an agent nondeterministically selects a proposal from $\mathbf{R}$.

Example 3.2. Suppose a buying-selling situation in Section 1. A seller agent has the knowledge base P_s which consists of the following rules:

$$product(hdd, 200G, 120\$) \leftarrow not\ \neg\, product(hdd, 200G, 120\$), \tag{6}$$

$$product(hdd, 180G, 100\$) \leftarrow not\ \neg\, product(hdd, 180G, 100\$), \tag{7}$$

$$\leftarrow product(x, y, z),\ product(x, y, w),\ z \neq w, \tag{8}$$

$$\neg\, product(hdd, 200G, 120\$) \leftarrow product(hdd, 200G, x), x < 120\$,\ pay\_cash, \tag{9}$$

$$pay\_cash\,;\ pay\_card \leftarrow . \tag{10}$$

Here, the rules (6) and (7) represent products and their normal prices. They are represented by super-normal defaults because prices are subject to change. The rule (8) represents a constraint that the same product cannot have different

prices at the same time. The rule (9) represents if discount is made by payment with cash, the normal price is withdrawn. The rule (10) represents two options for payment. P_s has two answer sets:

$$S_1 : \{ product(hdd, 200G, 120\$),\ product(hdd, 180G, 100\$),\ pay\_cash \},$$
$$S_2 : \{ product(hdd, 200G, 120\$),\ product(hdd, 180G, 100\$),\ pay\_card \},$$

which represent the seller's initial belief.

A buyer agent has the knowledge base P_b which consists of rules:

$$product(hdd, 200G, 100\$) \leftarrow not\ \neg\ product(hdd, 200G, 100\$), \tag{11}$$
$$\leftarrow product(hdd, x, y),\ y > 100\$, \tag{12}$$
$$\leftarrow pay\_cash, \tag{13}$$
$$product(hdd, 180G, 100\$) \leftarrow not\ product(hdd, 200G, 100\$), \tag{14}$$
$$\neg\ product(hdd, 200G, 100\$) \leftarrow product(hdd, 180G, 100\$). \tag{15}$$

The rule (11) represents the intention of getting a HDD of 200GB with 100USD. It is represented by a super-normal default because the intention could be changed. The rule (12) represents a constraint that the budget is less than 100USD. The rule (13) represents a constraint that she does not pay by cash. The rule (14) represents that if a HDD with 200G is unavailable under the budget, the buyer would have an option of downgrading the specification. The rule (15) represents that the original request is withdrawn if the specification changes. P_b has two answer sets:

$$\{ product(hdd, 200G, 100\$) \},$$
$$\{ \neg\ product(hdd, 200G, 100\$),\ product(hdd, 180G, 100\$) \},$$

which represent the buyer's initial belief. With this setting, negotiation starts.

(1st round) First, the buyer asks the price of a HDD with 200GB. As $P_s \models_b product(hdd, 200G, 120\$)$, the seller replies the price:

$$G_s^1 :\ product(hdd, 200G, 120\$).$$

(2nd round) The buyer does not accept G_s^1 because it violates the constraint (12) and $P_b \cup \{G_s^1\}$ is inconsistent. As $P_b \models_b product(hdd, 200G, 100\$)$, the buyer returns the proposal

$$G_b^2 :\ product(hdd, 200G, 100\$)$$

to the seller. The seller evaluates G_b^2 in its knowledge base P_s and knows that $P_s \cup \{ product(hdd, 200G, 100\$) \}$ is consistent. Hence, G_b^2 is acceptable by concession in P_s. The seller then seeks a hypothesis H to accept G_b^2 and constructs the set $\mathbf{R}$ satisfying the condition:

$$\begin{aligned} r \in \mathbf{R} \quad \text{iff} \quad & head(r) = \{G_b^2\},\ body(r) \neq \emptyset, \\ & \emptyset \subseteq body^+(r) \subseteq S_1\ , \text{and} \\ & body^-(r) \subseteq Lit \setminus S_1 \end{aligned}$$

