Knowledge Assimilation in Domains of Actions: A Possible Causes Approach*

Renwei Li† and **Luís Moniz Pereira**‡
Department of Computer Science, Universidade Nova de Lisboa
2825 Monte da Caparica, Portugal
{renwei | lmp }@di.fct.unl.pt

ABSTRACT. One major problem in the process of knowledge assimilation is how to deal with inconsistency of new knowledge and the existing knowledge base. In this paper we present a formal, provably correct and yet computational methodology for assimilation of new knowledge into knowledge bases about actions and changes based on the slogan: What is believed is what is explained. Technically, we employ Gelfond and Lifschitz' action description language $\mathcal{A}$ to describe domains of actions. The knowledge bases on domains of actions are defined and obtained by a new translation from domain descriptions in $\mathcal{A}$ into abductive normal logic programs, where a time dimension is incorporated. The knowledge bases are shown to be both sound and complete with respect to their domain descriptions. In particular, we propose a possible causes approach (PCA) to knowledge assimilation about evolving domains of actions, in contrast to Ginsberg's possible worlds approach (PWA) and Winslett's possible models approach (PMA). A possible cause of new knowledge consists of abduced occurrences of actions and value propositions about the initial state of the domain of actions, that would allow to derive the new knowledge. We show how to compute possible causes with abductive logic programming, and present some techniques to improve search efficiency. We use examples to compare our possible causes approach with syntax-based approaches, such as Ginsberg's possible worlds approach, and semantics-based approaches, such as Winslett's possible models approach, to belief revision/update.

Introduction

The problem of knowledge assimilation is as follows. Let an agent have a knowledge base K about a domain. Suppose that the agent somehow observes

*Published in: *Journal of Applied Non-Classical Logic*, Vol. 7, No. 1-2, 1997, pp. 77-116.
†supported by a JNICT PRAXIS post-doctoral fellowship under the grant PRAXIS XXI/BPD/4165/94.
‡supported by the JNICT PRAXIS project (PROLOPPE) under the grant PRAXIS 3/31/TIT/24/94.

or is informed of a new fact N about the domain. Then, how should and will the agent assimilate the new knowledge N into her knowledge base K? Sometimes the task of assimilating new knowledge into the existing knowledge base is also called belief revision or update or change by some authors. One major problem in the process of knowledge assimilation is how to deal with inconsistency of new knowledge and the existing knowledge base.

A representative work on belief revision is the so-called *logic of epistemic change* [AGM 85, Gar 88] developed by Alchourrón, Gädenfors, and Makinson, and thus called the AGM theory. The AGM theory starts with eight postulates, which are assumed to be the properties that every belief revision operator should satisfy. As is well-known, some belief revision operators satisfying the AGM postulates give intuitively undesirable results.

In the literature some other belief revision/update operators have also been proposed. Most of them work on a finite set of formulas, called belief base, instead of infinite belief set[1], and can be roughly classified into two types[2]: syntax-based approach , e.g. [Gin 86, Nebel 91], and semantics-based approach , e.g. [Dalal 88, Win 88]. The essential idea of syntax-based approaches to belief base revision is similar to the partial meet revision of [AGM 85]: It selects set-inclusion maximal subsets of a belief base that do not allow for the derivation of the negation of the new sentence. The semantics-based approach to belief base revision arises to avoid the syntax-sensitivity of the syntax-based belief base revision [Win 88]. Semantics-based belief base revision operators operate on models of belief bases and select models satisfying the new sentence and differing minimally from the models of the original belief base. Among all these belief revision/update operations there are two basic common criteria: The consistency should be preserved and the change should be made *minimal*. Different approaches vary with different definitions of the minimality in change.

Now consider a domain where there are two people: Tom and $Mary$, and two objects: $Money$ and $Book$. In the beginning Tom has $Book$ while $Mary$ has $Money$. This scenario can be represented by the following first-order theory KB_1, called a belief base, whose deductive closure is referred to as a belief set:

$$KB_1 = \left\{ \begin{array}{ll} has(Tom, Book), & \neg has(Tom, Money) \\ \neg has(Mary, Book), & has(Mary, Money) \end{array} \right\} \quad (1)$$

We assume that either one of the two people can buy $Book$ from the other if he/she has $Money$ and the other has $Book$; either one can give $Book$ or $Money$ to the other; either one can steal an object from the other.

Suppose that $Mary$ is found to have $Book$, represented by the following formula N_1:

$$N_1 = has(Mary, Book) \quad (2)$$

[1] A belief base is any first-order theory. A belief set is the deductive closure of a belief base. We will not make distinctions between knowledge bases and belief bases.

[2] The syntax and semantics-based approaches are also called formula and model-based approaches, respectively, by some authors, e.g. in [LS 96].

Now we want to incorporate N_1 into KB_1. Note that $KB_1 \cup N_1$ is not consistent. A possible new knowledge base KB_2 is as follows:

$$KB_2 = \left\{ \begin{array}{ll} has(Tom, Book), & \neg has(Tom, Money) \\ has(Mary, Book), & has(Mary, Money) \end{array} \right\} \tag{3}$$

Note that KB_2 can be obtained by following the syntax-based approaches[3]: KB_2 is the union of the new knowledge N_1 and a maximal subset of KB_2 which do not allow for the derivation of the negation of N_1, or by following the semantics-based approaches: KB_2 is the model satisfying the new knowledge N_1 and differing minimally from the models of KB_1 (For example, the distance between KB_1 and KB_2 can be defined to be the number of propositions which have changed their truth values. In this case, the distance is 1).

It does not seem that we should believe both Tom and $Mary$ have $Book$ at the same time. Intuitively we know that $Book$ is different from $Money$, Tom is different from $Mary$, etc. In addition, it cannot be the case that both Tom and $Mary$ have the same object, i.e. in this domain we have:

$$has(X, Book) \wedge has(Y, Book) \rightarrow X = Y \tag{4}$$

$$has(X, Money) \wedge has(Y, Money) \rightarrow X = Y \tag{5}$$

Then, a possible new knowledge base KB_3 is as follows:

$$KB_3 = \left\{ \begin{array}{ll} \neg has(Tom, Book), & \neg has(Tom, Money) \\ has(Mary, Book), & has(Mary, Money) \end{array} \right\} \tag{6}$$

Note that KB_3 satisfies both the new knowledge N_1 and the constraints (4) and (5), and differs minimally from the models of KB_1. Now KB_3 is better than KB_2 as a candidate of the new knowledge base obtained by incorporating N_1 into KB_1. Is there any other possible new knowledge bases? No more will be found by the syntax-based approach or semantics-based approach. However, in reality $Mary$ may have bought $Book$ with her $Money$, which can be regarded as a cause of the new knowledge N_1. Thus, another possible new knowledge base KB_4 is as follows:

$$KB_4 = \left\{ \begin{array}{ll} \neg has(Tom, Book), & has(Tom, Money) \\ has(Mary, Book), & \neg has(Mary, Money) \end{array} \right\} \tag{7}$$

Among KB_3 and KB_4, we have no reasons for preference of one over the other. Thus, both KB_3 and KB_4 should be possible new knowledge bases obtained by incorporating N_1 into KB_1.

The above example can be made slightly more complicated by adding a time dimension. After a time dimension is added, knowledge becomes temporal. For example, suppose that on August 1, 1995 Tom has $Book$ while $Mary$ has

[3] The reader familiar with syntax or semantics-based approaches is encouraged to revise the knowledge base $\{P_1, \neg P_2, \neg P_3, P_4\}$ with the new knowledge P_3.

Money. And on August 3, *Mary* is found to have *Book*. Then, the knowledge base and the new knowledge can be represented by meta-predicates as follows:

$$KB_5 = \left\{ \begin{array}{l} holds(has(Tom, Book), 1995/8/1) \\ \neg holds(has(Tom, Money), 1995/8/1) \\ \neg holds(has(Mary, Book), 1995/8/1) \\ holds(has(Mary, Money), 1995/8/1) \end{array} \right\} \quad (8)$$

$$N_2 = holds(has(Mary, Book), 1995/8/3) \quad (9)$$

For time-stamped knowledge in KB_5, the following so-called common-sense inertia law is used: temporal facts will persist unless stated otherwise. By this inertia law we can infer, for example, that *Mary* has *Money* on August 2, 1995 since she has it on August 1, 1995. Repeating the use of the inertia law we can have the following knowledge about temporal facts on August 3, 1995:

$$KB_6 = \left\{ \begin{array}{l} holds(has(Tom, Book), 1995/8/3) \\ \neg holds(has(Tom, Money), 1995/8/3) \\ \neg holds(has(Mary, Book), 1995/8/3) \\ holds(has(Mary, Money), 1995/8/3) \end{array} \right\} \quad (10)$$

Now we have to deal with inconsistency between KB_6 and N_2, since one asserts that *Mary* does not have *Book* and the other asserts that *Mary* has *Book*. Then, similar conclusions can be drawn. That is, there are two possible knowledge bases KB_7 and KB_8 about temporal facts on August 3:

$$KB_7 = \left\{ \begin{array}{l} \neg holds(has(Tom, Book), 1995/8/3) \\ \neg holds(has(Tom, Money), 1995/8/3) \\ holds(has(Mary, Book), 1995/8/3) \\ holds(has(Mary, Money), 1995/8/3) \end{array} \right\} \quad (11)$$

$$KB_8 = \left\{ \begin{array}{l} \neg holds(has(Tom, Book), 1995/8/3) \\ holds(has(Tom, Money), 1995/8/3) \\ holds(has(Mary, Book), 1995/8/3) \\ \neg holds(has(Mary, Money), 1995/8/3) \end{array} \right\} \quad (12)$$

From the above example it can be seen that the two basic criteria, i.e. consistency preservation and minimal change, are not sufficient for knowledge assimilation in domains such as the above one. In many practical domains, there is often background knowledge, which should play some role in the process of knowledge assimilation, but has not been directly taken into account by either of syntax and semantics-based approaches. For example, in the above *Tom*-*Mary* domain, the background knowledge is a causal theory which characterizes relationships between changes in fluents and occurrences of actions. Occurrences of actions lead to changes in truth of fluents, and truth of fluents can only be changed by doing actions. It seems that the essential idea of existing belief revision/update operations could be summarized in a single sentence: what is believed is what is similar (to what has been believed). In the semantics-based approach, selecting models which satisfy the new knowledge and differ minimally from the models of the original knowledge base amounts to selecting models which satisfy the new knowledge and are maximally similar to the models of the original knowledge base. In this paper we propose another slogan for knowledge assimilation: what is believed is what is explained. Before a new fact N is added to the knowledge base K, the agent should look for all causes of N. If the agent admits a cause C of N, then the agent should also add all the effects of C besides N. Hence, our approach will be called *the possible causes approach* (PCA) to knowledge assimilation, in contrast with Ginsberg's possible worlds approach [Gin 86] and Winslett's possible models approach [Win 88].

Conceptually, the possible causes approach is very different from the above mentioned approaches. In the AGM theory, for example, if $K \cup N$ is consistent, then the new belief set will simply be $Cn(K \cup N)$. In our approach, however, if the cause C of N has another effect, say X, then the agent should also add X to her knowledge base. The PCA is especially suitable to domains of actions, where there is a clear cause-effect relation. In domains of actions, new knowledge is about changes in truth values of fluents. The possible causes for changes in truth values of fluents are the occurrences of actions. The process of looking for causes of a new fact can be done only when background knowledge about the cause-effect relations are given. In the rest of this paper we will elaborate on this idea. We will start with domain descriptions which describe the initial state of a domain under consideration and relations among actions and changes. Domain descriptions are employed not only for computations of causes of a new fact but also for construction of temporal databases which record historical information about the domain.

The rest of this paper is organized as follows: In Section 1 we review an action description language $\mathcal{A}$ proposed by Gelfond and Lifschitz [GL 93] for descriptions of action domains. Although we use $\mathcal{A}$ for domain descriptions, the ideas of this paper still apply to other languages to describe the cause-effect relations. In Section 2 we present a new transformation π from domain descriptions in $\mathcal{A}$ to abductive logic programs. The translation is different from all other translations in [GL 93, Den 93, Dung 93, BG 93, LP 96B] in that a real time dimension is incorporated so that one can represent and reason about narratives. In Section 3 we use the translation π to represent temporal knowledge bases. The temporal facts, stamped with time points, will be represented by meta-predicates. In this section, we will also prove the soundness and completeness theorems for the representation. In Section 4 we discuss how to incorporate new knowledge into a knowledge base. Before assimilating, the agent must look for all possible causes for the new knowledge. Then, the new knowledge base is defined to be the union of the old one and all the effects of the causes for the new knowledge. In this section we also consider some aspects of implementation. In Section 5 we compare our work with other belief revision/update operations, namely, possible worlds approach PWA [Gin 86] and possible models approach PMA [Win 88]. In the final section we conclude this paper with a summary and some remarks. A short version of this paper was published in [LP 96A].

