Reasoning About Knowledge Using Extensional Logics

by

Erann Gat

Thesis submitted to the Faculty of the
Virginia Polyetchnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science

in

Computer Science and Applications

Approved:

David P. Miller, Chairman

John W. Roach

Morton N. Nadler

October, 1987

Blacksburg, Virginia

Reasoning About Knowledge Using Extensional Logics

Erann Gat

Committe Chairman: David Miller
Computer Science

(ABSTRACT)

When representing statements about knowledge in a extensional logic, it occasionally happens that undesired conclusions arise. Such extraneous conclusions are often the result of substitution of equals for equals or existential instantiation within intensional operators such as Know. In the past, efforts at solving this problem have centered on modifications to the logic. In this thesis, I propose a solution that leaves the logic intact and changes the representation of the statements instead.

The solution presented here has four main points: 1) Only propositions can be known. 2) Relations rather than functions should be used to describe objects. 3) Temporal reasoning is often necessary to represent many real-world problems. 4) In cases where more than one label can apply to the same object, an agent's knowledge about labels must be explicitly represented.

When these guidelines are followed, statements about knowledge can be represented in standard first-order predicate logic in such a way that extraneous conclusions cannot be drawn. Standard first-order theorem provers (like Prolog) can then be used to solve problems which involve reasoning about knowledge

Acknowledgements

I would like to thank my advisor, David Miller for his patience and advice. In my random walk to generating this thesis, his guidance often set me back upon the path from which I would otherwise have strayed far and wide.

Thanks too go to the other members of my committee, John Roach, whose knowledge representation class inspired this research, and Morton Nadler, who kept reassuring me that there are still some worthwhile short theses to be written.

I am grateful also to Robert France and Terry Nutter for many interesting and stimulating discussions, and for pointing me in the right direction when I had neglected to do my homework. Forrest Norrod also provided much food for thought, though not necessarily on the topic of my Thesis.

I would like to thank Alison for putting up with me.

Finally, I would like to thank Mike and Mitch Bunnell for writing *Megaroids* and putting it in the public domain. It kept me sane on many long winter nights when I could no longer bear to look at a LISP prompt.

"There is no progress, no revolution of ages, in the history of knowledge, but at most a continuous and sublime recapitulation."

- Umberto Eco, *The Name of the Rose*

Table of Contents

1 Introduction

In order to perform most tasks you need knowledge. To call someone on the phone you must know the number. In order to go to the post office you need to know its location. If you don't know the requisite information, part of performing these tasks is acquiring the necessary knowledge: you can look up phone numbers in the phone book, and the location of the post office can be learned by consulting a map or someone familiar with the town. If AI is to achieve the goal of building an intelligent autonomous robot, then we must give it the ability to reason about its own knowledge and that of other intelligent agents, be they people or other robots.

For the most part, automated planning research has ignored the problems inherent in reasoning about knowledge. Most classic planners such as STRIPS [Fikes71] and NOAH [Sacerdoti77] and their descendents assume that the planning agent is omniscient and has instant access to all information necessary for solving the problem. More recently, Moore's integrated logic of knowledge and action [Moore85] and the FORBIN planner [Dean87] have begun to examine the role of knowledge and acquiring information in automated planning.

The reason that omniscent planners like STRIPS are inadequate is that they plan with their eyes closed, so to speak. They assume that they have perfect knowledge about the world and the results of the actions that they can perform. In the real world these are unjustified assumptions (to say the least). Real-world planning must include actions designed to gather or verify information during execution of the plan. For a planner to be truly robust it must also be able to reason about other intelligent agents who have knowledge, and the actions that they are likely to perform given that knowledge. For instance, a planner working on military strategy would have to take into account what the enemy knew about its forces.

One of the classic problems in reasoning about knowledge is the failure of Leibniz's law and existential instantiation under certain circumstances. This problem has its roots in philosophy in the early days of symbolic logic. It has received a great deal of consideration from the philosophy research community, and almost none from AI researchers. As a result, most of the

proposed solutions to the problem are not well suited to practical implementation on a computer. It is the goal of this thesis to provide at least a partial solution to the problem which is implementable in a classic theorem-proving system.

Unfortunately, as often happens with problems of philosophy, there is considerable disagreement as to the proper characterization of the problem and its scope. For the purposes of this thesis I will refer to it as the "problem of referential transparency", adopting a term which, though not entirely appropriate (as I shall later argue), is often found in the literature.

The nature of the problem of referential transparency is best illustrated by means of some examples.

1.1 Examples of the Problem

The classic example of the failure of Leibniz's law is the following problem: suppose that John knows Mary's phone number, and that Mary's phone number is the same as Bill's phone number. If we represent these statements in standard first-order logic the straighforward way:

Knows (John , Phone-number(Mary))
Phone-number (Mary) = Phone-number (Bill)

we can conclude that John knows Bill's phone number, which might not be true.

There is a host of related classic problems. Consider the statements, "George IV wondered whether Walter Scott was the author of the Waverly novels," and "Sir Walter Scott is the author of the Waverly novels." If we translate the latter statement as:

Author (Waverly) = WalterScott

we come up with the absurd conclusion that George IV wondered whether Walter Scott was Walter Scott.

