Updating Epistemic Logic Programs*

Yan Zhang
Intelligent Systems Laboratory
School of Computing & Mathematics
University of Western Sydney
Penrith South DC, NSW 1797, Australia
yan@scm.uws.edu.au

Abstract

We consider the problem of updating nonmonotonic knowledge bases represented by epistemic logic programs where disjunctive information and notions of knowledge and belief can be explicitly expressed. We propose a formulation for epistemic logic program update based on a principle called minimal change and maximal coherence. The central feature of our approach is that during an update or a sequence of updates, contradictory information is removed on a basis of minimal change under the semantics of epistemic logic programs and then coherent information is maximally retained in the update result. Through various update scenarios, we show that our approach provides both semantic and syntactic characterizations for an update problem. We also investigate essential semantic properties of epistemic logic program update.
Keywords: Epistemic logic programs; non-monotonic reasoning; update

1 Introduction

Reasoning about knowledge system dynamics is one of the central topics in AI research. As a dominant research trend in this aspect, the problem of belief revision and update has been extensively studied in the last decade, and profound theoretical foundations to model agents' belief and knowledge change have been developed through different approaches, e.g. [1, 7, 8, 18]. One challenge of applying these results in practice is to develop computational models for modeling agents' activities in dynamic environments. On the other hand, logic programming has been proved to be one of the most promising logic based formulations for problem solving, knowledge representation and reasoning, and reasoning

*A preliminary version of this paper was published in IJCAI-2003 [24].

about actions and plans. Recent research on logic program update shows that logic programming also provides a computational framework for reasoning about the agents' state change [2, 9, 21, 26].

While various approaches and theories for belief revision and update and logic program update have been proposed towards the modeling of agents' dynamic behaviors, they usually are not used for modeling changes associated with agents' epistemic states, because notions of knowledge and belief are not explicitly expressible in the formalisms of belief revision and update and logic programs. As observed in [6], to precisely model agents' dynamic behaviors, it is essential that both epistemic and actual state changes should be taken into account. Representing and reasoning about agents' epistemic states is the main issue in the research of reasoning about knowledge [10, 20], in which *dynamic epistemic logics* deal with the reasoning problem related to agents' epistemic state change, e.g. [5, 13]. Nevertheless, these logics normally do not specifically consider the actual world change (i.e. belief revision and update), and do not concern the computational process for modeling epistemic state changes.

Gelfond's epistemic logic programs is a significant extension of traditional extended logic programs under answer set semantics. By combining *knowledge* and *belief* operators into logic rules, epistemic logic programming is a powerful representation formalism in logic programming paradigm. It can not only deal with more complex problems in reasoning with incomplete information while traditional disjunctive extended logic programs fail to handle that [12], but also provides a formal basis for modeling knowledge and belief explicitly with a declarative semantics, e.g. [19]. From a dynamics modeling perspective, when we use an epistemic logic program to represent an agent's knowledge base, it is a natural question whether we can develop an update formulation for epistemic logic programs that may be used to model agents' behaviors involving both epistemic and actual states changes.

Updating an epistemic logic program by another epistemic logic program presents some technical challenges. Since the semantics of knowledge and belief in an epistemic logic program is represented by the *world views* of the underlying epistemic logic programs, changes on the program involving knowledge and belief should be carefully considered, while the existing (extended) logic program update approaches do not take this issue into account. In this paper, we propose a formal approach for epistemic logic program update. We require our update formulation to meet two major criteria: (1) an update should be performed on a basis of minimal change principle under the semantics of epistemic logic programs; and (2) based on the minimal change principle, the update result should have a clear syntactic representation and contain maximal information from previous program(s). Clearly, to achieve these criteria, we need to develop our approach from both semantic and syntactic considerations. The main idea we will use in our development is called *minimal change and maximal coherence* which presents both semantic and syntactic features in an update procedure.

We should indicate that traditional minimal change principles used in belief revision and update [15, 23] are not suitable for developing an update approach for epistemic logic

programs. First, let us consider the classical AGM belief revision postulates and Katsuno and Mendelzon's belief update postulates [15]. As it has been shown in [9], extended logic program update generally does not satisfy these postulates. Since epistemic logic programs are an extension of of extended disjunctive logic programs with additional knowledge and belief modal operators, it would not be realistic to develop an update approach for epistemic logic programs satisfying these general postulates. On the other hand, Winslett's *possible model approach* [23] is one of the major approaches for model based belief update that satisfies all Katsuno and Mendelzon's update postulates. However, as we mentioned earlier, since we require that our update approach should reflect the syntactic nature of epistemic logic programs, a pure model based minimal change principle cannot achieve this purpose.

The paper is organized as follows. Section 2 presents a brief overview for epistemic logic programs. Section 3 develops a formalization for epistemic logic program update, while section 4 extends this formalization to handle update sequences. Section 5 investigates important semantic properties for our update approach. Finally, section 6 concludes the paper with discussions on related issues and future work.

2 Epistemic Logic Programs: An Overview

In this section, we present a general overview on epistemic logic programs. Gelfond extended the syntax and semantics of disjunctive logic programs to allow the correct representation of incomplete information in the presence of multiple extensions [12]. Consider the following disjunctive program about the policy of offering scholarships in some university[1]:

$\mathcal{P}$:
r_1: $eligible(x) \leftarrow highGPA(x)$,
r_2: $eligible(x) \leftarrow minority(x), fairGPA(x)$,
r_3: $\neg eligible(x) \leftarrow \neg fairGPA(x), \neg highGPA(x)$,
r_4: $interview(x) \leftarrow not\ eligible(x), not\ \neg eligible(x)$,
r_5: $fairGPA(mike)\ or\ highGPA(mike) \leftarrow$,

while rule r_4 can be viewed as a formalization of the statement: "a student whose eligibility is not decided by rules r_1, r_2 and r_3 should be interviewed by the committee". It is easy to see that $\mathcal{P}$ has two answer sets $\{highGPA(mike), eligible(mike)\}$ and $\{fairGPA(mike), interview(mike)\}$. Therefore the answer to query $interview(mike)$ is *unknown*, which seems too weak from our intuition. Epistemic logic programs will overcome this kind of difficulties in reasoning with incomplete information.

In epistemic logic programs, the language of (disjunctive) extended logic programs is expanded with two modal operators K and M. KF is read as "F is known to be true" and MF is read as "F may be believed to be true". For our purpose, in this paper we

[1]This example was due to Gelfond [12]

will only consider propositional epistemic logic programs where rules containing variables are viewed as the set of all ground rules by replacing these variables with all constants occurring in the language. The semantics for epistemic logic programs is defined by the pair $(\mathcal{A}, W)$, where $\mathcal{A}$ is a collection of sets of ground literals which is also simply called is a collection of *belief sets*, and W is a set in $\mathcal{A}$ called the agent's *working set of beliefs*. The truth of a formula F in $(\mathcal{A}, W)$ is denoted by $(\mathcal{A}, W) \models F$ and the falsity by $(\mathcal{A}, W) =|F$, and are defined as follows.

$(\mathcal{A}, W) \models p$ iff $p \in W$ where p is a propositional atom.
$(\mathcal{A}, W) \models KF$ iff $(\mathcal{A}, W_i) \models F$ for all $W_i \in \mathcal{A}$.
$(\mathcal{A}, W) \models MF$ iff $(\mathcal{A}, W_i) \models F$ for some $W_i \in \mathcal{A}$.
$(\mathcal{A}, W) \models F \wedge G$ iff $(\mathcal{A}, W) \models F$ and $(\mathcal{A}, W) \models G$.
$(\mathcal{A}, W) \models F \; or \; G$ iff $(\mathcal{A}, W) \models \neg(\neg F \wedge \neg G)$.
$(\mathcal{A}, W) \models \neg F$ iff $(\mathcal{A}, W) =|F$.
$(\mathcal{A}, W) =|F$ iff $\neg F \in W$ where F is a ground atom.
$(\mathcal{A}, W) =|KF$ iff $(\mathcal{A}, W) \not\models KF$[2].
$(\mathcal{A}, W) =|MF$ iff $(\mathcal{A}, W) \not\models MF$.
$(\mathcal{A}, W) =|F \wedge G$ iff $(\mathcal{A}, W) =|F$ or $(\mathcal{A}, W) =|G$.
$(\mathcal{A}, W) =|F \; or \; G$ iff $(\mathcal{A}, W) =|F$ and $(\mathcal{A}, W) =|G$.

It is worth mentioning that since belief set W allows both positive and negative propositional atoms, in Gelfond's semantics, $(\mathcal{A}, W) =| \varphi$ is not equivalent to $(\mathcal{A}, W) \not\models \varphi$ in general. For instance, $(\{\{a, b\}\}, \{a, b\}) \not\models c$, but we do not have $(\{\{a, b\}\}, \{a, b\}) =| c$ (i.e. $(\{\{a, b\}\}, \{a, b\}) \models \neg c$). Consequently, here K and M are *not* dual modal operators here[3]. Consider $\mathcal{A} = \{\{a, b\}, \{a, b, \neg c\}\}$. Clearly we have $\mathcal{A} \models \neg K \neg c$. But having $\mathcal{A} \models Mc$ seems to be wrong.

If a formula G is of the form KF, $\neg KF$, MF or $\neg MF$ (where F is a propositional formula), then its truth value in $(\mathcal{A}, W)$ will not depend on W. In this case we call G a *subjective formula*. If F is a propositional literal, then we call KF, $\neg KF$, MF, and $\neg MF$ *subjective literals*. On the other hand, if G does not contain K or M, then its truth value in $(\mathcal{A}, W)$ will only depend on W and we call G an *objective formula* or objective literal if G is a propositional literal. In the case that G is subjective, we simply write $\mathcal{A} \models G$ instead of $(\mathcal{A}, W) \models G$, and $W \models G$ instead of $(\mathcal{A}, W) \models G$ in the case that G is objective. Consider two formulas F and G. We write $F \models G$ if for each $\mathcal{A}$ and each $W \in \mathcal{A}$ such that $(\mathcal{A}, W) \models F$, we have $(\mathcal{A}, W) \models G$.