1 Domain description

In this section we will briefly introduce Gelfond and Lifschitz' action description language $\mathcal{A}$ [GL 93] to describe domains of actions. Domain descriptions will act as cause-effect constraints in our methodology for knowledge representation and assimilation.

1.1 Syntax

We begin with the alphabet of a domain. The alphabet Σ of a domain consists of two disjoint non-empty sets of symbols Σ_f and Σ_a, called *fluent names* and *action names*, respectively. Later we will also use parameterized fluent names and action names for convenience. It is of no theoretical interest to introduce sorts to detail the definitions of parameterized fluent names and action names. The fluent names correspond to tuples, and parameterized fluent names correspond to relation schemas in relational databases. We will use fluent names to represent temporal facts. Temporal facts may change with occurrences of actions.

An *action expression* is defined to be an action name. A *fluent expression* is defined to be a fluent name possibly preceded by $\neg$. A fluent expression is also called a *positive fluent* if it only consists of a fluent name; otherwise it is called a *negative fluent*. Fluent expressions and action expressions will be simply called fluents and actions, respectively, if no confusion arises.

In $\mathcal{A}$ there are two kinds of propositions: value propositions and effect propositions, simply called v-propositions, e-propositions.

A v-proposition is defined to be a statement of the form

$$F \textbf{ after } A_1; \ldots; A_m \tag{13}$$

where F is a fluent expression and $A_1, \ldots, A_m (m \geq 0)$ are action expressions. If $m = 0$, then we will write it as

$$\textbf{initially } F \tag{14}$$

In this paper we will only use v-propositions of the form (14). An e-proposition is defined to be a statement of the form

$$A \textbf{ causes } F \textbf{ if } P_1, \ldots, P_n \tag{15}$$

where A is an action expression, and each of $F, P_1, \ldots, P_n (n \geq 0)$ is a fluent expression. If $n = 0$, then we will write it as

$$A \textbf{ causes } F \tag{16}$$

Given the alphabet of a domain, its description D is a set of v-propositions and e-propositions.

Example 1.1 (Stolen Car Problem) *In the Stolen Car Problem (SCP) domain, the alphabet Σ_f of fluent names only includes $Stolen$, the alphabet Σ_a of action names includes $Wait$ and $Steal$. The domain description D_{scp} includes the following propositions:*

$$\textbf{initially } \neg Stolen$$
$$Steal \textbf{ causes } Stolen \textbf{ if } \neg Stolen$$

Example 1.2 (Tom-Mary) *In the Tom-Mary domain, the alphabet Σ_f of fluent names is as follows:*

$$\Sigma_f = \{has(X,Y) : X \in \{Tom, Mary\}, Y \in \{Money, Book\}\}$$

And the alphabet Σ_a of action names includes the following types of actions:

$give(X,Y,Z)$	—	X *gives* Y *to* Z
$buy(X,Y,Z)$	—	X *buys* Y *from* Z
$steal(X,Y,Z)$	—	X *steals* Y *from* Z

Then, this domain is described by the following domain description D_1:

$$\begin{array}{l}
\textbf{initially } has(Tom, Book), has(Mary, Money) \\
\textbf{initially } \neg has(Mary, Book), \neg has(Tom, Money) \\
give(X,Y,Z) \textbf{ causes } has(Z,Y) \textbf{ if } has(X,Y) \\
give(X,Y,Z) \textbf{ causes } \neg has(X,Y) \\
steal(X,Y,Z) \textbf{ causes } has(X,Y) \textbf{ if } has(Z,Y) \\
steal(X,Y,Z) \textbf{ causes } \neg has(Z,Y) \\
buy(X,Y,Z) \textbf{ causes } has(X,Y) \textbf{ if } has(X,Money), has(Z,Y) \\
buy(X,Y,Z) \textbf{ causes } has(Z,Money) \textbf{ if } has(X,Money), has(Z,Y) \\
buy(X,Y,Z) \textbf{ causes } \neg has(Z,Y) \textbf{ if } has(X,Money), has(Z,Y) \\
buy(X,Y,Z) \textbf{ causes } \neg has(X,Money) \textbf{ if } has(Z,Y)
\end{array}$$

The semantics of domain descriptions will be discussed in the following subsection.

1.2 Semantics

The semantics of $\mathcal{A}$ is defined by using states and transitions. A *state* is a set of fluent names. Given a fluent name F and a state σ, we say that F holds in σ if $F \in \sigma$; $\neg F$ holds in σ if $F \notin \sigma$. A *transition function* Φ is a mapping from the set of pairs (A, σ), where A is an action expression and σ is a state, into the set of states.

An *interpretation structure* is a pair (σ_0, Φ), where σ_0 is a state, called the *initial state* of the interpretation structure , and Φ is a transition function.

For any interpretation structure $M = (\sigma_0, \Phi)$ and any sequence of action expressions $A_1; \ldots; A_m$ in M, by $\Phi(A_1; \ldots; A_m, \sigma_0)$ we denote the state

$$\Phi(A_m, \Phi(A_{m-1}, \ldots, \Phi(A_1, \sigma_0) \ldots))$$

A v-proposition of the form (13) is true in an interpretation structure $M = (\sigma_0, \Phi)$ iff F holds in the state $\Phi(A_1; \ldots; A_m, \sigma_0)$. In particular, the v-proposition **initially** F is true in M iff F holds in the initial state σ_0.

An interpretation structure (σ_0, Φ) is a *model* of a domain description D iff every v-proposition of D is true in (σ_0, Φ), and for every action expression A, every fluent name F and every state σ, the following conditions are satisfied:

- If D includes an e-proposition A **causes** F **if** $P_1, \ldots, P_n$, whose preconditions $P_1, \ldots, P_n$ hold in σ, then $F \in \Phi(A, \sigma)$.
- If D includes an e-proposition A **causes** $\neg F$ **if** $P_1, \ldots, P_n$, whose preconditions $P_1, \ldots, P_n$ hold in σ, then $F \notin \Phi(A, \sigma)$.
- If D does not include any such e-propositions, then $F \in \Phi(A, \sigma)$ iff $F \in \sigma$.

Gelfond and Lifschitz [GL 93] observed that there can be at most one transition function Φ satisfying the above three conditions.

A v-proposition is *entailed* by a domain description D if it is true in every model of D. A domain description is *consistent* if it has a model. A domain description is *complete* if it has exactly one model. We will use $Mod(D)$ to denote the set of all models of D.

In this paper we will restrict our domain descriptions to sets of effect propositions and the value proposition of the form **initially** F. Furthermore, we will write

initially $F_1, F_2, \ldots, F_m$

to abbreviate a set of value propositions

initially F_1
initially F_2
...
initially F_m

When there is no confusion, by saying domain descriptions in $\mathcal{A}$ we mean domain descriptions in $\mathcal{A}$ as restricted as above. In Section 3 we will also use the following lemma.

Lemma 1.3 *Let $M = (\sigma_0, \Phi)$ be a model of a domain description D. Then for any two effect propositions of the forms*

A **causes** $\neg F$ **if** $P_1, \ldots, P_n$
A **causes** F **if** $Q_1, \ldots, Q_k$

in D, for any state $\sigma = \Phi(A_1; \ldots; A_m, \sigma_0)$, if $\{P_1, \ldots, P_n\} \subseteq \sigma$, then $\{Q_1, \ldots, Q_k\} \not\subseteq \sigma$, and if $\{Q_1, \ldots, Q_k\} \subseteq \sigma$, then $\{P_1, \ldots, P_n\} \not\subseteq \sigma$.

Proof: Straightforward. ■

Before we proceed on, we should make it clear that we will assume that (a) the information on the initial state of a domain need not be complete, but Stanford Murder Mystery [Baker 91] where information on the initial state is not available. Accomodating incomplete initial states will enhance the capability of our methodology. The second assumption is actually a kind of closed-world assumption (CWA) widely adopted in Artificial Intelligence and Databases. For example, consider a factory where all workers are robots. Effectors of all robots should be completely known, and thus CWA holds in such factories. Certainly there are domains where CWA does not hold. Open worlds are out of the scope of this paper.

2 From domain descriptions to logic programs

There have been several translations of domain descriptions in $\mathcal{A}$-like languages into logic programs [GL 93, Den 93, Dung 93, BG 93, LP 96B]. In these translations, the state transition is captured by a function of the form $Result(A, S)$ which denotes the new situation resulting from doing A in S. These translations are appropriate for proving branching-pace properties of domains of actions, but not suitable for problems where we are concerned about the real time (whether a property is true of a time point, instead of a situation or state). Note that a situation of the form $Result(A, S)$ actually represents a time period during which no changes occur.

In the following we present a new translation π from domain descriptions in $\mathcal{A}$ to abductive logic programs. The translation is different from all other translations in [GL 93, Den 93, Dung 93, BG 93, LP 96B] in that a real time dimension is incorporated so that one can represent and reason about actions and times, or narratives. The reader who is not familiar with abductive logic programming is referred to [KKT 93].

Let D be a domain description, the translation πD consists of the following logic programming rules:

1. Time dimension:

   For simplicity, we assume that time is structured on points, represented by linearly ordered set $(TP, \prec, succ, init)$, where TP is an infinite set of time points and $init \in TP$ and $succ(T) \in TP$ for any $T \in TP$. The time point $init$ acts as the starting time point from which we are interested in changes of fluents. Intuitively $succ(T)$ represents the next time point of T. We assume that $(TP, \prec, succ, init)$ is formulated as a set of logic programming rules. The following simple programming rules are sufficient for the purpose of this paper:

$$tp(init) \leftarrow \tag{17}$$

$$tp(succ(T)) \leftarrow tp(T) \tag{18}$$

$$init \prec succ(T) \leftarrow tp(T) \tag{19}$$

$$T \prec succ(T) \quad \leftarrow \quad tp(T) \tag{20}$$
$$T1 \prec T2 \quad \leftarrow \quad tp(T_1), tp(T_2), T1 \prec Ts, Ts \prec T2 \tag{21}$$

where the predicate $tp(T)$ means that T is a time point. The reader may add as many axioms in order to facilitate the execution of the program, provided that there is an initial time point and time points are linearly ordered. In our examples, we will often use natural numbers which have a straightforward correspondence to terms of the form $succ(T)$: $init$ corresponds to 0, $succ(init)$ to 1, etc. In the following examples we will often omit logic programming rules about the time dimension. We will assume that it will take a time unit to perform an action. This assumption implies that our notion of time is rather a kind of abstract "system time" than "real time".

2. Initialization:

$$holds(F, init) \leftarrow initially(F) \tag{22}$$

We will take the predicate $initially(F)$ as an abducible one. This logic programming rule says that a fluent F is true in the initial state if it is abduced to be initially true. The abduction is certainly constrained by the value propositions of D. The translation of value propositions will be presented later.

3. Law of Inertia:

$$\begin{aligned} holds(F, succ(T)) \quad \leftarrow \quad & holds(F, T), \\ & not\ noninertial(F, T) \end{aligned} \tag{23}$$

A fluent F is true of a time point T if it was true and inertial at the last time point.

4. For each effect proposition a **causes** f **if** $p_1, \ldots, p_n$, where f is a positive literal:

$$\begin{aligned} holds(f, succ(T)) \quad \leftarrow \quad & happens(a, T), \\ & holds(p_1, T), \ldots, holds(p_n, T) \end{aligned} \tag{24}$$

5. For each effect proposition a **causes** $\neg f$ **if** $p_1, \ldots, p_n$, where f is a positive literal:

$$\begin{aligned} noninertial(f, T) \quad \leftarrow \quad & happens(a, T), \\ & holds(p_1, T), \ldots, holds(p_n, T) \end{aligned} \tag{25}$$

The predicate $noninertial(f, T)$ can disable the applicability of the law of inertia (23).

6. For each value proposition **initially** f, where f is a positive literal, we have
$$\perp \;\leftarrow not\ holds(f, init) \tag{26}$$
where $\perp$ is regarded as a special literal, and is always interpreted to be logical falsity, which is never true in any model. We should point out that (26) can't be simply represented as
$$holds(f, init) \leftarrow$$
since by predicate completion semantics it will exclude the possibility that other fluents may be true in the initial state. The reader may consult (28) and (30) in Section 3 for the technical reason.

7. For each value proposition **initially** $\neg f$, where f is a positive literal, we have
$$\perp \;\leftarrow holds(f, init) \tag{27}$$
The logic programming rules (26) and (27) function as integrity constraints. The constraint (26) says that $holds(f, init)$ must be true in every model of the logic program, while (27) says that $holds(f, init)$ must be false in every model of the logic program.

Note that the logic program πD has a built-in time dimension. In the foregoing section we will use the above translation for representation of temporal knowledge bases and prove the soundness and completeness related to the above translation π.

