Brave Induction

Chiaki Sakama[1] and Katsumi Inoue[2]

[1] Department of Computer and Communication Sciences
Wakayama University, Sakaedani, Wakayama 640-8510, Japan
sakama@sys.wakayama-u.ac.jp
[2] National Institute of Informatics
2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo 101-8430, Japan
ki@nii.ac.jp

Abstract. This paper considers the following induction problem. Given the background knowledge B and an observation O, find a hypothesis H such that a consistent theory $B \wedge H$ has a minimal model satisfying O. We call this type of induction *brave induction*. Brave induction is different from explanatory induction in ILP, which requires that O is satisfied in every model of $B \wedge H$. Brave induction is useful for learning *disjunctive* rules from observations, or learning from the background knowledge containing indefinite or incomplete information. We develop an algorithm for computing brave induction, and extend it to induction in answer set programming.

1 Introduction

Logical foundations for *induction* is one of the central topics in machine learning, and different theories of induction have been proposed in the literature [1–4, 13–15, 19, 21, 25]. A typical induction task constructs hypotheses to explain an observation (or examples) using the background knowledge. More precisely, given the background knowledge B and an observation O, a hypothesis H *covers* O under B if

$$B \wedge H \models O \tag{1}$$

where $B \wedge H$ is consistent. This style of induction is often called *explanatory induction* [8] and is usually used in *inductive logic programming* (ILP) [20].

By the definition, explanatory induction requires that a possible solution H together with B logically entails O. In other words, O is true in *every* model of $B \wedge H$. This condition is often too strong for building possible hypotheses, however. Suppose that there are 30 students in a class. Of which 20 are European, 7 are Asian, and 3 are American. The situation is represented by the background knowledge B and the observation O:

$B:\ student(1) \wedge \cdots \wedge student(30)$,
$O:\ euro(1) \wedge \cdots \wedge euro(20) \wedge asia(21) \wedge \cdots \wedge asia(27) \wedge usa(28) \wedge \cdots \wedge usa(30)$,

where each number represents individual students. In this case, the following clause, saying that every student is either European, Asian, or American, appears a good hypothesis:

$$H: \ \ euro(x) \vee asia(x) \vee usa(x) \ \leftarrow \ student(x)\,. \tag{2}$$

Unfortunately, however, H does not satisfy the relation $B \wedge H \models O$. In fact, $B \wedge H$ has many models in which O is not true. An instance of such a model is: $\{\, student(1), \ldots, student(30),\ euro(1), \ldots, euro(30) \,\}$.

Explanatory induction in ILP has been mainly used for learning *Horn* theories. When the background knowledge B and a hypothesis H are Horn theories, $B \wedge H$ has the unique minimal model (or the least model), and the relation (1) represents that O is true in the least model. In this case, the relation (1) is necessary and sufficient to make O an explanatory consequence of $B \wedge H$. On the other hand, when B or H contains *indefinite* information, $B \wedge H$ becomes a *non-Horn* theory which has multiple minimal models in general. An observation O might be true in some minimal models of $B \wedge H$ but not every one. In this case, however, the relation (1) excludes a hypothesis H due to the existence of a single minimal model in which O is not true. As a result, meaningful hypotheses might be unqualified as presented above.

To cope with the problem, this paper introduces a weak form of induction called *brave induction*. In contrast to explanatory induction, brave induction defines that a hypothesis H covers O under B if O is true in *some minimal model* of $B \wedge H$. By the definition, brave induction is weaker than explanatory induction, and the hypothesis (2) becomes a solution of brave induction.

This paper introduces a logical framework of brave induction and develops a procedure for computing hypotheses in brave induction. The proposed framework is further extended to induction from *nonmonotonic* logic programs in *answer set programming* [16]. The rest of this paper is organized as follows. Section 2 introduces the framework of brave induction and develops its computational method. Section 3 applies brave induction to nonmonotonic logic programming. Section 4 discusses related issues, and Section 5 concludes the paper.

2 Brave Induction

2.1 Logical Framework

We first introduce a logical framework of induction considered in this paper. A *first-order language* $\mathcal{L}$ consists of an alphabet and all formulas defined over it. The definition is the standard one in the literature [20]. For induction we use a *clausal language* which is a subset of $\mathcal{L}$.

A *clausal theory* (or simply a *theory*) is a finite set of *clauses* of the form:

$$A_1 \vee \cdots \vee A_m \vee \neg A_{m+1} \vee \cdots \vee \neg A_n \ \ (n \geq m \geq 0)$$

where each A_i $(1 \leq i \leq n)$ is an atom. Any variable in a clause is assumed to be universally quantified at the front. A clause of the above form is also written as

$$A_1 \vee \cdots \vee A_m \leftarrow A_{m+1} \wedge \cdots \wedge A_n\,. \tag{3}$$

$A_1 \vee \cdots \vee A_m$ is the *head* of the clause, and $A_{m+1} \wedge \cdots \wedge A_n$ is the *body*. Given a clause C of the above form, $head(C)$ represents the set $\{ A_1, \ldots, A_m \}$ and $body(C)$ represents the set $\{ A_{m+1}, \ldots, A_n \}$. A clause C is often identified with the set of literals $\{ A_1, \ldots, A_m, \neg A_{m+1}, \ldots, \neg A_n \}$. A *Horn clause* is a clause of the form (3) with $m \leq 1$. A *Horn theory* is a finite set of Horn clauses. A theory is identified with the conjunction of the clauses in it. A theory, a clause or an atom is *ground* if it contains no variable. A *ground substitution* θ replaces variables $x_1, \ldots, x_k$ occurring in a clause C (resp. an atom A) to ground terms $t_1, \ldots, t_k$ in $C\theta$ (resp. $A\theta$). A ground clause C is *prime* with respect to a theory T if $T \models C$ but $T \not\models C'$ for any $C' \subset C$. A *conjunctive normal form* (CNF) formula is a conjunction of disjunction of literals, and a *disjunctive normal form* (DNF) formula is a disjunction of conjunction of literals. A CNF formula or a DNF formula is *ground* if it contains no variable. A DNF formula $F = c_1 \vee \cdots \vee c_k$ is *irredundant* if $F \not\equiv F'$ for any $F' = c_1 \vee \cdots \vee c_{i-1} \vee c_{i+1} \vee \cdots \vee c_k$ $(1 \leq i \leq k)$.

The *domain* of a theory T is given as the *Herbrand universe* HU and an *interpretation* of T is defined as a subset of the *Herbrand base* HB. An interpretation I *satisfies* a ground clause (3) if $\{ A_{m+1}, \ldots, A_n \} \subseteq I$ implies $\{ A_1, \ldots, A_m \} \cap I \neq \emptyset$. An interpretation I satisfies a theory T if I satisfies every clause in T. In this case, I is a *model* of T and $Mod(T)$ represents the set of all models of T. A model $M \in Mod(T)$ is *minimal* if $N \subseteq M$ implies $M \subseteq N$ for any $N \in Mod(T)$. The set of minimal models of T is written as $MM(T)$. A theory T *entails* a formula F (written as $T \models F$) if F is true in any $I \in Mod(T)$. A theory T is *consistent* if $Mod(T) \neq \emptyset$; otherwise, T is *inconsistent*. A conjunction C of ground atoms is identified with the set of ground atoms in C.