where S_1 is selected because S_1 is closer to the answer set $\{\, product(hdd, 200G, 100\$), \neg\, product(hdd, 200G, 120\$), product(hdd, 180G, 100\$),\ pay\_cash \,\}$ of $P_s \cup \{\, G_b^2 \,\}$ than S_2 is. The set $\mathbf{R}$ includes rules such that

$$
\begin{aligned}
r_1 :\ & product(hdd, 200G, 100\$) \leftarrow product(hdd, 200G, 120\$),\\
r_2 :\ & product(hdd, 200G, 100\$) \leftarrow product(hdd, 180G, 100\$),\\
r_3 :\ & product(hdd, 200G, 100\$) \leftarrow pay\_cash,\\
r_4 :\ & product(hdd, 200G, 100\$) \leftarrow not\ pay\_card,\\
r_5 :\ & product(hdd, 200G, 100\$) \leftarrow product(hdd, 200G, 120\$),\\
& \qquad\qquad\qquad\qquad\quad product(hdd, 180G, 100\$),\\
r_6 :\ & product(hdd, 200G, 100\$) \leftarrow product(hdd, 200G, 120\$),\ pay\_cash,\\
& \quad \cdots\cdots\cdots
\end{aligned}
$$

Among them, non-minimal rules, for instance, r_5 and r_6, are eliminated. The seller also considers the context C as

$$C : \{\, product(hdd, 200G, 120\$),\ pay\_cash,\ pay\_card \,\},$$

because the present deal between the buyer and the seller is on the product $product(hdd, 200G, 120\$)$ and its payment. Then, rules containing any literal $L \notin C$, for instance r_2, are eliminated. After the elimination, the seller selects rules such that $P_s \cup \{r_i\}$ is consistent. Since r_1 violates the constraint (8), two rules r_3 and r_4 remain as candidates. Then, the seller constructs possible hypotheses as $H_1 = \{r_3\}$ and $H_2 = \{r_4\}$. Both $P_s \cup H_1$ and $P_s \cup H_2$ have two answer sets:

$$
\begin{aligned}
&\{\, product(hdd, 200G, 100\$),\ \neg\, product(hdd, 200G, 120\$),\\
&\qquad\qquad\qquad\qquad\quad product(hdd, 180G, 100\$),\ pay\_cash \,\},\\
&\{\, product(hdd, 200G, 120\$),\ product(hdd, 180G, 100\$),\ pay\_card \,\},
\end{aligned}
$$

so that $P_s \cup H_i \models_b product(hdd, 200G, 100\$)$ for $i = 1, 2$. For the seller H_1 and H_2 have the same meaning, so he takes H_1 as a hypothesis and returns the condition

$$G_s^2 :\ pay\_cash$$

as a counter-proposal.

(3rd round) The buyer does not accept G_s^2 because it violates the constraint (13). The buyer then returns the critique

$$G_b^3 :\ reject$$

to the seller. As there is no way to meet the request of the buyer, the seller makes a new proposal

$$G_s^3 :\ product(hdd, 180G, 100\$)$$

which satisfies $P_s \models_b G_s^3$.

(4th round) As $P_b \models_b G_s^3$, the buyer accepts the proposal and an agreement is reached.

4 Related Work

Several studies use logic-based *abduction* or *abductive logic programming* [7] as a representation language of negotiating agents [9, 14, 16]. Kakas and Moraitis [9] propose a negotiation protocol which integrates abduction within an argumentation framework. In their negotiation protocol, an agent considers another agent's goal and searches for conditions under which the goal is acceptable. They use abduction to seek conditions to support arguments. In their protocol, counter-proposals are chosen among candidates based on preference knowledge of an agent. In [14] an abductive logic program is used for specifying dialogue primitives and negotiation protocol. Once a dialogue is uttered by an agent, another agent that observed the utterance thinks and acts according to a given *observe-think-act cycle*. There are two important differences between [9, 14] and our present work. First, those studies have no mechanism of constructing new counter-proposals in response to a proposal made by an agent. The behavior of an agent is completely pre-specified in either a knowledge base of an agent or a negotiation protocol, and possible responses are prepared in advance. In our framework, proposals are newly constructed using induction. It enables us to build proposals that are independent of particular negotiation protocols. Most theories of automated negotiation specify possible responses in advance, while [16] is an exception. Sakama and Inoue [16] propose methods for building new proposals by *extended abduction* and *relaxation*. Extended abduction is an extension of abduction proposed by Inoue and Sakama [6], which can not only introduce hypotheses to a knowledge base but remove hypotheses from it to explain an observation. Relaxation is a technique of weakening constraints in database queries. They use extended abduction to compute *conditional proposals* and use relaxation to compute *neighborhood proposals*. An essential difference from our present work is that they use (extended) abduction for computing conditions of accepting a proposal, while we use induction for that purpose.

