A RESOLUTION METHOD FOR QUANTIFIED MODAL LOGICS OF KNOWLEDGE AND BELIEF

Christophe Geissler
École Nationale Superieure
des Telecommunications
Paris, France

Kurt Konolige
Artificial Intelligence Center
SRI International
Center for the Study of
Language and Information
Stanford University

ABSTRACT

B-resolution is a sound and complete resolution rule for quantified modal logics of knowledge and belief with a standard Kripke semantics. It differs from ordinary first-order binary resolution in that it can have an arbitrary (but finite) number of inputs, is not necessarily effective, and does not have a most general unifier covering every instance of an application. These properties present obvious obstacles to implementation in an automatic theorem-proving system. By using a technique similar to ***semantic attachment***, we obtain a very natural expression of B-resolution that is potentially efficient, and easily understood and controlled. We have implemented the method and used it to solve the Wise Man Puzzle.

Introduction

Modal logics with a possible-world semantics have been widely used to formalize various accounts of belief and knowledge in Artificial Intelligence (AI) (Moore [17], Levesque [14], and McCarthy [15]) and more recently in Computer Science in general (Halpern and Moses [7]). These logics are important both as an analytic tool in analyzing systems, and as a means of endowing artificial agents with the ability to reason about the knowledge and belief of other agents. In this latter category we include query answering (Levesque [14]), dialogue understanding (Appelt [2], Cohen and Perrault [3], and Grosz [6]), and multiagent planning systems (Konolige [9], Rosenschein and Genesereth [19]). It is widely recognized (see, for example, Moore [17]) that efficient and conceptually transparent proof methods are needed for these systems. By *efficient* we mean that computer automation of the methods produces those proofs needed for reasoning about belief within allowable time and space limitations; by *conceptually transparent* we mean that the action of the theorem prover is readily understandable, and the proofs clear and direct, so that it is easy to check and modify the behavior of the system.

While there has been a good deal of useful work on decision procedures for propositional modal logics (see Halpern and Moses [8]), fewer results have been obtained for quantified modal logics. Hilbert-style and natural deduction axiomatizations (Kuo [13]) exist, but there are no serious proposals to automate them. As an alternative, McCarthy [15] proved theorems about a modal system by axiomatizing its possible-worlds semantics in a first-order system; subsequently Moore [17] used this technique to efficiently automate proofs. However, this is not a direct proof technique, because it involves reasoning about possible worlds and other objects of the semantic domain, rather than manipulating beliefs directly.

In this paper we present an efficient, direct proof method for a modal logic of belief that is based on Robinson's resolution principle ([18]). First we briefly review the modifications to first-order resolution that are necessary to establish the B-resolution rule. This rule has several properties which present problems for implementation in an automatic theorem-prover: it is non-effective, so we must find a way to apply it incrementally; and we must also develop techniques for controlling the size of the search space it generates.

The key idea we use to solve both problems is the concept of *semantic attachment* (Weyhrauch [22]). To illustrate this technique, consider the statements "A believes P" and "A doesn't believe $P \vee Q$." These are inconsistent in possible-worlds semantics, because there is no world compatible with A's beliefs in which P is true and $P \vee Q$ is false. We can show this inconsistency by deducing a contradiction from P and $\neg(P \vee Q)$. Thus we can prove facts about belief statements by attaching to their meaning and performing a computation (in this case, deduction). The structure of reasoning is clear, and it is easy to understand and control the often confusing embedding of agents reasoning about other agents' beliefs.

The Resolution Method

Language preliminaries

We assume a modal language L built on a first-order language with function symbols. The modal atoms are of the form $[S]\phi$, where S is a term denoting an agent and ϕ is a formula denoting a proposition. The intending meaning is that a believes or knows ϕ.

The semantics of L are the standard Kripke possible-world models with an accessiblity relation for each agent. It is well-known that various properties of knowledge and belief can be expressed by placing conditions on the accessibility relations (Halpern and Moses [8]). For simplicity of exposition we will limit ourselves to the system K, which has no restrictions, although versions of the resolution method have been derived for all the important systems (T, $K4$, $S4$, $K5$, $K45$, $S5$, *etc.*).