An *epistemic logic program* is a finite set of rules of the form:

$$F \leftarrow G_1, \cdots, G_m, not\ G_{m+1}, \cdots, not\ G_n. \tag{1}$$

In (1), $m, n \geq 0$, F is of the form $F_1 \; or \; \cdots \; or \; F_k$ ($k \geq 1$) and $F_1, \cdots, F_k$ are objective literals, $G_1, \cdots, G_m$ are objective or subjective literals, and $G_{m+1}, \cdots, G_n$ are objective

[2]We denote $(\mathcal{A}, W) \not\models \varphi$ iff $(\mathcal{A}, W) \models \varphi$ does not hold.
[3]K and M are called *dual* if $\neg K \neg \varphi$ is logically equivalent to $M\varphi$.

literals. For an epistemic logic program $\mathcal{P}$, its semantics is given by its *world view* which is defined in the following steps:
Step 1. Let $\mathcal{P}$ be an epistemic logic program not containing modal operators K and M and negation as failure not. A set W of ground literals is called a *belief set of* $\mathcal{P}$ iff W is a minimal set of satisfying conditions: (i) for each rule $F \leftarrow G_1, \cdots, G_m$ from $\mathcal{P}$ such that $W \models G_1 \wedge \cdots \wedge G_m$ we have $W \models F$; and (ii) if W contains a pair of complementary literals then $W = Lit$, i.e. W is an inconsistent belief set[4].
Step 2. Let $\mathcal{P}$ be an epistemic logic program not containing modal operators K and M and W be a set of ground literals in the language of $\mathcal{P}$. By $\mathcal{P}_W$ we denote the result of (i) removing from $\mathcal{P}$ all the rules containing formulas of the form $not\ G$ such that $W \models G$ and (ii) removing from the rules in $\mathcal{P}$ all other occurrences of formulas of the form $notG$.
Step 3. Finally, let $\mathcal{P}$ be an arbitrary epistemic logic program and $\mathcal{A}$ a collection of sets of ground literals in its language. By $\mathcal{P}_{\mathcal{A}}$ we denote the epistemic logic program obtained from $\mathcal{P}$ by (i) removing from $\mathcal{P}$ all rules containing formulas of the form G such that G is subjective and $\mathcal{A} \not\models G$, and (ii) removing from rules in $\mathcal{P}$ all other occurrences of subjective formulas.

Now we define that a collection $\mathcal{A}$ of sets of ground literals is a *world view* of $\mathcal{P}$ if $\mathcal{A}$ is the collection of all belief sets of $\mathcal{P}_{\mathcal{A}}$. Consider the program $\mathcal{P}$ about the eligibility of scholarship discussed at the beginning of this section, if we replace rule r_4 with the following rule:

$$r_4'\text{: } interview(x) \leftarrow \neg K eligible(x), \neg K \neg eligible(x),$$

then the epistemic logic program that consists of rules r_1, r_2, r_3, r_4', and r_5 will have a unique world view:

$$\{\{highGPA(mike), eligible(mike), interview(mike)\},\\ \{fairGPA(mike), interview(mike)\}\},$$

which will result in a "*yes*" answer to the query $interview(mike)$.

3 Formalizing Epistemic Logic Program Updates

From this section, we begin to develop a formalization for epistemic logic program update. Consider the update of an epistemic logic program $\mathcal{P}_0$ by another epistemic logic program $\mathcal{P}_1$. Our approach consists of two stages: firstly, we update a world view of $\mathcal{P}_0$ by $\mathcal{P}_1$ - this will ensure a minimal change for the underlying update semantics; and secondly, based on the first stage result, we will derive a resulting program which retains the maximal syntactic information represented by $\mathcal{P}_0$.

[4]Note that in our context, a belief set is simply a set of ground literals. Here a belief set of a program is a belief set that satisfies the conditions (i) and (ii).

3.1 Preliminaries

To begin with, we first introduce some useful notions. Let $\mathcal{A}$ be a collection of belief sets and $\mathcal{P}$ an epistemic logic program. We define a pair $\mathcal{M} = (\mathcal{A}, W)$ where $W \in \mathcal{A}$, to be an *epistemic model induced from* $\mathcal{A}$. If $\mathcal{A}$ is a world view of $\mathcal{P}$, then $(\mathcal{A}, W)$ is also called an epistemic model of $\mathcal{P}$. Clearly, for each $\mathcal{A}$, there are $\|\mathcal{A}\|$ models of $\mathcal{P}$ induced from $\mathcal{A}$ (here $\|\mathcal{A}\|$ is the cardinality of set $\mathcal{A}$). We use $ind(\mathcal{A})$ to denote the set of all epistemic models induced from $\mathcal{A}$.

Consider a rule r of the form (1). We use $H(r)$ and $B(r)$ to denote the head and body parts of rule r respectively. For instance, for rule

$$r\text{: } a \text{ or } \neg b \leftarrow c, Kd, not\ \neg e,$$

we have $H(r) = \{a, \neg b\}$, and $B(r) = \{c, Kd, not \neg e\}$. Recall that $H(r)$ is *satisfied* in an epistemic model $\mathcal{M}$, i.e. $\mathcal{M} \models H(r)$, iff $\mathcal{M} \models l$, for some $l \in H(r)$. For $B(r)$, since it contains negation as failure operator *not*, we extend its satisfaction denotation as follows. $B(r)$ is *satisfied* in $\mathcal{M}$, denoted as $\mathcal{M} \models B(r)$, if (1) for each objective or subjective literal $l \in B(r)$ $\mathcal{M} \models l$, and (2) for each $not\ l \in B(r)$ $\mathcal{M} \not\models l$. r *is satisfied* in $\mathcal{M}$, denoted as $\mathcal{M} \models r$, if $\mathcal{M} \models B(r)$ implies $\mathcal{M} \models H(r)$. An epistemic logic program $\mathcal{P}$ is *satisfied* in $\mathcal{M}$ if each rule of $\mathcal{P}$ is satisfied in $\mathcal{M}$. $\mathcal{P}$ is *satisfied* in a collection of belief sets $\mathcal{A}$ if $\mathcal{P}$ is satisfied in all epistemic models induced from $\mathcal{A}$.

A collection of belief sets is *consistent* if each of its belief sets is consistent, and is *non-redundant* if it does not contains two belief sets W_1 and W_2 such that $W_1 \subseteq W_2$. An epistemic logic program is *consistent* if it has a world view and all of its world views are consistent. By $\mathbf{V}(\mathcal{P})$ we denote the set of all non-redundant collections of belief sets satisfying $\mathcal{P}$. We also denote the set of all world views of $\mathcal{P}$ as $\mathbf{A}(\mathcal{P})$. Clearly $\mathbf{A}(\mathcal{P}) \subseteq \mathbf{V}(\mathcal{P})$. Under the context of epistemic logic program update, we will only consider the problem of updating a consistent program by another consistent program or a sequence of consistent programs.

3.2 Minimal Change on World Views Updates

Let W and W_1 be two belief sets. We denote $Diff(W, W_1)$ to be the set $|(W \setminus W_1) \cup (W_1 \setminus W)|$[5]. Our method for updating world views shares a similar spirit of traditional model based update [23] by defining a closeness relation between world views, and we require that the resulting world view after the update should be as close as possible to the original world view.

Definition 1 ***(Closeness)*** *Let $\mathcal{A}$, $\mathcal{A}_1$ and $\mathcal{A}_2$ be three collections of belief sets. We say $\mathcal{A}_1$ is at least as close to $\mathcal{A}$ as $\mathcal{A}_2$, denoted as $\mathcal{A}_1 \leq_{\mathcal{A}} \mathcal{A}_2$, iff for any $W \in \mathcal{A}$ and $W_2 \in \mathcal{A}_2$, there exists some $W_1 \in \mathcal{A}_1$ such that $Diff(W, W_1) \subseteq Diff(W, W_2)$. We denote $\mathcal{A}_1 <_{\mathcal{A}} \mathcal{A}_2$ if $\mathcal{A}_1 \leq_{\mathcal{A}} \mathcal{A}_2$ and $\mathcal{A}_2 \not\leq_{\mathcal{A}} \mathcal{A}_1$.*

[5]For a set of literals W, $|W| = \{|l| \mid l \in W\}$ and here $|l|$ is l's corresponding propositional atom, i.e. $|l| = a$ if l is a or $\neg a$.

Proposition 1 *$\leq_{\mathcal{A}}$ defined in Definition 1 is a pre-ordering.*

Proof: From Definition 1, it is easy to see that $\leq_{\mathcal{A}}$ is reflexive. Now we prove the transitivity. Assume that $\mathcal{A}_1 \leq_{\mathcal{A}} \mathcal{A}_2$ and $\mathcal{A}_2 \leq_{\mathcal{A}} \mathcal{A}_3$. Clearly, we know that from $\forall W \in \mathcal{A}$, $W_2 \in \mathcal{A}_2$, $\exists W_1 \in \mathcal{A}_1$ $Diff(W, W_1) \subseteq Diff(W, W_2)$; and $\forall W \in \mathcal{A}$, $W_3 \in \mathcal{A}_3$, $\exists W_2 \in \mathcal{A}_2$ $Diff(W, W_2) \subseteq Diff(W, W_3)$. Then we have $\forall W \in \mathcal{A}$, $W_3 \in \mathcal{A}_3$, we always have particular W_1 in $\mathcal{A}_1$ and W_2 in $\mathcal{A}_2$ as indicated above such that $Diff(W, W_1)$ $\subseteq Diff(W, W_2) \subseteq Diff(W, W_3)$. This follows that $\forall W \in \mathcal{A}$ and $W_3 \in \mathcal{A}_3$, $\exists W_1 \in \mathcal{A}_1$ such that $Diff(W, W_1) \subseteq Diff(W, W_3)$. This proves $\leq_{\mathcal{A}}$'s transitivity. □

Now we consider how a world view $\mathcal{A}$ of some program can be updated by a specific program $\mathcal{P}$. Intuitively, the result of this type of update is a new collection of belief sets, say $\mathcal{A}'$, which should satisfy $\mathcal{P}$ and have a minimal difference from $\mathcal{A}$. Consider a simple example. Let $\mathcal{A} = \{\{a, b\}, \{a, c\}\}$ and $\mathcal{P} = \{\neg b \vee c \leftarrow Ka\}$. When we update $\mathcal{A}$ by $\mathcal{P}$, we would expect to obtain an intuitive result like $\mathcal{A}' = \{\{a, \neg b\}, \{a, c\}\}$, which, according to Definition 1, has a minimal difference from $\mathcal{A}$. However, note that $\mathcal{A}'' = \{\{\neg a, b\}, \{a, c\}\}$ also satisfies $\mathcal{P}$ and has a minimal difference from $\mathcal{A}$, but $\mathcal{A}''$ should not be a desirable result because no information is presented in $\mathcal{P}$ to change a.