Yet another example is the morning star problem. Certainly the morning star is equal to the evening star, since they are both the same object, namely the planet Venus. And yet, if John saw the morning star at 6:00 AM, we would not want to say that he also saw the evening star at the same time. Note that this problem does not involve knowledge, and indeed I will argue that the source of the difficulty, and hence the solution, are very different from the other examples. A related problem arises if John knows that the Morning Star is lit by the sun; it should not follow that he knows the same of the Evening Star (unless, of course, he knows that they are the same object.)

1.2 Outline of the Thesis

Chapter 2 presents previous work on the problem of referential transparency, and discusses why none of the proposed solutions are adequate in terms of the needs of AI. Technical and philosophical solutions are discussed.

Chapter 3 presents a series of practical solutions to the problem which do not suffer from the drawbacks of previous solutions. The situations in which each solution is applicable are also discussed.

Chapter 4 describes *Superchain*, a simple theorem prover which was written in order to verify that the proposed solution can actually work with a real system.

Chapter 5 presents some conclusions and commentary.

2 Previous Work

The problem of referential transparency did not originate with AI research, but has its roots back in the early days of formal logic starting in the late eighteenth century. At the time, philosophers were wrestling with the problem of capturing the regularities of human reasoning in a formal system. One of the most influential of these early researchers was Leibniz who developed the definition of equality which is still in common use today. Leibniz's law is a second order formula which states that if two things are equal then they have every property in common:

$$\forall x,y:((x=y) \equiv (\forall \Psi:\Psi x \equiv \Psi y))$$

This definition has a great deal of intuitive appeal, and usually works just as one might expect it to. And yet it would seem that Mary's phone number can be equal to Bill's and still have a property that Bill's does not necessarily have, namely that John can know one without knowing the other.

Past work on the problem of referential transparency has fallen in two major categories. Some researchers have considered the problem to be a purely technical one, and have attempted to find modifications to standard formal logic that will avoid the dificulties, while others consider the problem to be a philosophical one, and maintain that radical changes in the fundamental nature of formal logic must be made.

2.1 Technical Solutions

The most obvious solution to the problem of referential transparency is to simply abandon Leibniz's law of identity and disallow substitution of equals for equals. This is a rather drastic approach; substition of equals for equals is a powerful reasoning tool and should not be discarded lightly. Furthermore, there are cases where substitution of equals for equals is necessary in order to reach certain valid conclusions. Suppose John dialed Mary's phone number. We would want to be able to conclude that he also dialed Bill's phone number, even though he may not know that he dialed it.

A better solution is to note that substitution of equals for equals fails only inside certain operators (like Knows) and not others (like Dials). Thus we can distinguish *referentially opaque* operators, where substitution fails, from *referentially transparent* ones where it does not, and disallow substitution only within opaque operators [Quine61].

This solution, aside from being inelegant and difficult to mechanize, occasionally leads to erroneous conclusions. Suppose John knew Mary's mother, and that Mary's mother was Bill's sister. We would want to be able to conclude that John knew Bill's sister, though he may not be aware that he knows her.

Another technical solution relies on distinguishing between "knowing what" and "knowing that". The difference is best ilustrated by an example:

John knows Mary's phone number.
John knows that Mary's phone number is 555-1212.

Not only are there obvious semantic differences between the two statements, but there is also a syntactic difference: in the first statement, John knows an object, while in the second he knows a proposition. Such a dichotomy can cause severe problems in a formal system of deduction. It is argued then that many of these problems (including referential transparency) can be solved by allowing Knows to operate only on propositions and not objects [Hintikka62, Moore85]. Thus, "John knows Mary's phone number," becomes, "John knows that Mary's phone number is x, for some x," i.e.:

$\exists x$: Knows (John, Phone-number(Mary)=x)

In this case, Knows can be considered either a modal operator whose argument is a proposition or a syntactic predicate whose argument is the name (i.e. the Gödel number) of a proposition. If one takes the former position as Moore does then Knows must again be taken as an opaque operator with substitution expressly forbidden within its scope. If the latter position is taken, then some provision must be made for converting names of

propositions into the propositions themselves and vice versa. This is to allow deductions such as Moore's axiom M2:

$$Knows(A,P) \supset P$$

In practice, such conversion does not present a serious difficulty. Most automated deduction systems work directly on internal representations of propositions in ASCII or some other alphanumeric code which makes a perfectly legitimate Gödel numbering scheme.

However, this approach has other problems. For example, suppose that John knows that Mary and Bill have the same phone number, but doesn't know the actual number:

$$Knows\ (John,\ Phone\text{-}number(Mary) = Phone\text{-}number(Bill))$$

From this we could conclude that John knows Mary's phone number according to the formulation above by the rule of existential generalization. Disallowing "quantifying-in" (the application of existential generalization within operators) [Kaplan69] doesn't work because then if John knows that Mary's phone number is 555-1212 we cannot conclude that John knows Mary's phone number.

The usual fix for this is to allow quantifying-in for rigid designators like Mary and 555-1212, but not for functional designators like Phone-number(Mary) [Moore85]. This solution is awkward, and has its own problems. Suppose John knows that Mary's phone number is prime. From this it is reasonable to conclude that John knows that some number is prime, but we cannot do so under the restriction on quantifying-in.