For technical reasons we make one further assumption: the domain of each possible world is a subset of the domain of any accessible world. Rescinding this restriction is possible, but introduces further complications in the resolution method that we do not wish to address here.

Herbrand's Theorem

One version of Herbrand's Theorem is: *a set of universal sentences is unsatisfiable if and only if a finite subset of its instances are.* Stated in this form, it sanctions the "lifting" of proofs over ground sentences to those with universal variables. Unfortunately, Herbrand's Theorem is not true for modal logics with Kripke semantics, as we can see from the following counterexample:

$$\begin{array}{l} P(m(c)) \\ \neg[S]P(m(c)) \\ \forall x.Px \supset [S]Px \end{array} \tag{1}$$

We can construct a model as follows. Let P be the property of being non-Italian, let $m(x)$ denote the mayor of the city x, and c denote New York. Suppose S believes the mayor of New York is Fiorello LaGuardia (and not Ed Koch, the actual mayor); it is easy to confirm that all the sentences are satisfied.

Now if we substitute $m(c)$ for x in the third sentence, the resulting set is unsatisfiable. The reason is that, although x must refer to the same individual in all possible worlds, the substituted expression $m(c)$ need not. So even if a universal sentence is true in a model, some of its instances can be false.

Our solution to this problem is to redefine the meaning of "instance" by introducing a *bullet operator* ($\bullet$) whenever there is a substitution for variables inside the context of modal operators. $\bullet t$ always refers to whatever t denotes in the actual world, no matter what the context of interpretation; the bullet operator thus acts like a rigid designation operator for terms. In the above example, substituting $m(c)$ for x yields

$$P(m(c)) \supset [S]P(\bullet m(c)), \tag{2}$$

which is still satisfied by the original model, since $\bullet m(c)$ refers to Ed Koch even in the context of S's beliefs.

With this revised definition of substitution (and instance), Herbrand's Theorem is once more valid (see Konolige [11]).

Clause form

Converting to clause form is the same as for first-order logic, with modal atoms having different argument structures treated as if they were different predicate symbols. Thus $[S]\forall x.Px$, $[S]Pa$, and $[S]\exists x.Px$ are all considered to be different nilary predicates. Modal atoms with n free variables are n-ary predicates, *e.g.*, $[S](Px \wedge \exists y.Py)$ and $[S](\exists y.Py \wedge Px)$ are different unary predicates with the free variable x. Note that variables quantified under the scope of the modal operator remain unanalyzed or inert in B-resolution, and do not interact with variables quantified outside the operators. An example:

$$\forall x \exists y Rxy \supset [S]\exists z.Rxyz \quad \Rightarrow \quad \neg R(x, f(x)) \vee [S]\exists z.R(\bullet x, \bullet f(x), z) \tag{3}$$

Note that substitution of $f(x)$ for y in the modal context is done with $\bullet f(x)$. Also, in clause form we automatically insert a bullet operator before quantified-in variables (like x), to distinguish them from variables whose quantifiers are inside the scope of modal operators (like z).

B_K-resolution

Our resolution method is based on Stickel's *total narrow theory resolution* rule [21], which has the following form. Let L be a language that embeds a theory T, that is, the axioms of T contain a set of predicates P of L (but not necessarily all predicates of L). Suppose there is a decision procedure for determining a set of ground literals W in P to be unsatisfiable (according to T). Then

$$\frac{\begin{array}{c} L_1 \vee A_1 \\ L_2 \vee A_2 \\ \vdots \\ L_n \vee A_n \end{array}}{A_1 \vee A_2 \vee \ldots \vee A_n}, \quad \text{when } \{L_1, L_2, \ldots L_n\} \text{ is } T\text{-unsatisfiable} \tag{4}$$

is a resolution rule that is sound and complete for the theory T. This rule includes binary resolution as a special case, where L_1 and L_2 are complementary literals.