Let B, O and H are all consistent theories, where B, O, and H are respectively called a *background knowledge*, an *observation*, and a *hypothesis*. We assume that B, O and H have the same HU and HB. The task of induction is to construct H when B and O are given. Formally, given the background knowledge B and an observation O, a hypothesis H *covers* O under B if

$$B \wedge H \models O \tag{4}$$

where $B \wedge H$ is consistent. This type of induction is called *explanatory induction* [8] or *learning from entailment* [3], and is usually used in *inductive logic programming* (ILP) [20]. As presented in the introduction, however, explanatory induction is too strong for handling indefinite disjunctive information. To relax the condition of explanatory induction, we introduce a weak form of induction.

Definition 2.1. (brave induction) Let B be the background knowledge and O an observation. A hypothesis H *covers* O under B in *brave induction* if a consistent theory $B \wedge H$ has a minimal model satisfying O. H is called a *solution* of brave induction.

The above definition requires that an observation is satisfied in some minimal model of a consistent theory $B \wedge H$. This is in contrast to the definition of explanatory induction in which an observation must be satisfied in every minimal

model of $B \wedge H$. In this sense, explanatory induction of (4) is also called *cautious induction*, hereafter. These names are taken from brave/cautious reasoning in nonmonotonic logics [18] and disjunctive logic programs [6]. A formula F is a consequence of *cautious inference* in a theory T if it is true in every minimal model of T, while F is a consequence of *brave inference* in T if F is true in some minimal model of T.[3] When a theory contains indefinite or incomplete information, brave inference infers more results than cautious inference in general. Brave and cautious inferences have been used in different reasoning tasks of deduction and *abduction* in artificial intelligence. Thus, it is natural to apply brave inference to induction from non-Horn theories containing indefinite or incomplete information. The utility of brave induction has already been illustrated in the introductory example. Some properties of brave induction are provided. In what follows, B, O, and H represent the background knowledge, an observation, and a hypothesis, respectively.

Proposition 2.1 *Brave induction has a solution iff $B \wedge O$ is consistent.*

Proposition 2.2 *If H covers O under B in cautious induction, H is a solution of brave induction. The converse implication also holds when B is a Horn theory.*

Proposition 2.3 *If H is a solution of brave induction, $B \wedge H \wedge O$ is consistent.*

Proposition 2.4 *For any theory $H' \models H$ such that $B \wedge H'$ is consistent, if H is a solution of brave induction, so does H'.*

Proof. The result holds by $B \wedge H' \models B \wedge H$. □

Proposition 2.4 allows us to search the most specific solutions (under implication) rather than all theories. Brave induction is nonmonotonic, that is, two solutions cannot be merged in general.

Proposition 2.5 *The fact that both H_1 and H_2 are solutions of brave induction does not imply that $H_1 \wedge H_2$ is a solution.*

Example 2.1. Let $B = \{\, p(a) \leftarrow \,\}$ and $O = \{\, q(a) \vee r(a) \leftarrow, \ \leftarrow q(a) \wedge r(a) \,\}$. Then, both $H_1 = \{\, q(x) \leftarrow p(x) \,\}$ and $H_2 = \{\, r(x) \leftarrow p(x) \,\}$ cover O under B in brave induction, but $H_1 \wedge H_2$ is not.

In Example 2.1, H_1 and H_2 cover O under B in cautious induction, but $H_1 \wedge H_2$ is not. Thus, explanatory induction is also nonmonotonic.

Proposition 2.6 *If H_1 and H_2 are solutions of brave induction, so is $H_1 \vee H_2$.*

Cautious induction also satisfies Proposition 2.6.

Proposition 2.7 *The fact that H covers both O_1 and O_2 under B does not imply H covers $O_1 \wedge O_2$ under B in brave induction.*

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

Example 2.2. Let $B = \{\, p(x) \vee q(x) \leftarrow r(x), \quad s(a) \leftarrow \}$, $O_1 = \{\, p(a) \,\}$, and $O_2 = \{\, q(a) \,\}$. Then, $H = \{\, r(x) \leftarrow s(x) \,\}$ covers both O_1 and O_2 under B in brave induction, but H does not cover $O_1 \wedge O_2$ under B.

In cautious induction, on the other hand, if H covers both O_1 and O_2 under B, so does $O_1 \wedge O_2$. Thus, Proposition 2.7 provides a property that distinguishes brave induction from cautious one.

When B has a minimal model satisfying O, O is inferred by brave inference from B. In this case, $H = \mathit{true}$ covers O, which is a trivial and uninteresting solution. The problem of our interest is the case in which B has no minimal model satisfying O. In other words, $\neg O$ is derived from B under the *generalized closed world assumption* (GCWA) [17]. It is worth noting that explanatory induction in Horn theories finds a hypothesis H when a Horn theory B has no minimal model satisfying O. In this case, $\neg O$ is derived from B under the *closed world assumption* (CWA) [22]. Thus, brave induction in non-Horn theories is considered a natural extension of explanatory induction in Horn theories.

2.2 Computation

In this section, we develop an algorithm for computing brave induction. Throughout the section, we assume that (1) any observation O is a conjunction of ground atoms,[4] and (2) any hypothesis H is a finite clausal theory such that each clause has the non-empty head. The first condition is assumed as the normal problem setting in ILP [20]. The second condition is also natural because we are interested in getting any clause which derives an observation together with the background knowledge. The next proposition characterizes the brave induction problem.

Proposition 2.8 *Let B be the background knowledge, H a hypothesis, and O an observation. Then, $B \wedge H$ has a minimal model satisfying O iff there is a disjunction F of ground atoms such that $B \wedge H \models O \vee F$ and $B \wedge H \not\models F$.*[5]

Proof. ($\rightarrow$) Suppose that $B \wedge H$ has a minimal model M such that $M \models O$. Consider a disjunction F of ground atoms satisfying (i) $M \not\models F$ and (ii) $N \models F$ for any $N \in MM(B \wedge H)$ such that $N \not\models O$. Such F is constructed by picking up ground atoms from each $N \setminus M$. Then, $B \wedge H \models O \vee F$ holds. As $M \not\models F$, $B \wedge H \not\models F$.

