Nested Epistemic Logic Programs

Kewen Wang[1] and Yan Zhang[2]

[1] Griffith University, Australia
`k.wang@griffith.edu.au`
[2] University of Western Sydney
`yan@cit.uws.edu.au`

Abstract. Nested logic programs and epistemic logic programs are two important extensions of answer set programming. However, the relationship between these two formalisms is rarely explored. In this paper we first introduce the epistemic HT-logic, and then propose a more general extension of logic programs called *nested epistemic logic programs*. The semantics of this extension - named *equilibrium views* - is defined on the basis of the epistemic HT-logic. We prove that equilibrium view semantics extends both the answer sets of nested logic programs and the world views of epistemic logic programs. Therefore, our work establishes a unifying framework for both nested logic programs and epistemic logic programs. Furthermore, we also provide a characterization of the strong equivalence of two nested epistemic logic programs.

1 Introduction

Answer set programming (ASP) [6] was developed in the late of 1990s and has been widely recognized as a promising tool for effective knowledge representation and declarative problem solving [1]. ASP is based on the *answer set semantics* of logic programs introduced by Gelfond and Lifschitz [4, 5]. The formal systems for ASP may have different features such as default negation, explicit negation, disjunction and preference. Normal, general, extended and disjunctive logic programs are among the major ASP formalisms.

As many researchers (including [3, 8]) have noticed, the languages of logic programming are still insufficient in representing commonsense knowledge. Recently answer set semantics has been extended to nested logic programs [8], in which arbitrarily nested formulas are allowed. On the other hand, ASP is also expanded to epistemic logic programs by Gelfond [3], where belief operators can be explicitly presented so that incomplete information may be correctly represented in the extent of multiple belief sets. The semantics of epistemic logic programs is defined as the collection of its *world views*, which are generalizations of the answer sets for logic programs without nested expressions.

Having examined the syntax and semantics of nested logic programs and epistemic logic programs, people may observe an important fact: Although the world view semantics of epistemic logic programs generalizes the answer set semantics for disjunctive (extended) logic programs, it, however, cannot be used as the semantics for the epistemic logic programs with nested expressions containing belief operators. Hence, the following two problems remain open:

1. As two extensions of ASP, can nested logic programs and epistemic logic programs be unified in one common language?
2. Can nested expressions of formulas with belief operators be allowed in both the head and body of rules in an epistemic logic program?

We observe that existing nonmonotonic epistemic logics, such as [9], do not provide direct solutions to the above problems. As it will be illustrated in the following, since the world view semantics is defined based on a transformation from epistemic logic programs to disjunctive logic programs by eliminating the belief operators in the body of rules, it is not feasible to simply allow belief operators to occur in the head of rules in an epistemic logic program. Hence, it seems to be inevitable to develop a new approach to solve these problems instead of seeking for a straightforward extension of Gelfond's world view semantics.

This paper aims to solve these problems in a unified manner. In particular, we first introduce a new logic called *epistemic equilibrium HT-logic*. This logic is a natural integration of the equilibrium HT-logic [10, 11] and modal logic. Based on this logic, we then specify a more general extension of logic programs called *nested epistemic logic programs (NELPs)*. The semantics of this extension - named *equilibrium views* - is defined on the basis of the epistemic HT-logic. We prove that equilibrium view semantics extends both the answer sets of nested logic programs and the world views of epistemic logic programs. Therefore, our work establishes a unifying framework for both nested logic programs and epistemic logic programs. Furthermore, we also provide a characterization of the strong equivalence of two nested epistemic logic programs.The main results of this paper are summarized as follows:

1. Equilibrium view semantics extends the answer set semantics of nested logic programs;
2. Equilibrium view semantics extends the world view semantics of epistemic logic programs;
3. Two nested epistemic logic programs are strong equivalent if and only if they are equivalent in the epistemic HT-logic.

The rest of this paper is organized as follows. Section 2 proposes a new logic called epistemic HT logic and defines its semantics. Section 3 presents a new logic programming language (i.e. nested epistemic logic programs) which extends both the nested logic programs and epistemic logic programs. Section 4 proves two major results to show that the equilibrium view semantics generalizes both the answer set semantics for nested logic programs and the world view semantics for epistemic logic programs. Section 5 further proves a result about the strong equivalence of nested epistemic logic programs. Section 6 discusses how to add the second negation into the new class of programs. Finally, section 7 concludes the paper with some discussions.

2 Epistemic HT-Logic and Equilibrium Models

2.1 Syntax and Semantics

The language of our epistemic HT-logic will be the modal language which extends the classical propositional language by means of two belief operators K and M. We consider classical propositional formulas built from propositional atoms and the 0-place connective $\bot$ ("false") using the binary connectives $\vee$, $\wedge$ and $\rightarrow$. We use $\top$ for $\bot \leftarrow \bot$, and $\neg F$ for $F \rightarrow \bot$ where F is a formula. Modal formulas are obtained by addition of the following clauses to the usual inductive definition of propositional formulas:

– If F is a formula, then KF is a formula;
– If F is a formula, then MF is a formula.