For the modal logic K, this rule is rephrased as follows. Let Γ be a set of formulas of L; by $\Gamma^{\bullet}$ we mean the same formulas with the bullet operator uniformly replaced by a unary function not appearing in Γ. Then, in the case of ground clauses,

$$\frac{\begin{array}{c} [S]\phi_1 \vee A_1 \\ [S]\phi_2 \vee A_2 \\ \vdots \\ [S]\phi_n \vee A_n \\ \neg[S]\delta \vee A \end{array}}{A_1 \vee A_2 \vee \ldots \vee A_n \vee A}, \quad \text{when } \{\phi_1, \phi_2, \ldots \phi_n, \neg\delta\}^{\bullet} \text{ is } K\text{-unsatisfiable.} \tag{5}$$

is a sound resolution rule for K. If we are allowed to infer instances of any clause, then by Herbrand's Theorem for L it is also a complete rule. Because it is a rule for the logic K, we call this the B_K-resolution rule.

Implementation problems

The following problems must be solved to obtain an efficient implementation of B_K-resolution.

1. There is no decision procedure for unsatisfiability in quantified K.
2. Although we have given the resolution rule for the ground case, to be useful it must also be able to handle free variables in the arguments of the modal atoms. In this respect, B_K-resolution is more complicated than ordinary binary resolution, because in general there is no most general unifier covering all possible ground resolutions. For example, consider the following two clauses:

$$\begin{array}{l} [S](P{\bullet}a \wedge P{\bullet}b) \\ \neg[S]P{\bullet}x \end{array} \tag{6}$$

There are two substitutions for x which yield a resolvent (a/x and b/x), but no "most general" unifier.

3. The search space is exponential in the number of modal literals. Consider the following example:

$$\begin{array}{l} [S]r \vee A_1 \\ [S]p \vee A_2 \\ [S](p \supset q) \vee A_3 \\ \neg[S]q \\ \hline A_1 \vee A_2 \vee A_3 \end{array} \tag{7}$$

Only the last three clauses are needed for the resolution; indeed, including the first clause will not lead to a proof if A_1 cannot eventually be resolved away. In order to be complete in general theory resolution rules must be applied to a *minimal* set of unsatisfiable literals. If there are n clauses containing one modal literal each, there are 2^n possible B_K-resolutions that must be tried.

4. The above search space problem is compounded by the presence of variables, since a given clause may have to be used twice. For example, there is a resolution of the clauses

$$\begin{array}{l} Px \vee [S]P{\bullet}x \\ \neg[S](P{\bullet}a \wedge P{\bullet}b) \end{array} \tag{8}$$

yielding the resolvent $Pa \vee Pb$. However, this requires the first clause to be used twice in the belief resolution rule (5), as follows:

$$\begin{array}{l} Pa \vee [S]P{\bullet}a \\ Pb \vee [S]P{\bullet}b \\ \neg[S](P{\bullet}a \wedge {\bullet}b) \\ \hline Pa \vee Pb \end{array}$$

5. If there are several clauses with negative belief literals for the same agent, we may duplicate our efforts in deciding unsatisfiability each time. Consider again example (7), and suppose there is another clause with the negative belief literal $\neg[S](q \wedge p)$. A resolution using this clause and the positive belief clauses exists; however, in finding it we duplicate the work involved in deciding that $\{p, p \supset q, q\}$ is unsatisfiable.

A proof procedure for B_K-resolution

Semantic attachment

We now give a version of B_K-resolution which treats the problems just mentioned. The key idea is to replace the unsatisfiability condition of (5) with a recursive call to the theorem-prover, using as input the arguments of the modal atoms. If the recursive call is successful, then the resolution rule can be applied. Because it is not certain that the call will terminate, processing of the call must be interspersed with other activities of the theorem-proving process. At any given time, the theorem prover must "time-share" its attention between ordinary binary resolution and multiple invocations of the semi-decision procedure.

