PROCEEDINGS OF THE INTERNATIONAL CONFERENCE
ON FIFTH GENERATION COMPUTER SYSTEMS 1988,
edited by ICOT. © ICOT, 1988

Epistemic Logic Programming

Y.J. Jiang

Com. Sci. Dept., Univ. of Essex, Colchester, U.K.

Abstract

In this paper, a possible-worlds based Horn clausal form of epistemic logic is introduced. It is intended to serve *directly* as a programming language to reason about epistemic notions. For this purpose, an epistemic SLD-like proof procedure based on Konolige's B-resolution is developed. This proof procedure is then extended to deal with a stratified Horn auto-epistemic logic that allows quantifying-in variables. The concept of *epistemic negation as failure* is also introduced. An attempt to extend the single agent AE logic to multi-agents is also made.

Keywords: Epistemic notions, Modal logic programming, Intension/Extension, SLD Horn Clausal resolution, Auto-epistemic logic, Stratification, Non-monotonic reasoning, Negation as failure.

1. Introduction

Epistemic notions are concerned with beliefs and knowledge. They are important because our representation of the world is generally *incomplete* and *subjective*. This means that an effective logic paradigm should attempt to solve a problem on the basis of what it currently believes/knows and what it does not, rather than to wait for the complete state of the world which may never be obtained. This requirement is more crucial for distributed problem solving systems where communication and cooperation are essential (eg. multi-agent planning systems [Konolige 81]). In this case, an agent is further required to rely on its knowledge (or beliefs) about other agents's knowledge (or beliefs) in order to make a proper decision.

Epistemic notions are also important in commonsense reasoning. Often we draw conclusions based on our current autoepistemic reflection and may retract them later when new beliefs are aquired. If we regard a normal logic database as a set of beliefs of a single agent (eg. the database), then an autoepistemic view of the program could additionally allow the representation and introspection of beliefs about beliefs and non-beliefs of the agent itself. Thus epistemic notions are one level higher concepts that are inclusive notions of normal logic programming paradigms. In other words, a mechanization of these notions will also entail that of a normal logic programming paradigm. In this paper, we thus attempt to develop such a mechanization as the basis of an epistemic logic programming papradigm.

It is possible to model epistemic notions in a first order theory in a syntactic framework [Perlis 87, Konolige 81] in which beliefs are represented as quoted terms of a syntactic BEL predicate. This has the nice property of partially avoiding the *logic omnisience* problem of epistemic reasoning [Moore & Hendrix 79]. It also allows quantification over beliefs. The main problem of the syntactic approach however, is the enormous complexity involved in mechnizing epistemic notions which have to be indirectly modelled through first order encodings. First order Hilbert style systems of beliefs are usually used (eg. [Morgenstern 87]), although no serious proofs exist to automate them [Geissler&Konolige 86]. In addition, it is difficult to analyze the axioms of epistemic notions in the syntactic approach [Levesque 84].

As a solution to this last difficulty, McCarthy et al [78] and Moore [85a] proposed a first order axiomatization of possible-worlds semantics of epistemic notion at the expense of quantification over beliefs and logic omnisience. This has been shown by Sakakibara [87] to be implementable in Prolog. This approach nevertheless is still rather inefficient because it involves reasoning about possible worlds and other objects of semantic domains, rather than manipulating beliefs *directly*.

In this paper, we thus attempt to develop an alternative possible-worlds based mechanization that directly models epistemic notions. Traditionally, such a mechanization are restricted at the propositional level [Halpern&Moses 85]. Otherwise the introduction of quantified modal logic often requires much creative help from a user or gives rise to long proofs. Farinas del Cerro [83] proposed imitating classical clausal resolution in some modal logics (eg. temporal logics). The proposed method can be incorporated in a very efficient Prolog environment [86], but fails to treat many epistemic notions under consideration. In particular, quantifying-in beliefs are not allowed. Another classical approach to modal logic is taken by [Abadi & Manna 87]. Their system however is mainly concerned with temporal logics.