To capture such intuition of the world view update, we need to compare two collections of belief sets $\mathcal{A}'$ and $\mathcal{A}''$, where both of them are in $\mathbf{V}(\mathcal{P})$, in terms of which one is more consistent with $\mathcal{A}$ with respect to program $\mathcal{P}$. That is, if $\mathcal{A}'$ satisfies more rules r in $\mathcal{P}$ than $\mathcal{A}''$ in some sense under the consideration of $\mathcal{A}$.

To formalize this concept, we define a notion as follows. Let $\mathcal{A}$, $\mathcal{A}'$ be two collections of belief sets, and $\mathcal{P}$ a program, where $\mathcal{A}' \in \mathbf{V}(\mathcal{P})$. We define

1. $R(\mathcal{A}, \mathcal{P}, \mathcal{A}')^0 = \{r \mid r \in \mathcal{P}$, such that for each induced epistemic model $\mathcal{M}$ from $\mathcal{A}$ satisfying $\mathcal{M} \models B(r)$, there is an induced epistemic model $\mathcal{M}'$ from $\mathcal{A}'$ satisfying $\mathcal{M}' \models B(r) \wedge H(r)\}$;

2. $R(\mathcal{A}, \mathcal{P}, \mathcal{A}')^{i+1} = R(\mathcal{A}, \mathcal{P}, \mathcal{A}')^i \cup \Delta$, where $R(\mathcal{A}, \mathcal{P}, \mathcal{A}')^i \neq \emptyset$ and $\Delta = \{r \mid r \in \mathcal{P}$, where $B(r) = B'(r) \wedge B_1(r) \wedge \cdots \wedge B_k(r)$, such that for each induced epistemic model $\mathcal{M}$ from $\mathcal{A}$ satisfying $\mathcal{M} \models B'(r)$, there are rules $r_1, \cdots, r_k \in R(\mathcal{A}, \mathcal{P}, \mathcal{A}')^i$ such that for each induced epistemic model $\mathcal{M}'$ satisfying (1) $\mathcal{M}' \models B(r_j) \wedge H(r_j)$ implies $\mathcal{M}' \models B_j(r)$ $(j = 1, \cdots, k)$, and (2) $\mathcal{M}' \models B(r) \wedge H(r)\}$.

3. $R(\mathcal{A}, \mathcal{P}, \mathcal{A}') = \bigcup_{i=1}^{\infty} R(\mathcal{A}, \mathcal{P}, \mathcal{A}')^i$.

Let us take a closer look at notion $R(\mathcal{A}, \mathcal{P}, \mathcal{A}')$. Intuitively, $R(\mathcal{A}, \mathcal{P}, \mathcal{A}')$ tries to capture the rules that make required changes in $\mathcal{A}'$, by giving $\mathcal{A}$ in which some induced epistemic models make these rules' bodies hold. First, $R(\mathcal{A}, \mathcal{P}, \mathcal{A}')$ contains those rules from $\mathcal{P}$ that force direct changes of $\mathcal{A}$ in order to satisfy these rules in $\mathcal{A}'$ (condition 1 in $R(\mathcal{A}, \mathcal{P}, \mathcal{A}')$ definition). Then these rules form a basis to trigger more rules from $\mathcal{P}$ to be satisfied by $\mathcal{A}'$ so that more indiect changes are caused in $\mathcal{A}$. In particular, if a rule's body is only partially

satisfied in some induced epistemic model from $\mathcal{A}$, but other parts of its body have been derived from previous rules that cause changes of $\mathcal{A}$, then this rule will be triggered as well to derive further changes of $\mathcal{A}$ (condition 2 in $R(\mathcal{A}, \mathcal{P}, \mathcal{A}')$ definition). So $R(\mathcal{A}, \mathcal{P}, \mathcal{A}')$ actually contains the rules from $\mathcal{P}$ that enforce all direct and indirect changes of $\mathcal{A}$ in the present of $\mathcal{A}'$. To illustrate the intuition behind this definition clearly, let us consider the following example.

Example 1 Let $\mathcal{A} = \{\{a, b\}\}$, and $\mathcal{P}$ a program consisting of the following rules:

$r_1 : c \; or \; d \leftarrow Ka, not \; f,$
$r_2 : e \leftarrow b, Mc.$

Suppose $\mathcal{A}' = \{\{a, b, c, e\}, \{a, b, d, e\}\}$. Now we compute $R(\mathcal{A}, \mathcal{P}, \mathcal{A}')$ according to the previous definition. Firstly, it is easy to see that $R(\mathcal{A}, \mathcal{P}, \mathcal{A}')^0 = \{r_1\}$ since $\mathcal{A}$ has a unique induced epistemic model $\mathcal{M} = (\mathcal{A}, \{a, b\})$, $\mathcal{M} \models (Ka, not \; e)$, and for the induced epistemic models $\mathcal{M}' = (\mathcal{A}', \{a, b, c, e\})$ and $\mathcal{M}'' = (\mathcal{A}', \{a, b, d, e\})$ from $\mathcal{A}'$, $\mathcal{M}' \models (c \; or \; d) \wedge (Ka, not \; e)$ and $\mathcal{M}'' \models (c \; or \; d) \wedge (Ka, not \; e)$. Then we consider rule r_2. We observe that $\mathcal{M} \models b$, $\mathcal{M}' \models B(r_1) \wedge H(r_1)$ and $\mathcal{M}' \models Mc$, and $\mathcal{M}'' \models B(r_1) \wedge H(r_1)$ and $\mathcal{M}'' \models Mc$. Furthermore, $\mathcal{M}' \models e$ and $\mathcal{M}'' \models e$. So according to the specification of $R(\mathcal{A}, \mathcal{P}, \mathcal{A}')^{i+1}$, we have $R(\mathcal{A}, \mathcal{P}, \mathcal{A}')^1 = R(\mathcal{A}, \mathcal{P}, \mathcal{A}')^0 \cup \{r_2\}$. Therefore, $R(\mathcal{A}, \mathcal{P}, \mathcal{A}') = \{r_1, r_2\}$. □

We say that $\mathcal{A}'$ is *as $\mathcal{A}$-consistent as $\mathcal{A}''$* with respect to $\mathcal{P}$ if $R(\mathcal{A}, \mathcal{P}, \mathcal{A}'') \subseteq R(\mathcal{A}, \mathcal{P}, \mathcal{A}')$, and $\mathcal{A}'$ is *maximally* $\mathcal{A}$-consistent with respect to $\mathcal{P}$ if there does not exist other $\mathcal{A}''$ such that $R(\mathcal{A}, \mathcal{P}, \mathcal{A}') \subset R(\mathcal{A}, \mathcal{P}, \mathcal{A}'')$ (proper set inclusion).

Definition 2 ***(Updating world views)*** *Let $\mathcal{A}$ be a world view of some program. A collection of belief sets $\mathcal{A}'$ (not necessary a world view of a particular program) is a* possible result of updating $\mathcal{A}$ by a program $\mathcal{P}$*, if $\mathcal{A}'$ satisfies the following conditions:*

1. *$\mathcal{A}' \models \mathcal{P}$ (i.e. $\mathcal{A}' \in \mathbf{V}(\mathcal{P})$);*
2. *$\mathcal{A}'$ is maximally $\mathcal{A}$-consistent with respect to $\mathcal{P}'$; and*
3. *there does not exist another collection of belief sets $\mathcal{A}''$ satisfying conditions 1 and 2, and $\mathcal{A}'' <_{\mathcal{A}} \mathcal{A}'$.*

We use $Res(\mathcal{A}, \mathcal{P})$ to denote the set of all possible results of updating $\mathcal{A}$ by $\mathcal{P}$.

Example 2 We first look at a simple example. Let $\mathcal{A} = \{\{a\}\}$. Consider the update of $\mathcal{A} = \{\{a\}\}$ with program $\mathcal{P}$ consisting of the following two rules:

r_1: $p \leftarrow Ka, not \; q,$
r_2: $q \leftarrow Ka, not \; p.$

It is easy to see that there are four collections of belief sets that satisfy $\mathcal{P}$[6]:

$\mathcal{A}_1 = \{\{a, p\}, \{a, q\}\}$,
$\mathcal{A}_2 = \{\{a, p\}\}$,
$\mathcal{A}_3 = \{\{a, q\}\}$, and
$\mathcal{A}_4 = \{\{\}\}$.

Then from the definition of $R(\mathcal{A}, \mathcal{P}, \mathcal{A}')$, we can see that

$R(\mathcal{A}, \mathcal{P}, \mathcal{A}_1) = \{r_1, r_2\}$,
$R(\mathcal{A}, \mathcal{P}, \mathcal{A}_2) = \{r_1\}$,
$R(\mathcal{A}, \mathcal{P}, \mathcal{A}_3) = \{r_2\}$, and
$R(\mathcal{A}, \mathcal{P}, \mathcal{A}_4) = \{\}$.

So $\mathcal{A}_1$ is the only maximally $\mathcal{A}$-consistent collection of belief sets with respect to $\mathcal{P}$. From Definition 2, we know that $\mathcal{A}_1 = \{\{a, p\}, \{a, q\}\}$ is the unique result of the update of $\mathcal{A}$ with $\mathcal{P}$. □

Example 3 Consider two epistemic logic programs $\mathcal{P}_1$ and $\mathcal{P}_2$ where

$\mathcal{P}_1$:
  $a \leftarrow$,
  $\neg b \leftarrow not\ c$,
  $c \leftarrow not\ \neg b$, and
$\mathcal{P}_2$:
  $b\ or\ c \leftarrow Ka$,

Clearly, $\mathcal{P}_1$ has one world view $\mathcal{A} = \{\{a, \neg b\}, \{a, c\}\}$. Updating $\mathcal{A}$ by $\mathcal{P}_2$, according to Definition 2, we obtain one resulting world views $\mathcal{A}' = \{\{a, b\}, \{a, c\}\}$. Note that $\mathcal{A}'' = \{\{a, c\}\}$ is not a result. Although $\mathcal{A}'' \models \mathcal{P}_2$, and $\mathcal{A}''$ is maximally $\mathcal{A}$-consistent with respect to $\mathcal{P}_2$, $\mathcal{A}' <_{\mathcal{A}} \mathcal{A}''$ because $Diff(\{a, \neg b\}, \{a, b\}) = \{b\} \subset \{b, c\} = Diff(\{a, \neg b\}, \{a, c\})$. Also note that $\mathcal{A}''' = \{\{\neg a, \neg b\}, \{a, c\}\}$ is not a result either because $\mathcal{A}'''$ is not maximally $\mathcal{A}$-consistent with respect to $\mathcal{P}_2$. □

Example 4 We consider another program $\mathcal{P}$ as follows:

$r_1 : a \leftarrow$,
$r_2 : b \leftarrow Ka, b$,
$r_3 : c \leftarrow Ka, \neg b$.