2.2 Philosophical Solutions

The philosophical solutions to the problem are based on the idea that the phone number that you know is a different sort of thing than the phone number you dial, the latter being a true number while the former is the concept of a number [Carnap47, Frege1892, Kripke80, McCarthy69,

McCarthy79]. This solution comes under many labels. Some distinguish between concept (or idea) and object, others sense and denotation, still others extension and intension. The distinction between these concepts is fuzzy and difficult to describe clearly. As an example, consider the following statements:

Pegasus is a horse.
Pegasus is a fictional character.

The first sentence speaks of Pegasus as if he were a material object, sharing many of the properties common to material objects, e.g., he has mass, he occupies space, etc. The second sentence speaks of Pegasus as if he were a concept, a collection of words on a page or neural impulses in someone's mind. The first sentence speaks of an object, an extension, while the second speaks of a concept or an intension.

Regardless of the terminology, the upshot is the same. If John knows Mary's phone number, what he really knows (it is maintained) is the concept of Mary's phone number, and not the phone number itself:

Knows (John, Concept (Phone-number (Mary)))

The problem of referential opacity is then solved by excercising care not to set extensions and intensions equal to one another. The intension of Mary's phone number is not equal to the intension of Bill's phone number, even though their extensions (the actual numbers) are equal.

This solution works, but it leads to an exceedingly messy system, with propositions awash with predicates to switch back and forth between objects and concepts and concepts of concepts. For example, McCarthy translates, "John knows that Mary's phone number is 555-1212," as:

true K(John, Equal (Telephone Mary, Concept1 "555-1212"))

"where K(P,Q) is the proposition that denot(P) [sic] knows the proposition Q and Concept1("555-1212") is some standard concept of that telephone number," [McCarthy79]. As if that weren't enough, he makes further distinctions

between john and "John" and John, being the concept of John, John's name, and John himself. Even McCarthy seems unsure of exactly what it all means when he writes, "The reader may be nervous about what is meant by concept. He will have to remain nervous; no final commitment will be made . . ." [McCarthy79].

Nevertheless, there are some philosophical grounds for believing that this is the best solution to the problem. The idea that some distinction must be made between concepts and objects is supported by the following statements:

The President of the United States is 76 years old.
The President of the United States is elected by the people.

The first sentence is talking about a particular individual, while the second is talking about the office of the president in the abstract. Thus, the phrase "The President" in the first sentence is an extension, while in the second it is an intension. It is maintained by many researchers that no system which deals only with objects or only with concepts can ever adequately represent the meanings of both sentences.

There are cases where this distinction becomes quite fuzzy. Consider the following statement:

The tallest person in the room wins a prize.

Is the phrase "The tallest person in the room" an extension or an intension? It depends on the state of affairs in the world. If the context of the statement is such that it is talking about a particular room with a particular group of people in it, then the phrase refers to a particular person, namely the tallest one in the room. On the other hand, if the statement is uttered in a context where there is not a particular group of people referred to, such as in planning for an event which will not occur until later, then the phrase is an intension. Thus the same phrase in the same sentence can be both an extension and an intension. This is clearly problematic for automated theorem provers.

While it may be true that philosophically it is necessary to distinguish between objects and concepts, as a practical matter there is to date no mechanized formal system of logic that can deal with both at once, though there are many mechanizable extensional logics [Morgan76]. Automatic theorem provers for many extensional logics have been successfully implemented. Therefore, it would be highly desirable if we could translate statements about knowledge into a standard extensional logic in such a way that we could draw all the desired conclusions and none of the undesired ones.

3 A Practical Solution

If we lay aside the abstruse philosophical issues of exactly what extensions and intensions are, and what it means to denote something, it is possible to formulate most of the examples given in the first chapter in a standard extensional logic in such a way that the undesired conclusions that result from substitution or quantifying-in cannot be drawn.

3.1 Functions: The Root of all Evil

I will argue that the source of the problem is not in misapplication of Leibniz's law or quantifying-in, but rather in the indiscriminate use of the equals sign to represent the word "is" and in the use of functions to represent objects.

It seems perfectly natural to represent phrases such as "Mary's phone number" as the "phone number of Mary" or "Phone-number(Mary)". And yet, there are problems with this representation. For one thing, this representation restricts Mary to having no more than one phone (unless we start adding such encumberances as Extention-phone-number(Mary) and Office-phone-number(Mary), or we use set-valued functions). Furthermore, this representation becomes quite awkward for sentences such as the following:

John knows the value of pi.
John knows the sum of 2 and 4.

These sentences might be represented as:

Knows (John, Value(π))
Knows (John, Sum (2,4))

or

Knows (John, 3.14159...)

Knows(John, 2+4)

Similar absurdities can arise in the case of Mary's phone number. Suppose John knows Mary's phone number, and Mary's phone number is 555-1212:

Knows (John, Phone-number (Mary))
Phone-number(Mary) = 555-1212

From this we could conclude

Knows (John, 555-1212)

which is quite meaningless.

The underlying reason for all these problem is that in an extensional logic, the interpretation of a function and its value are exactly identical according to the semantics of the logic. Thus,

Knows (John, Phone-number(Mary))

and

Knows (John, 555-1212)

are equivalent statements (assuming that Mary's phone number is in fact 555-1212); their interpretations as defined by the semantics of the logic are the same. Therefore, as the latter formulation is clearly inadequate to represent the meaning of the sentence, so must the former be as well.