There are also some non-resolution based modal proof procedures which work on natural forms of modal formulae. This has the virtue of added clarity, since formulae do not need to be rephrased in unnatural and sometimes long clausal forms. These proof procedures are either tableau-based (eg. [Wallen 87]) or sequent-based (eg. [Jackson & Reighleit 87]). The former however has to manipulate the set of formulae as a whole with no 'cut' operator; while the latter tends to be less goal-oriented. In addition, both approaches normally introduce explicit indices for possible worlds.

Despite the possibly unnatural form of clauses, clause-based resolution still remains as an important proof strategy for modal logics. In [Konolige 86], Konolige introduced a clausal resolution based proof theory called *B-resolution* for quantified epistemic notions. However their mechanization is lack of control of search space and involves recursive clausalization in resolving epistemic formulae. In addition, B-resolution is only applied to a set of monotonic epistemic logics.

Nevertheless, B-resolution appears to provide a foundation for mechanizing epistemic logics in the same spirit as Robinson's resolution for first order logic. Like Robinson's resolution, unrestricted B-resolution would also inevitably generate a huge search space for general clausal form of epistemic logics. In this paper, we thus first try to develop a SLD-like [Lloyd 84] B-resolution for a Horn clausal epistemic logic. This logic is intended as the theoretical basis of an epistemic logic programming paradigm in the same spirit as a Horn clausal logic to Prolog-like paradigms. In other words, we want to write down epistemic formulae as a program whose execution is *directly* based on the proof theory of the epistemic logic. We also attempt to extend the mechanization of the nonmonotonic epistemic logic to an non-monotonic autoepistemic predicate logic that allows quantifying-in variables.

The paper is organized as follows. In section 2, we introduce Konolige's B-resolution and analyze its problems. We then attempt to solve these problems in Section 3. This is followed by an extension to AE reasonning.

2. B-resolution

In [Konolige 86], Konolige has proved a set of epistemic resolution rules called *B-resolution* rules to be sound and complete for the corresponding class of epistemic logics as follows.

B-resolution:

Let $\Gamma=\{\gamma_1, \gamma_2 ...\}$, $\Sigma=\{\sigma_1,\sigma_2,..\}$ and $\Delta=\{\delta_1,\delta_2 ...\}$ be finite sets of formulae.

Given the following clauses (omiting the agents and the terms),

$$\begin{array}{l} A_1 \text{ v } B\gamma_1 \\ A_2 \text{ v } B\gamma_2 \\ \quad \cdot \\ \quad \cdot \\ AA_1 \text{ v } \neg B\delta_1 \\ AA_2 \text{ v } \neg B\delta_2 \\ \quad \cdot \\ \quad \cdot \\ AB_1 \text{ v } \sigma_1 \\ AB_2 \text{ v } \sigma_2 \\ \quad \cdot \\ \quad \cdot \end{array}$$

We can derive the following clause,

$A_1 \text{v} A_2 \text{v..v} AA_1 \text{v} AA_2 \text{v..v} AB_1 \text{v} AB_2 \text{v..}$,

if only if the following set of formulae is unsatisfiable

1. $\{\Gamma, \neg\delta_1\}$ for K, T epistemic logic;
2. $\{\Gamma, B\Gamma, \neg\delta_1\}$ for K4, S4 epistemic logic;
3. $\{\Gamma, B\Gamma, \neg\delta_1, \neg B\Delta\}$ for K45 epistemic logic;
4. $\{\Gamma, B\Gamma, \neg\delta_1, \neg B\Delta, \neg B\neg\Sigma\}$ for S5 epistemic logic.

In addition, in the case of T, S4 and S5 epistemic logics, we also need to add the following *knowledge rule of inference:*

$$\frac{B\phi \text{ v } A}{\phi \text{ v } A}$$

For the B-resolution to be effective, Geissler&Konolige's solution [86] is to apply a *semantic attachment* technique (a kind of Stickle's meta theory resolution [85]) to recursively check the unsatisfiability condition. Suppose, each time a negative B-literal participates in a B-resolution, another refutation procedure (or a view which is similar to Kripke's device of auxiliary tableaux [59]) is invoked (or opened) using the indicated sets of sentences. Then the execution of deductions in the parent refutation proof is intermixed with execution in the child view which is being used to check unsatisfiability. If at some point a child refutation succeeds, we can construct a resolvent with bindings returned to a disjunction of auxiliary remainder literals in the parent refutation. This allows free variables to perform a schematic refutation. In addition, by keeping track of the clauses in a resolution proof in a view, we need only consider the remainer literals of these clauses in the parent view.