($\leftarrow$) Suppose that $B \wedge H \models O \vee F$ holds for a disjunction F of ground atoms and $B \wedge H \not\models F$. If $B \wedge H$ has no minimal model satisfying O, $B \wedge H \models O \vee F$ implies $B \wedge H \models F$. This contradicts the assumption that $B \wedge H \not\models F$. □

Step 1: Computing ground hypotheses

By Proposition 2.8, a solution of brave induction is obtained by computing H satisfying

$$B \wedge H \models O \vee F \tag{5}$$

[4] A conjunction O is identified with the set of ground atoms in it.

[5] Related results are shown in [9, Theorem 4.5] in the context of circumscription, and in [12, Corollary 3.5] in terms of abduction.

and

$$B \wedge H \not\models F\,. \tag{6}$$

By (5), it holds that

$$B \wedge \neg O \models \neg H \vee F. \tag{7}$$

$\neg H \vee F$ is thus obtained by deduction from $B \wedge \neg O$. This technique is *inverse entailment* that is originally proposed by Muggleton for induction in Horn theories [19], and is later extended by Inoue to full clausal theories [13].

As H is a clausal theory, put

$$H = (\Sigma_1 \leftarrow \Gamma_1) \wedge \cdots \wedge (\Sigma_k \leftarrow \Gamma_k) \tag{8}$$

where Σ_i $(i = 1, \ldots, k)$ is a disjunction of atoms and Γ_i $(i = 1, \ldots, k)$ is a conjunction of atoms. It then becomes

$$\neg H = (\neg\Sigma_1 \wedge \Gamma_1) \vee \cdots \vee (\neg\Sigma_k \wedge \Gamma_k). \tag{9}$$

Since F is a disjunction of ground atoms, every formula $\neg H \vee F$ in (7) is a disjunctive normal form. From $B \wedge \neg O$, a number of DNF formulas could be deduced. Among them, we take DNF formulas obtained as follows. First, compute *prime CNF* formulas with respect to $B \wedge \neg O$. A prime CNF formula is a conjunction of prime clauses and is obtained by a system of *consequence-finding*. For this purpose, *SOL-resolution* by Inoue [10] is used. Second, construct a DNF formula as follows: given a prime CNF formula $c_1 \wedge \cdots \wedge c_k$, produce an irredundant DNF formula $d_1 \vee \cdots \vee d_l$ where d_i $(1 \leq i \leq l)$ contains a literal from each c_j $(1 \leq j \leq k)$. Then, $B \wedge \neg O \models d_1 \vee \cdots \vee d_l$ holds, and we identify the DNF formula $\neg H \vee F$ of (7) with $d_1 \vee \cdots \vee d_l$. After deriving such ground DNF formulas of the form $\neg H \vee F$, the next problem is to extract $\neg H$ from $\neg H \vee F$. This is simply done as follows. By the assumption, Σ_i in H is non-empty, so that $\neg H$ is a DNF formula in which each disjunct $\neg\Sigma_i \wedge \Gamma_i$ of (9) contains at least one negative literal. On the other hand, F is a disjunction of ground atoms. Thus, from the DNF formula $\neg H \vee F$, $\neg H$ is extracted by selecting disjuncts containing negative literals. As such, any ground DNF formula $\neg H$ is obtained. From this $\neg H$, we can obtain a clausal theory H such that $B \wedge H$ has a minimal model satisfying O (by Proposition 2.8).

Step 2: Generalization

H is a clausal theory containing no variable, so that H is generalized in the next step. Two cases are considered: (a) O contains a single predicate, and (b) O contains multiple different predicates. In case of (a), we apply Plotkin's *least generalization under subsumption (LGS)* [21] to H. The LGS of any finite set of clauses exists and is computed by the LGS algorithm in [20, 21]. The result of LGS is written as $lgs(H)$. In case of (b), let n be the number of different predicates appearing in O. Then, O is partitioned into disjoint subsets:

$$O = O_1 \wedge \cdots \wedge O_n \tag{10}$$

where O_i $(1 \leq i \leq n)$ is a conjunction of ground atoms having the same predicate. Correspondingly, H is partitioned as

$$H = H_1 \wedge \cdots \wedge H_n \tag{11}$$

where H_i $(1 \leq i \leq n)$ is a conjunction of clauses whose heads contain the predicate in O_i. The LGS of each H_i is then computed and collected as

$$lgs(H) = lgs(H_1) \wedge \cdots \wedge lgs(H_n)\,. \tag{12}$$

Note that the equation (12) also represents the result of (a) when $n = 1$.

Step 3: Constructing a weak form of hypotheses

When an observation has some specific property, $lgs(H_i)$ is combined into a weaker formula.

Definition 2.2. (synchronous) Let $pred(A)$ be the predicate of an atom A, and $const(A)$ the set of constants in A. Given a set S of atoms, suppose two atoms A_1 and A_2 in S such that $pred(A_1) = p_1$, $pred(A_2) = p_2$, and $p_1 \neq p_2$. Then, p_1 and p_2 are *synchronous* in S if $const(A_1) \cap const(A_2) \neq \emptyset$. Otherwise, p_1 and p_2 are *asynchronous* in S. A set S is *asynchronous* if p_1 and p_2 are asynchronous in S for any different predicates p_1 and p_2 in S.

An observation O is *asynchronous* if O is an asynchronous set. Suppose that O is partitioned into n disjoint sets as (10) and H is partitioned as (11). In this case, take the *greatest specialization under implication (GSI)* [20] of clauses $lgs(H_1), \ldots, lgs(H_n)$. The GSI of any finite set of clauses exists and is computed by simply taking a disjunction as

$$gsi(lgs(H_1), \ldots, lgs(H_n)) = lgs(H_1) \vee \cdots \vee lgs(H_n). \tag{13}$$

By $lgs(H_i) \models gsi(lgs(H_1), \ldots, lgs(H_n))$ for $i = 1, \ldots, n$, the GSI (13) provides a formula which is weaker than each $lgs(H_i)$.

Step 4: Optimization

Hypotheses computed in the above two steps generally contain clauses or atoms that are useless or have no direct connection to explaining the observation O. To extract useful information, a method for optimization is provided.

Definition 2.3. (isolated) Let $term(A)$ be the set of terms appearing in an atom A. Then, two atoms A_1 and A_2 are *linked* if $term(A_1) \cap term(A_2) \neq \emptyset$. Given a clause C, an atom $A \in body(C)$ is *isolated* in C if there is no atom $A'(\neq A)$ in C such that A' and A are linked.

For any clause C in $lgs(H_i)$ $(1 \leq i \leq n)$,

1. remove any atom A from $head(C)$ such that $pred(A)$ is not included in O,
2. remove any atom A from $body(C)$ such that A is isolated in C.

Procedure: BRAIN

Input : the background knowledge B and an observation O;
Output : hypotheses $H^\wedge$ and $H^\vee$.