In addition, we structure the semi-decision procedure so that it accepts free variables in formulas, and eventually returns substitutions covering all proofs that can be found with instantiations of these variables.

The idea of showing validity or unsatisfiability of a predication by means of a computation that reflects the intended meaning of the predicate is called *semantic attachment* (Weyhrauch [22]). In belief resolution, we compute the unsatisfiability of a set of modal literals by performing deductions on their arguments. This process is a generalization of semantic attachment in two ways. First, we show the unsatisfiability of a *set* of modal literals, rather than a single atom. Second, by allowing variables, we are able to perform many different instances of semantic attachment at once. Without this ability, belief resolution would not be efficient in the presence of variables, because we would have to first chose an instantiation of the modal literals without knowing whether it would lead to a resolution or not.

An example

Our implementation of (5) has much in common with Kripke's [12] device of auxiliary tableaux. We define a structure called a *view*, which is an annotated instantiation of the theorem-proving process. Here is a short example to illustrate the basic idea. Assume initial clauses:

1. $[S]Pa$
2. $\neg Pb$
3. $Qx \vee Px \vee [S]P{\bullet}x$
4. $\neg[S](Pa \wedge P{\bullet}y) \vee Qy$
5. $\neg Qb$

Note that we have added a bullet operator to each variable under the scope of a belief atom. Ordinary resolution work as usual, for example, 2 and 3 can be resolved to yield:

6. $Qb \vee [S]P{\bullet}b$ 2,3

Clause 4 contains a negative belief literal, and we open a new view in an attempt to resolve it:

$\underline{\text{view } S\text{, rems } (0, Qy)}$
1. $\neg Pa \vee \neg Pn(y) \vee Ans(0,y)$

This is view *for* S, the agent of the belief. The clause is derived from $\neg(Pa \wedge P{\bullet}y)$; note the substitution of the function n for the bullet operator. The *Ans* predicate keeps track of the input free variable y; 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 Qy will be returned with an appropriate binding for y as a deduced clause of the original proof.

We can add the arguments of positive belief atoms to the view, as in clause 1 (of the original clause set). The view now contains:

$\underline{\text{view } S\text{, rems } (0, Qy)}$
1. $\neg Pa \vee \neg Pn(y) \vee Ans(0,y)$
2. Pa

These two clauses can be resolved, yielding:

$\underline{\text{view } S\text{, rems } (0, Qy)}$
1. $\neg Pa \vee \neg Pn(y) \vee Ans(0,y)$
2. Pa
3. $\neg Pn(y) \vee Ans(0,y)$ 1,2

Clause 6 has a positive belief literal, so we add its argument also:

$\underline{\text{view } S\text{, rems } (0, Qy)\ (1, Qb)}$
1. $\neg Pa \vee \neg Pn(y) \vee Ans(0,y)$
2. Pa
3. $\neg Pn(y) \vee Ans(0,y)$ 1,2
4. $Pn(b) \vee Ans(1)$

Clause 4 contains an answer predicate with a new index. The remainder of the original clause containing the positive belief atom (6) is inserted into the indexed remainder list. Note that the bullet operator was replaced with the same function n as in clause 1.

Clauses 3 and 4 resolve, yielding a clause containing just answer predicates:

view S, rems $(0,Qy)$ $(1,Qb)$		
1.	$\neg Pa \vee \neg Pn(y) \vee Ans(0,y)$	
2.	Pa	
3.	$\neg Pn(y) \vee Ans(0,y)$	1,2
4.	$Pn(b) \vee Ans(1)$	
5.	$Ans(0,b) \vee Ans(1)$	3,4

Now we gather up the remainders indexed by the answer predicates in the answer clause, namely, Qb (index 1) and Qy (index 0). Using the substitution b/y generated by the Ans-predicate, we return $Qb \vee Qb$ $(= Qb)$ as the result.

