Epistemic Logic and its Applications: Tutorial Notes

Ernest Davis*
New York University
New York, New York

Leora Morgenstern
IBM Watson Research Center
Hawthorn, New York

1 Introduction

It is to the advantage of a thinking creature to be aware of knowledge and thought and to be able to reason about them. If the creature has to interact with other thinking creatures, then reasoning about their mental processes is often necessary to understand and predict their behavior. Even if the creature is alone, stranded on a desert island, the ability to reason about its own mental life will be valuable. A Robinson Crusoe robot will need to make plans involving the gaining and maintaining of knowledge, such as, "If I want to learn my way around the island, I should go to the top of the big hill," or "If I want to avoid being eaten, I should keep a close watch."

The distribution of information among autonomous agents, the transferral of information between agents, and the gain and loss of information by agents over time are critical characteristics of many environments. This is particularly true of environments where computers are applied, since the control and distribution of information is a primary function of computer systems and a central issue in computer science. It can be very valuable in analyzing any such environment to be able to represent explicitly and to reason about the state of information and the dynamics of information. Formalisms that support such representation and such reasoning are called *logics of knowledge* or *epistemic logics.*

The applications for epistemic logics in artificial intelligence, in computer science generally, and in other fields are numerous:

AI Applications

Planning

In many cases, an agent is initially ignorant of information that he needs to achieve a given goal, but he knows how to acquire the information. In that case, he may plan to acquire the necessary information and then to use that information in further steps of planning. For example, if Debby wants to read *Moby Dick* but does not know where her copy is on her bookshelf, then she can form the plan, "Look through the book case; take the book from its place; read it." Here the purpose of the first step of looking through the shelf is to determine the place of the book, a datum needed for the second step of the plan, grasping the book.

In order to construct or analyze such plans, a reasoner needs to be able to reason about

*The preparation of these notes was supported in part by NSF grant #IRI-9001447

- What knowledge the agent has and what knowledge he lacks. For example, Debby knows that the book is on the shelf; she does not know where.
- What knowledge it needs to carry out specific actions and specific plans. For example, Debby needs to know where the book is in order to grasp it.
- How actions produce knowledge. For instance, if *Moby Dick* is on the shelf, then after scanning the shelf, Debby will know where *Moby Dick* is.

We will study this kind of planning in section 2.3.

Motivation Analysis

A large part of the interpretation of natural language text and a large part of the understanding necessary for correct social behavior is *motivation analysis*, determining the reason for other agents' perceived behaviors. To carry out motivation analysis it is necessary to understand how an agent's knowledge and beliefs affect how he pursues his goals. For example, if an intelligent system sees or is told that Jack anxiously reads through the Help Wanted ads every morning, it should infer that Jack wishes to know about job openings; further, that Jack probably wishes to apply for a job; further, that Jack is probably either unemployed or profoundly dissatisfied with his current position. If the system hears Sarah ask it for the time, it should infer that Sarah probably does not know the time.

Active Perception

To use high-level reasoning to guide the focus of a sensor, the reasoner must know the powers of the sensor and how knowledge can be gained through its perceptions. For example, a robot who wishes to determine whether a dark patch on a perceived object is a mark or a shadow may decide to look at the point where the object casting the shadow would have to be. To do this, it must deduce that if there is an object casting the shadow, then it will be able to see the object if it looks in the proper direction. Another example: As before, Debby is looking for *Moby Dick* on her bookshelf. She can know that scanning will work if she knows (1) that a particular trajectory of the sensor will bring every book on the shelf into view; (2) that if she scans slowly enough, she will be able to recognize *Moby Dick* when she passes it.

Speech Acts

The right choice of what to say in a given situation often depends on what the person you are talking to knows. For example, if a person points to your dog and says, "What is that?" then the correct answer may be

- "That's a dog" if the questioner is a two-year old.
- "A Yorkshire terrier" if the questioner is an adult.
- "Yes, he's had that limp since Tuesday" if the questioner is your vet.

Pursuing a conversation often requires a large degree of mutual understanding and common knowledge among the conversants. Suppose that, in the course of conversation, Bob asks Karen, "And what did *she* say to *him*?" and Karen answers, "She said she wouldn't trust him as far as she could kick him." After Karen answers, Bob must know that Karen knows who are the people and the circumstances under discussion; otherwise, Bob will worry about some confusion between Karen and himself. Therefore, when she answer thus, Karen must know that Bob will know that Karen will know the subject of the discussion. Therefore, when Bob asks the question, Bob must know that Karen will know that Bob will know that Karen will know the subject of discussion. And so on.