Step 1 : Compute ground and irredundant DNF formulas $\neg H \vee F$ from $B \wedge \neg O$, and extract $\neg H$ from $\neg H \vee F$.
Step 2 : Compute $lgs(H)$.
Step 3 : If O is asynchronous and is partitioned into $O = O_1 \wedge \cdots \wedge O_n$, compute $gsi(lgs(H_1), \ldots, lgs(H_n))$.
Step 4 : If $B \wedge lgs^*(H_i)$ is consistent, put $H^\wedge = lgs^*(H_1) \wedge \cdots \wedge lgs^*(H_n)$ and $H^\vee = lgs^*(H_1) \vee \cdots \vee lgs^*(H_n)$.

Fig. 1. An algorithm for brave induction

The first reduction eliminates atoms in the head which do not contribute to the derivation of observations. The second reduction eliminates atoms in the body which have no connection to the observation. Let $lgs^*(H_i)$ be the result of such reduction over $lgs(H_i)$. When $B \wedge lgs(H_i)$ is consistent, the reduction is performed as far as $B \wedge lgs^*(H_i)$ is consistent. The algorithm (called BRAIN) for computing hypotheses is summarized in Figure 1.[6]

Now we show that BRAIN computes a solution for brave induction.

Lemma 2.9 *Let B be the background knowledge and O an observation. Let $H^\wedge$ be a clausal theory obtained by* BRAIN. *If $B \wedge H^\wedge$ is consistent, $B \wedge H^\wedge$ has a minimal model satisfying O.*

Proof. (i) Suppose first that O contains a single predicate. Then, for each clause $C_i = \Sigma_i \leftarrow \Gamma_i$ $(1 \leq i \leq k)$ of (8), there is a ground substitution θ_i such that $lgs(H)\theta_i \subseteq C_i$ for the $lgs(H)$ of (12). Thus, $lgs(H) \models H$ and $B \wedge lgs(H) \models B \wedge H$. So $lgs(H)$ satisfies the relation (5). By $H^\wedge \models lgs(H)$, $H^\wedge$ also satisfies (5). Selecting a disjunction F of ground atoms such that $B \wedge H^\wedge \not\models F$, $H^\wedge$ satisfies the relation (6). Hence, the result holds by Proposition 2.8. (ii) Next, suppose that O contains multiple different predicates. Then, for each H_i of (11), $lgs(H_i) \models H$. This implies $lgs(H) \models H$ and $lgs(H)$ satisfies the relation (5). The rest of the proof is the same as (i). □

Lemma 2.10 *Let B be the background knowledge and O an asynchronous observation. Let $H^\vee$ be a clausal theory obtained by* BRAIN. *If $B \wedge H^\vee$ is consistent, $B \wedge H^\vee$ has a minimal model satisfying O.*

Proof. Let $O = O_1 \wedge \cdots \wedge O_n$. By Lemma 2.9, $B \wedge lgs^*(H_i)$ has a minimal model M_i satisfying O_i $(1 \leq i \leq n)$. Putting $M = \bigcup_{1 \leq i \leq n} M_i$, M satisfies O. As $B \wedge lgs^*(H_i) \models B \wedge lgs^*(H_1) \vee \cdots \vee lgs^*(H_n)$, M is a model of $B \wedge H^\vee$ satisfying O. Since the head of $H^\vee$ consists of atoms with different predicates, for any

[6] BRAIN is named after BRAve INduction.

ground instance of $H^\vee$, we can select an atom $A \in M$ from the head of each clause whenever $A \in O$. Since O is an asynchronous set, two atoms A_1 and A_2 with different predicates are not selected from the same ground instance of $H^\vee$. Hence, M is a minimal model of $B \wedge H^\vee$. □

By Lemmas 2.9 and 2.10, we have the next result.

Theorem 2.11. *Any hypothesis computed by* Brain *becomes a solution of brave induction.*

Example 2.3. Consider the background knowledge B and the observation O:

$$B:\ teacher(0) \wedge student(1) \wedge \cdots \wedge student(30),$$
$$O:\ euro(1) \wedge \cdots \wedge euro(20) \wedge asia(21) \wedge \cdots \wedge asia(27) \wedge usa(28) \wedge \cdots \wedge usa(30).$$

Brain computes candidate hypotheses as follows. First, $B \wedge \neg O$ entails the prime CNF formula $B \wedge \neg O$. From this, the ground and irredundant DNF formula $\neg H_1 \vee \neg H_2 \vee \neg H_3$ is obtained where

$$H_1 = (\neg B \vee euro(1)) \wedge \cdots \wedge (\neg B \vee euro(20)),$$
$$H_2 = (\neg B \vee asia(21)) \wedge \cdots \wedge (\neg B \vee asia(27)),$$
$$H_3 = (\neg B \vee usa(28)) \wedge \cdots \wedge (\neg B \vee usa(30)).$$

The LGS of each H_i becomes

$$lgs(H_1) = \neg teacher(0) \vee \neg student(x) \vee euro(x),$$
$$lgs(H_2) = \neg teacher(0) \vee \neg student(y) \vee asia(y),$$
$$lgs(H_3) = \neg teacher(0) \vee \neg student(z) \vee usa(z).$$

Then, $lgs(H) = lgs(H_1) \wedge lgs(H_2) \wedge lgs(H_3)$. On the other hand, as O is asynchronous, the greatest specialization becomes $gsi(lgs(H_1), \ldots, lgs(H_n)) = lgs(H_1) \vee lgs(H_2) \vee lgs(H_3)$.

Finally, the atom $teacher(0)$ is isolated in each $lgs(H_i)$ $(i = 1, 2, 3)$, so that it is remove from the body of each clause. As a result, $H^\wedge$ becomes

$$(euro(x) \leftarrow student(x)) \wedge (asia(x) \leftarrow student(x)) \wedge (usa(x) \leftarrow student(x)),$$

and $H^\vee$ becomes

$$euro(x) \vee asia(x) \vee usa(x) \leftarrow student(x).$$

Thus, $H^\wedge$ and $H^\vee$ become two solutions of brave induction.

In Example 2.3, $H^\wedge$ also becomes a solution of cautious induction, but $H^\vee$ is a solution inherent to brave induction.

3 Brave Induction in Nonmonotonic Logic Programming

As presented in Section 2, brave induction is useful for learning theories with indefinite or incomplete information. Incomplete information is also represented as *default* rule in logic programming. In this section, we consider brave induction in *nonmonotonic logic programs*.