1.	$[S]Pa$	
2.	$\neg Pb$	
3.	$Qx \vee Px \vee [S]P{\bullet}x$	
4.	$\neg[S](Pa \wedge P{\bullet}y) \vee Qy$	
5.	$\neg Qb$	
6.	$Qb \vee [S]P{\bullet}b$	2,3
7.	Qb	1,4,6

Clauses 5 and 7 resolve to give the null clause, completing the proof.

Views

Formally, a view is an annotated, finite set of clauses. The annotation is a list of remainders to be used in returning a result from the view. We perform four operations on views.

Opening a view. Let C be a clause of the form $\neg[S]\phi \vee A$. A view for S may be created. Into it we insert clauses formed from ϕ as follows. Let W be the set of clauses resulting from putting $(\neg\phi)^{\bullet}$ into clause form, and let $\mathbf{x}$ be the free variables of ϕ. We insert each member of W into the view, disjoining the answer predicate $Ans(0,\mathbf{x})$. We also add the annotation $(0, A)$ to the remainder list.

Adding a positive belief literal. Let C be a clause of the form $[S]\phi \vee A$, and let $\mathbf{x}$ be the free variables of ϕ. To any existing view for S we may add the clauses formed by converting $\phi^{\bullet}$ to clause form and disjoining $Ans(n,\mathbf{x})$, where n is a new index. (n, A) is added to the remainder list.

Stepping a view. A resolution step may be performed in any view. This includes using one of the four operations described here to create and manipulate embedded views.

Returning an answer. If a clause containing only answer literals is deduced in a view, we may assert a new clause in the proof containing the view. Let

$$\begin{array}{l} Ans(0,\mathrm{a})\vee \\ Ans(n_1,\mathrm{a}^1_1)\vee \;\cdots\; \vee Ans(n_1,\mathrm{a}^1_{i_1})\vee \\ \quad\vdots \\ Ans(n_k,\mathrm{a}^k_1)\vee \;\cdots\; \vee Ans(n_k,\mathrm{a}^k_{i_k}) \end{array}$$

be the answer clause, and let $A_n(\mathrm{a})$ be nth remainder with a substituted for its free variables (a may itself contain variables). The returned clause is:

$$\begin{array}{l} A_0(\mathrm{a})\vee \\ A_{n_1}(\mathrm{a}^1_1)\vee \;\cdots\; \vee A_{n_1}(\mathrm{a}^1_{i_1}) \\ \quad\vdots \\ A_{n_k}(\mathrm{a}^k_1)\vee \;\cdots\; \vee A_{n_k}(\mathrm{a}^1_{i_k}) \end{array}$$

Note that only one $Ans(0,\mathrm{a})$-predicate is allowed in the answer clause. Multiple answer predicates are allowed for positive belief atoms, because more than one instance of these atoms may participate in B_K-resolution.

The use of the *Ans*-predicate allows us to perform a schematic proof, where the input sentences can have free variables. At the end of a proof, the answer predicates give the necessary instantiations of the free variables. Thus we have "lifted" B_K-resolution from the ground case.

If these rules are added to a refutation system using ordinary resolution, we can prove the following result. Let W be a set of clauses of L, and suppose there is a set of ground instances $W\theta$ such that B_K-resolution derives the ground clause C. Then there is a sequence of applications of the view rules on W that returns a clause C' having a ground instance C. Thus these rules faithfully implement B_K-resolution, and together with ordinary resolution form a sound and complete system for K.

Agent terms

We have implicitly assumed that in modal atoms of the form $[S]\phi$, S is a ground term. However, we may easily lift to the more general case of variables, because the agent term is not in a modal context. There are two modifications to the rules. First, in opening a view, $\mathbf{x}$ is a list of all variables in both ϕ and S. Second, we may add a positive belief literal $[S']\psi$ to any view for S, if S and S' have a most general unifier θ. When adding clauses obtained from ψ, we must also disjoin the answer predicate $Ans(0, x\theta)$. This is to assure that a result is returned only if all the participating clauses have unifiable agent terms.