Grice's (1967) well known *Maxim of Quantity*, "Make your contribution as informative as is required but not more informative than is required," depends for its use on an understanding of what information is already held and what information is required; that is, on an understanding of the knowledge and the knowledge goals of the other speaker. The same holds in many cases for the interpretation of indirect speech acts; for distinguishing the new from the given in discourse analysis; and for many other aspects of discourse analysis.

Intelligent CAI

Ideally, an automated tutor would maintain a model of what a student knows, how he reasons, and how he learns. The model could be initialized by combining generic information about what students at his level characteristically and can do with data specific to the individual (e.g. test scores.) The tutor should be able to update and correct the model using its knowledge of what the student has been taught (by itself and others) and by observing the student's actions and responses. It should be able to use the model to plan an effective teaching strategy.

Non-monotonic logic

An important category of defeasible inference is the *autoepistemic* inference, in which an agent infers that ϕ must be false from the fact that he does not know ϕ to be true. For example, I know that Mark Twain was never governor of New York, not because I have ever read specifically that he was not, nor because I can enumerate all the governors, nor because I know everything that Mark Twain ever did, but just because I may safely presume that I would have heard about it if he had been. Characterizing autoepistemic inferences requires a theory that combines a logic of knowledge with a logic of defeasible inference; the form of such a logic and the scope of the autoepistemic inference as a model for defeasible inference generally are areas of current research.

Design of Intelligent Systems

At the meta-level, it may be able be possible to custom design an AI program for a specific application from specifications which describe what kinds of things the program can know; how this knowledge is gathered and how it can be augmented; and how the program should behave under various states of knowledge. For example, Rosenschein and Kaelbling (1986) describe a program which accepts as input a high-level description of a reactive robot, in terms of what the robot learns from its sensors and how it should respond, and generates as output a circuit diagram that implements this behavior.

Meta-level reasoning at run-time about a knowledge base's own knowledge and own reasoning capacities may likewise be important. For example, the controller for an inference engine may use a high-level characterization of the knowledge in a knowledge base to choose to pursue one path of inference over another.

Applications in General CS

Distributed Systems

Epistemic logic is a promising tool for analyzing distributed systems; this application has been very extensively studied. In this approach, the state of a distributed system is characterized using epistemic concepts, such as,

- The information available to each processor (what the processor "knows");
- The information needed by the processor (what it wants to know);
- What each processor knows about what the other processors know;
- What is potentially known when the information of all the processors has been pooled;
- What is common knowledge among the processors.

The dynamics of the distributed system is characterized in terms of the transfer of information among the processors though communication. The validation of a protocol P consists, in general, in proving a result of the form, "If P is executed under circumstances C, then, when P is complete, such and such a state of knowledge is attained in the system."

For example, one problem in distributed systems is to develop a protocol allowing the processors to come to a joint decision on a question and to commit to it. This is often modelled through the parable of the *Byzantine generals.*

> Two Byzantine armies lie on opposite sides of a Saracen army. If the two armies both attack simultaneously, they can defeat the Saracens; if only one attacks, then the Saracens will first defeat the attacking army, then defeat the other army. The Byzantine generals must therefore coordinate their attacks by sending messengers back and forth. Unfortunately, the general sending a messenger cannot ever be sure that he will get through.

It is then possible to prove the following depressing theorem (Halpern and Moses, 84):

> No prearranged communications protocol can guarantee that both generals will attack, even if all the messengers do, in fact, get through.

The proof is essentially as follows: Consider the last message to be sent in a given protocol; let us say that this is A sending message M to B. The protocol must specify what B should do if he receives various messages at this point and what he should do if he fails to receive a message. Moreover, there must be some circumstances under which B will attack and some under which he will fail to attack; otherwise, why send a message? Now, one of the two following possibilities must hold:

1. It is agreed that B will not attack if he does not get a message, but will attack if he gets message M. Now, if A sends M, he cannot be sure whether B will get M and attack, or whether B will not get M and not attack.
2. It is agreed that B will attack if he does not get a message, but will not attack if he gets message N. Now, if A sends N, he cannot be sure whether B will get N and not attack or whether B will fail to get N and attack.

Thus, if A's message makes any difference, A cannot be sure that B will attack.