Lets illustrate the ideas with a short example. Assume the following initial set of clauses:

0. B(agent,w(c)) v g(k)
1. B(agent,p(a))
2. ¬p(b)
3. q(x) v p(x) v B(agent,p(x) v r(x))
4. ¬B(agent,p(y) v r(y)) v q(y)
5. ¬q(b)

Ordinary resolution work as usual, for example, 2+3 yield:

6. q(b) v B(agent, p(b) v r(b))

6+5 yield:

7. B(agent,p(b) v r(b))

Clause 4 contains a negative belief literal, following B-resolution, we open a new view in a attempt to resolve it:

View agent, rems(0,q(y))

---

8. ¬p(y) v ans(0,y)
9. ¬r(y) v ans(0,y)

This is a view for the agent of the belief. The clauses within the view are obtained by changing disjunctive literals into conjunctive literals. The *ans* predicate keeps track of the input free variable y to allow schematic resolution; it also contains the additional argument "0" to indicate that it is connected to the remainder (rems) indexed by 0. If a proof is found in the view, the remainder q(y) will be returned with an appropriate binding for y as a deduced clause of the original proof.

we can now add the arguments of positive belief literals to the view following B-resolution, as in clauses 0 and 7. The view now contains:

View agent, rems(0,q(y)) rems(1,g(k))

---

8. ¬p(y) v ans(0,y)
9. ¬r(y) v ans(0,y)
10. p(b) v r(b)
11. w(c) v ans(1)

Only the first three clauses resolve (using ordinary binary resolution) with each other, yield:

12. ans(0,b)

ie. a proof for the view is found. Because clause 11 is not involved in the proof, so the rems(1,g(k)) will not be returned to the parent view.

Now using the substitution b/y generated by the ans-predicate, we return q(b) as the result to the original view in which this new view is created. In this case, it is the initial view.

12. q(b).

clauses 12 and 5 resolve to give the null clause to complete the proof.

The main problems of B-resolution are its lack of control and the need to clausalize B-literals each time a new view is opened. The clausalization process can be exponential to the size of the literals in concern [Lakemayer 87]. The control problem on the other hand is complicated by the inability of applying a direct set-of support or linear resolution strategy. We will attempt to solve these problems in the next section.

3. Epistemic SLD

We first want to obtain a clausal form of epistemic logic. Because quantifiers are bounded to the modal B operators, we use a simplified intensional scheme developed in [Jiang 88] to generate a syntactic characterization of epistemic formulae. The basic idea is to attach an integer level (default being zero) to each quantified variable denoting its nesting of B-operators so that normal skolemization process can be applied on the resultant formulae. For example, ∀x B(a, ∃y p(x,y)) would become $\forall x$ B(a, $\exists y_1$ p(x,y_1) which can then be skolemized into B(a, p(x,f(x)$_1$)). Thus the syntactic transformation will preserve the context information of the original formulae. In fact, as shown in [Jiang 88], we can also attach levels to constants to allow complicated de ref and de dicto readings of sentences. In particular, we can either reject or accept the constant domain assumption of possible worlds (or the Barcan Formulae $\forall x$B(p(x)) $\leftrightarrow$ B($\forall x$p(x))) by either levelling universal variable or not levelling. To capture the correct level of intension however, introspective literals need to increment the level of intension to the non-rigid terms of the original literal. A semantic characterization of the scheme is given in [Jiang 88].

Theorem: A epistemic sentence is unsatisfiable iff its skolemized form is.

Definition: An epistemic clause is an epistemic formulae that has a standard clausal form in the scope of every B-operator of positive polarity and has a standard conjunctive normal form in the scope of every B-operator of negative polarity; eg. B(a,p v ¬B(c,q&r)) is an epistemic clause and B(a,p&q) is not.

Now by applying the standard clausal transformation rules and the modal transformation rule B(a, p&q) = B(a,p)&B(a,q), we can transform every epistemic formulae into an epistemic clause.

Theorem: There is an effective procedure which can transform every epistemic formula into a satisfiability-preserving set of formulae in an epistemic clausal form.