3.2 Using Relations Instead of Functions

In order to motivate the solution that I will propose, let us consider John's friend Bob who, unlike John, does not know Mary's phone number. Yet, despite the fact that he does not know it, he is nonetheless aware that Mary has

a phone, and that there is such a thing as Mary's phone number. He is similarly aware that 555-1212 is a phone number, i.e. if John were to tell him that Mary's phone sumber is 555-1212 we would not expect Bob to respond by saying, "Golly, I never in my wildest dreams imagined that a phone number could look like that!"*

So Bob does not know Mary's phone number despite the fact that he has the concept of her phone number and the number itself firmly entrenched in the engrams of his mind. What Bob does not have is a *mental connection* between his internal concept of Mary's phone number and his internal concept of 555-1212. So in order to express John's knowledge of Mary's phone number we need to say that John has a mental connection between Mary's phone number and 555-1212. This sort of connection, despite its flowery name, is quite easy to express in first-order logic in term of a relational predicate:

Knows (John, Phone-of-Mary(555-1212))

or, a bit more elegantly:

Knows (John, Phone-of (Mary, 555-1212))

Now, this statement says that John knows that Mary's phone number is 555-1212. In order to represent simply, "John knows Mary's phone number," we just existentially quantify over the number to obtain:

$\exists x$: Knows (John, Phone-of (Mary, x))

This says that John knows that Mary's phone number is x, for some x, which is not an unreasonable rendition of, "John knows Mary's phone number." Note

---

* For the purposes of this discussion I am ignoring the fact that 555-1212 is in fact the number for directory assistance and thus it would actually be quite remarkable if it were Mary's phone number.

that we have now implicitly adopted the convention of allowing only propositions to be known.

When we want to say that Mary's phone number is the same as Bill's we have several options. Since we no longer restrict Mary to having a single phone we can write any of:

$$\exists x: \text{Phone-of (Mary,x)} \wedge \text{Phone-of (Bill,x)}$$
$$\forall x: \text{Phone-of (Mary,x)} \supset \text{Phone-of (Bill,x)}$$
$$\forall x: \text{Phone-of (Mary,x)} \equiv \text{Phone-of (Bill,x)}$$

or several other variations with subtle differences in meaning. The first formula states that Mary and Bill have at least one phone in common. The second says that any phone that Mary has is also Bill's phone, without commenting on whether either of them has any phones at all. The third states that any phone that either Mary or Bill has also belongs to the other one. For the purposes of this discussion I will use the third formulation.

We can now quantify-in and substitute equals for equals to our hearts' content, but we cannot conclude that John knows Bill's phone number unless we add the premise that John knows that Bill's and Mary's phone number are the same. That, together with an appropriate axiomatization of the operator Know such as in [Moore85], will allow us conclude that John knows Bill's phone number, as we should be able to.

Now suppose that John dials Mary's phone number, but does not know that it is Bill's phone number. We should still be able to conclude that he dialed Bill's phone number, which we can if we formulate John's dialing as:

$$\exists x: \text{Dialed (John, x)} \wedge \text{Phone-of(Mary, x)}$$

We can then easily conclude:

$$\exists x: \text{Dialed (John, x)} \wedge \text{Phone-of(Bill, x)}$$

These statements translate as, "John dialed a number, and that number has the property of being Mary's (or Bill's) phone number," which is a reasonable rendition of, "John dialed Mary's (Bill's) phone number."

In general, to represent a statement of the form, "A knows B's C," (where "B's C" represents phrases such as, "Mary's phone number", "Bill's weight" or "Waverly's author") we write:

$$\exists x: \text{Knows (A, B-of (C, x))}$$

To represent statements of the form, "A VERB B's C" where "verb" is an extensional verb such as "dialed" or "wrote", we write:

$$\exists x: \text{VERB (A, x)} \wedge \text{B-of (C, x)}$$

Notice that these forms do not cover cases such as the morning-star problem where there are no functions to convert into relations. I will discuss how to deal with such cases in the following sections. We have, however, now solved the Walter Scott example. The formulation becomes:

Wondered (George, Author-of (Waverly, Scott))
Author-of (Waverly, Scott)

We could also add a statement that Scott was the only author of Waverly:

$$\forall x: \text{Author-of (Waverly, x)} \supset (x = \text{Scott})$$

This example illustrates one appropriate use of the equals sign, namely as a device for asserting the uniqueness of an object with certain properties.

This formulation is similar to one proposed by Bertrand Russell [Russell56], though his presentation is not as concise.* Russell's solution seems to have been largely overlooked by the AI community, possibly because it was presented as part of a philosophical paper concerning the nature of denotation. In any case, the solution presented here and in subsequent chapters is more comprehensive, and more suitable for application to practical systems.

3.2.1 Analysis

At this point the question naturally arises of why this solution works, and what the extent of its effectiveness is. This is a very tricky question to answer. In effect what we are asking is: to what extent does this solution produce logical conclusions that are consistent with intutition? Since we cannot define intuition precisely, it is impossible to answer the question precisely. Nonetheless, we can provide some intuitive arguments that this solution will not lead us into unanticipated trouble.

Consider the following two formulations of, "John knows Mary's phone number."