Security

The central issue in computer security is one of controlling access to knowledge; we wish to guarantee that, unless an agent knows P (the password) he cannot find out Q (the information). In the case of public key encription schemes, where a decryption key D is publically known but the corresponding encription ken E is to be kept private (or vice versa), the issue is to guarantee that E cannot be computed from D without inordinate computational resources.

Zero-knowledge interactive proofs (Goldwasser, Micali, and Rackoff, 85) add another twist. Here one person holding the encryption key E must convince another person holding D that he does, indeed, know E; but he must do so in a way that does not tell the other anything that will help him compute E.

Economics and Game Theory

Epistemic logic has also been applied outside of computer science, particularly to economics and game theory. In economics, it arises through the fact that the value of an investment to you is influenced heavily by how you think other people will gauge its value. In particular, if you can buy an investment (a share of stock, say) at price P, then that is *prima facie* evidence that the seller thinks that it is worth less than P, and, assuming that the seller is not a fool, this should presumably influence one's own thinking. The analysis of such situations thus naturally leads to considering imbedded levels of belief and considering how knowing other peoples' beliefs should affect one's own (Aumann, 76), (Geanakopolos, 90).

Another point where epistemic logic meets economics is in analyzing the economics of information as a commodity of value and of information processing as an economic activity (Allen, 90).

2 Representation and Inference

In this tutorial, we will focus on three questions:

- Representation: How can facts about the knowledge of agents be expressed symbolically?
- Inference: Given a collection of facts about knowledge, what further facts can be reasonably deduced?
- Semantics: What does a sentence in our representation actually mean?

A framework which specifies the form of answers to these three questions for a given domain is called a *logic* of the domain. logic of knowledge and belief is called an *epistemic logic.*

The most familiar logics, both in mathematical logic and in AI, are the extensional logics, such as propositional logic and predicate calculus, which characterize objects and their properties. Such logics allow the expression of statements such as "Joe is a boy," "All men are mortal," "Every person has two parents," and so on.

But the relation "A knows ϕ" is very different in its logical structure than a relation like "X is blue" and epistemic logics must therefore address a number of issues that do not arise in extensional logics:

- The predicate "blue" applies to objects: the ball is blue, the Empire State Building is not blue. It is not meaningful to say that the fact "George Washington was born in 1732" is blue. The operator "know" applies to sentences: One may know, or fail to know, that George Washington was born in 1732; that the Empire State Building is not blue. One cannot know an object (except in the sense of being acquainted with it).

In particular, the operator "know" is self-embedding; one may know facts about knowledge. Mark knows that Leslie know that Star Trek is on at 7:00; Sam knows that he does not know how to spell "protocol". By contrast the fact that "the ball is blue" cannot be said to be blue.

- The predicate "blue" is *referentially transparent*; that is, it applies to an object independently of how that object is named. Given that Lake Superior is the largest fresh-water lake, we can infer that Lake Superior is blue if and only if the largest fresh-water lake is blue. The operator "know" is *referentially opaque*; substituting an equal term inside a knowledge operator may change the truth of a sentence. It is possible that Andrea knows that Lake Superior lies partly in the US, but does not know that the largest fresh-water lake lies partly in the US (for example, if she thinks that the Caspian Sea is fresh-water.)

2.1 Propositional Modal Epistemic Logics

We begin with a very simple logic for characterizing primitive pieces of knowledge of a single agent at a single moment. In this logic, we will have three kinds of symbols.

i. Propositional atoms: A propositional atom is a symbol that denotes a particular fact about the external world. Examples: the atom "Frog-Kermit" denotes the fact that Kermit is a frog. "All-men-are-mortal" denotes the fact that all men are mortal.

ii. Boolean connectives. There are five of these: $\neg$ (not), $\vee$ (or), & (and), $\Rightarrow$ (implies), $\Leftrightarrow$ (if and only if).

iii. Modal operator. The symbol "know".

We also use parentheses to clarify grouping.

A *sentence* in the logic is built up by applying Boolean connectives and the modal operator "know" to propositional atoms. Examples:

Frog-Kermit.
Frog-Kermit $\Rightarrow$ (Green-Kermit & Croaks-Kermit).
know(Frog-Kermit).
know(Frog-Kermit $\Rightarrow$ Green-Kermit).
know(Frog-Kermit) $\vee$ know($\neg$Frog-Kermit).
know($\neg$(know(Frog-Kermit & Green-Kermit)) $\vee$ Frog-Kermit)

Formally, we define a sentence as follows:

Definition 1: In the propositional modal epistemic logic, a sentence is either

- a propositional atom; or
- an expression of the form $\neg\phi$; $\phi \vee \psi$, $\phi \& \psi$, $\phi \Rightarrow \psi$, $\phi \Leftrightarrow \psi$, or know($\phi$), where ϕ and ψ are (recursively) sentences.