Now to solve the control problem of B-resolution, lets first investigate B-resolution's inability to enforce a set of support strategy by an example.

query: ¬B(a,p)

Refutated query: B(a,p)

Database:
B(a,¬p v q)
¬B(a,q)

If we use the refuted query as the set-of-support, because the B-clause is positive, we cannot apply the B-resolution to open a view. On the other hand, the theorem/query would be proved in a general B-resolution-based system by opening a view for the negative B-clause ¬B(a,q). Thus B-resolution with direct set-of-support will not be complete.

To solve this problem, we propose an indirect kind of set of support strategy by associating a set of support for each view. The basic idea is to keep the refuted query/goal as the set of support of the initial view. To resolve a negative B-literal ¬B(a,p) in a set of support derivation in a view, we open a new view with *p* as its set of support. However everytime we attempt to resolve a positive B-literal B(a,q) in a view, we look for a negative B-literal of a unifiable agent in the view to open a new view with *q* as its set of support. Thus in the above example, to resolve B(a,p), we use ¬B(a,q) to open a view with *p* as its set of support. In this subview, we will then be able to obtain a proof.

Theorem (completeness): Let S and T be sets of epistemic formulae, if S+T is unsatisfiable and S is satisfiable, then there exists an indirect set of support B-resolution proof D with T as the set-of-support of the initial view consisting of S+T.

Because the set of support strategy is compatible with linear resolution, we can also add linear resolution to each view in a B-resolution.

Theorem: Let S and T be sets of formulae, if S is satisfiable and S+T is unsatisfiable, then there exists an indirect set of support with linear B-resolution proof with T as the set-of-support of the initial view consisting of S+T.

However the resultant system is still very inefficient compared with its first order counterpart even though we omit the recursive nature of B-resolution. This is because the set of support for each view can be exponentially expanding as subviews are generated. Consider the following partial view with clause 1 as the set of support:

1. ¬a
2. a v ¬B(ag,p v q v r)

The resolution between 1 & 2 will produce a negative B-literal which would then result a view with ¬p, ¬q and ¬r as its set of support. This is because multiple clauses may be generated when negation is passed through a negative B-literal. For this reason and for the reason that Horn clauses can be implemented efficiently, a restricted form of epistemic logic — epistemic Horn logic, is proposed.

Definition: An epistemic Horn clause is an epistemic clause in which every B-literal of positive polarity is in standard Horn clausal form and every B-literal of negative polarity is in a conjunctive normal form of unit literals in which there is at most a negative literal; eg. B(a,p v ¬q) & ¬B(a, c & b) is an epistemic Horn clause; whilst

$\neg B(a,c \vee \neg b)$ is not.

The basic idea of the restriction is that when negation (if any) is passed through a B-operator, the resultant clause will be a single Horn clause. We thus have the following theorem. **Theorem:** The set of support of every view in a proof structure of the epistemic Horn logic is always a single Horn clause.

Theorem: For any epistemic Horn logic proof (or view) structure, every view of the proof will be in standard Horn clausal form.

We only need to show that the remainer literals of a view will be in a Horn clausal form for this theorem to be true. Since a view is always opened by a negative B-literals with clauses added to the view from positive B-literals and there is at most one positive literal in a standard Horn clausal form, thus there is at most one positive literal in the remainer clause of each view.

Since each view contains only Horn clauses and its set of support is a single Horn clause, we can thus have a sound and complete linear *input* (SLD-like) B-resolution strategy with selection function for a consistent set of Horn epistemic formulae.

Theorem: The epistemic SLD-like proof system is sound and complete for a consistent set of epistemic Horn clauses.

Thus epistemic reasoning can be performed by applying recursively a SLD-like procedure in each view. However unlike standard SLD procedure, the epistemic Horn clauses in a view need not be definite clauses; ie. it can have negative clauses. A consequence is that we can have incomplete or disjunctive answers to a proof in a view. This is because each view can contain ans-predicates for input variables to capture the incomplete answers of a proof. However it should be noted that there can only be one ans-literal in a proof of a view for the remainder literals of the negative B-literal that opened the view; although there can be many ans-literals for the remainder literals of the positive B-literals added to the view. This is because the input variable from the negative B-literal must not get disjunctive answers from a semantical point of view. For example, given $B(p(a) \vee P(b))$, we cannot conclude $\exists x\ B(p(x))$ although we can conclude that $B(\exists x\ p(x))$. On the other hand, input variables from positive literals can get disjunctive answers. For example, given $\neg B(p(a) \vee P(b))$, and $\forall x\ B(p(x)) \vee q(x)$, we can obtain $q(a) \vee q(b)$.