$\exists x$: Knows (John, Phone (Mary) = x)
$\exists x$: Knows (John, Phone-of (Mary, x))

The similarity between these two statements is more that just superficial. The formal interpretations of these statements are almost exactly the same. Both the Phone function and the diadic Phone-of predicate are defined in terms of a set of ordered pairs. Both of the propositions inside the Knows operators simply state the fact that the ordered pair (Mary, x) belongs to this set. Why then does the first formulation lead to trouble while the second does

---

* For example, Russel's formulation of, "The father of Charles II was executed," is, "It is not always false of x that x begat Charles II and that x was executed and that 'if y begat Charles II, y is identical with x' is always true of y."

not?

The salient difference between the two formulations is that in the first one, the function expression, "Phone (Mary)", is a syntactic object in its own right. The problems arise when this syntactic object is manipulated by substitution or by existential generalization. In the second formulation there is no syntactic object which denotes Mary's phone number, and thus we cannot get into trouble by manipulating it. There is only an existentially quantified variable which has the property of being Mary's phone number.

Now the formal similarity of the two formulations comes to our aid. Because the formal interpretations are nearly identical (the only difference being that functions are usually restricted to being single-valued) we can still draw all the conclusions we could before, except those that involve manipulating the syntactic object, "Phone (Mary)." Furthermore, any assertion that we made previously about, "Phone (Mary)," we can now make about the variable x. Thus, anything that could be expressed using the old system can also be expressed using the new.

Finally, the sorts of conclusions that we used to draw from substituting functional expressions, we now draw using modus ponens as in the examples given above. Because the formulation for statements like, "A knows B's C" is now syntactically different from the formulation for, "A knows that B's C is the same as D's E", we are assured that we can never conclude the former from the latter by quantifying-in.

3.3 Knowing People

If we restrict ourselves as we have to knowing only propositions, how do we represent, "John knows Mary."? It is an unfortunate artifact of English that the word "know" is used for both knowing facts and knowing people, even though these are two very different things. Many languages have different words for the two meanings: *saber* and *conocer* in Spanish, or *wissen* and *kennen* in German. But there is no reason to carry the overuse of the word "know" into our formal representations. To represent "John knows Mary" we simply use a different operator and write:

Conoce (John, Mary)

I have appealed to Spanish for a word to label my personal-know operator. If John knows Mary's mother, and Mary's mother is Bill's sister, we write:

$$\exists x: \textit{Conoce}\ (\text{John}, x) \wedge \text{Mother}\ (x, \text{Mary})$$
$$\forall x: \text{Mother}\ (x, \text{mary}) \supset \text{Sister}\ (x, \text{Bill})$$

and we can conclude that John knows Bill's sister (in the *conocer* sense) even though he may not know (in the *saber* sense) that he knows her.

The *conocer* operator is applicable in all cases where the word "know" means "to be familiar with," as in, for example, "John knows *Romeo and Juliet.*" In cases such as knowing Mary's phone number, where the sentence intends to convey that a particular fact or aspect of an object is known, the statement can always be converted into one which states that a proposition is known.

3.4 Referential Transparency and Temporal Reasoning

The morning star example is a bit trickier. We cannot apply the techniques used above because there is no function that we can convert into a relation. (There is also no "Know" operator.) We could say that John saw x such that x has the property of being the morning star

$$\exists x: \text{Saw(John, x)} \wedge \text{morning-star(x)}$$

but this seems a bit specious. If we carry this to an extreme we get the following:

$$\exists x,y: \text{Saw(x,y)} \wedge \text{John(x)} \wedge \text{morning-star(y)}$$

which, aside from being unnecessarily awkward, doesn't solve the problem of the unwanted conclusion that John saw the Evening-star.

The trouble in this case is that the problem requires reasoning about time. The example refers to an object which has different labels attached to it depending on the time of day. It is true that the morning star and the evening star are the same object, namely Venus, but each of the two labels only applies at certain points in time. Venus is the morning star, but only in the morning. Similarly, Venus is only the evening star in the evening. If we ignore time, as standard first-order logic does, then we cannot represent, "John saw the morning star at 6:00 AM," because we cannot represent the time at which the observation took place.

Sometimes the reliance on temporal reasoning can be hidden, as in the following problem:

> **Holmes knows that Mr. Hyde is a murderer.**
> Dr. Jekyll is Mr. Hyde,

from which we do not want to conclude that Holmes knows that Dr. Jekyll is a murderer. Superficially, this example involves no temporal reasoning; all the verbs are present tense. And yet, the manner in which Dr. Jekyll "is" Mr. Hyde requires temporal reasoning: Dr. Jekyll becomes Mr. Hyde in a process which extends over time. As in the case of Venus, Jekyll and Hyde are two labels which apply to the (presumably) same object, but each is only appropriate at certain times depending on the situation.

The traditional philosphical dichotomy between intensions and extensions is intimately related to temporal reasoning. McDermott [McDermott81] suggests that intensional objects can be represented as objects changing in time within the context of a robust temporal logic. These dynamic objects, which McDermott calls *thingi* (or singular *thingum*), can undergo transitions in their properties as time passes.