KF is read as "F is known to be true" and MF is read as "F may be believed to be true". If a is an atom in the classical propositional logic, then an objective literal is either a or $\neg a$, and both Kb and Mb are called *subjective literals* for any objective literal b. Similarly, F is a *subjective formula* if F contains at least one belief operator. An *epistemic theory* is a (finite) set of formulas in the language of epistemic HT-logic.

The HT-logic (i.e. the logic of here-and-there) is also known as "the logic of present and future", which is basically a three-valued logic. Pearce first used this logic to characterize the answer set semantics of logic programs [11]. More recently, Lifschitz, Pearce and Valverde characterize the strong equivalence of logic programs through the HT-logic [7]. In the following we extend the semantics of the HT logic to the epistemic HT-logic. So in our logic, we will have two tenses (H and T) and two belief operators (K and F).

Let $\mathcal{A}$ be a collection of sets of (ground) atoms. An *epistemic HT-interpretation* is defined as an ordered tuple $(\mathcal{A}, I^H, I^T)$ where I^H, I^T are sets of atoms with $I^H \subseteq I^T$. If $I^H = I^T$, we say $(\mathcal{A}, I^H, I^T)$ is *total*. Notice that we do not require $I^H \in \mathcal{A}$ or $I^T \in \mathcal{A}$.

For any epistemic HT-interpretation $(\mathcal{A}, I^H, I^T)$, any tense $t \in \{H, T\}$, and any formula F, we define when $(\mathcal{A}, I^H, I^T, t)$ *satisfies* F, denoted as $(\mathcal{A}, I^H, I^T, t) \models F$, as follows:

– for any atom F, $(\mathcal{A}, I^H, I^T, t) \models F$ if $F \in I^t$.
– $(\mathcal{A}, I^H, I^T, t) \not\models \bot$.
– $(\mathcal{A}, I^H, I^T, t) \models KF$ if $(\mathcal{A}, J^H, J^T, t) \models F$ for all $J^H, J^T \in \mathcal{A}$ with $J^H \subseteq J^T$.
– $(\mathcal{A}, I^H, I^T, t) \models MF$ if $(\mathcal{A}, J^H, J^T, t) \models F$ for some pair $J^H, J^T \in \mathcal{A}$ with $J^H \subseteq J^T$.
– $(\mathcal{A}, I^H, I^T, t) \models F \wedge G$ if $(\mathcal{A}, I^H, I^T, t) \models F$ and $(\mathcal{A}, I^H, I^T, t) \models G$.
– $(\mathcal{A}, I^H, I^T, t) \models F \vee G$ if $(\mathcal{A}, I^H, I^T, t) \models F$ or $(\mathcal{A}, I^H, I^T, t) \models G$.
– $(\mathcal{A}, I^H, I^T, t) \models F \rightarrow G$ if, for every tense t' with $t \leq t'$, $(\mathcal{A}, I^H, I^T, t') \not\models F$ or $(\mathcal{A}, I^H, I^T, t') \models G$.
– $(\mathcal{A}, I^H, I^T, t) \models \neg F$ if $(\mathcal{A}, I^H, I^T, t) \models F \rightarrow \bot$

It is easy to see that if F does not contain any belief operators, $(\mathcal{A}, I^H, I^T, t) \models F$ is irrelevant to the collection $\mathcal{A}$; if F is a subjective literal (either Ka or Ma), $(\mathcal{A}, I^H, I^T, t) \models F$ is irrelevant to I^H and I^T. For example, take $\mathcal{A} = \{\{a, b\}, \{a, c\}\}$ and then $(\mathcal{A}, I^H, I^T, t) \models Ka$ for any sets of atoms I^H and I^T with $I^H \subseteq I^T$.

Epistemic HT-logic has the following basic properties which will be used in subsequent sections.

Lemma 1. *For any epistemic HT-interpretation* $(\mathcal{A}, I^H, I^T)$*, if* $(\mathcal{A}, I^H, I^T, H) \models F$*, then* $(\mathcal{A}, I^H, I^T, T) \models F$*.*

The intuition behind this lemma is obvious: If a statement is true "here", then it is also true "there". This property is guaranteed by the condition $I^H \subseteq I^T$.

Finally, we say that an epistemic HT-interpretation $(\mathcal{A}, I^H, I^T)$ *satisfies* a formula F, denoted $(\mathcal{A}, I^H, I^T) \models F$, if $(\mathcal{A}, I^H, I^T, H) \models F$.

A *model* of an epistemic theory E is an epistemic HT-interpretation $(\mathcal{A}, I^H, I^T)$ by which every formula in E is satisfied.

2.2 Epistemic Equilibrium Logic

Pearce's equilibrium logic is a kind of minimal model reasoning based on the HT-logic and its semantics is defined as the set of the equilibrium models [11]. An interesting result about the equilibrium logic is that it provides a characterization of the answer sets for nested logic programs [7]. In this subsection, we generalize the notion of equilibrium models to our epistemic HT-logic and the resulting logic is called *epistemic equilibrium logic*.