So far nothing has been said about unification. Because terms are additionally associated with a level of intension, we can thus first define a strict rule of unification.

Strict Unification: Two terms are unifiable if they and their level of intension are unifiable in the stardard sense. Since a composite structure of a term and its level of intension is also a term, there is no need to change the standard unification algorithm.

Because some terms may be *rigid*, ie. have the same interpretation in all possible worlds, we can treat their level of intension (indicated by a '_' sign) similarly as the '_' operator in Prolog, ie. matchable to anything. In the case the Barcan formulae are assumed, the universal variable will be treated as *rigid* with their level of intension denoted by the '_' sign. The strict unification will thus be the same irrespect of the assumption of the Barcan formulae.

Sometimes, it is also useful to allow certain relationships of level of intension to be unifiable. For example, often it is useful to deduce an intensional meaning from an extensional meaning. Thus given $\exists x B(p(x))$, we can deduce $B(\exists x P(x))$ but not the other way around. This example also shows that the relationships may be directional. To define these relationships, we specify a set of builtin predicates which can be used in the strict unification algorithm in a similar fashion as many-sorted unification. For example, Unify(1,2) would allow any two unifiable terms to be unifiable if the level of intension of one term is 1 and the other is 2. On the other hand, Goal-unify(j,i) $\leftarrow$ j>i would only allow two unifiable terms to be unifiable if the goal term's level is greater than the other term's level. This is in fact the case of the intension-from-extension example mentioned above.

4. An autoepistemic predicate logic

Autoepistemic (AE) logic is a nonmonotonic logic that is concerned with the reasoning of beliefs of an ideally rational agent who reflects upon his own beliefs [Moore 85b]. Originally it was only defined for propositional case. Otherwise, the logic is restricted to closed epistemic formulae in the sense that no quantifying-in variables are allowed. Although there were attempts made to include quantifying-in beliefs, eg. [Niemela 88], however the proposed proof procedures are normally restricted to function free epistemic formulae. As a result, existential quantified variables are not allowed. With the assumption of Barcan Formulae, the AE predicate logic in this case is essentially reduced to one worse than closed AE logic due to the lack of existential quantifiers.

Although AE logic is famed for its ability to reason about incompleteness, however most of the incomplete knowledge are normally quantifying-in type of epistemic formulae. For example, the statement "I believe that I know all the teachers" can be represented in agent *I*'s belief as:

$$\forall x\ \mathrm{Teacher}(x) \rightarrow B(I,\ \mathrm{Teacher}(x)).$$

Another impact of the introduction of quantifying-in to AE logic is that it will invalidate Konolige's equivalence relationship [87] between AE logic and default logic [Reiter 80]. In fact, AE quantified logic becomes more powerful than the default logic. The translation of the latter to the former is still possible and local, but not conversely. In this section, we thus attempt to extend closed AE logic to include quantifying-in variables.

It is noted by Moore [84] that an AE logic can be formalized in a transitive and euclidean possible worlds semantics. In particular, the Stalnaker's three stable set conditions [80] of an AE theory can be charaterized by a complete S5 kripke structure for the theory. As a consequence, an AE extension T of a set A of premises can be characterized by the following non-constructive proof-theoretic fixpoint equation [Konolige 87]:

$$T = \{\Phi \mid A \cup BT_0 \cup \neg B\neg T_0 \vdash_{k45} \Phi\}$$

where T_0 is the consequential closure of the ordinary (first order non-modal sentences) formulae in the AE extension. In section 3, we have already developed an efficient resolution based proof mechanization of the monotonic quantified K45 logic. It is thus natural to extend this mechanization to the AE predicate logic. Unlike tableau-based [Niemela 88] or sequent-based [Jackson & Reichgelt 88] AE-like proof theories, the (SLD-like) resolution based approach is more goal oriented.