3.1 Answer Set Programming

Answer set programming (ASP) [16] represents incomplete knowledge in a logic program and realizes nonmonotonic default reasoning. In ASP a logic program is described by an *extended disjunctive program* (EDP). An EDP (or simply a *program*) is a set of rules of the form:

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

($n \geq m \geq l \geq 0$) where each L_i is a positive/negative literal, i.e., A or $\neg A$ for an atom A. *not* represents *default negation* or *negation as failure* (NAF). $not\, L$ is called an *NAF-literal*. Literals and NAF-literals are called *LP-literals*. The symbol ";" represents disjunction and "," represents conjunction. The above rule is read "If all $L_{l+1}, \ldots, L_m$ are believed and all $L_{m+1}, \ldots, L_n$ are disbelieved, then some of $L_1, \ldots, L_l$ is believed". The left-hand side of the rule is the *head*, and the right-hand side is the *body*. For each rule r of the above form, $head(r)$, $body^+(r)$ 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 disjunction of literals and a conjunction of (NAF-)literals in a rule are identified with its corresponding sets of literals. 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. Two literals L_1 and L_2 have the same *sign* if both L_1 and L_2 are positive literals (or negative literals). A set S of ground literal is *consistent* if $L \in S$ implies $\neg L \notin S$ for any literal L; otherwise, S is *inconsistent*. Let L_0 be a ground literal and S a set of ground literals. Then, $L_1 \in S$ is *relevant* to L_0 if either (i) $const(L_0) \cap const(L_1) \neq \emptyset$, or (ii) for some literal $L_2 \in S$, $const(L_1) \cap const(L_2) \neq \emptyset$ and L_2 is relevant to L_0. Otherwise, $L_1 \in S$ is *irrelevant* to L_0.

The semantics of an EDP is defined by the *answer set semantics*. Let Lit be the set of all ground literals in the language of a program. Suppose a program P and a set of literals $S(\subseteq Lit)$. 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 that 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. Given an EDP P and a set S of 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*.

Example 3.1. The program:

$$
\begin{aligned}
&tea\,;\ coffee \leftarrow,\\
&milk \leftarrow tea,\ not\ lemon,\\
&lemon \leftarrow tea,\ not\ milk,\\
&milk \leftarrow coffee,
\end{aligned}
$$

has the three answer sets: $S_1 = \{\, tea,\ milk \,\}$, $S_2 = \{\, tea,\ lemon \,\}$, and $S_3 = \{\, coffee,\ milk \,\}$, which represent possible options for drink.

3.2 Brave Induction in ASP

In this section, we consider the following problem setting:

- the background knowledge B is given as a consistent EDP,
- an observation O is given as a consistent set of ground literals,
- a hypothesis H is a consistent set of rules.

Then, brave induction in ASP is defined as follows.

Definition 3.1. (brave induction in ASP) Let B be the background knowledge and O an observation. A hypothesis H *covers* O under B in *brave induction* if $B \cup H$ has a consistent answer set S such that $O \subseteq S$.[7]

Cautious induction, by contrast, requests that $O \subseteq S$ holds for *every* consistent answer set S of $B \cup H$. Brave induction in ASP has properties similar to those of clausal theories. As the case of clausal theories, the problem of our interest is the case when B has no answer set including O.

In case of brave induction from clausal theories, inverse entailment is used for computing hypotheses. However, it is known that inverse entailment in classical logic is not applied to nonmonotonic logic programs [23]. We then consider another method for computing possible hypotheses.

Step 1: Computing ground hypotheses

Given an observation O, let $\Theta = \{\, L \mid L \in Lit \text{ and } pred(L) \text{ appears in } O \,\}$. Suppose that the background knowledge B has an answer set S. Then, construct a finite and consistent set R of ground rules satisfying the following conditions. For any rule $r \in R$,

[7] In nonmonotonic logic programming, logical connectives in classical logic are not used. So we write $B \cup H$ instead of $B \wedge H$.

1. $head(r) = \{L\}$ for any $L \in O$,
2. $body^+(r) = \{\, L \mid L \in S$ and L is relevant to the literal in $head(r) \,\}$.
3. $body^-(r) = \{\, L \mid L \in Lit \backslash (S \cup \Theta)$ and L is relevant to the literal in $head(r)$ and appears in $Ground(P) \,\}$.

The third condition requires that no rule contains default negation of literals in $S \cup \Theta$. The reason is that if $body^-(r)$ contains literals from S, $body(r)$ may contain both L in $body^+(r)$ and $not\, L$ in $body^-(r)$, which makes the rule meaningless. Also, if $body^-(r)$ contains literals from Θ, r may contain a *negative loop* that would make a program inconsistent. By its construction, different hypotheses are constructed by different answer sets in general.

Step 2: Generalization

The notion of LGS is extended to rules containing default negation. It is done by syntactically viewing rules as "clauses". That is, identify disjunction ";" with the classical one "$\vee$, and any NAF-literal "$not\, p(t_1, \ldots, t_n)$" with a new atom "$not\_p(t_1, \ldots, t_n)$" with the predicate "$not\_p$". $\neg p$ is also considered a predicate "$\neg\_p$" and is considered a predicate different from p. With this setting, the LGS of a finite set of rules is defined in the same manner as the one in clausal theories [24]. The generalization phase is similar to the case of clausal theories. If O contains a single predicate, compute $lgs(R)$. Else if O contains multiple different predicates, O is partitioned into disjoint subsets $O = O_1 \cup \cdots \cup O_n$ where O_i $(1 \leq i \leq n)$ is a set of ground literals having the same predicate. Correspondingly, R is partitioned as $R = R_1 \cup \cdots \cup R_n$ where R_i $(1 \leq i \leq n)$ is a set of ground rules whose heads have the predicate in O_i. The LGS of each R_i is then computed and collected as $lgs(R) = lgs(R_1) \cup \cdots \cup lgs(R_n)$.

Step 3: Constructing a weak form of hypotheses

The GSI of two rules $Head_1 \leftarrow Body_1$ and $Head_2 \leftarrow Body_2$ is also defined as $Head_1\,;\, Head_2 \leftarrow Body_1,\, Body_2$, by taking the disjunction/conjunction of two heads/bodies. When an observation O is an asynchronous set, the GSI of $lgs(R_1), \ldots, lgs(R_n)$ is constructed as

$$gsi(lgs(R_1), \ldots, lgs(R_n))$$
$$= head(lgs(R_1))\,;\, \cdots\,;\, head(lgs(R_n)) \leftarrow body(lgs(R_1)),\, \ldots,\, body(lgs(R_n))\,.$$

Step 4: Optimization

The notion of "isolated literal" in a rule is defined by replacing a clause with a rule, and an atom with a literal in Definition 2.3. Then, for any rule r in $lgs(R_i)$ $(1 \leq i \leq n)$, remove any literal L from $body(r)$ such that L is isolated in r. Let $lgs^*(R_i)$ be the result of such reduction over $lgs(R_i)$. When $B \cup lgs(R_i)$ is consistent, the reduction is performed as far as $B \cup lgs^*(R_i)$ is consistent.

The algorithm of brave induction in ASP (called $\textsc{Brain}^{not}$) is sketched in Figure 2. In what follows, we show that $\textsc{Brain}^{not}$ computes hypotheses for brave induction in ASP. We say that O is *independent* of B if every predicate in O appears nowhere in B.

Procedure: BRAIN^{not}

Input : the background knowledge B and an observation O;
Output : hypotheses $H^{\wedge}$ and $H^{\vee}$.