Definition 1. *An* epistemic equilibrium model *of an epistemic theory Γ is a total epistemic HT-interpretation $(\mathcal{A}, I, I)$ such that*

(i) *$(\mathcal{A}, I, I)$ is a model of Γ.*
(ii) *for every proper subset J of I, $(\mathcal{A}, J, I)$ is not a model of Γ.*

Epistemic equilibrium logic is the logic whose semantics is defined through epistemic equilibrium models. To provide a unifying characterization for the semantics of both nested logic programs and epistemic logic programs, we need the following definition.

Definition 2. *Let $\mathcal{A}$ be a collection of sets of atoms occurring in an epistemic theory Π. We say $\mathcal{A}$ is an equilibrium view if $\mathcal{A}$ is a maximal collection that satisfies*

$$\mathcal{A} = \{I \mid (\mathcal{A}, I, I) \text{ is an equilibrium model of } \Pi\}.$$

To illustrate our definitions, let us look at the following examples.

Example 1. Let Π_1 be the epistemic theory containing only the formula $\{Ka \vee b\}$. Then both $(\{\{a\}\}, \{a\}, \{a\})$ and $(\{\{b\}\}, \{b\}, \{b\})$ are epistemic equilibrium models of Π_1. Moreover, $\mathcal{A}_1 = \{\{a\}\}$ and $\mathcal{A}_2 = \{\{b\}\}$ are equilibrium views of Π_1. But $\mathcal{A}_3 = \{\{a\}, \{b\}\}$ is not an equilibrium view of Π_1 since $(\mathcal{A}_3, \{a\}, \{a\})$ is not an epistemic model of Π_1 (note that $(\mathcal{A}_3, \{a\}, \{a\}) \not\models Ka$). $\mathcal{A}_4 = \{\{a, b\}\}$ is not an equilibrium view of Π_1 either because $(\mathcal{A}_4, \{a, b\}, \{a, b\})$ is not an epistemic equilibrium model.

Example 2. Let $\Pi_2 = \{K(a \vee b)\}$ be an epistemic theory. Then $\{\{a\}, \{b\}\}$ is the unique equilibrium view of Π_2. Note that although $(\{\{a\}\}, \{a\}, \{a\})$ and $(\{\{b\}\}, \{b\}, \{b\})$ are epistemic equilibrium models of Π_1, neither $\{\{a\}\}$ nor $\{\{b\}\}$ is an equilibrium view of Π_2 since they are not maximal.

Example 3. Let $\Pi_3 = \{a, Ka \rightarrow b \vee c\}$ be an epistemic theory. It is easy to see that Π_3 has a unique equilibrium view $\{\{a, b\}, \{a, c\}\}$.

It is worth to mentioning that differently from the standard propositional modal logic, for instance S5, the epistemic equilibrium logic can be viewed as a kind of minimal model reasoning about epistemic concepts (i.e knowledge and belief). In this way, the equilibrium view semantics shares the same spirit of Gelfond's world view semantics. However, as will be shown next, epistemic equilibrium logic is general enough to characterize the semantics of nested epistemic logic programs, while the world view semantics cannot.

3 Nested Epistemic Logic Programs (NELPs)

As we will see in the next section, the epistemic HT-logic is actually a very general extension of both nested logic programs (NLPs) and epistemic logic programs (ELPs). To make this comparison more direct, we generalize both the syntax of NLPs and the syntax of ELPs by introducing a class of logic programs called *nested epistemic logic programs* or *NELP*. This language corresponds to a subset of the language of the epistemic HT-logic.

The *atom* is understood as in propositional logic. *Elementary formulas* are propositional atoms and the 0-place connective $\bot$ ("false") and $\top$ ("true"). *NELP formulas* are built from elementary formulas using negation as failure "*not*", conjunction ",", disjunction ";", and the two belief operators K and M.

An *NELP rule* is an expression of the form

$$F \leftarrow G$$

where F and G are NELP formulas called the *head* and the *body* of the rule. For any rule r, its head and body are denoted $head(r)$ and $body(r)$, respectively. A *nested epistemic logic program (abbreviated NELP)* is a (finite) set of NELP rules.

For our purpose, in this paper we 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.

Let us think of NELP rules as epistemic formulas by replacing every "*not*" with "$\neg$", every comma with "$\wedge$", every semicolon ";" with disjunction "$\vee$", and transforming every rule $head \leftarrow body$ into the implication $body \rightarrow head$. Accordingly we can turn every nested epistemic logic program Π into an epistemic theory. When no confusion is caused, we will not distinguish a nested epistemic logic program Π and its corresponding epistemic theory. Note that the negation "$\neg$" corresponds to the negation as failure rather than the strong negation (or classical negation) in logic programming. For simplicity, the second negation will not be considered until in Section 6. For example, we may use Π to denote both the NELP $\{Ka; b \leftarrow c, Md, not e\}$ and its corresponding epistemic theory $\{c \wedge Md \wedge \neg e \rightarrow Ka \vee b\}$.