Example 2.1 *Consider the Tom-Marry example of Example 1.2. The translation* πD_1 *is as follows:*

$$\begin{array}{rcl}
holds(F, init) & \leftarrow & initially(F) \\
holds(F, succ(T)) & \leftarrow & holds(F, T), \\
 & & not\ noninertial(F, T) \\
holds(has(Z, Y), succ(T)) & \leftarrow & happens(give(X, Y, Z), T), \\
 & & holds(has(X, Y), T) \\
holds(has(X, Y), succ(T)) & \leftarrow & happens(steal(X, Y, Z), T), \\
 & & holds(has(Z, Y), T) \\
holds(has(X, Y), succ(T)) & \leftarrow & happens(buy(X, Y, Z), T), \\
 & & holds(has(X, Money), T), \\
 & & holds(has(Z, Y), T) \\
holds(has(Z, Money), succ(T)) & \leftarrow & happens(buy(X, Y, Z), T), \\
 & & holds(has(X, Money), T), \\
 & & holds(has(Z, Y), T)
\end{array}$$

$$
\begin{array}{rcl}
noninertial(has(X,Y),T) & \leftarrow & happens(give(X,Y,Z),T), \\
 & & holds(has(X,Y),T) \\
noninertial(has(Z,Y),T) & \leftarrow & happens(steal(X,Y,Z),T) \\
noninertial(has(Z,Y),T) & \leftarrow & happens(buy(X,Y,Z),T), \\
 & & holds(has(X,Money),T), \\
 & & holds(has(Z,Y),T) \\
noninertial(has(X,Money),T) & \leftarrow & happens(buy(X,Y,Z),T), \\
 & & holds(has(Z,Y),T) \\
\perp & \leftarrow & not\ holds(has(Tom,Book),init) \\
\perp & \leftarrow & not\ holds(has(Mary,Money),init) \\
\perp & \leftarrow & holds(has(Mary,Book),init) \\
\perp & \leftarrow & holds(has(Tom,Money),init)
\end{array}
$$

3 Representation of temporal knowledge

3.1 Domain evolutions and temporal knowledge bases

In practice a domain is always evolving: Actions are performed in real time and thus fluents are changed in real time. A temporal knowledge base is a collection of temporal facts or (some kinds of) rules of inference, from which we can determine at which time points which fluents are true.

We start with the concept of *history*, then employ abductive logic programming for the representation of temporal knowledge.

A domain history H is a subset of $\Sigma_a \times TP$, where Σ_a is the alphabet of action names and TP the set of time points . By an element $(a,t) \in H$ we mean that the action a happens at the time point t.

Note that for any $t \in TP$, there might be $a_1, \ldots, a_m$ such that $(a_1,t) \in H, \ldots, (a_m,t) \in H$. In this case we consider $\{a_1, \ldots, a_m\}$ as a concurrent action. In this paper we will not consider concurrent actions. Thus we assume that

For any $t \in TP$ there is at most one action a such that $(a,t) \in H$

We should point out that the results of this paper can be easily generalized for concurrent actions, if some additional means, for example, the inheritance principle of [BG 93], are adopted to deal with effects of concurrent actions.

A domain evolution E is defined to be a pair (D,H), where D is a domain description and H a history.

Given a domain evolution (D,H), we can have an abductive logic program $KB(D,H)$ defined as follows:

$$KB(D,H) = \pi D \cup \{\ \perp\ \leftarrow not\ happens(a,t) : (a,t) \in H\}$$

with abducibles $initially(F)$ for fluent F.

The logic program $KB(D,H)$ will be simply called *the knowledge base generated from* D *and* H. The literal $false$ is always interpreted as logical falsity, and all rules with $false$ as heads function as integrity constraints. The logic program $KB(D,H)$ looks similar to the event calculus [KS 86, SK 95], but it is technically different from the event calculus. We will not deviate into comparisons between the logic program $KB(D,H)$ and the event calculus. The predicate $initially(F)$ is taken as an abducible predicate used to capture the incomplete knowledge about the initial situation/time. Following Denecker [Den 93], we will define the semantics of an abductive normal logic program with integrity constraints to be a first-order theory which consists of the integrity constraints and the first-order theory obtained by completing the non-abducible predicates together with Clark's Equality Theory (CET). We will write $COMP(KB(D,H))$ to denote the predicate completion semantics of the logic program $KB(D,H)$. We will also write $KB(D,H) \models L$ to stand for $COMP(KB(D,H)) \models L$. For all the acyclic logic programs [AB 90], the predicate completion semantics coincide with other semantics such as the stable model semantics [GL 88] and the well-founded model semantics [vGRS 91]. In the case of abductive logic programs, the semantical coincidence still holds [Den 93].

Proposition 3.1 *Let D be any domain descriptio, and H any history. The logic program $KB(D,H)$ is acyclic in the sense of [AB 90].*

Proof: It suffices to give a level mapping for predicates in $KB(D,H)$. The level mapping can be given by adapting those in [AB 90, Den 93, LP 96C]. ■

Corollary 3.2 *Let D be any domain descriptio, and H any history. For the knowledge base $KB(D,H)$, the completion semantics $COMP(KB(D,H))$ coincides with its generalized stable model semantics [KakM 90] and generalized well-founded model semantics [PAA 91].*

3.2 Soundness and completeness of knowledge bases

Note that $\mathcal{A}$, as the situation calculus [MH 69], has an infinite tree-like time structure. Given a history H, the translation π only takes into account a special path in the tree corresponding to H. In order for π be correct, we require that π be correct with respect to any history. In this section we will prove the soundness and completeness theorems about the correctness of the translation π.

Given a history H, we can always have a unique sequence of actions

$$A_1; A_2; \ldots; A_m$$

and a sequence of time points

$$T_1 \prec T_2 \prec \ldots \prec T_m$$

such that $H = \{(A_i, T_i) : 1 \leq i \leq m\}$. The sequence of actions $A_1; A_2; \ldots; A_m$ will be called *the trajectory generated from* H, *denoted by* $Traj(H)$. A pre-history of H is defined to be a subset H' of H such that $Traj(H')$ is a prefix of $Traj(H)$.

Definition 3.3 (Generated States) *Given an interpretation structure* $M = (\sigma_0, \Phi)$, *a history* H, *and a time point* T, *we can derive a state* S *as follows: if* $H' = \{(A_1, T_1), \ldots, (A_k, T_k)\}$, *with* $T_1 \prec \ldots \prec T_k$, *is a pre-history of* H *such that* $T_k \prec T$ *and there is no time point* Tx *and action* Ax *with* $T_k \prec Tx \prec T$ *and* $(Ax, Tx) \in H$, *then* $S = \Phi(A_1; A_2; \ldots; A_k, \sigma_0)$ *is called the state generated from* H *and* T *in* M, *denoted by* $GS_M(H, T)$.

If T *is a time point such that* $(A, T) \notin H$ *for any action* A, *and* $(A, Tx) \notin H$ *for any action* A *and any time point* Tx *with* $T \prec Tx$, *then the state* $GS_M(H, T)$ *is called the state generated from* H *in* M, *denoted by* $GS_M(H)$.

For example, let $M = (\sigma_0, \Phi)$ be an interpretation structure , and a history

$$H = \{(a, 1), (a, 6), (b, 4), (c, 10)\}$$

Then, $GS_M(H, 5) = \Phi(a; b, \sigma_0)$ and $GS_M(H) = \Phi(a; b; a; c, \sigma_0)$.

From the above definition, it is easy to see that $GS_M(H, init) = \sigma_0$ for any interpretation structure M and history H. Note that the states generated from a history corresponds to nodes in the path picked out by the history.

Definition 3.4 *Let* D *be a domain description, and* H *a history. The logic program* $KB(D, H)$ *is said to be sound with respect to* H *iff the following condition is satisfied: For every model* $M = (\sigma_0, \Phi)$ *of* D, *every fluent* F *and time point* T, *if* $COMP(KB(D, H)) \models holds(F, T)$, *then* $F \in GS_M(H, T)$.

The translation π *is said to be sound iff for every history* H', $KB(D, H')$ *is sound with respect to* H'.

Definition 3.5 *Let* D *be a domain description, and* H *a history. The logic program* $KB(D, H)$ *is said to be complete with respect to* H *iff the following condition is satisfied: For every fluent* F, *and time point* T, *if* $F \in GS_M(H, T)$ *for every model* M *of* D, *then* $COMP(KB(D, H)) \models holds(F, T)$.

The translation π *is said to be complete iff for every history* H', $KB(D, H')$ *is complete with respect to* H'.

Let (D, H) be a domain evolution. The semantics $COMP(KB(D, H))$ of $KB(D, H)$ is a first-order theory which includes the following equivalences in addition to Clark Equality Theory CET:

The completed $holds(F, T)$ predicate:

$$\forall F, T : holds(F, T) \leftrightarrow E_1 \vee E_2 \vee E_3 \vee \ldots \vee E_m \tag{28}$$

where

$$
\begin{aligned}
E_1 &\equiv T = init \wedge initially(F) \\
E_2 &\equiv \exists S : T = succ(S) \wedge \\
&\quad\; holds(F,S) \wedge \neg noninertial(F,S) \\
E_i &\equiv \exists S : F = f \wedge T = succ(S) \\
&\quad\; happens(a,S) \wedge \\
&\quad\; holds(p_1,S) \wedge \ldots \wedge holds(p_n,S)
\end{aligned}
$$

such that there is an exact correspondence between all disjuncts $E_i, i \geq 3$ and all effect propositions of the form a **causes** f **if** $p_1, \ldots, p_n$ with f being a positive fluent.

The completed $noninertial(F,T)$ predicate:

$$\forall F, T : noninertial(F,T) \leftrightarrow \bigvee_i E_i \tag{29}$$

where

$$
\begin{aligned}
E_i \equiv\;& F = f \wedge happens(a,T) \wedge \\
& holds(p_1,T) \wedge \ldots \wedge holds(p_n,T)
\end{aligned}
$$

such that there is an exact correspondence between all the disjuncts E_i and all the effect propositions of the form a **causes** $\neg f$ **if** $p_1, \ldots, p_n$ with f being a positive fluent.

The completed $\perp$:

$$\forall F, T : \perp \leftrightarrow (\bigvee_i E_i) \vee (\bigvee_j F_j) \vee (\bigvee_k G_k) \tag{30}$$

where

$$E_i \equiv \neg holds(f, init)$$

such that there is an exact correspondence between all the disjuncts E_i and all the value proposition of the form **initially** f with f being a positive fluent, and

$$F_j \equiv holds(f, init)$$

such that there is an exact correspondence between all the disjuncts F_j and all the value propositions of the form **initially** $\neg f$ with f being a positive fluent, and

$$G_k \equiv \neg happens(a,t)$$

such that there is an exact correspondence between the disjunct G_k and an element (a,t) of the history H.

Since $\perp$ is always interpreted as logical falsity and each disjunct is a literal, the equivalence (30) is reduced to a set of formulas of the form

$$holds(f, init) \tag{31}$$

such that there is an exact correspondence between them and all the value propositions of the form **initially** f with f being a positive fluent, and

$$\neg holds(f, init) \tag{32}$$

such that there is an exact correspondence between them and all the value propositions of the form **initially** $\neg f$ with f being a positive fluent, and

$$happens(a, t) \tag{33}$$

such that there is an exact correspondence between the disjunct G_k and an element of the history H.

An interpretation structure (U, M) for a first-order theory consists of a universe U and a mapping M such that (i) for any constant v, $M(v) \in U$; (ii) for any n-ary function symbol f, $M(f)$ is an n-ary function on U; (iii) for any n-ary predicate P, $M(P)$ is an n-ary relation on U. When no confusion arises, we often write M for the interpretation structure (U, M). An assignment is a mapping from variables to universe U. Given an interpretation structure and an assignment, a first-order formula can be interpreted to be *true* or *false*, called *truth values*, as usual. The truth values of any first-order sentence (formula without free variables) does not depend on assignments. Given a first-order theory, an interpretation structure is called its *model* if every formula of the theory is interpreted to be true. In the following proofs we will use the following abused notations without confusion: Let M be any interpretation structure for the first-order theory $COMP(KB(D, H))$.

- Let F be any sentence. We will write $M \models F$ to denote F is true in M.
- Let f and s be any two elements of the universe of M. Then, we write $M \models holds(f, s)$ to denote $\langle f, s\rangle \in M(holds)$, where $M(holds)$ is a binary relation on U as the interpretation of the predicate $holds$.
- Let f be a fluent symbol, and s any element of the universe of M, we write $M \models holds(f, s)$ to denote $\langle M(f), s\rangle \in M(holds)$, where $M(f)$ is the interpretation of the fluent symbol f.

Given a model M of $COMP(KB(D, H))$, if $M \models holds(F, T)$, then we say that F is true at time point T in M.