Step 1 : Select an answer set S of B and construct a set R of rules.
Step 2 : Compute $lgs(R)$.
Step 3 : If O is asynchronous and is partitioned into $O = O_1 \cup \cdots \cup O_n$, compute $gsi(lgs(R_1), \ldots, lgs(R_n))$.
Step 4 : If $B \cup lgs^*(R_i)$ is consistent, put $H^{\wedge} = lgs^*(R_1) \cup \cdots \cup lgs^*(R_n)$ and $H^{\vee} = \{\, head(lgs^*(R_1)) \,;\, \cdots \,;\, head(lgs^*(R_n)) \leftarrow body(lgs^*(R_1)),\, \ldots,\, body(lgs^*(R_n)) \,\}$.

Fig. 2. An algorithm for brave induction in ASP

Proposition 3.1 *Let P be a consistent program. Suppose a consistent set R of rules such that for any $r \in R$, every predicate in $head(r)$ appears nowhere in P. Then, $P \cup R$ is consistent.*

Lemma 3.2 *Let B be the background knowledge and O an observation. Let $H^{\wedge}$ be a set of rules obtained by* BRAIN^{not}*. If O is independent of B, $B \cup H^{\wedge}$ has an answer set U such that $O \subseteq U$.*

Proof. Let S be an answer set of B. For any rule r in R, $head(r) \leftarrow body^+(r)$ is in R^S. Here, $body^+(r) \subseteq S$, $head(r) = \{L\}$, and $pred(L)$ appears nowhere in B. Put $T = S \cup \{\, L \mid L \in head(r) \text{ and } r \in R^S \,\}$. By $B^T \cup R^T = B^S \cup R^T$, T becomes a minimal closed set of $B^T \cup R^T = (B \cup R)^T$. Since O is independent of B, every predicate in $head(r)$ appears nowhere in P and $B \cup R$ is consistent (Proposition 3.1). As R contains rules having every literal in O, T is a consistent answer set of $B \cup R$ such that $O \subseteq T$. Next, we show that $B \cup lgs(R)$ has an answer set such that $O \subseteq U$. Let $R = R_1 \cup \cdots \cup R_n$. By the definition, $lgs(R_i)\theta \subseteq r$ for any $r \in R_i$ $(1 \leq i \leq n)$ with some ground substitution θ. Then, for any rule $r \in R_i$, $body^-(r) \cap S = \emptyset$ implies $body^-(lgs(R_i)\theta) \cap S = \emptyset$. So $B^S \cup R^S \subseteq B^S \cup lgs(R)^S$. Since O is independent of B, $lgs(R)^S \setminus R^S$ is a set of NAF-free rules whose heads have predicates appearing nowhere in B. Put $V = \{\, L \mid r \in lgs(R)^S \setminus R^S,\ head(r) = \{L\} \text{ and } body^+(r) \subseteq S \,\}$. Then, $B^{S \cup V} \cup lgs(R)^{S \cup V} = B^S \cup lgs(R)^S$ has a minimal closed set $U = S \cup V$. Since O is independent of B, $B^S \cup lgs(R)^S$ is consistent (Proposition 3.1). As $lgs(R)$ contains rules having every literal in O, U is a consistent answer set of $B \cup lgs(R)$ such that $O \subseteq U$. When $B \cup lgs^*(R_i)$ is consistent, U also becomes a consistent answer set of $B \cup H^{\wedge}$. Hence, the result follows. □

Lemma 3.3 *Let B be the background knowledge and $O = O_1 \cup \cdots \cup O_n$ an asynchronous observation. Let $H^{\vee}$ be a set of rules obtained by* BRAIN^{not}*. If O is independent of B, $B \cup H^{\vee}$ has an answer set U such that $O \subseteq U$.*

Proof. Let $r = gsi(lgs(R_1), \ldots, lgs(R_n))$. For some answer set S of B, $body^+(r\theta) \subseteq S$ and $body^-(r\theta) \subseteq Lit \setminus (S \cup \Theta)$ hold for any ground instance $r\theta$ of r. Since

$head(r)$ consists of literals with different predicates, for any ground instance of $r\theta$, a set T of literals is constructed in a way that a literal $L \in T$ is selected from the head of each $r\theta$ whenever $L \in O$. Since O is an asynchronous set, two literals L_1 and L_2 with different predicates are not selected from the same ground instance of $gsi(lgs(R_1), \ldots, lgs(R_n))$. In this way, we can construct a minimal closed set $U = S \cup T$ of $B^U \cup \{r\}^U$. Since O is independent of B, $B^U \cup \{r\}^U$ is consistent (Proposition 3.1). Then, U becomes a consistent answer set of $B \cup \{\, gsi(lgs(R_1), \ldots, lgs(R_n)) \,\}$ and $O \subseteq U$. When $B \cup lgs^*(R_i)$ is consistent, U also becomes a consistent answer set of $B \cup H^\vee$. □

By Lemmas 3.2 and 3.3, we have the next result.

Theorem 3.4. *Any hypothesis computed by* BRAIN^{not} *becomes a solution of brave induction.*

Example 3.2. There are two couples, Adam and Nancy, and Bob and Jane. They plan to go to either sea or mountain on this weekend. Each couple can select one of them, but a husband and a wife go to the same place. The situation is represented as the background knowledge B:

$$
\begin{aligned}
&s(x) \leftarrow \mathit{not}\ m(x),\\
&m(x) \leftarrow \mathit{not}\ s(x),\\
&c(a,n) \leftarrow, \quad c(b,j) \leftarrow,\\
&\leftarrow c(x,y), s(x), m(y),\\
&\leftarrow c(x,y), s(y), m(x)
\end{aligned}
$$

where the predicates s, m and c mean *sea*, *mountain* and *couple*, respectively, and the constants a, n, b and j mean Adam, Nancy, Bob and Jane, respectively. B has four answer sets: $S_1 = \{\, c(a,n),\ c(b,j),\ s(a),\ s(n),\ s(b),\ s(j) \,\}$, $S_2 = \{\, c(a,n),\ c(b,j),\ s(a),\ s(n),\ m(b),\ m(j) \,\}$, $S_3 = \{\, c(a,n),\ c(b,j),\ m(a),\ m(n),\ s(b),\ s(j) \,\}$, and $S_4 = \{\, c(a,n),\ c(b,j),\ m(a),\ m(n),\ m(b),\ m(j) \,\}$.

Suppose the observation that Adam and Nancy are tanned, but Bob and Jane are not. It is represented as:

$$O = \{\, t(a),\ t(n),\ \neg t(b),\ \neg t(j) \,\}$$

where the predicate t mean *tanned*.