Now based on the epistemic HT-logic introduced in Section 2, we define the semantics of nested epistemic logic programs as follows.

Definition 3. *Let Π be a nested epistemic logic program and $\mathcal{A}$ be a collection of sets of atoms. We say $\mathcal{A}$ is an* equilibrium view *of Π if $\mathcal{A}$ is an equilibrium view of the corresponding epistemic theory Π.*

Example 4. Consider the nested epistemic logic program $\mathit{\Pi}$:

$Ka; Kb \leftarrow,$
$c \leftarrow Ka, not\, Mb,$
$d \leftarrow Kb, not\, Ma.$

It is easy to see that $\mathit{\Pi}$ has two equilibrium views $\{\{a, c\}\}$ and $\{\{b, d\}\}$. Note that if we change the first rule in $\mathit{\Pi}$ to be $a; b \leftarrow$, then the modified program will only have one equilibrium view $\{\{a\}, \{b\}\}$.

4 Epistemic Equilibrium Logic and Logic Programs

The class of NELPs contains two major classes of logic programs: *nested logic programs* and *epistemic logic programs*. Thus NELPs generalize most classes of logic programs including normal logic programs and disjunctive logic programs. In this section we will prove that the equilibrium view semantics of NELPs extends both the world view semantics of epistemic logic programs and answer set semantics of nested logic programs.

4.1 Equilibrium Views and Answer Sets of Nested Logic Programs

An *NLP rule* is a special NELP rule which contains no belief operators, and a *nested logic program (NLP)* is a set of NLP rules. Similarly, an *NLP* formula is an NELP formula containing no belief operators.

To define the answer sets of nested logic programs, we first define when a set S of atoms *satisfies* an NLP formula F, denoted as $S \models F$, recursively as follows:
$S \models F$ if F is an atom and $F \in S$,
$S \models \top$,
$S \models \bot$,
$S \models (F, G)$ if $S \models F$ and $S \models G$,
$S \models (F; G)$ if $S \models F$ or $S \models G$,
$S \models not\, F$ if $S \not\models F$.

We say a nested logic program $\mathit{\Pi}$ is *closed* under a set of atoms S if, for every rule r, $body(r)$ implies $head(r)$. Then the definition of *answer sets* is defined in two steps:

- Let $\mathit{\Pi}$ be a nested logic program without negation as failure *not*. A set S of atoms is an *answer set* of $\mathit{\Pi}$ if S is minimal set closed under S;
- For an arbitrary nested logic program $\mathit{\Pi}$, the *reduct* $\mathit{\Pi}^S$ with respect to a set S of atoms is obtained by replacing every maximal occurrence of a formula of the form $not\, F$ in $\mathit{\Pi}$ with $\bot$ if $S \models F$ and with $\top$ if $S \models not\, F$. We say S is an *answer set* of $\mathit{\Pi}$ if S is an answer set of $\mathit{\Pi}^S$.

As we have noted before, a nested logic program is also a nested epistemic logic program and thus any nested logic program $\mathit{\Pi}$ can be assigned two semantics: *answer sets* and *equilibrium views*. The following result states that these two semantics coincide for nested logic programs and thus the equilibrium view semantics generalizes the answer set semantics.

Theorem 1. *Let Π be a nested logic program and S be a set of atoms. Then S is an answer set of Π if and only if there exists an equilibrium view $\mathcal{A}$ of Π (as a nested epistemic logic program) such that $S \in \mathcal{A}$.*

Proof. ($\Rightarrow$) If S is an answer set of Π, let $\mathcal{A}$ denote the set of minimal models of Π^S. Then we have $S \in \mathcal{A}$ by the definition of answer sets. We need only to prove that $\mathcal{A}$ is an equilibrium view of Π (as a nested epistemic logic program). Since Π contains no belief operators, $(\mathcal{A}, S, S)$ is an equilibrium model of Π in the epistemic HT-logic if and only if (S, S) is an equilibrium model of Π in ordinary HT-logic. Again, by Lemma 3 in [7], (S, S) is an equilibrium model of Π in ordinary HT-logic if and only if S is an answer set of Π. Thus $(\mathcal{A}, S, S)$ is an equilibrium model of Π in the epistemic HT-logic.
($\Leftarrow$) Using the above argument, we have that if $(\mathcal{A}, S, S)$ is an equilibrium model of Π in the epistemic HT-logic then S is an answer set of Π.