Definition 3.6 *Let M be a model of $COMP(KB(D, H))$, and S any element of the universe of M. Define*

$$state_M(S) = \{F : F \textit{ is a fluent symbol and } M \models holds(F, S)\}$$

Notice that $state_M(S)$ denotes all fluents interpreted to be true at some time point T of M, and thus corresponds to a generated state from a history and a time point in an interpretation structure for a domain description D. Hence we can define a transition function Φ on it.

Definition 3.7 (Non-inertial) *Given an interpretation structure M of the first-order theory $COMP(KB(D,H))$, a fluent F is said to be non-inertial at a time point T in M iff there exists an effect proposition*

$$A \textbf{ causes } \neg F \textbf{ if } P_1, \ldots, P_n$$

with F being a positive literal, such that $M \models holds(P_1,T), \ldots, holds(P_n,T)$ and $(A,T) \in H$, otherwise it is said to be inertial at time point T in M.

The following is the main lemma for the soundness theorem of the translation π.

Lemma 3.8 *Let D be a consistent domain description, H a history, F a fluent, and T a time point. Then, for any model $I = (\sigma_0, \Phi)$ of D, there is a model M of $COMP(KB(D,H))$ such that $F \in GS_I(H,T)$ iff $M \models holds(F,T)$.*

Proof: Given a model $I = (\sigma_0, \Phi)$ of D, we want to construct a Herbrand model M of the first-order theory $COMP(KB(D,H))$ such that $F \in GS_I(H,T)$ iff $M \models holds(F,T)$.

Let U be the Herbrand universe. We will be particularly interested in two subsets of U, called *actions* and *points*, respectively, defined as follows:

- all action expressions are actions;
- *init* is a point;
- $succ(T)$ is a point whenever T is a point.

Points will also be called time points if no confusion arises. Notice that there are some elements in U which are neither actions nor time points. For example, $succ(A)$ for action A also appears in U, since U is the Herbrand universe, but it is neither an action nor a time point.

The basic idea of the following proof is as follows: We define M such that for any time point $T = succ(succ(\ldots succ(init)\ldots))$, we have

- $state_M(T) = GS_I(H,T)$
- $noninertial(F,T)$ is true whenever F is non-inertial at time point T.

Now we proceed to construct a Herbrand model M. Define $state_M(T)$ for any element $T \in U$ as follows:
(i) If $T = init$, then

$$state_M(init) = \sigma_0$$

(ii) If $T = succ(S)$ for some point S, then

$$state_M(T) = GS_I(H,T)$$

(iii) If T is not a point in U, then

$$state_M(T) = \emptyset$$

Then, M is defined as follows:

$$\begin{aligned}
M(initially) &\equiv \{\langle F\rangle : F \in \sigma_0\}\\
M(happens) &\equiv \{\langle A,T\rangle : (A,T) \in H\}\\
M(noninertial) &\equiv \{\langle F,T\rangle :\ F \text{ is non-inertial at } T\ \}\\
M(holds) &\equiv \{\langle F,T\rangle : F \in state_M(T) \text{ for all } F \text{ and } T \text{ of } U\ \}
\end{aligned}$$

Now we want to show

1. For any fluent F and any time T, $F \in GS_I(H,T)$ iff $M \models holds(F,T)$.
2. M is a model of $COMP(KB(D,H))$.

To prove 1: Let $GS_I(H,T) = \Phi(A_1; A_2; \ldots; A_k, \sigma_0)$, which implies that there is a pre-history $H' = \{(A_1,T_1),\ldots,(A_k,T_k)\}$ of the history H such that $T_k \prec T$ and there is no time point Tx and action Ax with $T_k \prec Tx \prec T$ and $(Ax,Tx) \in H$. Then, we will have

$$\begin{aligned}
F \in GS_I(H,T) &\Leftrightarrow F \in state_M(T)\\
&\Leftrightarrow \langle F,T\rangle \in M(holds)\\
&\Leftrightarrow M \models holds(F,T)
\end{aligned}$$

To prove 2: It suffices to prove that (28), (29), and (30) are all satisfied in M.

The proof for (30) is straightforward by using the previous result, since the equation (30) is equivalently reduced to a set of formulas of the form (31), (32), and (33).

Notice that the expression "P is true at T" is equivalent to $\langle P,T\rangle \in M(holds)$. By definition we also have $(A,T) \in H$ is equivalent to $\langle A,T\rangle \in M(happens)$. Thus, the completed definition (29) of $noninertial(F,T)$ is a rephrased representation of Definition 3.7. Thus, (29) is satisfied in M.

Now it remains to show that the completed definition (28) of $holds(F,T)$ is satisfied in M. For this purpose we need to check all the ground instances of (28). On the right-hand side of (28) there are three types of disjuncts. We will analyze them one by one.

- T is not a point in M.

  Notice that $state_M(T) = \emptyset$ by definition. On the other hand, the right-hand side of (28) collapses to $\perp$. Thus, (28) is satisfied in M.

- $T = init$.

  The completed definition (28) of $holds(F,T)$ collapses to

  $$holds(F, init) \leftrightarrow initially(F)$$

  Since $state_M(init) = \sigma_0$ and $M(initially) = \{\langle F\rangle : F \in \sigma_0\}$, the above equivalence is trivially satisfied in M.

- $T = succ(S)$ for some some point S. By assumption on the history H we know that at any time point there is at most one action which happens. That is, no concurrent actions are allowed to happen. Thus, the the completed definition (28) of $holds(F, T)$ collapses to:

$$\forall F, T : holds(F, T) \leftrightarrow E_2 \vee E_3 \vee \ldots \vee E_m$$

  where

$$\begin{aligned} E_2 &\equiv holds(F, S) \wedge \neg noninertial(F, S) \\ E_i &\equiv happens(a, S) \wedge \\ &\quad\ holds(p_1, S) \wedge \ldots \wedge holds(p_n, S) \end{aligned}$$

  such that there is an exact correspondence between the disjunct $E_i, i \geq 3$ and all the effect propositions a **causes** F **if** $p_1, \ldots, p_n$ in D with F being a positive fluent. Note that either F is non-inertial at S in M, or F is not non-inertial at S in M. Now we consider the following two exclusive cases:

  – F is non-inertial at S in M. By Definition 3.7, there exists an effect proposition

$$A \textbf{ causes } \neg F \textbf{ if } P_1, \ldots, P_n$$

    such that $P_1, \ldots, P_n$ are true at S and $(A, S) \in H$. Thus, by definition we have $\langle F, S \rangle \in M(noninertial)$, which implies $M \not\models E_2$. Since $(A, S) \in H$, by definition we have $\langle A, S \rangle \in M(happens)$. Thus, $M \models happens(A, S)$. By assumption on the history H, there is no other action B such that $(B, S) \in H$. On the other hand, the domain description D is assumed to be consistent. Thus, there is no effect proposition A **causes** F **if** $Q_1, \ldots, Q_k$ such that $Q_1, \ldots, Q_k$ are true at S by Lemma 1.3. Thus, for any $i \geq 3$, $M \not\models E_i$. Thus, $M \not\models holds(F, T)$. Now it remains to verify that $\langle F, T \rangle \notin M(holds)$. By definition of $M(holds)$, it suffices to verify $F \notin state_M(T)$. Note that $F \notin state_M(T)$ iff $F \notin GS_I(H, T)$. On the other hand, $GS_I(H, T) = \Phi(A_1; A_2; \ldots; A_k, \sigma_0)$ for some pre-history $H' = \{(A_1, T_1), \ldots, (A_k, T_k)\}$ of H such that $T_k \prec T$ and there is no time point Tx and action Ax with $T_k \prec Tx \prec T$ and $(Ax, Tx) \in H$. Since $(A, S) \in H$ and $T = succ(S)$, we have $A_k = A$ and $T_k = S$. Since D is consistent and there is an effect proposition A **causes** $\neg F$ **if** $P_1, \ldots, P_n$ such that $P_1, \ldots, P_n$ are true at S, we have $P_1, \ldots, P_n \in GS_I(H, S)$ by the result 1. Since I is a model of D, then we have $F \notin \Phi(A, GS_I(H, S))$. Note that $\Phi(A, GS_I(H, S)) = \Phi(A, \Phi(A_1; A_2; \ldots; A_{k-1}, \sigma_0)) = GS_I(H, T) = state_M(T)$. Thus, $F \notin state_M(T)$, which implies that $\langle F, T \rangle \notin M(holds)$. Therefore, the equation (28) is satisfied in M.

– F is not non-inertial at S in M. Then, we will have that $M \models \neg noninertial(F, S)$. By Definition 3.7, there does not exist any effect proposition A **causes** $\neg F$ **if** $P_1, \ldots, P_n$ with F being a positive literal, such that $P_1, \ldots, P_n$ are true at S and $(A, S) \in H$. But this does not exclude the possibility that there is an action B such that $(B, S) \in H$ and there is an effect proposition B **causes** F **if** Q_1, ..., Q_k with Q_1, ..., Q_k being true at S.

  Suppose that there is an action B such that $(B, S) \in H$ and there is an effect proposition B **causes** F **if** $Q_1, \ldots, Q_k$ with $Q_1, \ldots, Q_k$ being true at S. Then, we will have that $state_M(T) = GS_I(H, T) = \Phi(B, GS_I(H, S))$. Since $Q_1, \ldots, Q_k$ are true at S, by the result 1 we have that $Q_1, \ldots, Q_k \in GS_I(H, S)$. Since I is a model of D, the effect proposition B **causes** F **if** $Q_1, \ldots, Q_k$ is satisfied in I, which makes $F \in \Phi(B, GS_I(H, S))$, i.e. $F \in state_M(T)$. Thus we have $\langle F, T\rangle \in M(holds)$. Thus we have $M \models holds(F, T)$. It remains to show that the right-hand side of (28) is also satisfied in M. Since $(B, S) \in H$ and $Q_1, \ldots, Q_k$ are true at S, we have $M \models happens(B, S)$ and $M \models holds(Q_1, S) \wedge \ldots \wedge holds(Q_k, S)$. Therefore (28) is satisfied.

  Suppose that there is not any action B such that $(B, S) \in H$ and there is an effect proposition B **causes** F **if** $Q_1, \ldots, Q_k$ with $Q_1, \ldots, Q_k$ being true at S. Then, for any effect proposition B **causes** F **if** $Q_1, \ldots, Q_k$ in D, we have $M \not\models happens(B, S) \wedge holds(Q_1, S) \wedge \ldots \wedge holds(Q_k, S)$. Thus (28) is reduced to

$$\forall F, T : holds(F, T) \leftrightarrow holds(F, S) \wedge \neg noninertial(F, S)$$

  Since F is not non-inertial at S, we have $M \models \neg noninertial(F, S)$. By using the facts that F is not non-inertial at S and there is not any action B such that $(B, S) \in H$ and there is an effect proposition B **causes** F **if** $Q_1, \ldots, Q_k$ with $Q_1, \ldots, Q_k$ being true at S, it is straightforward to prove that $F \in GS_I(H, S)$ iff $F \in GS_I(H, T)$. Thus we have $F \in state_M(S)$ iff $F \in state_M(T)$, which implies $M \models holds(F, S)$ iff $M \models holds(F, T)$. Thus, (28) is satisfied. ■

Theorem 3.9 (Soundness Theorem) *Let D be a domain description. For any history H, every model $I = (\sigma_0, \Phi)$ of D, every fluent F and time point T, if $COMP(KB(D, H)) \models holds(F, T)$, then $F \in GS_I(H, T)$.*

Proof: If D is inconsistent, then D has no model. Thus the theorem holds trivially. Now assume D is consistent. Since D is consistent, D has at least one model. Then, for any model $I = (\sigma_0, \Phi)$ of D, by Lemma 3.8 there is a model M of $COMP(KB(D, H))$ such that $F \in GS_I(H, T)$ iff $M \models holds(F, T)$. if $COMP(KB(D, H)) \models holds(F, T)$, then for any model M of $COMP(KB(D, H))$, $M \models holds(F, T)$. Thus we can conclude that if $COMP(KB(D, H)) \models holds(F, T)$, then for any model $I = (\sigma_0, \Phi)$ of D, $F \in GS_I(H, T)$. ■

For completeness, we need a new concept: e-consistency [Den 93]. A domain description D is e-consistent iff for each pair of effect propositions

$$a \textbf{ causes } f \textbf{ if } p_1, \ldots, p_n$$
$$a \textbf{ causes } \neg f \textbf{ if } p_{n+1}, \ldots, p_m$$

in D, there exists i and j such that p_i is the complement of p_j.