BRAIN^{not} constructs candidate hypotheses as follows. First, an answer set of B, for instance S_2, is selected. A set R of rules is then constructed as:

$$
\begin{aligned}
&t(a) \leftarrow c(a,n),\ s(a),\ s(n),\ \mathit{not}\ m(a),\ \mathit{not}\ m(n),\\
&t(n) \leftarrow c(a,n),\ s(a),\ s(n),\ \mathit{not}\ m(a),\ \mathit{not}\ m(n),\\
&\neg t(b) \leftarrow c(b,j),\ m(b),\ m(j),\ \mathit{not}\ s(b),\ \mathit{not}\ s(j),\\
&\neg t(j) \leftarrow c(b,j),\ m(b),\ m(j),\ \mathit{not}\ s(b),\ \mathit{not}\ s(j).
\end{aligned}
$$

Next, the $lgs(R)$ is constructed as

$$
\begin{aligned}
&t(x) \leftarrow c(a,n),\ s(x),\ \mathit{not}\ m(x),\\
&\neg t(y) \leftarrow c(b,j),\ m(y),\ \mathit{not}\ s(y).
\end{aligned}
$$

Since O is an asynchronous set, $gsi(lgs(R_1), \ldots, lgs(R_n))$ is also constructed as

$$t(x)\,;\, \neg\, t(y) \leftarrow c(a,n),\, c(b,j),\, s(x),\, m(y),\, \mathit{not}\; m(x),\, \mathit{not}\; s(y).$$

Finally, isolated literals $c(a,n)$ and $c(b,j)$ are removed, and $H^{\wedge}$ and $H^{\vee}$ become
$H^{\wedge} = \{\, t(x) \leftarrow s(x),\, \mathit{not}\; m(x), \quad \neg t(y) \leftarrow m(y),\, \mathit{not}\; s(y)\,\}$ and
$H^{\vee} = \{\, t(x)\,;\, \neg\, t(y) \leftarrow s(x),\, m(y),\, \mathit{not}\; m(x),\, \mathit{not}\; s(y)\,\}$, respectively.

4 Discussion

There are some induction frameworks which relax explanatory induction in different ways. De Raedt and Dehaspe [3, 4] introduce the framework of *learning from satisfiability* (LFS). Given the background knowledge B and an observation O, a hypothesis H covers O under B in LFS if $B \wedge H \wedge O$ is consistent. In other words, H covers O under B in LFS if $B \wedge H$ has a model satisfying O. By the definition, LFS is weaker than brave induction. That is, if a hypothesis H covers O under B in brave induction, H covers O under B in LFS (cf. Proposition 2.3). The converse implication does not hold in general. Compared with brave induction, LFS does not require the minimality of models. So any theory H becomes a solution as far as it is consistent with $B \wedge O$. This implies that any hypothesis which has no connection to $B \wedge O$ may become a solution. For instance, let $B = \{\, p(a)\,\}$ and $O = \{\, q(a)\,\}$. Then, $H_1 = \{\, r(b)\,\}$, $H_2 = \{\, s(x) \leftarrow r(x)\,\}$, $H_3 = \{\, \neg\, s(c)\,\}, \ldots$ are all solutions in LFS. Note that none of H_1, H_2, and H_3 becomes a solution of brave induction. As seen in the above example, LFS appears too weak for building useful hypotheses. Since it generally produces tons of useless hypotheses, additional conditions must be introduced to reduce the hypotheses space for practical usage. Brave induction is considered as a restricted version of LFS, that is, we imposed the condition of *minimality* on models of $B \wedge H$ satisfying O. Moreover, in [3] the authors say:

> One open question for further research is how learning from satisfiability (which employs a monotonic logic) could be used for inducing nonmonotonic logic programs.

This paper provides a solution in the context of brave induction.

Confirmatory induction or *descriptive induction* [15] provides a different method for induction. Given the background knowledge B and an observation O such that $B \wedge O$ is consistent, a hypothesis H *covers* E under B in confirmatory induction if $\mathit{Comp}(B \wedge O) \models H$ where Comp represents Clark's predicate completion. When B is a set of definite clauses, any hypothesis H induced in explanatory induction and the *closed world assumption* becomes a solution of confirmatory induction [1]. For instance, given $B = \{\, \mathit{human}(\mathit{Socrates})\,\}$ and $O = \{\, \mathit{mortal}(\mathit{Socrates})\,\}$, both $H_1 = (\mathit{mortal}(x) \leftarrow \mathit{human}(x))$ and $H_2 = (\mathit{human}(x) \leftarrow \mathit{mortal}(x))$ become solutions of confirmatory induction, but only H_1 is the solution of explanatory induction and brave induction. On the other hand, explanatory induction and brave induction are not always stronger

than confirmatory induction. For instance, let $B = \{ p(x) \leftarrow q(x) \}$ and $O = p(a)$. Then, $H = q(a) \wedge q(b)$ becomes a solution of explanatory induction and brave induction, while H is not a solution of confirmatory induction. Thus, there is no relation between brave induction and confirmatory induction in general. Generally speaking, confirmatory induction does not explain why particular individuals are observed under the background knowledge, and the aim is to learn relationships between any of the concepts [8]. For induction in full clausal theories, Inoue [13] introduces *CF-induction* which extends Muggleton's inverse entailment to full clausal theories. However, CF-induction is cautious induction and is often too strong for learning indefinite theories. Induction in answer set programming is introduced by Sakama [25], but it is also cautious induction.

Brave induction guarantees the existence of a minimal model of $B \wedge H$ in which an observation O is satisfied. In this case, H covers the *positive* observation O under B. In ILP, on the other hand, *negative* observations as well as positive ones are also handled. Given a negative observation N, it is required that H *uncovers* N under B. This condition is logically represented as $B \wedge H \not\models N$. Definition 2.1 is extended to handle negative observations as follows.

Definition 4.1. Let B be the background knowledge, P a positive observation, and N a negative observation. A hypothesis H is a solution of brave induction if $B \wedge H$ has a minimal model M such that $M \models P$ and $M \not\models N$.

By putting $O = P \wedge \neg N$, the above definition reduces to Definition 2.1 and negative observations are handled within the framework of this paper.[8]

In this paper, we introduced induction algorithms which produce clauses or rules that define more than one predicate. The problem is known as *multiple predicate learning* (MPL) [2]. In MPL the order of learning different clauses affects the results of the learning tasks and even the existence of solutions, especially in the presence of negative observations [1]. We do not discuss details of the problem in this paper, and just impose the condition of consistency on $B \wedge lgs^*(H_i)$ in the first-order case. In case of ASP, independence of O with respect to B guarantees the consistency of $B \cup \{ lgs^*(R) \}$ as far as B is consistent. Further discussion for MPL is left for future study.