This logic is known as the *propositional modal epistemic logic:* "propositional" because the base sentences are propositional atoms; "modal" because it uses the operator "know", which affects the mode of a sentence (changes a sentence which is merely stated to one that is known); and "epistemic" because it is about knowledge. (Other modal operators include propositional attitudes, such as "X believes that ϕ" or "X hopes that ϕ"; temporal operators such as "ϕ is always true" or "ϕ will be true at some future time"; ontic operators, such as "ϕ is necessarily true"; deontic operators, such as "It is obligatory that ϕ should be true"; and others. A modal logic with an appropriate logical structure can be developed for each of these.)

2.1.1 Semantics: Knowledge Base Model

It's all very well to write down sentences about knowledge, but without a specific definition of what they mean, there is no way to tell whether a given set of sentences applies or what it entails. The precise meaning of a logic is specified by a *semantics*, which consists of a idealized model of the world and an account of when a sentence in the logic is true in the model. In this tutorial, we will look at a number of models of knowledge.

The first and simplest model is this. Consider a knowledge base which contains a collection of true propositional sentences with no knowledge operators. We adopt the following definition:

Definition 2: know(ϕ) is true if it is possible to determine that ϕ given the sentences in the knowledge base.

Example: Suppose that the knowledge base $\mathcal{B}$ contains the following three sentences.

Frog-Kermit.
Frog-Kermit) $\Rightarrow$ (Green-Kermit & Croaks-Kermit).
All-men-are-mortal $\Leftrightarrow$ $\neg$(Socrates-was-a-tailor).

Then the following sentences are all true:

- know(Frog-Kermit) — Frog-Kermit is in $\mathcal{B}$.
- know(Croaks-Kermit & Green-Kermit) — "Croaks-Kermit & Green-Kermit" is a logical consequence of the first two sentences of $\mathcal{B}$.
- $\neg$know(All-men-are-mortal) — All-men-are-mortal cannot be derived from $\mathcal{B}$.
- $\neg$know($\neg$All-men-are-mortal) — $\neg$All-men-are-mortal likewise cannot be derived from $\mathcal{B}$.
- know(know(Frog-Kermit)) — We have determined from the knowledge base that know(Frog-Kermit) is true, hence it is known.
- know(Frog-Kermit) $\vee$ know(All-men-are-mortal). — A disjunction is true if either disjunct is true. In this case, as we have seen, the first disjunct is true.
- know($\neg$know(All-men-are-mortal)) — It can be proven that All-men-are-mortal cannot be proven from $\mathcal{B}$.

2.1.2 Axioms for Modal Logic

We can now ask, "What are the important general properties of this definition of knowledge?" The following have been found to be particularly important:

Veridicality: What is known is true. This is a consequence of our specifying that the sentences in the knowledge base were true. Of course, there is no way to tell whether this condition holds just by looking at the knowledge base; one must know what the propositional atoms mean, and whether they hold in reality.

The property of veridicality can be expressed formally in the following axiom:

K.1 $\text{know}(\phi) \Rightarrow \phi$.

K.1 is actually an axiom schema, meaming that if any sentence is substituted for ϕ in K.1, the resulting instance is an axiom. Thus, the following two sentences and all sentences like them are axioms.

$\text{know}(\text{Frog-Kermit}) \Rightarrow \text{Frog-Kermit}$.
$\text{know}(\text{Frog-Kermit} \vee \neg\text{know}(\text{Green-Kermit})) \Rightarrow [\text{Frog-Kermit} \vee \neg\text{know}(\text{Green-Kermit})]$.

Consequential Closure: If the agent knows a collection of sentence Γ and ϕ is a logical consequence of Γ, then the agent also knows ϕ. This property derives from our use of implicit knowledge, defining the agent as "knowing" any consequence of the information in his knowledge base.

In a broader setting, this property clearly does not apply to "knowledge" in the usual sense. It cannot be that an agent's knowledge is closed under logical consequence; if it were, then we would all be perfect reasoners and anyone who knew the axiom of a mathematical field would know all the theorems. Even in the narrower domain of propositional reasoning, the property is unrealistic; determining whether a conclusion is a consequence of a set of axioms in the propositional calculus is co-NP-complete. This property is essentially an idealization of the fact that an agent can be expected to do some level of reasoning with his knowledge. For many simple theories of knowledge it is by far the simplest and most natural idealization.