Suppose $\mathcal{A} = \{\{\neg a, \neg b, \neg c\}\}$ is a world view of some program, we consider the update of $\mathcal{A}$ by $\mathcal{P}$. Then from Definition 2, it can be verified that $\mathcal{A}' = \{\{a, \neg b, c\}\}$ is the unique result from this update. In particular, we have $R(\mathcal{A}, \mathcal{P}, \mathcal{A}') = \{r_1, r_3\}$, from which we can see that $\mathcal{A}'$ is maximally $\mathcal{A}$-consistent with respect to $\mathcal{P}$.

Note that $\mathcal{A}'' = \{\{a, b, \neg c\}\}$ is not maximally $\mathcal{A}$-consistent with respect to $\mathcal{P}$ because $R(\mathcal{A}, \mathcal{P}, \mathcal{A}'') = \{r_1\}$ which is a proper subset of $R(\mathcal{A}, \mathcal{P}, \mathcal{A}') = \{r_1, r_3\}$. □

[6]We assume that a, p, q are the only atoms in the language of $\mathcal{P}$.

Now we study some basic properties of world view updates. Given a world view (collection of belief sets) $\mathcal{A}$ and a program $\mathcal{P}$, we denote $Min(\mathbf{V}(\mathcal{P}), \leq_{\mathcal{A}})$ to be a subset of $\mathbf{V}(\mathcal{P})$ containing all minimal elements with respect to the ordering $\leq_{\mathcal{A}}$. The following result directly follows from Definition 2.

Proposition 2 $Res(\mathcal{A}, \mathcal{P}) \subseteq Min(\mathbf{V}(\mathcal{P}), \leq_{\mathcal{A}})$.

From the above three examples, we observe that for each case, all changes of a world view are specifically enforced by the underlying update program and no any extra change will be generated. Considering Example 1 for instance, although rules r_1 and r_2 conflict with each other, each of them still makes a corresponding change once its body is satisfied. To characterize this property in general, we first introduce a useful notion.

Let $\mathcal{P}$ be a program and a rule $r \in \mathcal{P}$. We use $head\_literal(r)$ and $neg\_literal(r)$ to denote the sets of propositional literals occurring in the head and negative body of rule r respectively. By $pos\_literal(r)$, we mean the set of literals occurring in the body with positive modal operators K or M in front, or without K or M in front, *together with* the negations of all literals that have $\neg K$ or $\neg M$ in front. For instance, if $\neg K \neg a$ occurs in the body of rule r, then a will be in $pos\_literal(r)$. Let rule $r_p \in \mathcal{P}$. We define a notion $D(r_p)$ as follows:

$D(r_p)^0 = \{r_p\}$;
$D(r_p)^i = D(r_p)^{i-1} \cup \{r \mid$ there exists some $r' \in D(r_p)^{i-1}$ such that
$head\_literal(r') \cap pos\_literal(r) \neq \emptyset$
(if $pos\_literal(r) \neq \emptyset$) and
$head\_literal(r') \not\subseteq neg\_literal(r)\}$;
$D(r_p) = \bigcup_{i=0}^{\infty} D(r_p)^i$.

Intuitively, $D(r_p)$ represents a sequence of rules $r_1, r_2, \cdots, r_k$ ($r_1 = r_p$) in $\mathcal{P}$ that forms a *causal chain* initiated by rule r_p. That is, if rule $r_1 = r_p$ is triggered, then possibly rules $r_2, \cdots, r_k$ will be triggered as well.

Note that condition $head\_literal(r') \not\subseteq neg\_literal(r)$ is necessary in the above definition in order to capture actual causal changes triggered by r_p. Consider the following program:

$r_1 : a \ or \ b \ or \ c \leftarrow$,
$r_2 : d \leftarrow Mc, not \ b$.

We have $D(r_1) = \{r_1, r_2\}$. Clearly, applying rule r_1 will trigger rule r_2 under certain circumstance - when c is possibly derived, d will be also derived consequently. So this progam has a unique world view $\{\{a, d\}, \{b\}, \{c, d\}\}$.

It is also worth mentioning that in an epistemc logic program, even if $pos\_literal(r) \cap neg\_literal(r) \neq \emptyset$, rule r may still contribute to the generation of the world view of the underlying program, and therefore it will play a role in generating certain world view update result. This will be shown in the following example.

Example 5 Let $\mathcal{A} = \{\{\}\}$ and $\mathcal{P}$ a program as follows:

$r_1 : a \; or \; b \leftarrow$,
$r_2 : c \leftarrow Ma, not \; a$.

Here $pos\_literal(r_2) \cap neg\_literal(r_2) = \{a\}$. We consider the update of $\mathcal{A}$ by $\mathcal{P}$. First we can see that $D(r_1) = \{r_1, r_2\}$, and $\mathcal{P}$ has a unique world view $\mathcal{A}' = \{\{a\}, \{b, c\}\}$. Then by the definition of $R(\mathcal{A}, \mathcal{P}, \mathcal{A}')$, we can see that $R(\mathcal{A}, \mathcal{P}, \mathcal{A}') = \{r_1, r_2\}$. Clearly, $\mathcal{A}'$ is maximally $\mathcal{A}$-consistent with respect to $\mathcal{P}$. In fact, it is not hard to observe that $\mathcal{A}'$ is the unique result of updating $\mathcal{A}$ by program $\mathcal{P}$. □

Let us take a closer look at Example 3 once again. In Example 3, we have two causal chains starting from rule r_1 in program $\mathcal{P}$: $D(r_1) = \{r_1, r_2\}$ and $D'(r_1) = \{r_1, r_3\}$. Giving $\mathcal{A} = \{\{\neg a, \neg b, \neg c\}\}$, we can see that each rule in $R(\mathcal{A}, \mathcal{P}, \mathcal{A}') = \{r_1, r_3\}$ is also represented in $D'(r_1) = \{r_1, r_3\}$. In other words, all changes from $\mathcal{A}$ to $\mathcal{A}'$ are triggered by certain rules in some causal chains from $\mathcal{P}$. In general, we have the following result.

Proposition 3 *Let $\mathcal{A}, \mathcal{P}$ be specified as in Definition 2, and $\mathcal{A}' \in Res(\mathcal{A}, \mathcal{P})$. If an induced epistemic model $\mathcal{M}' \in ind(\mathcal{A}')$ which is not in $ind(\mathcal{A})$, then there exists a rule $r_p \in \mathcal{P}$ such that for some $\mathcal{M} \in ind(\mathcal{A})$, the following conditions hold: (1) $\mathcal{M} \models B(r_p)$, and (2) for some $D(r_p)$ and $r \in D(r_p)$, $\mathcal{M}' \models B(r) \wedge H(r)$.*

Proof: If for some $\mathcal{M} \in ind(\mathcal{A})$, $\mathcal{M} \models B(r_p)$, then from the definitions of $R(\mathcal{A}, \mathcal{P}, \mathcal{A}')$ and $D(r_p)$, we know that for each $r' \in D(r_p)$, $r' \in R(\mathcal{A}, \mathcal{P}, \mathcal{A}')$ iff for some $\mathcal{M}' \in ind(\mathcal{A}')$, $\mathcal{M}' \models B(r') \wedge H(r')$. Since we know that $\mathcal{M}' \notin ind(\mathcal{A})$, this implies that $R(\mathcal{A}, \mathcal{P}, \mathcal{A}') \neq \emptyset$. Otherwise we would have $\mathcal{A} = \mathcal{A}'$ from Definition 2. So there is a rule $r_p \in R(\mathcal{A}, \mathcal{P}, \mathcal{A}')^0$ such that $\mathcal{M} \models B(r_p)$ for some $\mathcal{M} \in ind(\mathcal{A})$. This infers that for some $D(r_p)$ and some $r \in D(r_p)$ (note that r could be r_p), $\mathcal{M}' \models B(r) \wedge H(r)$. □

The following example further illustrates a situation where indirect changes may lead to multiple updating results during a world view update.

Example 6 Let $\mathcal{A} = \{\{e\}\}$ be a world view of some program. We consider the update of $\mathcal{A}$ by program $\mathcal{P}$:

$r_1 : d \leftarrow$,
$r_2 : b \; or \; c \leftarrow Kd, \neg Ma$,
$r_3 : a \; or \; c \leftarrow Kd, \neg Mb$.

In this program, we have $D(r_1) = \{r_1, r_2\}$ and $D'(r_1) = \{r_1, r_3\}$. Rule r_1 will cause a direct change on $\mathcal{A}$, and this change will further enforce an indirect change from rule r_2 or r_3, but not both. In particular, consider two collections of belief sets $\mathcal{A}_1 = \{\{b, d, e\}, \{c, d, e\}\}$ and $\mathcal{A}_2 = \{\{a, d, e\}, \{c, d, e\}\}$ respectively. We have $R(\mathcal{A}, \mathcal{P}, A_1) = \{r_1, r_2\}$, and $R(\mathcal{A}, \mathcal{P}, \mathcal{A}_2) = \{r_1, r_3\}$, both are maximally $\mathcal{A}$-consistent with respect to $\mathcal{P}$. Note that due to the

mutual exclusion of rules r_2 and r_3, there does not exist a consistent collection $\mathcal{A}'$ of belief sets that such that both r_2 and r_3 are in $R(\mathcal{A}, \mathcal{P}, \mathcal{A}')$. So from Definition 2, it is easy to see that $\mathcal{A}_1$ and $\mathcal{A}_2$ are the two possible results after updating $\mathcal{A}$ by $\mathcal{P}$. From Proposition 3, we know that indeed changes represented by $\mathcal{A}_1$ are caused by rules in $D(r_1)$ while changes represented by $\mathcal{A}_2$ are caused by rules in $D'(r_1)$. □

3.3 Maximal Coherence and Resulting Programs

As discussed earlier, during the second stage of an update procedure, we need to derive a resulting program which should retain the maximal syntactic information represented by the initial program. This is achieved by introducing the concept of coherence. Let $\mathcal{M} = (\mathcal{A}, W)$ be an epistemic model and r a rule of the form (1) such that $\mathcal{M} \models r$. We denote $\mathcal{S}(\mathcal{M}, H(r)) = \{l \mid l \in H(r) \cap W \text{ and } \mathcal{M} \models B(r)\}$. Intuitively, if $\mathcal{M} \models B(r)$, then $\mathcal{S}(\mathcal{M}, H(r))$ presents all literals occurring in $H(r)$ that are true in W. For example, let $\mathcal{M} = (\{\{a, b\}, \{a, c\}, \{a, b, c\}\}, \{a, b, c\})$ and $r : b \text{ or } c \leftarrow Ka$, then $\mathcal{S}(\mathcal{M}, H(r)) = \{b, c\}$. In the case of $\mathcal{M} \not\models B(r)$, we have $\mathcal{S}(\mathcal{M}, H(r)) = \emptyset$.