This formalism allows us to formulate the Morning Star example in such a way that the problem of John seeing the Evening Star at 6:00 AM does not arise. We model the Morning Star as a thingum which we will call Venus. Now we set Venus equal to the Morning Star in the morning, and the Evening Star in the evening and the problem is solved. McDermott's logic provides the

machinery for reasoning about such formulations, the details of which are rather involved and quite beside the point.

Unfortunately, temporal reasoning, though it is a very powerful (even necessary) tool for solving certain sorts of problems, is of limited use when reasoning about knowledge. For example, we might model Dr. Jekyll and Mr. Hyde as a thingum (lets call him Person37 for no particular reason) which periodically switches back and forth between two states, S_{jekyll} and S_{hyde}. During the transition, Person37 undergoes various changes, notably in his physical form and the label by which he is referred. (Such a change is called a vtrans.) However, Holmes' knowledge about Person37 does not change during these transitions, and so we are back to square one. A solution to this problem (which also works for the Morning Star example) is presented in section 3.5.

3.4.1 Ignoring Time

If time is not taken into account, the distinction between intensions and extensions disappears. Consider the example given in Chapter 2 about the president, repeated below:

> The President of the United States is 76 years old.
> The President of the United States is elected by the people.

The second sentence refers to the president as an intension because it talks about all presidents at all times. The first sentence talks about the current president. If we restrict ourselves to one instant in time the two notions become equivalent.

Of course, it may well be that a philosophical distinction still remains, but from the standpoint of AI, all that matters is that the computer draw all and only the correct conclusions from the facts that it has been given. So let us consider some possible conclusions that one might draw from the above statements in the case where time is taken into account and the case where it is not.

Suppose we are told that Abraham Lincoln was once the President of the United States. From this, and the above statements, we should conclude that

Abraham Lincoln was elected by the people, but not that he is 76 years old. On the other hand, if we do not consider time, then we cannot be told that anyone *was once* the president; we can only be told who is the president in the static world we are considering, and then it is quite correct to conclude that that individual is both elected by the people and 76 years old (at least, to the extent that it is meangful to say that someone is 76 years old in a static world.)

If we ignore time then, many (but not all) of our problems go away. Unfortunately, so does our ability to express many important facts about the world, and so ignoring time is not the way to go about solving the problem of referential transparency.

3.5 Referential Transparency and Labels

Alas, there are problems, as we have seen, where temporal reasoning (or even the lack of it if we ignore time) won't help us. For a fresh example, suppose that Bill occasionally refers to Mary by her middle name, Jane, but John does not know this. As usual, John knows Mary's phone number. Does he also know Jane's? Does he know (in the *conocer* sense) Jane if he knows Mary? Surely John knows the person referred to by the label, "Jane." And yet, if you were to ask John, "Do you know Jane?" he would answer that he did not. However, if you then told him that Jane is just another name for Mary, he would then maintain, "Oh yes, I know her!" Perhaps we should abandon Moore's axiom M3 which states that if an agent knows a fact, he always knows that he knows it (and he knows that he knows that he knows it, and so on).

To solve this problem we must go back to some of the fundamental assumptions about first-order logic. First-order logic models the universe as being made up of objects and sets of objects with certain properties. The objects are referred to by labels. It is possible to know a fact about an object without knowing the label by which it is referred; the princess in the fairy tale knew things about Rumplestiltskin for a long time before she learned his label. So John knows Mary and her phone number, but he does not know that Mary's name is Jane:

$$\sim \text{Knows (John, name-of (Mary, "Jane"))}$$

When you ask John whether he knows Jane, what you are really inquiring about is the truth or falsehood of the following:

$$\exists x: \text{Knows (John, name-of (x,"Jane"))} \wedge \textit{Conoce (John, x)}$$

in other words, "Does John know (in the *conocer* sense) a person whose name he knows is Jane?" Thus John can know Jane, and still correctly answer, "No," when asked if he knows her. Natural language is slippery stuff.

Such logical acrobatics are necessary when there is no *a priori* standard for labelling objects which all agents are aware of. It seems obvious to the person typing the data base that Mary's name is "Mary", but from the point of view of a theorem prover, the designator Mary may as well be Person653 or Foobar. This is not a problem as long as we assume that all agents agree on what labels to attach to various objects. But as soon as we allow agents to call the same object by two different names, we must explicitly express knowledge about labels.

This solution can be applied in a straightforward fashion to the Jekyll/Hyde and Morning Star problems (with a little help from temporal logic). In the former case, Holmes knows that the person whose label is Mr. Hyde is a murderer, but he is not aware that this person occasionally vtranses into Dr. Jekyll (and vice versa):

Knows (Holmes, Murderer (Person37))
Knows (Holmes, Label-of (Person37, "Hyde"))
~Knows (Holmes, Label-of (Person37, "Jekyll"))

The referential transparency problem does not arise because Dr. Jekyll is not equal to Mr. Hyde any more.

In the Morning Star example, we can use the situation calculus (which is much simpler than McDermott's system, and also more limited) to model the temporal aspects of the problem and say that John saw Venus in situation S, and that he is aware that Venus' label (at the time) was the Morning Star.

True (Saw (John, Venus) , S)
Knows (John, True (Label-of (Venus, "Morning-star"), S))

Here again, the Morning Star is not equal to the Evening Star; they are not even objects in the representation, but merely labels for the object Venus.