We can express the property of consequential closure formally as follows. We use the fact that, in many logical theories, inference can be characterized by repeated use of the single rule *modus ponens,* "From α and $\alpha \Rightarrow \beta$ infer β", given a suitable set of logical axioms. In such a theory, saying that an agent's knowledge is closed under logical consequence is equivalent to saying that the agent knows all logical axioms and that the agent's knowledge is closed under *modus ponens.* We thus have the following axiom schemata:

K.2 If ϕ is a logical axiom, then $\text{know}(\phi)$ is an axiom.

K.3 $[\text{know}(\phi) \mathbin{\&} \text{know}(\phi \Rightarrow \psi)] \Rightarrow \text{know}(\psi)$.

Positive Introspection: Suppose that the agent knows ϕ; that is, he has been able to derive ϕ. Then he should be aware that he has derived ϕ; that is, he knows that he knows ϕ. We thus have the following axiom:

K.4 $\text{know}(\phi) \Rightarrow \text{know}(\text{know}(\phi))$.

Negative Introspection: Suppose that the agent does not know ϕ; that is, ϕ is not a consequence of his knowledge base. Then, the agent can in principle figure out that ϕ does not follow from his knowledge base. That is, the agent knows that he does not know ϕ.

K.5 $\neg\text{know}(\phi) \Rightarrow \text{know}(\neg\text{know}(\phi))$

This property relies on two aspects of our model. First, we are using propositional logic as the base of our epistemic logic, and it is always in principle possible to determine whether or not a

K.1 know(ϕ) $\Rightarrow$ ϕ.

K.2 If ϕ is a logical axiom, then know(ϕ) is an axiom.

K.3 [know(ϕ) & know($\phi \Rightarrow \psi$)] $\Rightarrow$ know(ψ).

K.4 know(ϕ) $\Rightarrow$ know(know(ϕ)).

K.5 $\neg$know(ϕ) $\Rightarrow$ know($\neg$know(ϕ))

Table 1: Logical Axioms of the Modal Theory S5

particular conclusion follows from particular premises in propositional logic. In richer logics, such as the predicate calculus, it is not, in general, computationally possible to establish that a conclusion is not a consequence of a set of premises. Thus an agent may be unable either to derive ϕ — so he does not know ϕ — or to establish that he will not be able to derive ϕ — so he does not know that he does not know ϕ.

Second, we posited that the statements in the knowledge base are all true. In a more general setting, one has to admit the possibility that some statements in the knowledge base are false. In such a case, one can still achieve the property of veridicality, by changing the definition of knowledge to, "ϕ is known if ϕ is a consequence of the true statements in the knowledge base." In that case, however, a statement ϕ which is a consequence of the false statements in the knowledge base is not known, but the agent cannot become aware that it is not known, since he has no way to know that it is not true. For example, a person who believes firmly that the world is flat does not know that the world is flat — one cannot know that the world is flat, since it is not — but he does not know that his belief does not constitute knowledge. Thus, the axiom of negative introspection fails.

Table 1 summarizes these five axioms of knowledge. The modal logic characterized by these axioms is known as S5. (The terminology is purely historical.) The reasonableness of these axioms has been extensively discussed in the literature; see, for example, [Lenzen, 78].

The modal theory S5 is *sound* and *complete* with respect to knowledge base models. Soundness means that any knowledge base model satisfies the axioms of S5; this we have essentially established this by our arguments above deriving the axioms from the model. Completeness is the following property: Let $\mathcal{T}$ be a set of sentences that are consistent with S5. Then there is a knowledge base model in which all the sentences of $\mathcal{T}$ are true.

2.1.3 State-based definition of knowledge

We next look at a quite different model of knowledge.

Consider a device D that can be in one of a collection of states. We will say, "D knows ϕ at time T," if,

i. the state of D at time T is S; and

ii. whenever D is in state S, ϕ is true.

For example, let $D1$ be a mercury thermometer in a room whose state at any moment is the level of the mercury. Suppose that at time t0, the mercury points to 20 degrees Centigrade. Then at t0, $D1$ knows that the temperature of the room is 20 degrees, because whenever the mercury points to 20 degrees, the temperature is 20 degrees. $D1$ also knows

That the temperature is greater than 10 degrees.
That the temperature in degrees Centigrade is a square number.
That any ice open in the room is melting.
That any planar map can be colored with four colors.
...

because all these statements are true when the mercury points to 20 degrees. On the other hand $D1$ does not know in S that it is raining outside, or that the President of the US is a Democrat, because these statements are sometimes true when the mercury points to 20 and sometime false.