Definition 3 ***(Subsumption)*** *Given two collections of belief sets $\mathcal{A}_1$ and $\mathcal{A}_2$ and a program $\mathcal{P}$. $\mathcal{A}_2$ is subsumed by $\mathcal{A}_1$ with respect to $\mathcal{P}$, denoted as $\mathcal{A}_2 \subseteq_{\mathcal{P}} \mathcal{A}_1$, iff $\mathcal{A}_1 \models \mathcal{P}$ implies $\mathcal{A}_2 \models \mathcal{P}$ and for each $\mathcal{M}_2 \in ind(\mathcal{A}_2)$ there exists some $\mathcal{M}_1 \in ind(\mathcal{A}_1)$ such that for each $r \in \mathcal{P}$, $\mathcal{S}(\mathcal{M}_2, H(r)) \subseteq \mathcal{S}(\mathcal{M}_1, H(r))$.*

Given $\mathcal{A}_1 \models \mathcal{P}$, if $\mathcal{A}_2$ is subsumed by $\mathcal{A}_1$ with respect to $\mathcal{P}$, then each rule r in $\mathcal{P}$ is also satisfied in $\mathcal{A}_2$ without increasing the satisfied literals of $H(r)$ in the epistemic models induced from $\mathcal{A}_2$. Informally, $\mathcal{A}_2 \subseteq_{\mathcal{P}} \mathcal{A}_1$ means that $\mathcal{A}_2$ is *not a larger* collection of belief sets than $\mathcal{A}_1$ to satisfy $\mathcal{P}$. For instance, let $\mathcal{P} = \{r : a \text{ or } b \leftarrow Kc\}$, $\mathcal{A}_1 = \{\{a, b, c, \}, \{b, c, d\}\}$ and $\mathcal{A}_2 = \{\{a, b\}, \{b, c, e\}\}$, then $\mathcal{A}_2 \subseteq_{\mathcal{P}} \mathcal{A}_1$. Clearly, $\mathcal{A}_2$ is *smaller* than $\mathcal{A}_1$ in the sense that fewer literals in $H(r)$ are satisfied in belief sets of $\mathcal{A}_2$.

Definition 4 ***(Coherence)*** *Let $\mathcal{P}$ and $\mathcal{P}'$ be two programs and $\mathcal{A} \in \mathbf{V}(\mathcal{P})$. $\mathcal{P}'$ is* coherent *with $\mathcal{P}$ with respect to $\mathcal{A}$ iff for each $\mathcal{A}' \in Res(\mathcal{A}, \mathcal{P}')$, $\mathcal{A}'$ is consistent and $\mathcal{A}' \subseteq_{\mathcal{P}} \mathcal{A}$.*

The intuitive meaning of coherence is explained as follows. Given the collection of belief sets $\mathcal{A}$ and programs $\mathcal{P}$ and $\mathcal{P}'$ where $\mathcal{A} \models \mathcal{P}$. Updating $\mathcal{A}$ by $\mathcal{P}'$ will change $\mathcal{A}$ to other possible collections of belief sets that satisfy $\mathcal{P}'$. Coherence ensures that such changes do not violate the satisfaction of $\mathcal{P}$, i.e. for each $\mathcal{A}' \in Res(\mathcal{A}, \mathcal{P}')$ $\mathcal{A}' \models \mathcal{P}$. Furthermore, it also ensures that $\mathcal{A}'$ is subsumed by $\mathcal{A}$ with respect to $\mathcal{P}$ (i.e. $\mathcal{A}'$ is not larger than $\mathcal{A}$ in terms of $\mathcal{P}$'s satisfaction). Definition 4 also implies that to be coherent, both program $\mathcal{P}$ and $\mathcal{P}'$ must be consistent.

Now the resulting program can be specified by the following definition.

Definition 5 ***(Resulting programs)*** *Let $\mathcal{P}_1$ and $\mathcal{P}_2$ be two programs. An epistemic logic program $\mathcal{P}'$ is a* possible resulting program *after updating $\mathcal{P}_1$ by $\mathcal{P}_2$, iff there exists some $\mathcal{A} \in \mathbf{A}(\mathcal{P}_1)$, $\mathcal{P}' = \mathcal{P}_1^* \cup \mathcal{P}_2$, where $\mathcal{P}_1^*$ is a maximal subset of $\mathcal{P}_1$ such that for each $\mathcal{A}' \in Res(\mathcal{A}, \mathcal{P}_2)$ $\mathcal{P}_1^*$ is coherent with $\mathcal{P}_2$ with respect to $\mathcal{A}'$.*

Example 7 Example 3 continued. Note that $\mathcal{A} = \{\{a, \neg b\}, \{a, c\}\}$ is the unique world view of $\mathcal{P}_1$. After updating $\mathcal{A}$ by $\mathcal{P}_2$, we have $Res(\mathcal{A}, \mathcal{P}_2) = \{\mathcal{A}'\} = \{\{a, b\}, \{a, c\}\}$ as shown in Example 1. Then it can be verified that $\{a \leftarrow\}$ is the only maximal subset of $\mathcal{P}_1$ that is coherent with $\mathcal{P}_2$ with respect to $\{\{a, b\}, \{a, c\}\}$. Therefore, from Definition 5, we have

$\mathcal{P}'$:
$a \leftarrow$,
$b \; or \; c \leftarrow Ka$,

from which, we conclude that $\mathcal{P}'$ has a unique world view $\{\{a, b\}, \{a, c\}\}$. □

Example 8 Consider a scenario that John is being investigated in relation to a crime. The following rules have been applied to decide whether or not John is innocent. If it is not believed that John is involved in the crime, then John is clear. If it is known that John is clear, then John is innocent. On the other hand, if it is not believed that John is clear, then John is involved in the crime, and if it is known that John is involved in the crime, then John is a suspect. These statements can be represented in the following epistemic logic program:

$\mathcal{P}_1$:
$clear(john) \leftarrow \neg Mbe\_involved(john)$,
$innocent(john) \leftarrow Kclear(john)$,
$be\_involved(john) \leftarrow \neg Mclear(john)$,
$suspected(john) \leftarrow Kbe\_involved(john)$.

Now suppose that John is either not clear or not being involved in the crime:

$\mathcal{P}_2$:
$\neg clear(john) \; or \; \neg be\_involved(john) \leftarrow$.

We consider to update $\mathcal{P}_1$ by $\mathcal{P}_2$. It is easy to see that $\mathcal{P}_1$ has two world views:

$\mathcal{A}_1 = \{\{clear(john), innocent(john)\}\}$ and
$\mathcal{A}_2 = \{\{be\_involved(john), suspected(john)\}\}$.

We first update $\mathcal{A}_1$ by $\mathcal{P}_2$. From Definition 2, we have $Res(\mathcal{A}_1, \mathcal{P}_2) = \{\mathcal{A}_{11}, \mathcal{A}_{12}\}$, where
$\mathcal{A}_{11} = \{\{\neg clear(john), innocent(john)\},$
$\{clear(john), \neg be\_involved(john), innocent(john)\}\}$, and
$\mathcal{A}_{12} = \{\{\neg clear(john), be\_involved(john), suspected(john)\},$
$\{\neg be\_involved(john), suspected(john)\}\}$.
Then it can be verified that a program consisting of the following two rules

$\mathcal{P}^*$:
$innocent(john) \leftarrow Kclear(john)$,
$suspected(john) \leftarrow Kbe\_involved(john)$

is the maximal subset of $\mathcal{P}_1$ that is coherent with $\mathcal{P}_2$ with respect to $\mathcal{A}_{11}$ and $\mathcal{A}_{12}$. Therefore, from Definition 5 we have:

$\mathcal{P}'$:
$innocent(john) \leftarrow Kclear(john)$,
$suspected(john) \leftarrow Kbe\_involved(john)$,
$\neg clear(john)$ *or* $\neg be\_involved(john) \leftarrow$

is a resulting program after updating $\mathcal{P}_1$ by $\mathcal{P}_2$. From the update of $\mathcal{A}_2$ by $\mathcal{P}_2$, on the other hand, we will obtain exactly the same resulting program $\mathcal{P}'$. That is, $\mathcal{P}'$ is the unique resulting program after updating $\mathcal{P}_1$ by $\mathcal{P}_2$, which has a unique world view $\{\{\neg clear(john)\}, \{\neg be\_involved(john)\}\}$. It is also noted that program $\mathcal{P}_1 \cup \mathcal{P}_2$ is not consistent as it does not have a world view. □

4 Handling Update Sequences

In this section, we extend our previous update formulation to handle update sequences where more than two programs are involved. Let $\mathcal{P}_1, \cdots, \mathcal{P}_k$ be consistent programs. $\mathbf{P} = (\mathcal{P}_1, \cdots, \mathcal{P}_k)$ is called an *update sequence*. Informally performing this update sequence means that program $\mathcal{P}_1$ is sequentially updated by $\mathcal{P}_2$, $\cdots$, and $\mathcal{P}_k$. From our intuition, during this process we would like to assign $\mathcal{P}_j$ a higher priority of persistence than $\mathcal{P}_i$ where $j > i$ since $\mathcal{P}_j$ represents the agent's newly received knowledge comparing to $\mathcal{P}_i$. Therefore, after the performance of this update sequence, it is desirable to achieve a result which maximally contains information represented by $\mathcal{P}_k$, $\mathcal{P}_{k-1}$, $\cdots$, $\mathcal{P}_1$ with *progressively decreasing priorities*. In other words, we would like to have a minimal change principle for performing an update sequence that is of progressively decreasing priority degrees on $\mathcal{P}_k$, $\mathcal{P}_{k-1}$, $\cdots$, $\mathcal{P}_1$. Similarly to our previous update formulation, we will formalize this principle for handling update sequence from both semantic and syntactic considerations. The idea is illustrated by the following example.