Because of the non-monotonic nature (characterized by fixpoint operations) of the AE logic, the deployment of the monotonic K45 mechanization to the AE logic need to be augmented by the extensions of BT_0 and $\neg BT_0$. This however present several difficulties. For start, the second extension (called *negative* introspection) is itself not even semi-decidable [Konolige 85]. Furthermore, these extensions are part of the AE extension itself, thus involving making assumptions and checking for groundness. For example, from $Bp \rightarrow p$, an AE extension could assume P to be true and be justified through groundness.

To solve these problems, we propose the following solutions. First, instead of having a general negation as failure type of extension, we adopt an epistemic Negation As finite Failure (NAF) rule of inference (omitting the increment of level of intension for clarity reason) for the negative introspection. In this case, the AE extension will then be made semi-decidable instead of non-semi-decidable.

Epistemic NAF: If $\neg \vdash p$, then $\vdash \neg Bp$

The finite failure restriction is reasonable since an agent can only conclude non-beliefs through a finite process upon which he can judge. This by no means relaxes the assumption of an ideal rational agent. As far as reasoning is concerned, an agent has still infinite power. The difference here is that the infinite power will not be able to obtain conclusive finite result from an infinite process.

The epistemic NAF rule is in fact similar to Clark's NAF [Clark 78] except that it is made at an epistemic level. It is nevertheless more intuitively sound than Clark's NAF. After all if we cannot prove something, we should not deduce that it is false; rather it is commonsense to deduce that we do not believe it. Furthermore, we also offer an interpretative semantics to NAF notions in the same way to default reasoning [Konolige 87]. As a result, it also makes it clearer to express non-beliefs about non-beliefs than Clark's NAF rule. In this way, we could replace the negative literal $\neg p$ in traditional logic programming paradigm with $\neg Bp$. The resultant system then operates on an epistemic level.

However like Clark's NAF, the epistemic NAF can still result inconsistent beliefs at epistemic level. For example, given $Bp \vee Bq$, the epistemic NAF could result $\neg Bp$ & $\neg Bq$. From an autoepistemic point of view, this inconsistency would result an empty AE extension. To solve this problem, we restrict our AE logic to AE Horn logic. This restriction is reasonable and practical as normally a Horn logic can yield an efficient proof mechanization.

To characterize the positive introspection, we introduce another rule of inference (again omitting the increment of level of intension for clarity reason):

PI: If ⊢ p, then ⊢ Bp.

This is in fact the kind of rule of inference used in McDermott's modal non-monotonic logics [82]. As pointed out by Moore [85b], instead of treating the inference as part of a AE theory, ie. Bp∈T if p∈T, the mere use of inference as a rule would result no AE extension of Bp→p that contains p. On the other hand, by assuming p and justifying it, a Moore's AE extension would contain p. Moore argued for this pecularity that McDermott's B is more like knowledge rather than belief. It is felt that this argument seems ill-founded. Afterall, a belief of an agent is a knowledge as far as the agent is concerned. Thus believing p to be true as a belief of an agent can only be justified iff p is itself a belief of the agent.

Konolige [87] proposed several restricted definitions of AE extension to avoid Moore's pecularity. They are moderately grounded and strongly grounded extensions.

Definition: A moderately grounded AE extension T is

$$T = \{\Phi \mid A \cup BA \cup \neg B\neg T_0 \vdash_{k45} \Phi\}$$

Definition: A strongly grounded AE extension T is

$$T = \{\Phi \mid A' \cup BA' \cup \neg B\neg T_0 \vdash_{k45} \Phi\}$$

where A's is the set of sentences of A whose ordinary part is contained in T.

The moderately grounded extension would avoid Moore's pecularity for Bp→p but not of the set of sentences that contains intermediately derived Bp, eg {¬Bp→q,Bp→p}. The latter problem can be avoided through the stronly grounded AE extension.

Despite the strongly groundness restriction, Moore's AE logic is still suffered with the problem of 'theoremhood' in the sense that there could be multiple (eg. {¬Bp→q,¬Bq→p}) and empty (eg. {Bp}) AE extensions. The former may be justified by assuming that an agent can have alternative belief sets as far as the agent is concerned. The latter however is less justified. Given some beliefs of an agent, the possibility of an empty AE extension or belief set appears to be intuitively contradictory. Afterall, the agent did have some beliefs before his reflection.