Fixing a particular device D and time T, we define the modal operator "know(ϕ)" to mean "D knows ϕ at time T" as defined above. It is easily checked that this operator satisfies the axioms of S5 described in the previous section. Let S be the state of D at time T.

K.1 Suppose that D knows ϕ at time T. Then ϕ is true whenever S holds. In particular, ϕ is true at time T.

K.2 If ϕ is a logical axiom, then ϕ is always true. So ϕ is certainly always true when S holds. So D knows ϕ at T.

K.3 Suppose that D knows ϕ in S and D knows $\phi \Rightarrow \psi$ in T. That means, by definition, that both ϕ and $\phi \Rightarrow \psi$ are true whenever S holds. But if ϕ and $\phi \Rightarrow \psi$ are true at a given time, then so is ψ. Thus ψ is true whenever S holds; that is, D knows ψ.

K.4 Suppose that D knows ϕ at time T. Then D knows ϕ whenever S is true; that is, D knows in T that D knows ϕ.

K.5 Suppose that D does not know ϕ in T. Then D cannot know ϕ at any time when S holds; that is, D knows in T that D does not know ϕ.

Thus, though this model appears very different from the knowledge base model, its logical properties are identical. Our arguments above constitute a proof that the axioms of S5 are *sound* for the state-based definition of knowledge; it is also possible to show that they are *complete;* that is, that, given any set of sentences $\mathcal{T}$ consistent with the axioms S5, it is possible to construct a state-based model in which all the sentences in $\mathcal{T}$ are true.

One strength of this definition is that it is natural to speak of multiple devices and the interactions of knowledge between them. Consider two different devices $D1$ and $D2$, with two different states. According to our definition, the statement "At time T, $D1$ knows that $D2$ knows ϕ," means "Whenever $D1$ is in its current state, $D2$ knows ϕ", which means "Whenever the current state of $D1$ holds, $D2$ is in some state $S2$ such that, whenever $S2$ holds, ϕ is true."

For example, let $D1$ be a digital thermometer that displays the temperature in a room in degrees, rounded down from the true value. Let $D2$ be a digital thermometer that displays temperatures in tenths of a degree, rounded down from the true value. Now let us say that at time t0, the true temperature "temp0" is 25.87 so that $D1$ displays "25" and $D2$ displays "25.8". Then at time t0,

$D1$ knows that $25 \leq$ temp0 < 26.
$D2$ knows that $25.8 \leq$ temp0 < 25.9.
$D2$ knows that $D1$ knows that $25 \leq$ temp0 < 26.
$D1$ knows that one of the following is true: Either
    $D2$ knows that $25.0 \leq$ temp0 < 25.1; or
    $D2$ knows that $25.1 \leq$ temp0 < 25.2; or
    $D2$ knows that $25.2 \leq$ temp0 < 25.3; or

...or
$D2$ knows that $25.9 \leq \text{temp0} < 26.0$.

We can add this ability to speak of multiple states of knowledge to our modal language by placing the names of the individual devices as subscripts to the "know" operator. Thus, we could write the first three sentences above,

$\text{know}_{D1}(25 \leq \text{temp0} < 26)$.
$\text{know}_{D2}(25.8 \leq \text{temp0} < 25.9)$.
$\text{know}_{D2}(\text{know}_{D1}(25 \leq \text{temp0} < 26))$

We can also speak of the *combined* knowledge of two devices. Let $D3$ be an outdoor thermometer, and suppose that at time t0 $D3$ reads 10 degrees. Then at t0, $D1$ and $D3$ combined know that it is colder outside than in, though neither device knows this by itself. In general, $D1$ and $D2$ combined know ϕ at time T, if, at any time when both $D1$ and $D2$ are in their current states, ϕ is true.

Another useful concept in reasoning about multiple agents is that of their *common* knowledge (also called *mutual* knowledge.) A and B have common knowledge of ϕ if

A know ϕ;
B knows ϕ;
A knows that B know ϕ;
B knows that A knows ϕ;
A knows that B knows that A knows ϕ;
B knows that A knows that B know ϕ;
and so on.

That is, there is a perfect understanding about ϕ between A and B.