Example 9 Consider an update sequence $\mathbf{P} = (\mathcal{P}_1, \mathcal{P}_2, \mathcal{P}_3)$, where

$\mathcal{P}_1$:
r_1: $b \leftarrow Ka$,
r_2: $\neg b \leftarrow Ka$,
$\mathcal{P}_2$:
r_3: $a \leftarrow$,
$\mathcal{P}_3$:
r_4: $\neg a$ *or* $\neg b \leftarrow$.

To perform update sequence $\mathbf{P}$, we first update $\mathcal{P}_1$ by $\mathcal{P}_2$, which, according to the approach developed earlier, will generate two resulting programs: $\mathcal{P}' = \{r_1, r_3\}$ and $\mathcal{P}'' = \{r_2, r_3\}$. Now we consider to update $\mathcal{P}'$ and $\mathcal{P}''$ by $\mathcal{P}_3$ respectively. However, this time, we cannot simply apply our previous update approach because both $\mathcal{P}'$ and $\mathcal{P}''$ contain rules from $\mathcal{P}_1$ and $\mathcal{P}_2$ where rules from $\mathcal{P}_2$ should have a higher priority to be maintained during the change. For instance, consider the update of $\mathcal{P}'$ by $\mathcal{P}_3$. $\mathcal{P}'$ has one world view $\mathcal{A}' = \{\{a, b\}\}$. Then updating $\mathcal{A}'$ by $\mathcal{P}_3$ will generate a new world view $\mathcal{A}'' = \{\{\neg a, b\}, \{a, \neg b\}\}$. Clearly, both $\{r_1\}$ and $\{r_3\}$ are the maximal subsets of $\mathcal{P}'$ that are coherent with $\mathcal{P}_3$ with respect to $\mathcal{A}''$. However, as we discussed above, we should only have a final resulting program $\{r_3\} \cup \mathcal{P}_3$ because $\mathcal{P}_2$ has a higher priority than $\mathcal{P}_1$ to be persistent during the update. Updating $\mathcal{P}''$ by $\mathcal{P}_3$, on the other hand, will result in a program $\mathcal{P}'' \cup \mathcal{P}_3$ as there is no conflict between $\mathcal{P}''$ and $\mathcal{P}_3$. □

Let $\mathbf{P} = (\mathcal{P}_1, \cdots, \mathcal{P}_k)$ be an update sequence and $\mathcal{P}'$ a subset of $\mathcal{P}_1 \cup \cdots \cup \mathcal{P}_k$. We denote $\mathcal{P}'[\mathbf{P}, i] = \{r \mid r \in \mathcal{P}_i\}$ $(1 \leq k)$. Note that $\mathcal{P}'[\mathbf{P}, i] \subseteq \mathcal{P}_i$. Now our ideas of performing update sequences are formalized by the following definitions.

Definition 6 ***(Dominant subsets)*** *Let $\mathbf{P} = (\mathcal{P}_1, \cdots, \mathcal{P}_k)$ be an update sequence and $\mathcal{P}'$ and $\mathcal{P}''$ two subsets of $\mathcal{P}_1 \cup \cdots \cup \mathcal{P}_k$. We say that $\mathcal{P}'$* dominates *$\mathcal{P}''$ with respect to $\mathbf{P}$, denoted as $\mathcal{P}' \ll_{\mathbf{P}} \mathcal{P}''$, if (i) $\mathcal{P}'' \subset \mathcal{P}'$; or (ii) there exist some i, j where $1 \leq i < j \leq k$ such that for all l $(j \leq l \leq k)$, $\mathcal{P}''[\mathbf{P}, l] \subseteq \mathcal{P}'[\mathbf{P}, l]$, for some l' $(j \leq l' \leq k)$ $\mathcal{P}''[\mathbf{P}, l'] \subset \mathcal{P}'[\mathbf{P}, l']$, and $\mathcal{P}'[\mathbf{P}, i] \subset \mathcal{P}''[\mathbf{P}, i]$ (proper set inclusion).*

Let $\mathbb{P}$ be the collection of all epistemic logic programs. For each $i \geq 1$, we define π_i to be a i-ary *update selection function* if π_i is a mapping:

$$\pi_i : \underbrace{\mathbb{P} \times \cdots \times \mathbb{P}}_{i} \longrightarrow \mathbb{P}$$

Definition 7 ***(Well formed update selection functions)*** *For each $i \geq 1$, π_i is a* well formed *update selection function iff the following conditions hold:*

1. *if $i = 1$, then for any update sequence $\mathbf{P} = (\mathcal{P}_1)$, $\pi_1(\mathbf{P}) = \mathcal{P}_1$;*
2. *if $i = 2$, then for any update sequence $\mathbf{P} = (\mathcal{P}_1, \mathcal{P}_2)$, $\pi_2(\mathbf{P}) = \mathcal{P}'$, where $\mathcal{P}'$ is a resulting program as described in Definition 5;*
3. *if $i = k$ and $k > 2$, then for any update sequence $\mathbf{P} = (\mathcal{P}_1, \cdots, \mathcal{P}_k)$, $\pi_k(\mathbf{P}) = \mathcal{P}^* \cup \mathcal{P}_k$, where $\mathcal{P}^*$ is obtained as follows:*
   - *(a) let $\mathbf{P}_1 = (\mathcal{P}_1, \cdots, \mathcal{P}_{k-1})$, π_{k-1} be a $(k-1)$-ary well formed update selection function, and $\pi_{k-1}(\mathbf{P}_1) = \mathcal{P}'$;*
   - *(b) let $\mathcal{A}$ be a world view of $\mathcal{P}'$, and $\mathcal{P}^*$ is a maximal subset of $\mathcal{P}'$ such that for all $\mathcal{A}' \in Res(\mathcal{A}, \mathcal{P}_k)$, $\mathcal{P}^*$ is coherent with $\mathcal{P}_k$ with respect to $\mathcal{A}'$;*

(c) there does not exist another maximal subset $\mathcal{P}^\dagger$ of $\mathcal{P}'$ that satisfies condition (b) and $\mathcal{P}^\dagger \ll_{\mathbf{P}_1} \mathcal{P}^*$[7].

Example 10 Consider an irrigation system which has the following general rules to decide whether the plants should be watered:

> *If it is likely to be raining next day, then we do not need to water the plants*;
> *If there is no evidence showing that soil is dry, then we do not need to water the plants*.

It is also assumed that currently the soil is dry or it will not be raining next day. This scenario can be represented by the following program $\mathcal{P}_1$:

$\mathcal{P}_1$:
r_1: $\neg watering \leftarrow not\ \neg to\_be\_raining$,
r_2: $\neg watering \leftarrow not\ dry$,
r_3: $dry\ or\ \neg to\_be\_raining \leftarrow$.

$\mathcal{P}_1$ has one world view:

$$\{\{dry, \neg watering\}, \{\neg to\_be\_raining, \neg watering\}\},$$

from which it is concluded that we do not need to water the plants. However, from a conservative viewpoint for plants' growth, this result is rather optimistic because r_3 does not represent an exclusive disjunctive information. Therefore, we consider to update $\mathcal{P}_1$ by $\mathcal{P}_2$:

$\mathcal{P}_2$:
r_4: $watering \leftarrow \neg K \neg dry, M \neg to\_be\_raining$,
r_5: $\neg watering \leftarrow \neg dry$,
r_6: $\neg watering \leftarrow to\_be\_raining$,

which says that if it is not known that the soil is moist (not dry) and it is believed that it will not be raining the next day, then we water the plants; we do not need to water the plants if the soil is moist or it will be raining next day. After a period of time, suppose new information is further received that is represented by $\mathcal{P}_3$ as follows:

$\mathcal{P}_3$:
r_7: $\neg dry\ or\ to\_be\_raining \leftarrow$.

[7]Note that here we refer to the same world view $\mathcal{A}$ of $\mathcal{P}'$ as in (b) when we say $\mathcal{P}^\dagger$ is coherent with $\mathcal{P}_k$ with respect to all $\mathcal{A}'$ in $Res(\mathcal{A}, \mathcal{P}_k)$.

Now we consider the update sequence $\mathbf{P} = (\mathcal{P}_1, \mathcal{P}_2, \mathcal{P}_3)$. Let $\mathbf{P}_1 = (\mathcal{P}_1, \mathcal{P}_2)$. Then according to Definition 7, the unique well formed update selection function π_2 gives the result: $\pi_2(\mathbf{P}_1) = \mathcal{P}' = \{r_3, r_4, r_5, r_6\}$, where rules r_1 and r_2 are removed from the initial program $\mathcal{P}_1$. It is noted that $\mathcal{P}'$ has one world view $\{\{dry, watering\}, \{\neg to\_be\_raining, watering\}\}$, from which it is concluded that we need to water the plants. Finally, we obtain a unique resulting program $\pi_3(\mathbf{P}) = \{r_4, r_5, r_6, r_7\}$, from which it is concluded that we no longer need to water the plants, i.e. $\pi_3(\mathbf{P}) \models \neg watering$. □

5 Semantic Characterizations

In this section, we study important semantic properties of our approach for epistemic logic program update. Without specific declaration, in the rest of this section we will only consider consistent programs, and the update sequences consisting of consistent programs. Also, when we mention two arbitrary update selection functions π_k and π_{k+1}, we always mean that π_{k+1} is an arbitrary extension of an arbitrary selection function π_k, i.e. π_{k+1} is formed based on π_k.

Let $\mathbf{P} = (\mathcal{P}_1, \cdots, \mathcal{P}_k)$ be an update sequence and $\mathcal{P}$ a program, by $(\mathbf{P}, \mathcal{P})$ we denote the update sequence $(\mathcal{P}_1, \cdots, \mathcal{P}_k, \mathcal{P})$. By $Body(\mathcal{P})$ and $Head(\mathcal{P})$ we denote the sets of all objective literals occurring in bodies and heads of rules in $\mathcal{P}$ respectively[8].