Formally, extended abduction is defined as follows. An *abductive framework* is a pair $\langle K, \Gamma \rangle$ in which both K and Γ are first-order theories.[3] Given an observation G as a formula, a pair (E, F) is an *explanation* of G (with respect to $\langle K, \Gamma \rangle$) if (1) $(K \setminus F) \cup E \models G$, (2) $(K \setminus F) \cup E$ is consistent, and (3) both E and F consist of instances of elements from Γ. They also introduce the notion of *anti-explanations* to *un*explain negative observations. The above definition appears similar to the notion of "conditional acceptance by concession" which is defined as the relation $(K \setminus J) \cup H \models G$ in Definition 2.5 of this paper. However, there is an important difference between two definitions. In extended abduction, a hypothesis space Γ is given in advance. An explanation (E, F) is selected from the direct product $\Gamma \times \Gamma$. In our Definition 2.5, a set J is selected as a subset of weak belief Γ in a knowledge base K, while a hypothesis H is *newly* built by a knowledge base K and an observation G. This difference comes from the inherent characteristics of abduction and induction [3]. In (extended) abduc-

[3] The paper [6] introduces the framework in the context of autoepistemic logic, and another paper [15] uses it in the context of abductive logic programming.

tion, the goal is to compute causes of some observed events using a background knowledge base. In this case, possible causes are extracted from information in the knowledge base. In induction, on the other hand, the goal is to discover unknown general rules that would lie between observed events and the current belief in a knowledge base. We make use of this style of inference in the context of negotiation. A proposal given by another agent is not always explained using information included in a knowledge base only. In this case, an agent tries to bridge the gap between the proposal and its current belief. The method in [16] extracts conditions to satisfy a given proposal, which is useful for making the "Subsumption" style of proposals in Section 2.2. By contrast, our method proposed in this paper is useful for making the "Implication" style of proposals, which is more general than the subsumption style. To the best of our knowledge, no study characterizes the process of making proposals in terms of induction. On the other hand, [16] proposes another method for building proposals based on relaxation, which is different from both abduction and induction.

Meyer et al. [10] introduce a logical framework for negotiating agents. They introduce two different modes of negotiation: *concession* and *adaptation*. Concession weakens an initial demand of an agent, while adaptation expands an initial demand to accommodate a demand of another agent. They provide rational postulates to characterize negotiated outcomes between two agents, and describe methods for constructing outcomes. Compared with our present work, they consider classical propositional theories and provide logical conditions for negotiated outcomes to satisfy, but they do not provide a method of constructing proposals in negotiation. Amgoud et al. [1] develop a formal theory of argumentation-based negotiation. It provides a protocol that allows agents to exchange offers and arguments, and to make concessions when necessary. In their framework, offers that can be exchanged during a negotiation dialogue are given in advance, and concessions are made by proposing/accepting less preferred offers. Preference between offers is defined by the existence of arguments supporting them. So it does not construct new offers nor modify the current belief in a knowledge base. In the context of answer set programming, Foo et al. [2, 4] formalize one-to-one negotiation between two extended logic programs. In their framework, two agents exchange answer sets to produce a common belief set. Their goal is coordinating belief sets of two agents, and it has no mechanism of constructing new proposals.