4.2 Equilibrium Views and World Views of Epistemic Logic Programs

Epistemic logic programs were first proposed by Gelfond [3] in order to overcome difficulties in reasoning about disjunctive information through disjunctive logic programs. It turns out that epistemic logic programs can be used as an effective formulation to represent and reason about agents' epistemic states and hence have great potential in agent programming. The semantics for epistemic logic programs is based on the pair $(\mathcal{A}, S)$, where $\mathcal{A}$ is a collection of sets of ground literals and S is a set in $\mathcal{A}$. The truth of an NELP formula F in $(\mathcal{A}, S)$ is denoted by $(\mathcal{A}, S) \models F$ and the falsity by $(\mathcal{A}, S) =\!| F$, and are defined as follows:
$(\mathcal{A}, S) \models F$ iff $F \in S$ where F is a ground atom.
$(\mathcal{A}, S) \models KF$ iff $(\mathcal{A}, S_i) \models F$ for all $S_i \in \mathcal{A}$.
$(\mathcal{A}, S) \models MF$ iff $(\mathcal{A}, S_i) \models F$ for some $S_i \in \mathcal{A}$.
$(\mathcal{A}, S) \models (F, G)$ iff $(\mathcal{A}, S) \models F$ and $(\mathcal{A}, S) \models G$.
$(\mathcal{A}, S) \models (F; G)$ iff $(\mathcal{A}, S) \models \neg(\neg F, \neg G)$.
$(\mathcal{A}, S) \models \neg F$ iff $(\mathcal{A}, S) =\!| F$.
$(\mathcal{A}, S) =\!| F$ iff $\neg F \in S$ where F is a ground atom.
$(\mathcal{A}, S) =\!| KF$ iff $(\mathcal{A}, S) \not\models KF$.
$(\mathcal{A}, S) =\!| MF$ iff $(\mathcal{A}, S) \not\models MF$.
$(\mathcal{A}, S) =\!| (F, G)$ iff $(\mathcal{A}, S) =\!| F$ or $(\mathcal{A}, S) =\!| G$.
$(\mathcal{A}, S) =\!| (F; G)$ iff $(\mathcal{A}, S) =\!| F$ and $(\mathcal{A}, S) =\!| G$.
An *epistemic logic program* or *ELP* is a finite set of ELP rules of the form:

$$F_1; F_2; \cdots; F_k \leftarrow G_1, \cdots, G_m, not G_{m+1}, \cdots, not G_n. \tag{1}$$

Here $F_1, \cdots, F_k$ are (objective) formulae, $G_1, \cdots, G_m$ are (objective) formulae or subjective formulaes, and $G_{m+1}, \cdots, G_n$ are (objective) formulae.

Note that ELP also allows nested expressions but in a restricted form.

For an epistemic logic program Π, its semantics is given by its *world view* which is defined in the following steps:

Step 1. Let Π be an epistemic logic program containing neither belief operators K and M nor negation as failure *not*. A set S of ground literals is called a *belief set* of Π iff S is a minimal set of satisfying conditions: (i) for each rule $F_1; F_2; \cdots; F_k \leftarrow G_1, \cdots, G_m$ from Π such that $S \models (G_1, \cdots, G_m)$ we have $S \models (F_1; F_2; \cdots; F_t)$; and (ii) if S contains a pair of complementary literals, then S is the set Lit of all literals (called inconsistent belief set).
Step 2. Let Π be an epistemic logic program not containing modal operators K and M and S be a set of ground literals in the language of Π. By Π_S we denote the result of (i) removing from Π all the rules containing formulas of the form $notG$ such that $S \models G$ and (ii) removing from the rules in Π all other occurrences of formulas of the form $notG$. S is a *belief set* of Π iff S is a belief set of Π_S.
Step 3. Finally, let Π be an arbitrary epistemic logic program and $\mathcal{A}$ a collection of sets of ground literals in its language. By $\Pi_{\mathcal{A}}$ we denote the epistemic logic program obtained from Π by (i) removing from Π 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 Π all other occurrences of subjective formulas. S is a *belief set* of Π iff S is a belief set of $\Pi_{\mathcal{A}}$.

Example 5. The following epistemic logic program Π:

$a \leftarrow$
$b; c \leftarrow$
$d \leftarrow Ka$
$e \leftarrow Mb$

has a unique world view $\{\{a, b, d, e\}, \{a, c, d, e\}\}$.

Observe that the world view of the above program Π is also the equilibrium view of Π. This is not surprising because we will show that, for any epistemic logic program Π, $\mathcal{A}$ is a world view of Π if and only if $\mathcal{A}$ is an equilibrium view of Π.

The negation $\neg$ in epistemic program actually corresponds to the strong negation in extended logic program. In the rest of this section our discussion is temporarily restricted to epistemic programs that do not contain $\neg$. In Section 6, we will see that the results here are all valid for arbitrary epistemic programs.

We first introduce some notations. By viewing an epistemic logic program Π as an epistemic theory in epistemic HT-logic, $(\mathcal{A}, J, I) \models \Pi$ means that each rule is satisfied in the epistemic HT-interpretation $(\mathcal{A}, J, I)$. Under the world view semantics, on the other hand, we say that a rule of the form (1) *is satisfied* in a pair $(\mathcal{A}, S)$ if the fact $(\mathcal{A}, S) \models (G_1, \cdots, G_m)$, $(\mathcal{A}, S) \not\models G_{m+1}, \cdots, (\mathcal{A}, S) \not\models G_n$ implies $(\mathcal{A}, S) \models F_1; \cdots; F_k$. $(\mathcal{A}, S) \models \Pi$ means that each rule of Π is satisfied in $(\mathcal{A}, S)$. If Π does not contain any belief operators, we simply use $S \models \Pi$ to denote $(\mathcal{A}, S) \models \Pi$ (recall that the truth values of objective formulas are irrelevant to $\mathcal{A}$.