Controlling the search space

Avoiding redundancies

We now address the implementation problems mentioned in the previous section. All of the methods mentioned here maintain the soundness and completeness of B_K-resolution.

1. The view rules split each possible B_K-resolution into a sequence of effective steps. These steps may be interspersed with other activities of the theorem-prover, including ordinary resolution.

2. The use of answer predicates allows a schematic proof within views, so that free variables in the input can be tolerated. Separate proofs are found whenever there is no unifying instance of the input variables that allows a single schematic proof. Consider again example (6). If we open a view for the negative belief atom, and add the positive one, we get:

   $\underline{\text{view } S\text{, rems } (0,x)}$
   1. $\neg Pn(x) \vee Ans(0,x)$
   2. $Pn(a)$
   3. $Pn(b)$

   There are two proofs, one with a/x and one with b/x. Note that if there are no free variables or remainders when we add a clause, we can forgo the answer predicate.

3. We do not need to separately consider all possible combinations of modal literals that could lead to B_K-resolvents. The proof structure of the view takes care of this: only the remainders of those clauses that participated in the proof are returned in the result. Consider again example (7). We open a view for the negative belief literal, and add the arguments of all three positive belief literals. The view looks like this:

   $\underline{\text{view } S\text{, rems } (1,A_1)\ (2,A_2)\ (3,A_3)}$
   1. $\neg q$
   2. $r \vee Ans(1)$
   3. $p \vee Ans(2)$
   4. $\neg p \vee q \vee Ans(3)$

Two resolutions yield $Ans(2) \vee Ans(3)$, returning the result $A_2 \vee A_3$. Although the clause $[S]r \vee A_1$ was added to the view, it was never used in the proof.

4. Although several instances of the same clause may be needed to form a B_K-resolvent, we need only add its belief literal *once* to the view. Consider again example (8). We open a view for the negative belief literal, and add the positive one, obtaining:

   $$\begin{array}{ll} \multicolumn{2}{l}{\underline{\text{view } S\text{, rems } (1, Px)}} \\ 1. & \neg Pn(a) \vee \neg Pn(b) \\ 2. & Pn(x) \vee Ans(1, x) \end{array}$$

   By two resolutions of the second clause, we get:

   $$\begin{array}{lll} \multicolumn{3}{l}{\underline{\text{view } S\text{, rems } (1, Px)}} \\ 3. & \neg Pn(b) \vee Ans(1, a) & 1,2 \\ 4. & Ans(1, b) \vee Ans(1, a) & 2,3 \end{array}$$

   This is a particularly nice result, since the necessity of using multiple copies of a clause in resolution gives rise to nasty control problems.

5. With a little care in indexing the *Ans*-predicates, we can eliminate the redundancies caused by performing the same deductions on the arguments of positive belief literals in different views. Suppose we create only one view for each agent S, but we allow any negative belief literal $\neg[S]\phi$ to be added to this view, in the same way as positive literals are added. We keep track of the answer index so that that we can identify it as arising from a negative belief literal. Resolution are performed as usual within the view. However, to return an answer, we apply the following condition: exactly one *Ans*-predicate arising from a negative belief literal must appear in the answer clause. For example, consider the following clause set:

   $$\begin{array}{ll} 1. & [S]\forall x.Px \\ 2. & [S](\forall x.Px \supset Qx) \\ 3. & A_0 \vee \neg[S]Qa \\ 4. & A_1 \vee \neg[S]Qb \end{array}$$

   We open a single view, inserting all belief literals:

   $$\begin{array}{ll} \multicolumn{2}{l}{\underline{\text{view } S \text{ rems } (0, A_0)\ (1, A_1)}} \\ 1. & Px \\ 2. & \neg Px \vee Qx \\ 3. & \neg Qa \vee Ans(0) \\ 4. & \neg Qb \vee Ans(1) \end{array}$$

Resolving 1 and 2 yields Qx, which can be resolved separately against 3 and 4, returning A_0 and A_1, respectively. However, any resolutions which contain *both* 3 and 4 as ancestors will have $Ans(0)$ and $Ans(1)$ predicates, and so will not generate any result clauses.