Lemma 3.10 *Let D be an e-consistent domain description and H a history. Assume M is a model of $COMP(KB(D,H))$. Then, for any time point T, and any fluent symbol F, $F \in state_M(succ(T))$ iff*

- *$F \in state_M(T)$ and F is inertial at time point T, or*
- *there is an effect proposition A **causes** F **if** $P_1, \ldots, P_n$ such that P_1, $\ldots$, $P_n \in state_M(T)$ and $(A,T) \in H$.*

Proof: It follows from Definitions 3.6 and 3.7 and the equality (28) of the completed predicate $holds(F,T)$. ■

Theorem 3.11 (Completeness Theorem) *Let a domain description D be e-consistent. For every history H, every fluent F, and time point T, if $F \in GS_I(H,T)$ for every model I of D, then $COMP(KB(D,H)) \models holds(F,T)$.*

Proof: Assume that $F \in GS_I(H,T)$ for every model I of D, and M is any arbitrary model of $COMP(KB(D,H))$. We want to prove that $M \models holds(F,T)$. It suffices to construct a model $I = (\sigma_0, \Phi)$ of D such that for any fluent G and time point T, $M \models holds(G,T)$ iff $G \in GS_I(H,T)$.

$I = (\sigma_0, \Phi)$ is defined as follows:

- Φ is defined such that it satisfies the effect propositions of D. Notice that there is a unique transition function Φ since D is e-consistent.
- σ_0 is defined to be $state_M(init)$.

It is easy to see that I defined above is a model of D. Now we need to prove that for any fluent G and time point T, $M \models holds(G,T)$ iff $G \in GS_I(H,T)$. That is, we need to prove that for any time point T,

$$GS_I(H,T) = state_M(T) \tag{34}$$

We prove it by induction on complexity of T. For $T = init$, $GS_I(H, init) = \sigma_0$ by Definition 3.3, whereas $\sigma_0 = state_M(init)$ by definition of I. Now assume that the equality (34) holds for

$$T = S = succ(succ(\ldots succ(init)))$$

we want to prove that it also holds for $T = succ(S)$. We need to consider two cases: either there is an action A such that $(A,S) \in H$, or there is no action A such that $(A,S) \in H$.

Suppose that there is no action A such that $(A, S) \in H$. In this case, we have $GS_I(H, succ(S)) = GS_I(H, S)$. By the equality (28) of the completed predicate $holds(F, T)$, we have

$$\forall F, S : holds(F, succ(S)) \leftrightarrow holds(F, S) \wedge \neg noninertial(F, S)$$

Since there is no actions A such that $(A, S) \in H$, by the equality (29) of the completed predicate $noninertial(F, S)$, we have that the right-hand side of $noninertial(F, S)$ is empty. Thus, we have $\forall F, S : holds(F, succ(S)) \leftrightarrow holds(F, S)$, which makes $state_M(succ(S)) = state_M(S)$. By the induction hypothesis we have $GS_I(H, S) = state_M(S)$. Hence, we have $GS_I(H, succ(S)) = state_M(succ(S))$.

Suppose that there is an action A such that $(A, S) \in H$. In this case, we have $GS_I(H, succ(S)) = \Phi(A, GS_I(H, S))$. By the induction hypothesis we have $GS_I(H, succ(S)) = \Phi(A, state_M(S))$. Now we want to prove

$$\Phi(A, state_M(S)) = state_M(succ(S))$$

By Lemma 3.10 we have that for any fluent symbol F, $F \in state_M(succ(S))$ iff

- $F \in state_M(S)$ and F is inertial at time point S, or
- there is an effect proposition A **causes** F **if** $P_1, \ldots, P_m$ such that $P_1, \ldots, P_m \in state_M(S)$ and $(A, S) \in H$.

Since I is a model of D, the above two disjuncts correspond to the requirements that must be satisfied by Φ in the state $\sigma = state_M(S)$. Therefore, $F \in \Phi(A, state_M(S))$ iff $F \in state_M(succ(S))$. ∎

Notice that the translation π is not complete in general. The condition of being e-consistent is necessary. When a domain description D is not e-consistent, no transition function exists, and thus D has no models. On the other hand, $KB(D, H)$ may be consistent.

Remarks The above proofs look similar to those of [Den 93]. But there are some technical differences arising from the real time dimension. As discussed before, a time dimension is incorporated into our logic program $KB(D, H)$, where the time always elapses no matter whether actions happen or not. In [Den 93], the new situation, denoted by $result(A, S)$, is obtained by doing action A in the situation S. The term $result(A, S)$ looks similar to $succ(S)$, but actually they are different. Another difference is about the inertiality of fluents. Definition 3.7 about the predicate $noninertial(F, S)$ is different from the corresponding definition in [Den 93], where the predicate $noninertial(F, A, S)$ is used.

4 Assimilation of new knowledge

For easy reference, we will simply write $KB(D, H) \otimes N$ to informally denote all possible new knowledge bases obtained by assimilating N into $KB(D, H)$

before we formally define it later. For convenience, we often write $N = L$ for $N = \{L\}$ when N is a singleton set. By saying that an action a initiates a fluent F we mean that F becomes true after a is done, and by a terminates F we mean that F becomes false after a is done. We start with an informal discussion, then present the formal definition, and conclude this section with computational considerations for practical implementation.

4.1 An informal discussion

Suppose that $N = holds(F, T)$ is a newly observed fact. Let's analyse how an agent should assimilate it into her knowledge base $KB(D, H)$.

The criterion for the knowledge assimilation is that the new knowledge $N = holds(F, T)$ should be in the new knowledge base if there is a cause for it, and all the cause-effect relations characterized by D should be maintained, otherwise the knowledge base would be either unsound or incomplete with respect to the domain evolution. In order to do so, first we have to look for all possible causes for the newly observed fact, then for each possible cause we add all its effects to the knowledge base to obtain a new possible knowledge base. The causes for newly observed facts may be the occurrences of actions. Since the initial situation of a domain may be partially known, we also regard truth values of fluents in the initial situation as possible causes. The process of assimilating N into $KB(D, H)$ is informally described as follows:

First the agent should check whether N is already stored in her knowledge base $KB(D, H)$, i.e., whether $COMP(KB(D, H)) \models N$. If it has been stored in $KB(D, H)$, the agent will do nothing. In this case, $KB(D, H) \otimes N \equiv KB(D, H)$.

Suppose $COMP(KB(D, H)) \not\models N$, i.e. N is not stored in the knowledge base. In this case, we assume that either the fluent F was true and inertial at the last time point or an action has happened at the last time point and a initiates F. Thus we need to consider two cases:

- The fluent F was true and inertial at the last time point. Then we have

  $$KB(D, H) \otimes holds(F, T) \equiv KB(D, H) \otimes holds(F, pred(T))$$

  where $pred(T)$, intuitively denoting the previous time point as of T, is a time point such that $T = succ(pred(T))$ for $T \notin init$. Thus, $KB(D, H) \otimes holds(F, T)$ is recursively reduced to $KB(D, H) \otimes holds(F, pred(T))$. If $pred(T)$ happens to be $init$, then we assume that the fluent F is true in the initial state, i.e., we add a value proposition **initially** F to the domain description D. If $pred(T)$ is not $init$, the agent recursively assimilates $holds(F, pred(T))$.

- An action a has happened at the last time point and a initiates F. In this case the agent only needs to revise the history H into $H \cup \{(a, pred(T))\}$, which amounts to addition of the following logic programming rule to $KB(D,H)$:

$$happens(a, pred(T)) \leftarrow$$

Now the agent has to deal with the following three possibilities:

1. There may be more than one action which can initiate F.
2. The agent assumes that A happens at the last time point and initiates F under an effect proposition A **causes** F **if** $P_1, \ldots, P_m$, but not all its preconditions $P_1, \ldots, P_m$ hold at the last time point $pred(T)$.
3. The domain description is not complete, i. e. the domain description has more than one model, which implies that the truth values of some fluents are not known in the initial state of the domain.

For the possibility (1), in case of no more preference information, the agent should not discriminate an action against another. That is, she should admit that all of them are possible. For the possibility (2), the agent recursively does $KB(D, H \cup \{(A, pred(T))\}) \otimes holds(P_i, pred(T))$ for all P_i. For the possibility (3), the agent should admit literals of the form $initially(G)$ as possible causes for N.

From this analysis it can be seen that possible causes for the truth of a fluent at a time point are among $\{happens(A,T) : A \in \Sigma_a, T \in TP\} \cup \{initially(F) : F \in \Sigma_f\}$. The possible causes from $\{happens(A,T) : A \in \Sigma_a, T \in TP\}$ are due to the incompleteness of our knowledge about the real history of an evolving domain, and the possible causes from $\{initially(F) : F \in \Sigma_f\}$ are due to the incompleteness of our knowledge about the initial state of the domain under consideration.

In the process of knowledge assimilation, possible causes for a newly observed fact play a central role. Thus, our approach to knowledge assimilation in domains of actions is referred to as a *possible causes approach (PCA)* in contrast to the *possible worlds approach (PWA)* [Gin 86] and *possible models approach (PMA)* [Win 88]. We will return to their comparison in Section 5.

In the following subsections we will formalize the above intuitive ideas.

4.2 Possible causes and knowledge assimilation

We will use abduction to define the concept of possible causes. The counterpart of possible causes in abduction is *explanation*. Hence, our approach can also be called explanation-based approach to knowledge assimilation.

Notation: Let D be a domain description, H a history, and N a set of newly observed facts. Let $\Delta = \{initially(F) : F \in \Sigma_f\} \cup \{happens(A,T) : A \in \Sigma_a, T \in TP\}$. Assume that δ is a finite subset of Δ. We will write

$KB(D,H) \dot{+} \delta$ to stand for $KB(D',H')$, where

$$\begin{aligned} D' &= D \cup \{\ \textbf{initially}\ \ F : initially(F) \in \delta\} \\ H' &= H \cup \{(A,T) : happens(A,T) \in \delta\} \end{aligned}$$

We assume no concurrent actions are allowed in H'.

Definition 4.1 (Possible Causes) *Let D be a consistent domain description, H a history, and N a set of literals. Let $\Delta = \{initially(F) : F \in \Sigma_f\} \cup \{happens(A,T) : A \in \Sigma_a, T \in TP\}$. A possible cause of N in $KB(D,H)$ is a subset δ of Δ such that*

- $KB(D,H) \dot{+} \delta \models N$;
- $KB(D,H) \dot{+} \delta$ *is consistent;*
- δ *is minimal in the sense that no subset δ' of δ exists such that δ' satisfies the first two conditions. we will write $KB(D,H) \uparrow N$ to stand for the set of all possible causes of N in $KB(D,H)$.*

Obviously, the possible causes defined above can be computed by abductive reasoning systems. Later in Subsection 4.4 we will see how to compute them by abductive logic programming.

Let δ be a possible cause of N in $KB(D,H)$. Assume that δ contains an element of the form $initially(F)$. If **initially** $F \in D$, we will often leave out $initially(F)$ in δ, when no confusion arises.

Example 4.2 *Consider the Stolen Car Problem (SCP) motivated from an example of [Kautz 86]. In the SCP domain, there is one fluent name: stolen, and two action names: wait and steal. The domain description D_{scp} is as follows:*

$$\begin{gathered} \textbf{initially}\ \ \neg stolen \\ steal\ \textbf{causes}\ \ stolen \end{gathered}$$

Let $H_{scp} = \emptyset$. Then, the logic program $KB(D_{scp}, H_{scp})$ is as follows:

$$\begin{aligned} holds(F, init) &\leftarrow initially(F) \\ holds(F, succ(T)) &\leftarrow holds(F,T), \text{not } noninertial(F,T) \\ holds(stolen, succ(T)) &\leftarrow happens(steal, T) \\ \perp &\leftarrow holds(stolen, init) \end{aligned}$$

Suppose that the car is found missing at time 3 (= succ(succ(succ(init)))). By the definition of possible causes, we will find that there are three possible causes of $holds(stolen, 3)$ in her knowledge base:

$$\begin{aligned} \delta_1 &= \{happens(steal, 0)\} \\ \delta_2 &= \{happens(steal, 1)\} \\ \delta_3 &= \{happens(steal, 2)\} \end{aligned}$$

In order to incorporate N into $KB(D,H)$, if there is a possible cause δ for N in $KB(D,H)$, then we can simply add δ into the knowledge base to have a new knowledge base. Now we want to question about whether there is always a possible cause δ for N in $KB(D,H)$. The answer turns out to be negative.

Proposition 4.3 *Let D be a domain description, and H a history. Let N be a set of literals. It does not hold that there is always a possible cause of N in $KB(D,H)$.*

Proof: It is sufficient to give a counter-example. Let H be empty and D be

$$\begin{array}{c} \textbf{initially } \neg p, \neg q \\ a \textbf{ causes } p \textbf{ if } q \\ b \textbf{ causes } q \textbf{ if } p \end{array}$$

It is clear that there is no possible cause for, e.g. $holds(p,5)$, in $KB(D,H)$. ■

We say that a sequence of actions $a_1;a_2;\ldots;a_n$ is a sub-sequence of b_1; b_2; …; b_m iff $a_1;a_2;\ldots;a_n$ is a sub-sequence of $b_2;\ldots;b_m$ by recursion, or $a_1=b_1$ and $a_2;\ldots;a_n$ is a sub-sequence of $b_2;\ldots;b_m$ by recursion.