Lemma 2. *Let $(\mathcal{A}, I^H, I^T)$ be an epistemic HT-interpretation and Π an epistemic logic program without containing negation as failure. $(\mathcal{A}, J, I) \models \Pi$ if and only if $(\mathcal{A}, J) \models \Pi$.*

Lemma 3. *Let Π be an epistemic logic program and $(\mathcal{A}, J, I)$ be an epistemic HT-interpretation. Then $(\mathcal{A}, J, I) \models \Pi$ if and only if $J \models (\Pi_{\mathcal{A}})_I$. Here $(\Pi_{\mathcal{A}})_I$ is obtained through Step 3 and Step 2 in the definition of the world views.*

Proof. ($\Rightarrow$) Suppose $(\mathcal{A}, J, I) \models \Pi$, we want to show $J \models (\Pi_{\mathcal{A}})_I$.

If $R \in \Pi$, we can assume that R is of the form (1). If R satisfies $(\mathcal{A}, I) \models G_i$ for every i with $r+1 \leq i \leq m$ and $(\mathcal{A}, I) \not\models G_j$ for every i with $m+1 \leq j \leq n$, we use $(R_{\mathcal{A}})_I$ to denote the reduction of R with respect to $\mathcal{A}$ and I: $F_1; \cdots; F_k \leftarrow G_1, \cdots, G_r$. In general, $(R_{\mathcal{A}})_I$ may be undefined. Note that $(\Pi_{\mathcal{A}})_I = \{(R_{\mathcal{A}})_I \mid R \in \Pi\}$.

For any rule of $((\Pi_{\mathcal{A}})_I)$, it must be of the form $(R_{\mathcal{A}})_I$ for some rule $R \in \Pi$. So we need only to prove that $J \models (R_{\mathcal{A}})_I$ for $R \in \Pi$.

By the definition of the program reduction, we have the following two facts:

1. $(\mathcal{A}, I) \models G_i$ for $r+1 \leq i \leq m$ and
2. $(\mathcal{A}, I) \not\models G_i$ for $m+1 \leq i \leq n$.

Suppose $J \models body((R_{\mathcal{A}})_I))$, we want to show $J \models head((R_{\mathcal{A}})_I)$. That is, $J \models (F_1; \cdots; F_k)$. Since the body of $(R_{\mathcal{A}})_I$ is now the conjunction of $G_1, \cdots,$ and G_r, we have $J \models G_i$ for all i with $1 \leq i \leq r$.

We prove that $(\mathcal{A}, J, I) \models body(R)$ by considering three different cases:

Case 1. Since $J \models G_i$ for $1 \leq i \leq r$, then $G_i \in J$ and thus $(\mathcal{A}, J, I, t) \models G_i$ for any $t \in \{H, T\}$.

Case 2. If $r+1 \leq i \leq m$, then G_i is a subjective literal and it is of form either KO'_i or MO'_i for some objective literal O_i. If $G_i = KO_i$, by $(\mathcal{A}, I) \models G_i$ for $r+1 \leq i \leq m$, $(\mathcal{A}, I) \models KO_i$. This means $(\mathcal{A}, I') \models O_i$ for all $I' \in \mathcal{A}$, which implies $(\mathcal{A}, I^H, I^T, t) \models O_i$ for all sets J^H, J^T of atoms with $J^H \subseteq J^T$. Thus $(\mathcal{A}, J, I, t) \models KO_i$ or $(\mathcal{A}, J, I, t) \models G_i$ for $r+1 \leq i \leq m$. Similarly, we also have $(\mathcal{A}, J, I, t) \models G_i$ for $r+1 \leq i \leq m$ if $G_i = MO_i$.

Case 3. If $(\mathcal{A}, I) \not\models G_i$ for $m+1 \leq i \leq n$, then $G_i \in I$. Since G_i is objective and $J \subseteq I$, $(\mathcal{A}, J, I) \not\models G_i$.

Combining Cases 1-3, we have $(\mathcal{A}, J, I) \models body(R)$. By $(\mathcal{A}, J, I) \models R$, we have $(\mathcal{A}, J, I) \models head(R)$. By $head(R) = head((R_{\mathcal{A}})_I)$ and thus $(\mathcal{A}, J, I) \models head((R_{\mathcal{A}})_I)$.