We finally remark the computational complexity issue. First, in case of clausal theories (CT), brave induction has a solution H iff $B \wedge O$ is consistent (Proposition 2.1). Then, given a ground clausal theory B and a ground observation O, deciding the existence of solutions in brave induction is NP-complete. In case of ASP, brave induction has a solution H if $B \cup O$ has a consistent answer set. The decision problem is Σ_2^P-complete [6]. Next, we consider the task of identifying whether a theory H is a solution of brave induction. To this end, we consider a complementary problem: a ground clausal theory $B \wedge H$ has no minimal model satisfying a conjunction O of ground atoms. This is a task of the *extended GCWA* and is known Π_2^P-complete [6], so that the identification problem is Σ_2^P-complete. Deciding whether the ground program $B \cup H$ has a

[8] Strictly speaking, $\neg N$ requires *Skolemization* when a clausal theory N contains variables. For detailed technique, see [13].

consistent answer set satisfying O is also Σ_2^P-complete [6]. On the other hand, in cautious induction, the decision problem for the existence of solutions has the same complexity in both CT and ASP. The decision problem for the identification of solutions is coNP-complete in CT, and Π_2^P-complete in ASP. Comparing those results, brave induction appears more expensive than cautious induction for identifying solutions in CT.

Brave and cautious inferences are widely used for commonsense reasoning from incomplete knowledge. In hypothetical reasoning, two different types of *abduction* under brave and cautious inferences are introduced by [7, 11] under the *stable model semantics* of logic programs. To the best of our knowledge, however, no studies introduce brave induction as a form of learning or learning from incomplete information. Since abduction and induction are both hypothetical reasoning which extend the background knowledge to explain observations, brave induction proposed in this paper has a right place and serves as a natural extension of brave abduction.

5 Conclusion

This paper introduced the framework of brave induction which is weaker than explanatory induction usually used in ILP. We developed an algorithm for computing brave induction in full clausal theories, and also extended the framework to induction in answer set programming. As argued in the paper, explanatory induction is often too strong for learning indefinite or incomplete theories. Learning from satisfiability, on the other hand, appears too weak as presented in Section 4. Brave induction has a position between learning from satisfiability and explanatory induction, and provides a moderate solution in the middle.

We are now seeking practical applications of brave induction. One of the candidates is *systems biology* which would have indefinite or incomplete information in the background knowledge and observations. A recent study [5] shows that an ILP approach is useful for finding causal relations between concentration changes of metabolities and enzyme activities. In [5] CF-induction [13] is used for learning hypotheses. Since CF-induction is cautious induction, there would be a room for brave induction to find new hidden hypotheses. A theoretical extension to *circumscriptive induction* [14] is also a topic for future research.

References

1. L. De Raedt and N. Lavrač. The many faces of inductive logic programming. In: *Methodologies for Intelligent Systems, 7th International Symposium*, Lecture Notes in Computer Science 689, pp. 435–449, Springer 1993.
2. L. De Raedt and N. Lavrač. Multiple predicate learning in two inductive logic programming setting. In: *Journal of the IGPL*, 4(2):227–254, 1996.
3. L. De Raedt. Logical settings for concept-learning. *Artificial Intelligence*, 95:187–201, 1997.
4. L. De Raedt and L. Dehaspe. Learning from satisfiability. In: *Proceedings of the 9th Dutch Conference on Artificial Intelligence*, pp. 303–312, 1997.

5. A. Doncescu, Y. Yamamoto, and K. Inoue. Biological systems analysis using inductive logic programming. In: *Proceedings of the 21st International Conference on Advanced Information Networking and Applications*, pp. 690–695, IEEE Computer Society, 2007.
6. T. Eiter and G. Gottlob. On the computational cost of disjunctive logic programming: propositional case. *Annals of Mathematics and Artificial Intelligence*, 15: 289–323, 1995.
7. T. Eiter, G. Gottlob, and N. Leone. Abduction from logic programs: semantics and complexity. *Theoretical Computer Science*, 189:129–177, 1997.
8. P. A. Flach and A. C. Kakas. Abductive and inductive reasoning: background and issues. In: P. A. Flach and A. C. Kakas (eds.). *Abduction and Induction — Essays on their Relation and Integration*, Kluwer Academic, 2000.
9. M. Gelfond, H. Przymusinska, and T. Przymusinski. On the relationship between circumscription and negation as failure. *Artificial Intelligence*, 38:75–94, 1989.
10. K. Inoue. Linear resolution for consequence finding. *Artificial Intelligence*, 56:301–353, 1992.
11. K. Inoue and C. Sakama. A fixpoint characterization of abductive logic programs. *Journal of Logic Programming*, 27(2):107–136, 1996.
12. K. Inoue. Automated abduction. In: *Computational Logic: Logic Programming and Beyond, Essays in Honour of Robert A. Kowalski, Part II*, Lecture Notes in Artificial Intelligence 2408, pp. 311–341, Springer, 2002.
13. K. Inoue. Induction as consequence finding. *Machine Learning*, 55:109–135, 2004.
14. K. Inoue and H. Saito. Circumscription policies for induction. In: *Proceedings of the 14th International Conference on Inductive Logic Programming*, Lecture Notes in Artificial Intelligence 3194, pp. 164–179, Springer, 2004.
15. N. Lachiche. Abduction and induction from a non-monotonic reasoning perspective. In: P. A. Flach and A. C. Kakas (eds.). *Abduction and Induction — Essays on their Relation and Integration*, Kluwer Academic, 2000.
16. V. Lifschitz. Answer set programming and plan generation. *Artificial Intelligence*, 138:39–54, 2002.
17. J. Minker. On indefinite data bases and the closed world assumption. In: *Proceedings of the 6th International Conference on Automated Deduction*, Lecture Notes in Computer Science 138, pp. 292–308, Springer, 1982.
18. D. McDermott. Nonmonotonic logic II: nonmonotonic modal theories. *Journal of the ACM*, 29:33–57, 1982.
19. S. Muggleton. Inverse entailment and Progol. *New Generation Computing*, 13:245–286, 1995.
20. S.-H. Nienhuys-Cheng and R. De Wolf. *Foundations of inductive logic programming*, Lecture Notes in Artificial Intelligence 1228, Springer, 1997.
21. G. D. Plotkin. A note on inductive generalization. In: B. Meltzer and D. Michie (eds.), *Machine Intelligence*, vol. 5, pp. 153–63, Edinburgh University Press, 1970.
22. R. Reiter. On closed world databases. In: H. Gallaire and J. Minker (eds.), *Logic and Data Bases*, pp. 55–76. Plenum, New York, 1978.
23. C. Sakama. Inverse entailment in nonmonotonic logic programs. In: *Proceedings of the 10th International Conference on Inductive Logic Programming*, Lecture Notes in Artificial Intelligence 1866, pp. 209–224, Springer, 2000.
24. C. Sakama. Nonmonotonic inductive logic programming. In: *Proceedings of the 6th International Conference on Logic Programming and Nonmonotonic Reasoning*, Lecture Notes in Artificial Intelligence 2173, pp. 62–80, Springer, 2001.
25. C. Sakama. Induction from answer sets in nonmonotonic logic programs. *ACM Transactions on Computational Logic*, 6(2):203–231, 2005.