Explicitly representing an agent's knowledge of labels solves many (if not all) problems of referential transparency which arise as the result of there being more than one label for a single object. In cases where certain labels are applicable only in certain situations, this scheme must be augmented with a suitable system of temporal reasoning.

4 Implementation

In order to verify the contention that the representation described in Chapter 3 can in fact be easily applied in a standard theorem prover, a small forward chainer was augmented with the ability to reason with a subset of Moore's knowledge axioms. The resulting system, called *Superchain*, was able to solve a large variety of problems which involve reasoning about knowledge.

To get a practical theorem prover to reason about knowledge, several problems had to be overcome. The most serious of these is the fact that some of Moore's axioms (notably M3 and M5) interact with each other to produce an infinite number of more or less useless theorems [Moore80]. I chose the obvious solution and eliminated M3, which is of questionable practical value anyway. Whereas Moore's system is equivalent to the modal logic S4, the system which results when M3 is removed is equivalent to the modal logic T [Hughes68].

I also did not implement M1, which is simply the axioms of standard propositional logic. All this paring away may seem like throwing out the baby with the bath water, but in fact the situation is not quite so serious. M2 and M4 are really the heart of Moore's deduction system, and these were both fully implemented. M5, which simply states that all axioms are known by all agents, was applied only to M2 and M4. M5 was implemeted implicitly within the deduction procedure, and explicit knowledge of axioms was not actually entered into the data base.

The Knows operator was implemented as a relational predicate which operated on Gödel numbers of propositions. All of the conversion between Gödel numbers and actual propositions was done implictly; since propositions were actually stored as lists of symbols which makes a legitimate Gödel numbering scheme, conversion was trivial.

M2 was implemented by simply modifying the forward chainer to add P to the data base whenever it adds an assertion of the form Know (A, P). This process occurs recursively to extract all relevant facts from an assertion with nested levels of knowing. A brief reflection will reveal that this process is guaranteed to terminate.

M4 was implemented by collecting all of the assertions known by a given agent, assembling them into a separate data base, and feeding that to the forward chainer. For each assertion P in the resulting set of conclusions, Know (A, P) was added to the master data base.

Another problem that had to be overcome was the scoping of existential operators. Because the forward chainer worked with skolemized expressions, an assertion such as "John knows Mary's phone number," looked like:

(Knows John (Phone-of Mary a))

The problem is that this can be the skolemized version of two different formulas:

$\exists x$: Knows (John, Phone-of (Mary, x))
Knows (John, $\exists x$: Phone-of (Mary, x))

The difficulty is particularly apparent in cases where there are nested knows operators. For example, suppose that Jim does not know Mary's phone number, but he knows that John knows it. The skolemized version of this is:

(Knows Jim (Knows John (Phone-of Mary a)))

which is of the form Know (A, Know (B, P)), which implies Know (A, P) (by M2, M5 and M3), i.e. Jim knows Mary's phone number. The problem, of course, is that by skolemizing, we have lost the existential quantifier in front of the nested Know operator which would keep such unwanted conclusions from bubbling back into Jim's knowledge.

The solution to this problem was rather *ad hoc*. A Knows operator with an existential quantifier in front was renamed "Noes", and a mechanism was added to intercept any Noes operator added to the master data base and change it to to a Knows operator. Now, to say that Jim knows that John knows Mary's phone number, we write:

(Knows Jim (Noes John (Phone-of Mary a)))

Now, when M2 is applied, the assertion (Noes John (Phone-of Mary a)) is inserted into the master data base, which triggers the deduction of (Knows John (Phone-of Mary a)) and all which that assertion entails. However, Jim's knowledge base contains only the first assertion, to which Jim cannot apply M2. Thus, the invalid conclusion that Jim knows Mary's phone number is blocked.

The resulting system was remarkably simple. It consisted of about thirty lines of MacScheme code in addition to a textbook forward chainer which was about 100 lines long. It ran on a Macintosh Plus and took about three to five seconds to solve problems of moderate size, for example:

> Jim knows that John knows Mary's phone number.
> Sam knows that John knows that Mary's phone number is the same as Bill's.

From these statements, *Superchain* was able to deduce that John knows Bill's phone number.

5 Conclusions

5.1 Summary of the Thesis

Many of the difficulties involved in reasoning about knowledge can be avoided by representing statements about knowledge in certain ways. When statements about knowledge are properly represented, reasoning about knowledge can be done by a standard theorem prover working with standard first-order logic. It is not necessary to distinguish between transparent and opaque operators, nor is it necessary to make explicit checks for rigid designators. If the "know" operator is treated as a syntactic predicate, it is not even necessary to use a modal logic; almost all of the examples presented here could be implemented directly in standard Prolog.

In most cases not involving time and where objects have standard labels, many of the classic problems of referential opacity and quantifying-in can be avoided simply by using relations rather than functions to represent objects. However, many problems require temporal reasoning, even in some cases where time is not explicitly mentioned in the problem. In such cases (which include the vast majority of real-world problems) the logic must be augmented with some apparatus to do temporal reasoning. In some cases, a simple system like the situation calculus is sufficient, while in others a more sophisticated system like McDermott's temporal logic must be used.