In the above example, thermometers $D1$ and $D2$ have common knowledge that the temperature is between 25 and 26. By contrast, consider a thermometer $D4$ which displays temperatures in degrees rounded to the nearest degree, (rather than rounded down.) Then $D1$ and $D4$ do not have any knowledge about the temperature in common! Suppose that the temperature is actually 25.87.

$D1$ displays 25.
$D2$ displays 26.
$D1$ knows that $D2$ either displays 25 or 26; thus $D1$ knows that $D2$ knows that the temperature is between 24.5 and 26.5.
$D1$ knows that $D2$ knows that $D1$ either displays 24, 25, or 26; thus $D1$ knows that $D2$ knows that $D1$ knows that the temperature is between 24 and 27. ...

In general, we can define common knowledge as follows:

Devices $D1$ and $D2$ have common knowledge of ϕ at time T if
the state of $D1$ at T is an element of some set of states $SS1$;
the state of $D2$ at T is an element of some set of states $SS2$;
whenever the state of $D1$ is in $SS1$, ϕ is true and the state of $D2$ is in $SS2$;
whenever the state of $D2$ is in SS2, ϕ is true and the state of $D1$ is in $SS1$.

By adding additional modal operators and suitable axioms, we could extend our modal theory to include combined knowledge and common knowledge.[1] However, we will not pursue this direction.

[1]However, it is not possible to give a complete axiomatization of common knowledge using axioms similar to K.1 through K.5.

Figure 1: Free tree of nodes

2.1.4 Applications of S5: Distributed Systems

The state-based model of knowledge and the modal theory S5 are the most common theories of knowledge used in the analysis of distributed systems. Each element in the system is consider an agent in itself, and what it knows is determined by its entire internal state. What the system as a whole knows is determined by the combined states of all the elements.

Consider the following simple example: A network consists of a free tree of elements with undirected links. We wish to turn it into a rooted tree, where every elements knows which of its neighbors has been chosen as its father, so that messages can be passed up or down the tree in a well defined manner. (Figure 1).

We consider that each node U knows the following facts about its place in a rooted tree.

- If V is a neighbor of mine, then either I am V's father or V is my father.
- If I am the root then I have no father.
- If I am not the root, then I have exactly one father.

Moreover, U knows who all its neighbors are.

Finally we require nodes to follow the following protocol:

If you know your relation to neighbor V, then tell V.

This protocol allows the information about the structure of the tree to be distributed among the nodes in any of a number of ways. If the systems manager tell one node that it is the root, then the structure of the tree will percolate down from the root to the leaves. If he tells all but one node that they are not the root, then the structure of the tree will percolate up from the leaves to the root.

Note the extensive use of the concept of "knowledge" in this description. It is certainly possible to describe the scenario and the protocol without the use of this concept, but using a logic of knowledge is intuitive and elegant.

We can use modal propositional logic to characterize any particular network and starting condition and to determine its behavior. We will use a sequence of times $T0$, $T1$, $T2$, ... We will use the following symbols:

- For each node U and time T, the modal symbol "$\text{know}_{U,T}(\phi)$" means "U knows ϕ at time T."

- For each pair of nodes U, V and time T, the modal symbol "$\text{message}_{U,V,T}(\phi)$" means "$U$ sends message ϕ at time T."
- For each pair of nodes U, V, the proposition atom "father-U-V" means "U is the father of V."
- For each node U, the propositional atom "root-U" means that U is the root.

Using this language, we can characterize the knowledge of every node. For example, the initial knowledge of node E in figure 1 is described in the rules:[2]

1. $\text{know}_{E,T}(\text{root-E} \Leftrightarrow (\text{father-E-D \& father-E-M \& father-E-N}))$

(E knows that he is the root if and only if he is the father of all his neighbors.)

2. $\text{know}_{E,T}(\neg(\text{father-D-E \& father-M-E}) \;\&$
$\neg(\text{father-D-E \& father-N-E}) \;\&$
$\neg(\text{father-M-E \& father-N-E}))$

(E knows that he has at most one father.)

3. $\text{know}_{E,T}((\text{father-D-E} \Leftrightarrow \neg\text{father-E-D}) \;\&$
$(\text{father-M-E} \Leftrightarrow \neg\text{father-E-M}) \;\&$
$(\text{father-N-E} \Leftrightarrow \neg\text{father-E-N}))$.

(E knows for each neighbor V that either he is the father of V or vice versa.)