Theorem 1 *Let $\mathbf{P} = (\mathcal{P}_1, \cdots, \mathcal{P}_k)$ be an update sequence, $\mathcal{P}$ a program, and π_k and π_{k+1} two k-ary and $(k+1)$-ary well formed update selection functions respectively. Then the following properties hold:*

1. $\mathbf{V}(\pi_{k+1}(\mathbf{P}, \mathcal{P})) \subseteq \mathbf{V}(\mathcal{P})$;
2. *if* $\mathbf{V}(\pi_k(\mathbf{P})) \subseteq \mathbf{V}(\mathcal{P})$, *then* $\pi_{k+1}(\mathbf{P}, \mathcal{P}) = \pi_k(\mathbf{P}) \cup \mathcal{P}$.

Proof: We prove Result 1. Since $\mathcal{P} \subseteq \pi_{k+1}(\mathbf{P}, \mathcal{P})$, this follows that for each $\mathcal{A} \in \mathbf{V}(\pi_{k+1}(\mathbf{P}, \mathcal{P}))$, $\mathcal{A} \models \mathcal{P}$. So $\mathcal{A} \in \mathbf{V}(\mathcal{P})$. That is, $\mathbf{V}(\pi_{k+1}(\mathbf{P}, \mathcal{P})) \subseteq \mathbf{V}(\mathcal{P})$.

Now we prove Result 2. According to Definition 7, $\pi_{k+1}(\mathbf{P}, \mathcal{P}) = \mathcal{P}^* \cup \mathcal{P}$, where $\mathcal{P}^*$ is a maximal subset of $\pi_k(\mathbf{P})$ satisfying conditions (b) and (c) in Definition 7. We show $\mathcal{P}^* = \pi_k(\mathbf{P})$. Since $\mathbf{V}(\pi_k(\mathbf{P})) \subseteq \mathbf{V}(\mathcal{P})$, it is clear that for each world view $\mathcal{A}$ of $\pi_k(\mathbf{P})$, $\pi_k(\mathbf{P})$ is coherent with $\mathcal{P}$ with respect to all $\mathcal{A}' \in Res(\mathcal{A}, \mathcal{P})$. That is, $\pi_k(\mathbf{P})$ itself is the maximal subset of $\pi_k(\mathbf{P})$ satisfying condition (b). On the other hand, it is easy to see that there does not exist any other subset $\mathcal{P}'$ of $\pi_k(\mathbf{P})$ such that $\mathcal{P}' \ll_{\mathbf{P}} \pi_k(\mathbf{P})$. So $\pi_k(\mathbf{P})$ also satisfies condition (c). This follows that $\pi_{k+1}(\mathbf{P}, \mathcal{P}) = \pi_k(\mathbf{P}) \cup \mathcal{P}$. □

Theorem 1 presents two essential properties of our update approach. Property 1 simply says that after update, the set of collections of belief sets that satisfy the resulting program

[8]Recall that all such literals are ground since we restrict to a propositional language.

shrinks, just like other knowledge base update and logic program update approaches, e.g. [9, 23, 26]. Property 2, on the other hand, provides a conditional syntactic expression for the result of an update sequence. The following theorem provides a sufficient condition under which the computation of a $(k+1)$-length update sequence can be reduced to the computation on a k-length update sequence.

Theorem 2 *Let $\mathbf{P}$ be an update sequence with a length of k, $\mathcal{P}$ and $\mathcal{P}'$ two programs, π_k, π_{k+1} two k-ary and $(k+2)$-ary arbitrary well formed update selection functions respectively. If $\mathbf{V}(\pi_k(\mathbf{P})) \subseteq \mathbf{V}(\mathcal{P}) \subseteq \mathbf{V}(\mathcal{P}')$, then $\pi_{k+2}(\mathbf{P}, \mathcal{P}, \mathcal{P}') = \pi_{k+1}(\mathbf{P}, \mathcal{P} \cup \mathcal{P}')$.*

Proof: From the condition, we have $\mathbf{V}(\pi_k(\mathbf{P})) \subseteq \mathbf{V}(\mathcal{P} \cup \mathcal{P}')$. Then from Theorem 1, $\pi_{k+1}(\mathbf{P}, \mathcal{P} \cup \mathcal{P}') = \pi_k(\mathbf{P}) \cup \mathcal{P} \cup \mathcal{P}'$. On the other hand, from $\mathbf{V}(\pi_k(\mathbf{P})) \subseteq \mathbf{V}(\mathcal{P})$ and Theorem 1, we have $\pi_{k+1}(\mathbf{P}, \mathcal{P}) = \pi_k(\mathbf{P}) \cup \mathcal{P}$. Now we consider $\pi_{k+2}(\mathbf{P}, \mathcal{P}, \mathcal{P}')$. From Definition 7, we have $\pi_{k+2}(\mathbf{P}, \mathcal{P}, \mathcal{P}') = \mathcal{P}^* \cup \mathcal{P}'$, where $\mathcal{P}^* \subseteq \pi_k(\mathbf{P}) \cup \mathcal{P}$ and $\mathcal{P}^*$ satisfies conditions (b) and (c) in Definition 7. That is, for some world view $\mathcal{A}$ of $\pi_k(\mathbf{P}) \cup \mathcal{P}$, $\mathcal{P}^*$ is the maximal subset of $\pi_k(\mathbf{P}) \cup \mathcal{P}$ that is coherent with $\mathcal{P}'$ with respect to all $\mathcal{A}' \in Res(A, \mathcal{P}')$. Note that since $\mathcal{A} \in \mathbf{V}(\pi_k(\mathbf{P}) \cup \mathcal{P})$, it follows that $\mathcal{A} \in \mathbf{V}(\mathcal{P}')$ as well. Therefore, $Res(\mathcal{A}, \mathcal{P}') = \{\mathcal{A}\}$. This follows that $\mathcal{P}^* = \pi_k(\mathbf{P}) \cup \mathcal{P}$. Clearly, there is no any other subset of $\pi_k(\mathbf{P}) \cup \mathcal{P}$ dominates $\pi_k(\mathbf{P}) \cup \mathcal{P}$ itself. So $\pi_k(\mathbf{P}) \cup \mathcal{P}$ also satisfies condition (c). Hence, we have $\pi_{k+2}(\mathbf{P}, \mathcal{P}, \mathcal{P}') = \pi_k(\mathbf{P}) \cup \mathcal{P} \cup \mathcal{P}'$. This proves our result. □

Now we investigate two important properties called *persistence* and *preservation* for epistemic logic program update. Informally, the persistence property ensures that if a literal is derivable from a program, then after updating this program, this literal is still derivable from the resulting program. The preservation property, on the other hand, says that if a literal is derivable from a program, then updating other program by this program will still preserve this literal's inference from the resulting program. While the persistence property is usually not valid for classical belief revision and update due to their nonmonotonicity, the preservation property, on the other hand, does hold for classical belief revision and update. It is easy to see that neither of these two properties holds for logic program update. Although persistence and preservation properties do not hold generally for epistemic logic program update, it is always worthwhile to explore their restricted forms because under certain conditions, these properties may significantly reduce the computational cost for answering a query to the update result. We first present the following lemma.

Lemma 1 *Let $\mathcal{P}$ be a program and L a ground literal. Suppose $\mathcal{P}'$ is a subset of $\mathcal{P}$ such that $\mathcal{P}' \models L$ and $Body(\mathcal{P}') \cap Head(\mathcal{P} \setminus \mathcal{P}') = \emptyset$. Then $\mathcal{P} \models L$.*

Proof: The proof for this lemma is involved in proving a result called *splitting theorem* for epistemic logic programs which inherits a similar feature of the splitting theorem for extended logic programs [17]. We first state the result as follows:

Result 1: *Let $\mathcal{P}$ be a program and $\mathcal{P}' \subseteq \mathcal{P}$. If $Body(\mathcal{P}') \cap Head(\mathcal{P} \setminus \mathcal{P}') = \emptyset$, then for any world view $\mathcal{A}$ of $\mathcal{P}$ and every belief set W in $\mathcal{A}$, there exists a belief set W' that is in some world view $\mathcal{A}'$ of $\mathcal{P}'$ such that $W' \subseteq W$.*

Now we prove this result. Suppose $\mathcal{A}$ is a world view of $\mathcal{P}$. Then by performing Step 3 transformation of the world view definition (see section 2), $\mathcal{P}_{\mathcal{A}}$ is a disjunctive extended logic program. Let ${\mathcal{P}'}_{\mathcal{A}}$ be the part obtained from the transformation on $\mathcal{P}'$. So we have $Body({\mathcal{P}'}_{\mathcal{A}}) \cap Head(\mathcal{P}_{\mathcal{A}} \setminus {\mathcal{P}'}_{\mathcal{A}}) = \emptyset$. Then from Theorem 4.3 in [26], we know that for each belief set W in $\mathcal{A}$, $W = W_{{\mathcal{P}'}_{\mathcal{A}}} \cup W'$, where $W_{{\mathcal{P}'}_{\mathcal{A}}}$ is a belief set of ${\mathcal{P}'}_{\mathcal{A}}$, and W' is a belief set of program $e(\mathcal{P}_{\mathcal{A}} \setminus {\mathcal{P}'}_{\mathcal{A}}, W_{{\mathcal{P}'}_{\mathcal{A}}})$, where $e(\mathcal{P}, X)$ is a disjunctive extended logic program obtained from a disjunctive extended logic program $\mathcal{P}$ by deleting (1) all rules with $not\ L$ occurring in the body and $L \in X$; and (2) L in the remaining rules with $L \in X$. So for each $W \in \mathcal{A}$, there is a corresponding $W_{{\mathcal{P}'}_{\mathcal{A}}}$ such that $W_{{\mathcal{P}'}_{\mathcal{A}}} \subseteq W$.