Definition 4.4 (Liveness) *Given a domain description D and a model (σ_0, Φ), a fluent F is said to be live in a sequence of actions $a_1;a_2;\ldots;a_n$ iff there is another sequence of actions $b_1;b_2;\ldots;b_m$ such that*

- $F \in \Phi(b_1;b_2;\ldots;b_m,\sigma_0)$
- $a_1;a_2;\ldots;a_n$ *is a sub-sequence of* $b_1;b_2;\ldots;b_m$.

To prove whether a property is live, one can employ the induction rules developed by Reiter [Rei 93]. The following proposition states the relationship between liveness of a fluent and existence of its possible causes.

Proposition 4.5 *There is no possible cause for N in $KB(D,H)$, iff N is not live in any sequence of actions corresponding to $GS_M(H)$ for any model M of D.*

Proof: Assume N is not live in any sequence of actions corresponding to $GS_M(H)$ for any model M of D. If there is a possible cause E for N in $KB(D,H)$, there is a $\delta \subseteq \Delta$, such that $KB(D,H) + \delta \models holds(F,T)$. From D and δ we can construct a new domain description D' such that $D \subseteq D'$ and D and D' have the same effect propositions. Thus, any model of D' is also a model of D, and any fluent F which is live in a sequence of actions in a model M of D' is also live in the same sequence of actions in the model M of D. From D,H and δ we can have a model M of D' such that N is live in the sequence of actions corresponding to $GS_M(H)$. Hence, there is a model M of D such that N is live in the sequence of actions corresponding to $GS_M(H)$. Vice versa. ■

For assimilation of N into $KB(D, H)$, when there is no possible cause for N in $KB(D, H)$, we will simply discard N because we assume that domain descriptions are ideal, faultless and not subject to revision. This assumption is essentially the same as that made about system descriptions in model-based diagnosis [Rei 87], where a system description is faultless and is not subject to any modification. To lift this assumption we could engage in domain description debugging, which is not discussed in this paper.

Definition 4.6 (Assimilation of New Knowledge) *Let D be a consistent domain description, H a history, and N a set of newly observed facts. Assume that δ is a possible cause of N in $KB(D, H)$. An assimilation of N into $KB(D, H)$ is defined to be $KB(D', H')$, where*

$$\begin{aligned} D' &= D \cup \{ \textbf{ initially } \ F : initially(F) \in \delta \} \\ H' &= H \cup \{(A, T) : happens(A, T) \in \delta \} \end{aligned}$$

We will write $KB(D, H) \otimes N$ to stand for $\{KB(D, H) \dot{+} \delta : \delta \in KB(D, H) \uparrow N\}$. The operator $\otimes$ is called the PCA knowledge assimilation operator.

Note that $KB(D, H) \otimes N$ is, in general, a set of knowledge bases. In the following we will also define $\{K_1, \ldots, K_n\} \otimes N$ to be $K_1 \otimes N \cup \ldots \cup K_n \otimes N$, where each K_i is a knowledge base.

From the above definition it can be seen that possible causes for new knowledge are essential. It is of interest to notice that there may be a possible cause for a subset of N when there is no possible cause for N. When there are no causes for a set of newly observed facts N, no causes for its members are incorporated. This is referred to as *either everything or nothing principle*, which means that either every member of N is incorporated or no member is incorporated, that is to say, nothing is believed if one of them cannot be explained.

Example 4.7 *Let H be empty, and D include*

$$\begin{array}{rl} \textbf{initially} & p, \neg q \\ a \textbf{ causes} & q \textbf{ if } p \\ a \textbf{ causes} & \neg p \textbf{ if } p \\ b \textbf{ causes} & p \textbf{ if } q \\ b \textbf{ causes} & \neg q \textbf{ if } q \end{array}$$

Let $N = \{holds(p, 2), holds(q, 2)\}$. It is easy to verify that there is no possible causes for N, but there are possible causes for $holds(p, 2)$ and $holds(q, 2)$, respectively. Thus, we have

$$\begin{aligned} KB(D, H) \otimes \{holds(p, 2), holds(q, 2)\} &= \{KB(D, H)\} \\ KB(D, H) \otimes holds(p, 2) &= \{KB(D, H)\} \\ (KB(D, H) \otimes holds(p, 2)) \otimes holds(q, 2) &= \left\{ \begin{array}{l} KB(D, \{(a, 0)\}), \\ KB(D, \{(a, 1)\}) \end{array} \right\} \end{aligned}$$

$$
\begin{aligned}
KB(D,H) \otimes holds(q,2) &= \left\{ \begin{array}{c} KB(D,\{(a,0)\}), \\ KB(D,\{(a,1)\}) \end{array} \right\} \\
(KB(D,H) \otimes holds(q,2)) \otimes holds(p,2) &= \left\{ \begin{array}{c} KB(D,\{(a,0),(b,1)\}), \\ KB(D,\{(a,1)\}) \end{array} \right\}
\end{aligned}
$$

Note that there is a possible cause $\{happens(b,1)\}$ *for* $holds(p,2)$ *in knowledge base* $KB(D,\{(a,0)\})$*, but no possible cause for* $holds(p,2)$ *in knowledge base* $KB(D,\{(a,1)\})$*.*

This example shows that the resulting knowledge bases depend on the order in which facts are incorporated. For different orders, for example, first $holds(p,2)$ then $holds(q,2)$, and first $holds(q,2)$ then $holds(p,2)$, the resulting knowledge bases are different. What is more, they are different from $KB(D,H) \otimes \{holds(p,2), holds(q,2)\}$. In the next subsection we will discuss more.

Example 4.8 *Consider the Stolen Car Problem of Example 4.2. Since there are three possible causes for* $holds(stolen,3)$*, then we have*

$$
KB(D_{scp},H_{scp}) \otimes holds(stolen,3) = \left\{ \begin{array}{l} KB(D_{scp},\{(steal,0)\}), \\ KB(D_{scp},\{(steal,1)\}), \\ KB(D_{scp},\{(steal,2)\}) \end{array} \right\}
$$

4.3 Some observations

We have given a basic definition of knowledge assimilation based on the possible causes approach. In the following we make a few observations:

- In general, there are more than one possible cause for given new knowledge. As a consequence, there are several resulting knowledge bases. This is due to uncertainty of causes of new knowledge. It is interesting to investigate how to discriminate against each other. This work is very similar to measurements, testing and discrimination studied in model-based diagnosis. After a possible cause is discovered, the agent may make some testing to acquire new knowledge, and then only keep those possible causes which predict the newly acquired knowledge. Another common way is to use a preference relation on possible causes. The actions are preferred if they are more frequently done. For the Tom-$Mary$ example, in the case that Tom and $Mary$ do not know each other, the action buy should be preferred to other actions, in the case that Tom and $Mary$ happen to be in the same family, the action $give$ is preferred, in the case that Tom and $Mary$ have records of criminal convictions, the action $steal$ may be preferred. Discrimination of possible causes is a pragmatic issue, which is out of the scope of this paper.

- New knowledge without causes is simply forgotten, that is, $KB(D,H) \equiv (KB(D,H) \otimes N)$ if there is no possible cause for N in $KB(D,H)$. As a consequence, there is no distinction between assimilating new knowledge without possible causes and assimilating new knowledge which is already stored in knowledge bases.

- As noticed before, $KB(D,H) \otimes N_1$ may include several knowledge bases. Thus, there may be possible causes for N_2 in some knowledge bases of $KB(D,H) \otimes N_1$, and there are no possible causes for N_2 in the other knowledge bases of $KB(D,H) \otimes N_1$. One may prefer to reject those knowledge bases in which there are no possible causes for N_2, and keep those knowledge bases in which there are possible causes for N_2. It seems very reasonable to adopt this suggestion.

- The *either everything or nothing principle* was accepted in Definition 4.6. It has pros and cons. For example, the fact that a liar sometimes tells the truth is a cons against the either-everything-or-nothing principle. It seems that this principle is very conservative. However, in absence of more information about credibility of a collection of knowledge, the either-everything-or-nothing principle might be a good choice.

  Another choice is as follows. Assume that N is a set of literals. Let M be a set-inclusion maximal subset of N such that there is a possible cause δ for M in $KB(D,H)$. Then, a new possible knowledge base is $KB(D,H) \dot{+} \delta$. That is, as much as possible of N is assimilated provided that it can be explained, and thus called *greedy principle*.

  We regard the greedy principle as a competing one with the either-everything-or-nothing principle. But we should point out that the greedy principle is computationally more expensive than the either-everything-or-nothing principle. Although the either-everything-or-nothing principle is accepted in PCA, we don't mean the greedy principle is incompatible with PCA. In fact, we can easily have two alternatives: the PCA based on the either-everything-or-nothing principle and the PCA based on the greedy principle.

- The past observations are not necessarily recorded in Definition 4.6. That is to say, N_1 is not necessarily stored in $(KB(D,H) \otimes N_1) \otimes N_2$. The causes for N_1 is kept as part of history, but N_1 itself is not kept.

  If an agent chooses to keep a past observation in her knowledge base, there is a very simple way to do it: In her knowledge base, she adds a constraint like this[4]:

$$\perp \leftarrow not\ N$$

[4] In the case that N is a set of literals $\{L_1, \ldots, L_k, \neg L_{K+1}, \ldots, \neg L_m\}$, the following m logic programming rules should be added:

$$\perp \leftarrow not\ L_1. \quad \ldots \perp \leftarrow not\ L_k. \quad \ldots \perp \leftarrow L_{k+1}. \quad \ldots \perp \leftarrow L_m.$$

Then, N is kept permanently in her knowledge base. Any addition of possible causes for newer knowledge won't falsify N, since $\bot$ is always interpreted to be falsity and the above logic programming rule effectively function as integrity constraints.

4.4 Computational considerations

In PCA, possible causes for new knowledge play a central role. The critical step in the process of knowledge assimilation is computation of possible causes. In this section we want to show how to compute possible causes with abductive logic programming [KKT 93] and how to improve computational efficiency.

In order to facilitate computation with abductive logic programming, we introduce a new predicate $occurs(A, T)$, for action A and time T, and add a new programming rule to $KB(D, H)$:

$$happens(A, T) \leftarrow occurs(A, T) \tag{35}$$

Now we take $occurs(A, T)$ to be an abducible predicate. The above rule means that if action A is abduced to occur at time T, it happens at T. As said before, this paper does not consider concurrent actions. Thus we add the following rule:

$$\bot \leftarrow A1 \neq A2, happens(A1, T), happens(A2, T) \tag{36}$$

The definitions for $A1 \neq A2$ should also be added such that every two syntactically different actions are not equal[5]. However, later we will find that the above rule is redundant and can be removed for efficiency improvement.

Let $KB_{occ}(D, H) = KB(D, H) \cup \{(35), (36)\}$. Now we take $initially(F)$ and $occurs(A, T)$ as abducible predicates. It is easy to see that KB_{occ} is still acyclic. The programming rules with $\bot$ as head in $KB_{occ}(D, H)$ are regarded as integrity constraints. Then it can be shown that the possible causes for N become the abductive answers to N in $KB_{occ}(D, H)$.

Proposition 4.9 *Given a domain description D, a history H, and a set N of newly observed facts. Then, δ is a possible cause for N in $KB(D, H)$ iff δ is an abductive answer to N in the abductive logic program $KB_{occ}(D, H)$.*

Proof: Immediate from the definition of possible causes. ∎

Hence, possible causes can be computed by an abductive query evaluation procedure for normal logic programs. Since KB_{occ} is acyclic, the completion semantics, stable model semantics, and well-founded model semantics coincide with the perfect model semantics. We have computed possible causes by using the latest version [DPS 96] of our REVISE system [DPN 94], an extended logic programming system for revising knowledge bases. The latest version of the REVISE system is based on a top-down query evaluation procedure [ADP 94] of well founded semantics with explicit negation (WFSX) [PA 92]. If explicit

[5] This is essentially the same as Reiter's unique name axioms.

negation is not used, then the WFSX semantics coincides with the WFS semantics. Since we only use acyclic normal logic programs, the use of REVISE is not essential for the purpose of this paper, and other abductive query evaluation procedures such as those in [DdS 92, SI 92] for abductive normal logic programs can also be used.

In what follows we introduce three techniques to improve search efficiency.

Proposition 4.10 *Let D be a consistent domain description, H a history, and $N = holds(F_n, T_n)$ new knowledge to be assimilated. Then, for any abductive answer δ to N in $KB_{occ}(D, H)$, we have*

$$\delta \subseteq \{initially(F) : F \in \Sigma_f\} \cup \{occurs(A, T) : A \in \Sigma_a, T \prec T_n\}$$

Proof: It suffices to prove that there is no element of the form $occurs(A, T)$ in δ with $T \not\prec T_n$.