The interesting point to note here is that we need open only a single view for each agent, instead of each negative belief literal. The view acts as a deductive testbed in which we try to show different combinations of belief and nonbelief are inconsistent for the agent.

It is possible to generalize this strategy to different agents sharing a set of common beliefs: a single view is created for all the agents. This is particularly useful when one has to deal with agent terms having variables, as in the following clause set:

1. $\neg Px \vee [x]q_1$
2. $\neg Px \vee [x](q_1 \supset q_2)$

$\vdots$

n. $\neg Px \vee [x](q_{n-1} \supset q_n)$

$n+1$. Pa

$n+2$. Pb

$n+3$. $\neg[a]q_n \vee \neg[b]q_n$

It is clear that a and b share many of the same beliefs, and that a great deal of effort will be saved if we assert these beliefs in the same view.

Heuristic control

We have investigated several refinements of the view rules that do not maintain completeness, but may be useful heuristic methods for controlling the size of the search space.

The first is to limit the depth of recursion of views. In a particular problem domain we can often judge whether or not it is useful to reason about agents reasoning about agents reasoning about agents ...and so on. By refusing to open views that are embedded beyond a certain depth, we can control inferences about nested reasoning. More fine-grained control is also possible, if we know that certain types of nested reasoning will be more useful than others. For example, if introspective reasoning is not required (an agent reasoning about his or her own beliefs) then we can refuse to open a view for S if it is embedded in a view for S.

A second method of control is to integrate the view rules into a set-of-support strategy. The most obvious method is to open a view only for negative belief literals in the set of support. The rationale is that we often have a large number of facts about an agent's beliefs, and we are trying to prove from these that the agent has some other belief. A negative literal $\neg[S]\phi$ will appear in the set of support when we are trying to prove that S has the belief ϕ.

Unlike in ordinary resolution, this set-of-support strategy is not complete because it does not permit inferences about lack of belief. For example, we cannot infer $\neg[S]p$ from $[S](p \supset q)$ and $\neg[S]q$, because there are no negative belief literals in the set of support.

Implementation

The view rules for quantified modal K have been implemented using a nonclausal connection-graph theorem prover developed by Stickel [20]. The implementation itself is of interest, especially the method of sharing the attention of the theorem-proving process between views (see Geissler and Konolige [5]).

In addition, we have incorporated theories of common belief, and a simple modal form of the situation calculus (McCarthy and Hayes [16]) as a logic of time. We have derived an automatic proof of the Wise Man puzzle that illustrates these ideas, showing the interaction between belief, action, and time. The proof is conceptually simple and easy to follow.

Other resolution systems for modal logics

Currently there are at least two other approaches to using resolution in a quantified modal logic, both for temporal logics. Fariñas-del-Cerro [4] describes a resolution method for restricted languages in which there are no quantifiers in modal contexts. Such languages are not suitable for knowledge and belief, because it is impossible to express, for example, the statement "Ralph knows that someone is a spy."

Abadi and Manna [1] derive sound and complete nonclausal resolution rules for propositional temporal logics, and are working on extending their techniques to the quantified case.

This work is interesting because it incorporates induction rules, a necessity for completeness in temporal logics containing both *next state* and *always* operators. When belief logics are extended to contain common belief operators, a similar problem surfaces (see Halpern and Moses [7]). We may be able to adopt a solution analogous to those found for temporal logics; currently we have only incomplete resolution rules for common belief.

A major difference between temporal logic resolution and the methods presented here is the use of semantic attachment. The temporal resolution rules are binary rules that transfer arguments in and out of the scope of modal operators; eventually a form results that can be resolved away. This type of resolution does not seem to result in perspicuous, easily-controlled proof methods.

Acknowledgements

We are grateful to Mark Stickel for his help in modifying the connection-graph theorem prover. This research was supported in part by Contract N00014-85-C-0251 from the Office of Naval Research, and in part by a gift from the Systems Development Foundation.