On the other hand, from the condition that $Body(\mathcal{P}') \cap Head(\mathcal{P} \setminus \mathcal{P}') = \emptyset$, we know that program $\mathcal{P}'$ must have a world view, otherwise it will conclude $\mathcal{P}$ has no world view that contradicts the consistency of $\mathcal{P}$. Now we show that the collection $\mathcal{A}'$ of all such $W_{{\mathcal{P}'}_{\mathcal{A}}}$ must be a world view of $\mathcal{P}'$. This is easy to see. Since each $W_{{\mathcal{P}'}_{\mathcal{A}}}$ in $\mathcal{A}'$ is generated from program ${\mathcal{P}'}_{\mathcal{A}}$, it simply follows that ${\mathcal{P}'}_{\mathcal{A}'} = {\mathcal{P}'}_{\mathcal{A}}$. Then as $\mathcal{A}'$ is the collection of all answer sets (belief sets) of program ${\mathcal{P}'}_{\mathcal{A}'}$, $\mathcal{A}'$ is a world view of $\mathcal{P}'$ according to the definition (see section 2).

Having **Result 1**, the lemma is proved as follows. Consider program $\mathcal{P}'$. Since $\mathcal{P}' \models L$, for each world view $\mathcal{A}'$ of $\mathcal{P}'$, $\mathcal{A}' \models L$. That is, for each $W' \in \mathcal{A}'$, $L \in W'$. Suppose $\mathcal{P} \not\models L$. Then there exists a world view $\mathcal{A}$ of $\mathcal{P}$ such that $\mathcal{A} \not\models L$. Then there must be some belief set $W \in \mathcal{A}$ such that $L \notin W$. However, since $Body(\mathcal{P}') \cap Head(\mathcal{P} \setminus \mathcal{P}') = \emptyset$, from the above result, there exists a world view $\mathcal{A}^*$ of $\mathcal{P}'$ such that $W^* \subseteq W$ for some $W^* \in \mathcal{A}^*$. Obviously this contradicts the fact $L \notin W$. So we have $\mathcal{P} \models L$. □

Form Lemma 1, we can prove the following two useful properties.

Theorem 3 ***(*Persistence property*)*** *Let $\mathbf{P}$ be an update sequence with a length of k, $\mathcal{P}$ a program, L a ground literal, and π_k and π_{k+1} two arbitrary k-ary and $(k+1)$-ary well formed update selection functions respectively. Suppose $\pi_k(\mathbf{P}) \models L$. Then $\pi_{k+1}(\mathbf{P}, \mathcal{P}) \models L$ if $\mathbf{V}(\pi_k(\mathbf{P})) \subseteq \mathbf{V}(\mathcal{P})$ and $Body(\pi_k(\mathbf{P})) \cap Head(\mathcal{P}) = \emptyset$.*

Proof: Since $\mathbf{V}(\pi_k(\mathbf{P})) \subseteq \mathbf{V}(\mathcal{P})$, from Theorem 1, we have $\pi_{k+1}(\mathbf{P}, \mathcal{P}) = \pi_k(\mathbf{P}) \cup \mathcal{P}$. On the other hand, from condition $Body(\pi_k(\mathbf{P})) \cap Head(\mathcal{P}) = \emptyset$ and Lemma 1, it follows $\pi_{k+1}(\mathbf{P}, \mathcal{P}) \models L$. □

Theorem 4 ***(*Preservation property*)*** *Let $\mathbf{P} = (\mathcal{P}_1, \cdots, \mathcal{P}_k)$ be an update sequence with a length of k, $\mathcal{P}$ a program, L a ground literal, and π_k and π_{k+1} two arbitrary k-ary and $(k+1)$-ary well formed update selection functions respectively. Suppose $\mathcal{P} \models L$. Then $\pi_{k+1}(\mathbf{P}, \mathcal{P}) \models L$ if $Head(\bigcup_{i=1}^{k} \mathcal{P}_i) \cap Body(\mathcal{P}) = \emptyset$.*

Proof: Clearly $\mathcal{P} \subseteq \Pi_{k+1}(\mathbf{P}, \mathcal{P})$. On the other hand, we have $\pi_{k+1}(\mathbf{P}, \mathcal{P}) = \mathcal{P}^* \cup \mathcal{P}$, where $\mathcal{P}^* \subseteq \pi_k(\mathbf{P}) \subseteq \bigcup_{i=1}^{k} \mathcal{P}_i$ and satisfies conditions (b) and (c) in Definition 7. So we have $Head(\mathcal{P}^*) \cap Body(\mathcal{P}) = \emptyset$. Then directly from Lemma 1, we have $\pi_{k+1}(\mathbf{P}, \mathcal{P}) \models L$. □

6 Related Work and Conclusions

As we indicated in Section 1, in recent years many formulations for (extended) logic program update have been proposed. Basically, there are three different types of approaches for logic program update: model based approach [2, 3, 4, 9], syntax based approach [21] and integrating both model and syntax based approach [25, 26]. Relations between these different approaches have been also thoroughly studied in [9, 16, 25, 26]. Since all these approaches were based on answer set semantics, they are generally not applicable for specifying epistemic logic program update. It is also not clear yet how these approaches can be extended to capture the precise semantics of program changes when knowledge and belief are explicitly presented. This is worth to studying further.

In this paper, we proposed a formulation for epistemic logic program update. Our update formalization was developed based on the principle of minimal change and maximal coherence. By using our approach, not only a minimal change semantics is embedded into the underlying update procedure, but also a maximal syntactic coherence is achieved after the update. We also investigated important semantic properties of our update approach.

Several interesting issues remain for our future work. As knowledge plays an important role in action theories, and epistemic logic programs may be used as a main component in such knowledge based action theories, e.g. [19, 22], we expect that these action theories could be enhanced by integrating our update approach on epistemic logic programs. Another interesting topic is that we can further generalize our current update framework to model dynamic behaviors of a multi-agent knowledge system. Suppose we have a number of epistemic logic programs where each program represents one agent's knowledge base. It is then an important question how these agents communicate with each other and make decisions. This problem involves conflict resolution, knowledge (belief) update and merging. One possible way to handle this problem is to develop a formulation of *mutual epistemic logic program update*: each agent uses his knowledge base to update the other's, and decisions could be made by merging all these resulting programs. We expect that research on this direction would lead to the development of a computational framework for modeling complex agents' activities involving epistemic representations such as negotiations and games [11, 14].

References

[1] C.E. Alchourron, P. Gardenfors and D. Makinson, On the logic of theory change: Partial meet contraction and revision functions. *Journal of Symbolic Logic*. 50 (1985) 510-530.

[2] J. Alferes, L. Pereira, H. Przymusinski, and T. Przymusinski, Dynamic updates of non-monotonic knowledge bases. *Journal of Logic Programming* 45(1-3) (2000) 43-70.

[3] J.J Alferes, L.M. Pereira, H. Przymusinska, and T. Przymusinski, LUPS - A language for updating logic programs. *Artificial Intelligence* 138 (2002) 87-116.

[4] J.J. Alferes and L.M. Pereira, Updates plus preference. In *Proceedings of JELIA2000*, 2000.

[5] A. Baltag, A logic for suspicious players: Epistemic actions and belief-updates in games. *Bulletin of Economic Research* 54 (2002) 1-45.

[6] C. Baral and Y. Zhang, Knowledge updates: Semantic and complexity issues. *Artificial Intelligence* 164 (2005) 209-243.

[7] R. Booth, T. Meyer and K. Wong, A bad day surfing is better than a good day working. In *Proceedings of the 10th International Conference on Knowledge Representation and Reasoning (KR 2006)*, pp 230-238. AAAI Press, 2006.

[8] J. Delgrande, J. Long and T. Schaub, Belief change based on global minimisation. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007)*, pp 2462-2476. AAAI Press 2007.

[9] T. Eiter, M. Fink, G. Sabbatini, and H. Tompits, On properties of update sequences based on causal rejection. *Theory and Practice of Logic Programming* 2 (2002) 711-767.

[10] R. Fagin, J.Y. Halpern, Y. Moses and M.Y. Vardi, *Reasoning about Knowledge*. MIT Press, 1995.

[11] N.Y. Foo, T. Meyer, Y. Zhang and D. Zhang, Negotiating logic programs. In *Proceedings of the 6th Workshop on Nonmonotonic Reasoning, Action and Change (NRAC 2005)*, pp 39-44, 2005.

[12] M. Gelfond, Logic programming and reasoning with incomplete information. *Annals of Mathematics and Artificial Intelligence* 12 (1994) 98-116.

[13] J.D. Gerbrandy, Dynamic epistemic logic. In *Proceedings of STASS Workshop on Logic, Language and Computation*, 1997.

[14] P. Harrenstein, W. van der Hoek, J.-J. Meyer and C. Witteveen, On modal logic interpretations of games. In *Proceedings of the 15th European Conference on Artificial Intelligence (ECAI 2002)*, pp 28-32. IOS Press 2002.

[15] H. Katsuno and A. Mendelzon, On the difference between updating a knowledge base and revising it. In *Proceedings of of KR-91*, pp387-394, 1991.

[16] J.A. Leite, *Evolving Knowledge Bases*. IOP Press, 2003.

[17] V. Lifschitz and H. Turner, Splitting a logic program. In *Proceedings of Eleventh International Conference on Logic Programming*, pp 23-37. MIT Press, 1994.

[18] P. Liberatore and M. Schaerf, Belief revision and update: Complexity of model checking. *Journal of Computer and System Sciences*. 62 (2001) 43-72.

[19] J. Lobo, G. Mendez, and S.R. Taylor Knowledge and the action description language $\mathcal{A}$. *Theory and Practice of Logic Programming* 1 (2001) 129-184.

[20] J.-J.Ch. Meyer and W. van der Hoek, *Epistemic Logic for AI and Computer Science*. Cambridge University Press, 1995.

[21] C. Sakama and K. Inoue, Updating extended logic programs through abduction. In *Proceedings of the 5th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'99)*, LNAI, Vol. 1730, pp 147-161. Springer, 1999.

[22] R. Scherl and H. Levesque, Knowledge, action, and the frame problem. *Artificial Intelligence* 114 (2003) 1-40.

[23] M. Winslett, Reasoning about action using a possible models approach. In *Proceedings of AAAI-88*, pp 89-93, 1988.

[24] Y. Zhang, Minimal change and maximal coherence for epistemic logic program updates. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI-03)*, pp 112-117. Morgan Kaufmann Publishers, Inc., 2003.

[25] Y. Zhang and N. Foo, A unified framework for representing logic program updates. In *Proceedings of the 20th National Conference on Artificial Intelligence (AAAI-2005)*, pp 707-712. AAAI Press, Inc., 2005.

[26] Y. Zhang, Logic program based updates. *ACM Transaction on Computational Logic*. 7 (2006) 421-472.