To solve this problem, we use a stratified AE predicate logic. We extend Gelfond's [87] definition for propositional logic to predicate logic.

Definition: An AE theory T is stratified if there is a partition $T=T_0+..+T_n$ such that

1. T_0 is ordinary (possibly empty).
2. Negative epistemic Horn clauses do not belong to T_k where k>0.
3. If a predicate p belongs to the positive literal of an epistemic Horn clause in T_k, then atoms with predicate p do not belong to $T_0,..,T_{k-1}$ and atoms Bf where f contains p do not belongs to $T_0,..,T_k$.

Follow the stratification definition, we have the following theorem, although the converse is not true.

Theorem: Every stratified AE extension is strongly-grounded and it is thus also a syntactic property of an AE logic.

Now Gelfond [87] has shown that every consistent stratified AE propositional theory has a unqiue AE extension. In [Jiang 88], we have shown that Herbrand theorem "a set of universal sentences is unsatisfiable iff a finite subset of its ground instances are" remains to be valid with the introduction of level of intension. Thus with unification and Gelond's propositional result, we can have the following theorem:

Theorem:: Every consistent stratified AE predicate theory has a unique AE extension.

In other words, a stratefied AE theory will define a unique and non-empty extension. It thus provides a better definition of 'theoremhood'.

IF we use K45+ENAF+PI to characterize the proof mechanization of our stratified AE Horn predicate logic, an AE extension T in this logic can thus be summarized as follows:

$$T = \{\Phi \mid A \vdash_{k45+ENAF+PI} \Phi\}.$$

We illustrate the AE logic with an example. It can be seen that the database is consistent since the agent's concept of concept of father of John is different from its concept of father of John.

Query: ¬teacher(Tom).

Goal: 0. teacher(Tom).

Database:

1. teacher(John).
2. teacher(father(John)).
3. ¬B(teacher($father(John)_1$)).
4. ¬teacher(x) v B(teacher(x)).

Resolve 0+4, yield:

5. B(teacher(Tom))

Apply ENAF and B-resolution, yield:

6. ¬B(teacher(Tom))

5+6 completes the proof.

If we add father(John)=Tom, then apply paramodulation and B-resolution in a view, we will not get a proof.

It may be noted that the AE logic described so far is for single agent, eg. the current database. In practice, it is often useful to allow a database to describe another database. This is also the case in distributed problem solving. The K45 system described in Section 3 is applicable to multiple agents. Thus it is tempting to add multiple agents to an AE theory.

We first define an stratified multiple agents AE theory as a stratified AE theory from every agent's point of view. Because of the existence of unique AE extension, we can thus add introspection of other agents from the current agent's point of view. Such introspection can be mechanized in exactly the same way as the current agent except that it is done as if the current agent is put in the shoe of the other agent. This introspection can also be be extened to nested agents if stratification is also extended correspondingly.

5. Conclusion

In this paper, we have presented an epistemic Horn logic with an epistemic SLD-like proof mechanization based on Konolige's B-resolution. This is intended as the theoretical foundation of *epistemic logic programming* in the sense it can serve directly as a programming language. We have also demonstrated how the K45 mechanization of the monotonic K45 epistemic Horn logic may be extended to a nonmonotonic AE logic that allows quantifying-in variables. An epistemic NAF rule of inference instead of Clark's NAF has also been introduced. In addition, we have also attempted to add multiagents reflection to AE logic.

Rather than attempting to map modal notions into a Prolog-like SLD interpreter (eg. [Fujita etal 86]), we have directly advocated the use of a direct epistemic SLD-like interpreter. After all Prolog is not equal to logic programming.

Acknowledgements

The author would like to thank Prof. R. Turner (Essex) for his helpful discussions. Special thanks must also be given to Prof. J. McCarthy (Stanford), Dr D. Perlis (Maryland), Prof. R. Reiter (Toronto), Dr L. Wallen (Austin), Dr A. Ramsay (Sussex) and Dr H. Reichgelt (Notingham) for taking up their valuable time to discuss some of the works reported in this paper. The author is supported by an ICL post-doctoral research fellowship.