($\Leftarrow$) Suppose $J \models (\Pi_{\mathcal{A}})_I$, we show $(\mathcal{A}, J, I) \models \Pi$ by considering the following three cases:

For any rule $R \in \Pi$, assume R is of form (1),

Case 1. If $(\mathcal{A}, I) \not\models G_i$ for some i with $r+1 \leq i \leq m$, then $G_i \notin I$ and thus $G_i \notin J$. This implies $(\mathcal{A}, J, I) \not\models G_i$. In this case, $(\mathcal{A}, J, I) \not\models body(R)$. So $(\mathcal{A}, J, I) \models R$.

Case 2. If $(\mathcal{A}, I) \models G_i$ for some i with $m+1 \leq i \leq n$, then $G_i \in I$. This implies $(\mathcal{A}, J, I, T) \models G_i$. In this case, $(\mathcal{A}, J, I) \not\models body(R)$. So $(\mathcal{A}, J, I) \models R$.

Case 3. If neither Case 1 nor Case 2, then $(\mathcal{A}, J, I) \models G_{r+1} \wedge \cdots \wedge G_m \wedge \neg G_{m+1} \wedge \cdots \wedge \neg G_n$ and thus $(R_{\mathcal{A}})_I$ is well defined.
If $(\mathcal{A}, J, I) \models G_1 \wedge \cdots \wedge G_r$, then $J \models G_1 \wedge \cdots \wedge G_r$. Since $J \models (R_{\mathcal{A}})_I$, we have $J \models head((R_{\mathcal{A}})_I)$. That is, $J \models head(R)$.

Lemma 4. *Let Π be an ELP and $(\mathcal{A}, J, I)$ be a epistemic HT-interpretation. Then $(\mathcal{A}, I, I)$ is an equilibrium model of Π if and only if I is a belief set of $(\Pi_{\mathcal{A}})_I$.*

The above lemma, obtained directly from Lemma 3, implies the following result.

Theorem 2. *Let Π be an epistemic logic program and $\mathcal{A}$ be a collection of sets of atoms. Then $\mathcal{A}$ is a world view of Π if and only if $\mathcal{A}$ is an equilibrium view of Π.*

5 Strong Equivalence of Nested Epistemic Logic Programs

Recently, researchers have addressed the problem of characterizing the strong equivalence of logic programs under the answer sets. In particular, the result in [7] shows that the strong equivalence of nested logic programs can be characterized in term of the equivalence of formulas in monotonic logic (the HT-logic). Here we are interested in extending this result to NELPs under the equilibrium view semantics. We say two NELPs Π_1 and Π_2 are *equivalent* if they have the same equilibrium views. A NELP Π_1 is said to be *strong equivalent* to another NELP Π_2 if, for every NELP Π, $\Pi_1 \cup \Pi$ and $\Pi_2 \cup \Pi$ are equivalent. It is well-known that equivalence of two programs does not implies their strong equivalence in general (see [2, 8, 7] for more examples of strongly equivalent programs).

Similarly, two theories Π_1 and Π_2 in the epistemic HT-logic is *equivalent* if they have the same set of models.

Theorem 3. *For any nested epistemic logic programs Π_1 and Π_2, the following conditions are equivalent:*

(1) *Π_1 is strongly equivalent to Π_2.*
(2) *Π_1 is equivalent to Π_2 in the epistemic HT-logic.*

By Theorem 2, it is easy to prove Theorem 3 since we have the following lemma, whose proof is similar to that of Theorem 1 in [7].

Lemma 5. *For any epistemic theories Γ_1 and Γ_2, the following conditions are equivalent:*

(1) *for every epistemic theory Γ, $\Gamma_1 \cup \Gamma$ and $\Gamma_2 \cup \Gamma$ have the same equilibrium models.*
(2) *Γ_1 is equivalent to Γ_2 in the epistemic HT-logic.*

For any two objective theories Γ_1 and Γ_2, they are equivalent in the logic of here-and-there if and only if they are equivalent in the epistemic HT-logic. Thus, by Theorem 2, the main result (Theorem 1) in [7] is a corollary of our Theorem 3:

Corollary 1. *For any nested logic programs Π_1 and Π_2, the following conditions are equivalent:*

(1) *Π_1 is strongly equivalent to Π_2.*
(2) *Π_1 is equivalent to Π_2 in the logic of here-and-there.*

By Theorem 2, the strong equivalence of epistemic logic programs under the world view semantics can also be verified by checking the equivalence of formulas in the epistemic HT-logic which is a monotonic logic.

Corollary 2. *For any epistemic logic programs Π_1 and Π_2, the following conditions are equivalent:*

(1) *Π_1 is strongly equivalent to Π_2.*
(2) *Π_1 is equivalent to Π_2 in the epistemic HT-logic.*

6 Adding Strong Negation in NELPs

In answer set programming, the syntax of logic programs usually allows both negation as failure and strong negation [5]. The second negation is denoted by $\neg$. It is well-known that this extension is very useful for representing and reasoning about incomplete information. In this section, we show how to add the second negation in NELPs. This can be done by an easy generalization of Section 5 in [7]. Technically, it is not hard to add the strong negation in the syntax of logic programs.