Let M be any model of $COMP(KB(D, H) \dot{+} \delta)$. Then M is also a model of $holds(F_n, T_n)$, since δ is an abductive answer to N in $KB(D, H)$. By the equality (28) of the completed predicate $holds(F, T)$, we have the completed $holds(F_n, T_n)$ as follows:

$$holds(F_n, T_n) \leftrightarrow E_1 \vee E_2 \vee E_3 \vee \ldots \vee E_m$$

where

$$\begin{aligned}
E_1 &\equiv T_n = init \wedge initially(F_n) \\
E_2 &\equiv \exists S : T_n = succ(S) \wedge holds(F_n, S) \wedge \neg noninertial(F_n, S) \\
E_i &\equiv \exists S : T_n = succ(S) \wedge happens(a, S) \wedge holds(p_1, S) \wedge \ldots \wedge holds(p_n, S)
\end{aligned}$$

such that there is an exact correspondence between the disjunct $E_i, i \geq 3$ and all the effect propositions a **causes** F_n **if** $p_1, \ldots, p_n$ with F_n being a positive fluent. From the right-hand side of the above equivalence about $holds(F_n, T_n)$ it can be seen that the truth value of $holds(F_n, T_n)$ has nothing to do with truth values of any other predicates $holds(F_n, T_x)$, $noninertial(F_n, T_x)$, and $happens(a, T_x)$ with $T_x \not\prec T_n$. Thus, if there is a δ_x such that

$$\begin{aligned}
&occurs(a, T_x) \in \delta_x \text{ with } T_x \not\prec T_n \\
&KB(D, H) \dot{+} \delta_x \models holds(F, T) \\
&KB(D, H) \dot{+} \delta_x \text{ is consistent}
\end{aligned}$$

then $\delta_x \setminus \{occurs(a, T_x)\}$ also satisfies the last two conditions above. Thus, by Definition 4.1 δ_x is not a possible cause since a subset of it satisfies the last two conditions above. ■

The above proposition implies that it is not necessary to consider all the possible abducibles in Δ defined before, and it is sufficient to define the abducibles to be

$$\Delta_{T_n} = \{initially(F) : F \in \Sigma_f\} \cup \{occurs(A, T) : A \in \Sigma_a, T \prec T_n\} \qquad (37)$$

This will effectively reduce the search space. The second observation is that we have assumed that no concurrent actions have happened in histories. This implies that if $(A,T) \in H$, then we need not consider $occurs(B,T)$ for any action B.

Proposition 4.11 *Let D be a consistent domain description and H a history. Then, for any new knowledge N, and any abductive answer δ to N in $KB_{occ}(D,H)$, we have that if $occurs(a,t) \in \delta$, then there is no other action b such that $(b,t) \in H$.*

Proof: Suppose $occurs(a,t) \in \delta$ and there is an action b such that $(b,t) \in H$. Then, we will have both $happens(a,t)$ and $happens(b,t)$ are true. Thus, $KB(D,H) \dot{+} \delta$ is not consistent, since $KB(D,H)$ includes an instance of the rule (36):

$$\perp \ \leftarrow a \neq b, happens(a,t), happens(b,t)$$

■

This proposition can reduce the search space by defining the abducibles as follows:

$$\Delta_H = \Delta_{T_n} \setminus \{occurs(A,T) : A \in \Sigma_a, \text{ there is } B \text{ such that } (B,T) \in H\} \quad (38)$$

where Δ_{T_n} is defined as in (37).

The third observation is about the consistency check. By Definition 4.1, $KB(D,H) \dot{+} \delta$ has to be consistent for δ to be a possible cause. To check the consistency of a knowledge base is very expensive. In the following we give a sufficient and cheap technique to check the consistency.

Lemma 4.12 *Let D and H be any domain description and any history, respectively. If D is consistent, then $KB(D,H)$ is also consistent.*

Proof: Since D is consistent, D has a model. By Lemma 3.8, $KB(D,H)$ has a model for any history. What is more, the model can be constructed as in the proof of Lemma 3.8. ■

Corollary 4.13 *Let D be a consistent domain description and H a history. Then, for any $\delta \subseteq \Delta_H \setminus \{initially(F) : F \in \Sigma_f\}$, $KB(D,H) \dot{+} \delta$ is consistent.*

Proof: Immediate from the above lemma. ■

Lemma 4.14 *Let D be a domain description only consisting of effect propositions. If D is e-consistent, then D is consistent.*

Proof: Straightforward. ■

Proposition 4.15 *Let D be an e-consistent domain description. If D is not consistent, then the value propositions of D are not satisfied in any interpretation structure of D. Furthermore, there are two v-propositions* **initially** *F and* **initially** *G such that $F = \neg G$ or $G = \neg F$.*

Proof: Suppose that the value propositions of D are satisfied by an interpretation structure $I = (\sigma_{0_I}, \Phi_I)$. Let $M = (\sigma_0, \Phi)$, where $\sigma_0 = \sigma_{0_I}$, and Φ is the unique transition function determined by the e-consistent effect propositions. It is then easy to see that M is a model for D, and thus D is consistent. Hence, we can conclude that if D is not consistent, the value propositions of D are not satisfied by any interpretation structure of D. Since the value propositions of D are not satisfied by any interpretation structure of D, there must be two value propositions **initially** F and **initially** G such that $F = \neg G$ or $G = \neg F$. ■

By the above proposition, the consistency check can be done in the following way: For each fluent F, we add a new programming rule to $KB(D, H)$:

$$\perp \ \leftarrow holds(F, init), not\ holds(F, init)$$

Actually, all the above programming rules in $KB(D, H)$ function as integrity constraints. After all the rules for $\perp$ are added, one can issue a query $\leftarrow \perp$ to a query evaluation procedure. If the answer to it is *yes*, the logic program $KB(D, H)$ is inconsistent, otherwise it is consistent. To check whether $KB(D, H)$ is consistent, the REVISE system can be directly used.

All the examples in this paper and some additional small-sized examples have been experimented with the latest version [DPS 96] of our REVISE system for logic programs with the WFSX semantics [PAA 93]. Although our examples are simple and small-sized, it looks that our methodology works well. We should point out that our methodolgy can be experimented with any other abductive normal logic programming systems such as those based on [DdS 92, SI 92].

5 Related work

There has been much work in belief revision and update. In the AGM theory [Gar 88] eight postulates are proposed for belief revision. Katsuno and Mendelzon [KatM 91] made a distinction between the belief revision in static domains and belief update in dynamic domains, and proposed other postulates for belief update operators. Since these postulate-based belief revisions/updates are too general and hardly used in practical domains, belief base revision/update operators, such as the the possible worlds approach PWA [Gin 86] and the possible models approach PMA [Win 88], have attracted more attentions. Since our PCA is conceptually and technically different from other belief revisions and updates, and is currently developed for domains of actions, it is extremely hard, if not impossible, to find its theoretical relations to other approaches. In what follows we will use examples from domains of actions to compare our work with the possible worlds approach PWA, and the possible models approach PMA.

5.1 The possible worlds approach

The possible worlds approach (PWA) [Gin 86] is a representative of syntax-based approaches to belief base revision/updates. Other similar approaches can be found in, e.g., [FUV 83, Nebel 91]. Its basic idea is as follows: The belief base of a belief set is taken as the representation of an epistemic state, and incorporation of new belief is done at the syntax level of the belief base by taking the union of the new belief and a selected subset of the belief base which does not allow for derivation of the negation of the new belief.

The PWA was originally proposed by Ginsberg [Gin 86] to evaluate counterfactuals from the AI perspective. Counterfactuals have been long discussed in philosophy and logic, but principally untreated in the AI literature until [Gin 86]. Typical examples of counterfactuals are, " If the electricity hadn't failed, dinner would have been ready on time", and "If the electricity hadn't failed, pigs would fly". In both cases, both the premise and the conclusion of the counterfactual are false. Nevertheless, the first counterfactual may well be true, while the second is unlikely to be so. Ginsberg showed that counterfactuals have wide applications in database updates, plan simulation, natural language understanding, diagnosis, and so on.

Given a set S of sentences as description of the current world, the truth value of a counterfactual of the form $\phi > \psi$ can be evaluated as follows: to construct all possible worlds W for ϕ in S, the counterfactual $\phi > \psi$ is said to be true just in case $T \cup \{\phi\} \models \psi$ for every $T \in W$. This can be rephrased in terms of belief revision/update: The evaluation of the truth value of $\phi > \psi$ amounts to evaluate whether ψ is true in all possible new belief sets obtained by revising/updating the current belief base S with the new belief ϕ. Let $*_G$ denote Ginsberg's PWA operator. Then, evaluating $\phi > \psi$ amounts to evaluating $S *_G \phi \models \psi$. Furthermore, Ginsberg gives a constructive definition of the new belief set represented by $S *_G \phi$ as follows. The models of $S *_G \phi$ are defined to be the union of models of the union of ϕ and maximal consistent subsets of S not implying $\neg\phi$. Formally, let $W_G(\phi, S) = \{T \subseteq S : T \not\models \neg\phi \text{ and } T \subset U \subseteq S \Rightarrow U \models \neg\phi\}$. Then, the PWA operator $*_G$ is defined as follows: $S *_G \phi = \{T \cup \{\phi\} : T \in W_G(\phi, S)\}$. In what follows, we need to protect some integrity constraints which are supposed to be true in every possible state. In this case, the PWA operator can be slightly modified as follows: Let IC be the integrity constraints, a set of formulas, then we have

$$S *_G \phi = \{T \cup \{\phi\} : T \in W_G^{IC}(\phi, S)\}$$

where

$$W_G^{IC}(\phi, S) = \{T \subseteq S : T \cup IC \not\models \neg\phi \text{ and } T \subset U \subseteq S \Rightarrow U \cup IC \models \neg\phi\}$$

In the following we write a set of formulas to stand for the conjunction of all the formulas in the set. Now let's consider the *Tom-Mary* scenario defined before. In this scenario, let IC_{TM} be the union of the unique names axioms[6]

[6] Reiter's unique name axioms (UNA) are often used in relational databases and logic programs. It says that two syntactically different terms are different. In this example, by UNA we have $Tom \neq Mary$. From IC_{TM} we can derive that $has(Tom, Book) \rightarrow \neg has(Mary, Book)$. We should point out that there is no need to use UNA if we assume that we are only restricted to Herbrand models. See [Rei 89] for more discussions.

and the following formulas:

$$\begin{array}{rcl} has(X, Book) \wedge has(Y, Book) & \rightarrow & X = Y \\ has(X, Money) \wedge has(Y, Money) & \rightarrow & X = Y \end{array}$$

By the PWA we will have the following computations for $KB_1 *_G N_1$, where KB_1 and N_1 are defined in (1) and (2):

$$\begin{array}{rcl} KB_1 & = & \left\{ \begin{array}{ll} has(Tom, Book), & \neg has(Tom, Money) \\ \neg has(Mary, Book), & has(Mary, Money) \end{array} \right\} \\ N_1 & = & has(Mary, Book) \\ W_G^{IC_{TM}}(N_1, KB_1) & = & \{ \neg has(Tom, Money), \quad has(Mary, Money) \ \} \\ KB_1 *_G N_1 & = & \left\{ \left\{ \begin{array}{l} \neg has(Tom, Money) \\ has(Mary, Book) \\ has(Mary, Money) \end{array} \right\} \right\} \end{array}$$

Combining with IC_{TM}, one can prove that

$$KB_1^{IC} *_G N_1 \models \neg has(Tom, Book)$$

which implies that KB_3 defined in (6) is derivable from the PWA. This is the best the PWA can do for this example. However, as discussed in the Introduction, another possible new knowledge base KB_4 cannot be derived:

$$KB_4 = \left\{ \begin{array}{ll} \neg has(Tom, Book), & has(Tom, Money) \\ has(Mary, Book), & \neg has(Mary, Money) \end{array} \right\}$$

Intuitively, KB_3 corresponds to the possible cause: Tom gives $Book$ to $Mary$ or $Mary$ steals $Book$ from Tom, while KB_4 corresponds to the possible cause: $Mary$ buys $Book$ from Tom. In our PCA, the counterparts in the real time line of both KB_3 and KB_4 can be derived.

In the literature syntax-based approaches to belief base revision/update have been criticized for being sensitive to the syntax of the description of the belief [Win 88]. Given different syntactical representations A and B of the same belief set $Cn(A)$ $(= Cn(B))$, the PWA could give two different results, i.e. $Cn(A *_G x) \neq Cn(B *_G x)$. This criticism has lead to the proposal of the possible models approach (PMA).