References

M. Abadi & Z. Manna (1987) "Temporal Logic Programming" IEEE Inter. Symp. on logic programming, San. Francisco, pp.4-16.

K. Clark (1978) "Negation as failure" in logic and database eds. H. Galliare and J. Minker, Plenum, New York, 293-322.

L. Farinas del Cerro (1983) "Temporal reasoning and termination of programs". IJCAI, Karlsruhe, West Germany, pp.926-929.

L.Farino del Cerro (1986) "MOLOG: A system that extends Prolog with modal logic". New Gen. Computing 4 pp.35-50.

Fujita et al (1986) "Logic programming based on temporal logic and its compilation to Prolog" 3rd International Conf. on Logic programming 86, London, July, 1986.

Geisler&Konolige (1986) "A resolution method for quantified modal logic" in Proc. theory of knowledge, ed. Y. Halpern.

M. Gelfond (1987) "On stratified autoepistemic theories" AAAI 87, pp.207-211.

Y.J. Halpern, Y. Moses (1985) "A guide to the modal logics of knowledge and belief: preliminary draft". IJCAI 85, Vol.1

P. Jackson, H. Reichgelt (1987) "A general proof method for first order modal logic". IJCAI-10, pp. 942-944.

P. Jackson, H. Reichgelt (1988) "A modal proof method for doxatic reasoning in incomplete theories", ECAI 88, pp.480-485.

Y.J. Jiang (1988) "Intension, logical omniscience, quantified beliefs and epistemic resolution" IMSIM 3, Torino, Italy, Oct., 1988.

K. Konolige (1981) "A first order theory of knowledge and action" Machine Intelligence 10.

K. Konolige (1985) "A theory of introspection" IJCAI 85.

K. Konolige (1986) "Resolution and quantified epistemic logics". 8th Int. Conf. on automated deduction, LNCS No 230, pp.199-208.

K. Konolige (1987) "On the relationship between default and autoepistemic logic" IJCAI 87.

S.A. Kripke (1959) "A completeness theorem in modal logic" J. of Symbolic Logic 24, pp.1-14.

G. Lakemayer (1987) "Tractable Meta-reasoning in propositional logics of belief" IJCAI 87.

H.J. Levesque (1984) "A logic of implicit and explicit belief". Proc. National Conf. on Artificial Intelligence. pp.198-202.

J.W. Lloyd (1984) "Foundations of logic programming" Spring-Verlag, 1984.

J. McCarthy et al (1978) "On the model theory of knowledge" Memo AIM-312, Stanford Univ.

D. McDermott (1982) "Non-monotonic logic II: non-motonic modal theories' in JACM 29 (1) 33-57.

R. Moore & G. Hendrix (1979) "On the computational semantics of belief sentences". SRI Technical Note 187, Menlo Park CA.

R.C. Moore (1984) "Possible worlds semantics for AE logic" SRI note 337, Menlo Park, 1984.

R.C. Moore (1985a) "A fomral theory of knwoledge and action" in Formal theories of the commonsense world, ed. Hobbs, J.R. and Moore,R.C, Ablex Pub. Corp., 1984.

R. C. Moore (1985b) "Semantic considerations of non-monotonic logic" AI 25 (1).

L. Morgenstern (1986) "A first order theory of planning, knowledge and action" Proc. of theory of knowledge, 1986.

I. Niemela "Autoepistemic Predicate logic" ECAI 88, pp.595-599.

D. Perlis (1987) "Languages with self-reference II: Knowledge, Belief and Modality". UMIACS TR 1987, also a shorter version in IJCAI 86.

R. Reiter (1980) "A logic for default reasoning" AI 13 (1-2).

Y. Sakakibara (1987) "Programming in modal logic: An extension of Prolog based on modal logic" Logic programming 87, Japan, pp.81-91.

R.C. Stalnaker (1980) "A note on non-monotonic modal logic" Dept. of Philo. Cornell University.

M. Stickle (1985) "Automated deduction by theory resolution" 9th IJCAI, 1985.

L. Wallen (1987) "Matric proof methods for modal logics" IJCAI 87.