A *literal* is an atom a or its strong negation $\neg a$. By allowing arbitrary literals in place of atoms, *extended NELP formula*, *extended NELP rule* and *extended NELP* can be defined in the same way as we defined NELP formula, NELP rule and NELP. Following [5], the semantic of an extended NELP can be defined through a simple syntactic translation.

Given an extended NELP Π, we introduce a new symbol a' for each atom a in Π. Then Π can be translated into a NELP Π' by replacing each negative literal $\neg a$ with a'. Note that Π' does not contain the strong negation. For any expression E, we use E' to denote the expression obtained by replacing every $\neg a$ with a'. Denote $Cons(\Pi) = \{\bot \leftarrow a, a' \mid a \text{ is literal in } \Pi\}$. Then we say $\mathcal{A}$ is an equilibrium view of Π if $\mathcal{A}'$ is an equilibrium view of $\Pi' \cup Cons(\Pi)$.

For nested logic programs (with strong negation) Π, a set X of literals is an answer set of Π iff X is X' is an answer set of $\Pi' \cup Cons(\Pi)$. Therefore, Theorem 1 is also true for nested logic programs with strong negation.

In the same way as in [5], the negation $\neg$ in an epistemic logic program can be eliminated by introducing new atom a' for each atom a. Thus, Theorem 2 is also true for epistemic logic program with strong negation.

Theorem 3 can also be generalized to extended NELPs.

Theorem 4. *For any extended NELPs Π_1 and Π_2, the following conditions are equivalent:*

(1) *Π_1 is strongly equivalent to Π_2.*
(2) *$\Pi_1 \cup Cons(\Pi_1)$ is equivalent to $\Pi_2 \cup Cons(\Pi_2)$ in the epistemic HT-logic.*

Proof. Π_1 is strongly equivalent to Π_2
if and only if
$\Pi_1' \cup Cons(\Pi_1)$ is strongly equivalent to $\Pi_2 \cup Cons(\Pi_2)$
if and only if
$\Pi_1 \cup Cons(\Pi_1)$ is equivalent to $\Pi_2 \cup Cons(\Pi_2)$ in the epistemic HT-logic.

7 Conclusions

In this paper, we introduced the epistemic HT-logic and based on this logic further developed a new type of logic programs called nested epistemic logic programs (NELPs). We showed that the equilibrium view semantics of NELPs generalizes the answer set semantics of nested logic programs as well as the world view semantics of epistemic logic programs. We also characterize the strong equivalence property of NELPs in terms of epistemic HT-logic.

Some important issues related to NELPs should be further investigated. Firstly, it is important to understand the computational properties of NELPs in detail from both theoretical and practical viewpoints. Secondly, as epistemic logic programs may be viewed as an effective formalism for representing and reasoning about agent's dynamic epistemic state, e.g. [12], it would be interesting to explore how this work can be improved by applying NELPs.

Acknowledgements. The authors would like to thank the three referees for their comments. In particular, the comments from one referee greatly helped the improvement of the paper.

References

1. C. Baral. *Knowledge Representation, Reasoning and Declarative Problem Solving*. Cambridge University Press, 2003.
2. E. Erdem and V. Lifschitz. Transformations of logic programs related to causality and planning. In *Proceedings of the 5th International Conference on Logic Programming and Nonmonotonic Reasoning (LNAI 1730)*, pages 107–116, 1999.
3. M. Gelfond. Logic programming and reasoning with incomplete information. *Annals of Mathematics and Artificial Intelligence*, 12:98–116, 1994.
4. M. Gelfond and V. Lifschitz. The stable model semantics for logic programming. In *Proceedings of the International Conference on Logic Programming*, pages 1070–1080. The MIT Press, 1988.
5. M. Gelfond and V. Lifschitz. Logic programs with classical negation. In *Proceedings of the International Conference on Logic Programming*, pages 579–597, 1990.
6. V. Lifschitz. Answer set planning. In *Proceedings of the International Conference on Logic Programming*, pages 23–37. The MIT Press, 1999.
7. V. Lifschitz, D. Pearce, and A. Valverde. Strongly equivalent logic programs. *ACM Transactions on Computationl Logic*, 2(4):426–541, 2001.
8. V. Lifschitz, L. Tang, and H. Turner. Nested expressions in logic programs. *Annals of Mathematics and Artificial Intelligence*, 25:369–389, 1999.
9. F. Lin and Y. Shoham. A logic of knowledge and justified assumptions. *Artificial Intelligence*, 57:271–289, 1992.
10. D. Pearce. From here to there: stable negation in logic programming. In D. Gabbay and H. Wansing, editors, *What is Negation?* Springer-Verlag, 1997.
11. D. Pearce. A new logical characterization of stable models and answer sets. In J. Dix, L. Pereira, and T. Przymusinski, editors, *Non-Monotonic Extensions of Logic Programming (LNAI 1216)*, pages 57–70. Springer-Verlag, 1997.
12. Y. Zhang. Minimal change and maximal coherence for epistemic logic program updates. In *Proceedings of IJCAI-03*, pages 112–117. 2003.