References

[1] Abadi, M. and Manna, Z. (1985). Nonclausal temporal deduction. Report No. STAN-CS-85-1056, Computer Science Department, Stanford University, Stanford, California.

[2] Appelt, D. (1985). *Planning English sentences.* Cambridge University Press, Cambridge, U. K.

[3] Cohen, P. R. and Perrault, C. R. (1979). Elements of a plan-based theory of speech acts. *Cognitive Science* **3**, pp. 177–212.

[4] Fariñas-del-Cerro, L. (1983). Temporal reasoning and termination of programs. In Proceedings of the Eighth International Joint Conference on Artificial Intelligence, Karlsruhe, West Germany, pp. 926-929.

[5] Geissler, C. and Konolige, K. (1986). Implementation of a resolution system for modal logic. Forthcoming Artificial Intelligence Center Tech Note, SRI International, Menlo Park, California.

[6] Grosz, B. J. (1981). Focusing and description in natural language dialogues. In *Elements of Discourse Understanding,* Cambridge University Press, Joshi, A. K., Webber. B, and Sag, I., Eds.

[7] Halpern, J. Y. and Moses, Y. (1984). Knowledge and common knowledge in a distributed environment. In Proceedings of the 3rd ACM Conference on Principles of Distributed Computing, pp. 50–61.

[8] Halpern, J. Y. and Moses, Y. (1985). A guide to the modal logics of knowledge and belief: preliminary draft. In Proceedings of the Ninth International Joint Conference on AI, Los Angeles, California, pp. 479–490.

[9] Konolige, K. (1980). A first-order formalization of knowledge and action for a multiagent planning system. Artificial Intelligence Center Tech Note 232, SRI International, Menlo Park, California.

[10] Konolige, K. (1984). A deduction model of belief and its Logics. Doctoral dissertation, Stanford University, Stanford, California.

[11] Konolige, K. (1986). Resolution methods for quantified modal logics. Forthcoming Artificial Intelligence Center Tech Note, SRI International, Menlo Park, California.

[12] Kripke, S. A. (1959). A Completeness Theorem in Modal Logic. *Journal of Symbolic Logic* **24**, pp. 1–14.

[13] Kuo, V. (1984). A formal natural deduction system about knowledge: modal logic W-JS. Unpublished manuscript, Stanford University.

[14] Levesque, H. J. (1982). A Formal Treatment of Incomplete Knowledge Bases. FLAIR Technical Report No. 614, Fairchild Laboratories, Palo Alto, California.

[15] McCarthy, J. *et. al.* (1978). On the model theory of knowledge. Memo AIM–312, Stanford University, Stanford.

[16] McCarthy, J. and Hayes, P. J. (1969). Some philosophical problems from the standpoint of Artificial Intelligence. In *Machine Intelligence 4*, B. Meltzer and D. Michie editors, Edinburgh University Press, Edinburgh, Scotland, pp. 120–147.

[17] Moore, R. C. (1980). Reasoning about knowledge and action. Artificial Intelligence Center Technical Note 191, SRI International, Menlo Park, California.

[18] Robinson, J. A. (1965). A machine-oriented logic based on the resolution principle. *J. Assoc. Comput. Mach. 12*, pp. 23–41.

[19] Rosenschein, J. S., and Genersereth, M. R. (1984). Communication and cooperation. Heuristic Programming Project Report 84–5, Stanford University, Stanford, California.

[20] Stickel, M. E. (1982). A nonclausal connection-graph resolution theorem-proving program. Proceedings of the AAAI-82 National Conference on Artificial Intelligence, Pittsburgh, Pennsylvania, pp. 229–233.

[21] Stickel, M. E. (1985). Automated deduction by theory resolution. *Proceedings of the Ninth International Joint Conference on Artificial Intelligence,* Los Angeles, California.

[22] Weyhrauch, R. (1980). Prolegomena to a theory of mechanized formal reasoning. *Artificial Intelligence* **13**, no. 1–2.