5 Conclusion

This paper introduced a logical framework of negotiating agents. An agent evaluates a proposal and constructs a counter-proposal by building hypotheses using induction, and concedes by abandoning a part of weak belief of its knowledge base. Such behavior of an agent is formalized using default logic and realized in answer set programming. The result of this paper shows that default reasoning and induction as well as abduction, which are all developed as commonsense reasoning for a single agent in AI, are also useful for social reasoning in multi-agent systems. They provide formal methodologies for reasoning about agents, and

are used as general inference mechanisms which are independent of particular negotiation protocols.

Several issues remain for further work. We used default logic to distinguish strong and weak beliefs. In realistic negotiation, however, there would be different degrees of preferences over beliefs, and simple distinction between strong and weak beliefs might be insufficient. For further refinement, a logic for prioritized reasoning is needed, together with the capability of revising the preferences attached to beliefs. We developed a method for computing proposals in answer set programming, which enables us to realize automated negotiation on top of the existing answer set solvers. Prototyping an implementation and evaluating the proposed framework on practical negotiating tasks are in the next step.

References

1. L. Amgoud, Y. Dimopoulos, and P. Moraitis. A unified and general framework for argumentation-based negotiation. In: *Proc. AAMAS'07*, pp. 1018–1025, ACM Press.
2. W. Chen, M. Zhang, and N. Foo. Repeated negotiation of logic programs. In: *Proc. 7th Workshop on Nonmonotonic Reasoning, Action and Change*, 2006.
3. P. A. Flach and A. C. Kakas (eds.). *Abduction and Induction — Essays on their Relation and Integration*. Kluwer Academic, 2000.
4. N. Foo, T. Meyer, Y. Zhang, and D. Zhang. Negotiating logic programs. In: *Proc. 6th Workshop on Nonmonotonic Reasoning, Action and Change*, 2005.
5. M. Gelfond and V. Lifschitz. Classical negation in logic programs and disjunctive databases. *New Generation Computing* 9:365–385, 1991.
6. K. Inoue and C. Sakama. Abductive framework for nonmonotonic theory change. In: *Proc. IJCAI-95*, pp. 204–210, Morgan Kaufmann.
7. A. C. Kakas, R. A. Kowalski, and F. Toni. The role of abduction in logic programming. In: *Handbook of Logic in AI and Logic Programming*, D. M. Gabbay, et al. (eds), vol. 5, pp. 235–324, Oxford University Press, 1998.
8. V. Lifschitz. Answer set programming and plan generation. *Artificial Intelligence* 138:39–54, 2002.
9. A. C. Kakas and P. Moraitis. Adaptive agent negotiation via argumentation. In: *Proc. AAMAS'06*, pp. 384–391, ACM Press.
10. T. Meyer, N. Foo, R. Kwok, and D. Zhang. Logical foundation of negotiation: outcome, concession and adaptation. In: *Proc. AAAI-04*, pp. 293–298, MIT Press.
11. S.-H. Nienhuys-Cheng and R. De Wolf. *Foundations of inductive logic programming*. Lecture Notes in Artificial Intelligence 1228, Springer, 1997.
12. S. Parsons, C. Sierra and N. Jennings. Agents that reason and negotiate by arguing. *Journal of Logic and Computation*, 8(3):261–292, 1988.
13. R. Reiter. A logic for default reasoning. *Artificial Intelligence* 13:81–132, 1980.
14. F. Sadri, F. Toni, and P. Torroni. An abductive logic programming architecture for negotiating agents. In: *Proc. 8th European Conference on Logics in AI*, Lecture Notes in Artificial Intelligence 2424, pp. 419–431, Springer, 2002.
15. C. Sakama and K. Inoue. An abductive framework for computing knowledge base updates. *Theory and Practice of Logic Programming* 3(6):671–715, 2003.
16. C. Sakama and K. Inoue. Negotiation by abduction and relaxation. In: *Proc. AAMAS'07*, pp. 1018–1025, ACM Press.
17. C. Sakama. Negotiation by induction (short paper). In: *Proc. 7th International Joint Conference on Autonomous Agents and Multiagent Systems*, forthcoming, 2008.