In cases where there are no standard labels for objects, facts about an agent's knowledge of labels must be represented explicitly. In these cases, queries about an agent's knowledge must often be couched in different terms than a straightforward translation of the natural language query. An agent may know P, but may answer "no" to the query, "Do you know P?" because P may contain labels other than those he is familiar with. In these cases the query, "Do you know P?" must be stated (roughly) as, "Do you know P, and do you know all the labels referred to in P?"

5.2 Future Work

The implementation presented in chapter 4 is incomplete. I beleive that it is complete enough to solve a large and interesting class of practical problems, but exactly what this class consists of remains to be seen. It would be of interest to attempt to categorize a larger number of examples to find out exactly what the practical limitations of the system are. For example, is it really useful to add Moore's M3 to the system and be able conclude that John knows that he knows that he knows that he knows Mary's phone number? What is the best way to keep axioms such as M3 from trailing off into infinite recursion while still keeping the system complete (assuming the arguable premise that completeness a desirable thing given the inevitable computational cost [Joslin86] [Chapman87])?

There is also a considerable amount of work to be done in the representation of and reasoning about such things as time, processes, and action, and integrating whatever solutions one finds to those problems back into a logic of knowledge. This Thesis has left that problem virtually untouched. First-order logic has severe problems in representing time and processes because it assumes that the world is made up of objects that remain intact though their properties may change. Socrates may be alive or dead, but he is always Socrates, even after the atoms that once made up his body are scattered to the winds. See [Hayes85] for some current work on the representation of processes (and the rest of the real world).

References

[Carnap47] Rudolph Carnap, *Meaning and Necessity*, Chicago: Univeristy of Chicago Press, 1947.

[Chapman87] David Chapman, "Planning for Conjunctive Goals", *Artificial Intelligence*, vol. 32, no. 3, pp. 333-378, Amsterdam: North Holland, July 1987.

[Dean87] Thomas Dean, James Firby, and David Miller, The Forbin Paper, Yale Technical Report YALEU/CSD/RR#550, July 1987.

[Fikes71] R. Fikes and Nills Nillson, "STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving," in *Artificial Intelligence 2*, pp. 189-208, 1971.

[Frege1892] Gottlob Frege, "On Sense and Nominatum" in *Readings in Philosophical Analysis* (eds. H. Feigle and W. Sellars), New York: Appleton-Century-Crofts, 1949.

[Gat87] Erann Gat, *The Trouble with Mary's Phone Number*, unpublished.

[Hayes85] Patrick Hayes, "The Second Naive Physics Manifesto," in *Formal Theories of the Commonsense World* (eds. J.R. Hobbs and R.C. Moore), pp. 1-36, Norwood, NJ: Ablex Publishing, 1985.

[Hintikka62] Jaakko Hintikka, *Knowlege and Belief*, Ithaca NY: Cornell University Press, 1962.

[Hughes68] G. E. Hughes and M. J. Cresswell, *An Introduction to Modal Logic*, London: Methuen, 1968.

[Joslin86] David Joslin, An Analysis of Conjunctive Goal Planning, Master's Thesis, Department of Computer Science, VIrginia Tech, October 1986.

[Kaplan69] David Kaplan, "Quantifying In", in *Words and Objections: Essays on the Work of W.V. Quine* (eds. D. Davidson et al.), pp. 206-242, New York: Humanities Press, 1969.

[Kripke80] Saul Kripke, *Naming and Necessity*, Cambridge Mass.: Harvard University Press, 1980.

[McCarthy69] John McCarthy and Patrick Hayes, "Some Philosophical Problems from the Standpoint of Artificial Intelligence", in *Machine Intelligence 4* (eds. B. Meltzer and D. Michie), pp. 463-502, Edinburgh: Edinburgh University Press, 1969.

[McCarthy79] John McCarthy, "First Order Theories of Individual Concepts and Propositions", in *Machine Intelligence 9* (eds. J. Hayes, D. Michie and L. Mikulich), pp. 129-147, Chichester England: Ellis Horwood, 1979.

[McDermott81] Drew McDermott, "A Temporal Logic for Reasoning About Processes and Plans," Research Report #196, Yale University Department of Computer Science, March 1981.

[Moore80] Robert Moore, "Reasoning About Knowledge and Action," SRI technical note #191, Menlo Park, CA: SRI International, 1980.

[Moore85] Robert Moore, "A Formal Theory of Knowlege and Action", in *Formal Theories of the Commonsense World* (eds J. Hobbs and R. Moore), pp. 319-358, Norwood NJ: Ablex Publishing, 1985.

[Morgan76] Charles Morgan, "Methods for Automated Theorem Proving in Nonclassical Logics", *IEEE Transactions on Computers*, vol. C-25, no. 8, August 1976.

[Quine61] Willard Quine, *From a Logical Point of View*, London: Oxford University Press, 1961.

[Russell56] Bertrand Russel, "On Denoting", in *Logic and Knowlege* (ed. Robert Marsh), pp. 39-56, London: George Allen and Unwin, 1956.

[Sacerdoti77] Earl Sacerdoti, *A Structure for Plans and Behavior*, American Elsevier, 1977.

[Vilain86] Marc Vilain and Henry Kautz, "Constraint Propogation Algorithms for Temporal Reasoning", *Proceedings of AAAI-86*.

The two page vita has been removed from the scanned document. Page 1 of 2

The two page vita has been removed from the scanned document. Page 2 of 2