5.2 The possible models approach

The possible models approach (PMA) is one representative of the semantics-based approaches to belief revision/update. The basic idea of the semantics-based approaches is as follows: the belief base K of a belief set is taken as the representation of an epistemic state as in the syntax-based approaches to belief revision/update, and incorporation of new belief N is done at the semantics level by selecting the models of the new belief N which are *closest* to the models of K [Borgida 85, Dalal 88, Win 88, KatM 89, KatM 91]. The approaches in the literature vary by different definitions of *closest models*. Among others there are two representative measures of differences between two models: cardinality-based measures such as Dalal's approach [Dalal 88], and subset-based measures such as Winslett's possible models approach [Win 88]. In this subsection we are restricted to the subset-based measures by considering the possible models approach PMA, but similar results can be obtained for comparison between PCA and Dalal's approach.

A model of a first-order theory will be represented by a set of predicate instances which are and only are interpreted to be true. And we write $mod(T)$ to denote the set of models of a first-order theory T. Let I and J be two models. We define $diff(I,J) = (I \setminus J) \cup (J \setminus I)$. Intuitively, $diff(I,J)$ denotes the atoms which have different truth values in I and J, respectively. For example, let $I = \{p,q\}$ and $J = \{p,r\}$, then we have $diff(I,J) = \{q,r\}$, which is the set of atoms which change their truth values between the models of I and J.

Let K be a knowledge base and N a new sentence. We denote Winslett's update operator with $*_W$. For each model I of K, from the models of N Winslett selects those closest to I by set inclusion. Formally, Winslett says that an interpretation J_1 is closer to I than J_2, denoted by $J_1 \preceq_I J_2$, iff $diff(J_1,I) \subset diff(J_2,I)$. Let IC be the integrity constraints[7]. Winslett defines $Incorporate(N,I)$ to be the set of all the minimal elements of $mod(N)$ with respect to $\preceq_I$ with IC being protected, that is,

$$\begin{aligned} Incorporate(N,I) &= \{J \in mod(N \wedge IC) : \\ &\qquad \text{No } J' \in mod(N \wedge IC) \text{ such that} J' \preceq_I J\} \end{aligned}$$

Then, the models $mod(K *_W N)$ of the new belief set are defined as follows:

$$mod(K *_W N) = \bigcup_{I \in mod(K \wedge IC)} Incorporate(N,I)$$

In the style of [Win 88], the knowledge bases are represented by fluent names which are known to be true or false. Let's again consider the Tom-$Mary$ example. Let KB_1, N_1, and IC_{TM} be the same as before. By the PMA operator $*_W$ we then have

$$mod(KB_1 *_W N_1) = \{\ has(Mary, Book),\quad has(Mary, Money)\ \}$$

Adding negative information to the above model, we can show that it is the same as KB_3. As discussed before, although KB_3 is an acceptable new possible knowledge base which corresponds to the possible cause: Tom gives $Book$ to $Mary$ or $Mary$ steals $Book$ from Tom, we cannot obtain another equally acceptable new possible knowledge base KB_4 which corresponds to another possible cause: $Mary$ buys $Book$ from Tom.

[7] In [Win 88] Winslett requires that the selected models satisfy some so-called protected formulas which are supposed to be satisfied in every possible state of beliefs. To decide which formulas should be protected is a pragmatic issue. It varies from one domain to another.

Summary and concluding remarks

This paper is a continuing experiment on using logic programs for representing and reasoning about actions and changes in line with [GL 93, Den 93, Dung 93, BG 93, LP 96B]. In this paper we have developed a formal, provably correct and yet computational methodology for assimilation of new knowledge into knowledge bases about actions and changes based on the slogan: What is believed is what is explained. The result of this paper is being used in a multi-agent system in which each agent, programmed by an agent programming language SICLE [LP 96D], is supposed to be embedded in a real-time world and is expected to be able to reason and act intelligently in order to realize its intentions.

Technically, we started with a simple action description language proposed by Gelfond and Lifschitz [GL 93], then used it to describe a domain of actions. We have presented a new translation π from domain descriptions to abductive normal logic programs where a time dimension is incorporated. Then by using the translation π we have defined knowledge bases about evolving domains of actions as abductive logic programs, showed the soundness and completeness of the knowledge bases with respect to their domain descriptions. In order to assimilate new knowledge into such knowledge bases, we proposed a possible causes approach PCA in contrast to Ginsberg's possible worlds approach PWA [Gin 86] and Winslett's possible models approach PMA [Win 88]. A possible cause of new knowledge consists of some abduced occurrences of actions and value propositions about the initial state of the domain of actions. To assimilate new knowledge into a knowledge base, all the effects of the actions which play the role of possible causes for the new knowledge, recursively together with all the effects of actions which play the role of possible causes for the preconditions of the actions whose occurrences are possible causes for the new knowledge, are also assimilated. In our PCA to knowledge assimilation, the possible causes play a central role. We have shown how to compute possible causes with abductive logic programming, and presented three techniques to improve the search efficiency. We have also made four observations on how to adapt the PCA in practice. We have employed examples to show that both the syntax-based and semantics-based approaches to belief revision/update sometimes give intuitively undesirable results or cannot give all intuitively desired results. All the examples of this paper and a few more examples have been experimented with the latest version [DPS 96] of the REVISE system.

Acknowledgements

We would also like to thank anonymous referees for their valuable comments which have improved the quality of this paper.

References

[AGM 85] Alchorrón, C., Gärdenfors, P. and Makinson, D., *On the logic of theory change: Partial meet contraction and revision functions*, J. of Symbolic Logic, 50:2, 1985, 510 – 530

[ADP 94] Alferes, J. J. , Damásio, C. V., andPereira, L. M., *SLX - A top-down derivation procedure for programs with explicit negation*, In: M. Bruynooghe (ed.), International Logic programming Symposium. MIT Press, 1994

[AB 90] Apt, K.R. and Bezem, M., *Acyclic programs*, Proc. of ICLP 90, MIT Press, 1990, 579–597

[Baker 91] Baker, A.B., *Nonmonotonic reasoning in the framework of situation calculus*, Artificial Intelligence, 49, 1991, 5–23

[BG 93] Baral, C. and Gelfond, M., *Representing concurrent actions in extended logic programming*, Proc. of l IJCAI'93, 1993, pp. 866–871

[Borgida 85] Borgida, A., *Language features for flexible handling of exceptions in information systems*, ACM Trans. on Database Systems, 10, 563-603, 1985

[CDT 91] Console, L., Dupre, D.T., and Torasso, P., *On the relationship between abduction and deduction*, J. of Logic and Computation, 1:5, 1991, 661–690

[Dalal 88] Dalal, M., *Investigations into a theory of knowledge base revision: Preliminary report*, Proc. of AAAI'88, 1988, 475 – 479

[DPS 96] Damásio, C. V., Pereira, L. M. and Schroeder, M., *REVISE Progress Report: A system description*, Department of Computer Science, Universidade Nova de Lisboa, submitted to FAPR'96, 1996

[DPN 94] Damásio, C. V., Pereira, L.M., and W. Nejdl, *REVISE: An Extended Logic Programming System for Revising Knowledge Bases*, Proc. of KR'94, 1994

[DdS 92] Denecker, M., De Schreye, D., *SLDNFA: an abductive procedure for normal abductive programs*, in K. R. Apt (ed.), Logic Programming: Proc. of 1992 Int'l Joint Conference and Symposium, 1992, 686-700

[Den 93] Denecker, M., *Knowledge Representation and Reasoning in Incomplete Logic Programming*, Ph.D. thesis, Department of Computer Science, K.U.Leuven, 1993

[Dung 93] Dung, P.M., *Representing actions in logic programming and its application in database updates*, Proc. of ICLP'93, MIT Press, 1993, 222–238

[FUV 83] Fagin, R., Ullman, J. D., and Vardi, Y., *On the semantics of updates in databases*, Proc. of 2nd ACM SIGART/SIGMOD Symp. on Principles of Database Systems, 1983, 352–365

[Gar 88] Gärdenfors, P., *Knowledge in Flux — Modeling the Dynamics of of Epistemic States*, MIT Press, Cambridge, Mass., 1988

[GL 88] Gelfond, M., and Lifschitz, V., *The stable model semantics for logic programming*, In. Proc. of 5th Logic Programming Symposium, R. Kowalski and K. Bowen, eds., 1988, 1070-1080

[GL 93] Gelfond, M. and Lifschitz, V., *Representing action and change by logic programs*, Journal of Logic Programming, Vol.17, 1993, 301–322

[Gin 86] Ginsberg, M., *Counterfactuals*, Artificial Intelligence, 30:1, 1986, 35 – 79

[KKT 93] Kakas, A.C., Kowalski, R.A., Toni, F., *Abductive logic programming*, J. of Logic and Computation, 2;6, 1993, 719-770

[KakM 90] Kakas, A.C., Mancarella, P., *Generalized stable models: A semantics for abduction*, Proc. of ECAI'90, 1990

[KatM 89] Katsuno, H. and Mendelzon, A., *A unified view of propositional knowledge base updates*, Proc. of 11th IJCAI, 1989, 1413 – 1419

[KatM 91] Katsuno, H. and Mendelzon, A., *On the difference between updating a knowledge base and revising it*, in: Allen. J., Fikes, R., and Sandewall, E. (eds.), Proc. of the 2nd Internat. Conf. on Principles of Knowledge Representation and Reasoning, 1991, 387–394

[Kautz 86] Kautz, H.A., *The logic of persistence*, Proc. of the AAAI86, 1986, 401–405

[KS 86] Kowalski, R.A. and Sergot, M., *A logic-based calculus of events*, New Generation Computing, 4, 1986, pp.67-75

[LP 96A] Li, R. and Pereira, L. M., *What is believed is what is explained (sometimes)*, Proc. of AAAI'96, pp. 550 – 555

[LP 96B] Li, R. and Pereira, L. M., *Temporal reasoning with abductive logic programming*, Proc. of ECAI'96, 1996, pp. 13 – 17

[LP 96C] Li, R. and Pereira, L. M., *Representing and Reasoning about Concurrent Actions with Abductive Logic Programs*, Technical Report, Department of Computer Science, Universidade Nova de Lisboa, 1996

[LP 96D] Li, R. and Pereira, L. M., *Knowledge-Based Situated Agents among Us*, In J. P. Muller, M. J. Wooldridge, and N. R. Jennings (eds.), Intelligent Agents III - Proc. of ATAL'96, the Third International Workshop on Agent Theories, Architectures, and Languages, Budapest, 1996

[LS 96] Liberatore, P. and Schaerf, M., *The complexity of model checking for belief revision and update*, Proc. of AAAI'96, pp. 556 – 561

[MH 69] McCarthy, J. and Hayes, P.J., *Some philosophical problems from the stand-point of artificial intelligence*, in: B. Meltzer and D. Michie, (eds.), Machine Intelligence, 4, Edinburgh, 1969, pp.463-502

[Nebel 91] Nebel, B., *Belief revision and default reasoning: Syntax-based approaches*, in: Allen. J., Fikes, R., and Sandewall, E. (eds.), Proc. of the 2nd Internat. Conf. on Principles of Knowledge Representation and Reasoning, 1991

[PAA 91] Pereira, L. M., Alferes, J. J., and Aparício, J. N., *Nonmonotonic Reasoning with Well Founded Semantics*, In: K. Furukawa (ed.), Proc. of 8th ICLP , MIT Press, 1991, 475-489

[PA 92] Pereira, L. M. and Alferes, J. J., *Well-founded semantics for logic programs with explicit negation*, in European Conf. on Artificial Intelligence, B. Neumann (ed.), John Wiley & Sons, 1992, 102-106

[PAA 93] Pereira, L. M., Aparício, J. N., and Alferes, J. J., *Non–monotonic Reasoning with Logic Programming*, Journal of Logic Programming 17(2,3 & 4), 1993, 227–263

[Rei 87] Reiter, R., *A theory of diagnosis from first principles*, Artificial Intelligence, 32:1,1987, pp. 57-96

[Rei 89] Reiter, R., *Towards a logical reconstruction of relational database theory*, in J. Mylopolous and M. Brodie (eds.), Readings in Artificial Intelligence and Databases, Morgan Kaufmann Publishers,Inc., 1989, pp. 301 – 326

[Rei 93] Reiter, R., *Proving properties of states in the situation calculus*, Artificial Intelligence, 64:2, 1993, pp. 337-351

[SK 95] Sadri, F. and Kowalski, R., *Variants of the event calculus*, Proc. of the ICLP95, 1995

[Satoh 88] Satoh, K., *Nonmonotonic reasoning by minimal belief revision*, In: Proc. of the International Conf. on Fifth Generation Computer Systems, 1988, 455–462

[SI 92] Satoh, K., Iwayama, N., *A query evaluation method for abductive logic programming*, in K. R. Apt (ed.), Logic Programming: Proc. of 1992 Int'l Joint Conference and Symposium, 1992, 671-685

[Win 88] Winslett, M., *Reasoning about action using a possible models approach*, AAAI'88, 1988, 89 – 93

[vGRS 91] Van Gelder, A. , Ross, K. and Schlipf, J.S., *The well-founded semantics for general logic programs*, J. ACM 38 1991, 620-650