The protocol for node E can be characterized in the rules:

4. $\text{know}_{E,T}(\text{father-E-D}) \Rightarrow \text{message}_{E,D,T}(\text{father-E-D})$
5. $\text{know}_{E,T}(\text{father-D-E}) \Rightarrow \text{message}_{E,D,T}(\text{father-D-E})$
6. $\text{know}_{E,T}(\text{father-E-M}) \Rightarrow \text{message}_{E,M,T}(\text{father-E-M})$
7. $\text{know}_{E,T}(\text{father-M-E}) \Rightarrow \text{message}_{E,M,T}(\text{father-M-E})$
8. $\text{know}_{E,T}(\text{father-E-N}) \Rightarrow \text{message}_{E,N,T}(\text{father-E-N})$
9. $\text{know}_{E,T}(\text{father-N-E}) \Rightarrow \text{message}_{E,N,T}(\text{father-D-E})$

Finally, the knowledge gained by E from receiving a message is characterized in the axioms

10. $\text{message}_{D,E,T}(\text{father-E-D}) \Rightarrow \text{know}_{E,T+1}(\text{father-E-D})$.
11. $\text{message}_{D,E,T}(\text{father-D-E}) \Rightarrow \text{know}_{E,T+1}(\text{father-D-E})$.
12. $\text{message}_{M,E,T}(\text{father-E-M}) \Rightarrow \text{know}_{E,T+1}(\text{father-E-M})$.
13. $\text{message}_{M,E,T}(\text{father-M-E}) \Rightarrow \text{know}_{E,T+1}(\text{father-M-E})$.
14. $\text{message}_{N,E,T}(\text{father-E-N}) \Rightarrow \text{know}_{E,T+1}(\text{father-E-N})$.
15. $\text{message}_{N,E,T}(\text{father-N-E}) \Rightarrow \text{know}_{E,T+1}(\text{father-N-E})$.

We state the analogous axioms for all the remaining nodes. We initialize the protocol by stating that at T0, H knows that it is the root.

$$\text{know}_{H,T0}(\text{root-H})$$

We can now prove that the knowledge disseminates down through the tree, and that every node will now his father by time T3.

[2] These and the remaining rules below are schemas: one instance for each value of T.

It should be noted that the proof only involves axioms A.2 and A.3 applied to sentence in the purely propositional calculus. Though we are reasoning about knowledge, the nodes do not themselves either about their own knowledge or about other nodes knowledge.

In section 2.2.4 we will show how this same information can be expressed more compactly using a more expressive language.

2.1.5 Application: Three wise men

A more frivolous example which illustrates more sophisticated reasoning is the well-known puzzle of the three wise men.

> The king summons three of his wise men and pastes on their forehead a small black dot. The wise men are facing one another, so that they can all see the dot on each others forehead, but they do not know the color of the dot on their own. He tells them, "On each of your foreheads, I have placed a dot which is either white or black. I have put a black dot on at least one of your foreheads. Do any of you know what the color on your forehead is?"
>
> The wise man all answer in unison, "No, your majesty".
>
> The king asks, "Do you now know what the color on your forehead is?"
>
> They again answer in unison, "No, your majesty."
>
> The king asks, "Do you now know what the color on your forehead is?"
>
> All three wise men answer, "Your majesty, I have a black dot on my forehead."
>
> How do the wise men determine this?

The solution is as follows:

After the first two questions, wise man A reasons as follows:

> Suppose that I have a white dot on my forehead. Then wise man B sees that I have a white dot and that C has a black dot. Now, in that case after the first question, wise man B could have reasoned to himself,
>
>> Suppose that I (B) have a white dot on my forehead. Then C sees the white dot on A and on me. Then C could reason to himself.
>>
>>> A's and B's dots are white but there is a black dot. The black dot must be mine.
>>
>> So C should have known the color of his dot before the king asked his first question, and should have answered the first question. But he did not answer it. This is a contradiction: therefore the supposition was wrong. I must have a black dot.
>
> So B should have been known the color of his own dot as soon as he saw that C did not answer the first question, and B should have been able to answer the second question. But he did not answer it. This is a contradiction: therefore, the supposition was wrong. I must have a black dot.

Our analysis of this puzzle proceeds as follows. We use the following symbols:

- For each wise man X and time T, the modal symbol "know$_{X,T}(\phi)$" means "U knows ϕ at time T."
- The propositional atom "speak-X-T" means that wise man X answers the question at time T.
- The propositional atom "black-X"

We now posit the following axioms:

1. black-A $\vee$ black-B $\vee$ black-C.
   (One of the wise men has a black dot.)
2. For each pair of wise men U, V where $U \neq V$ and each time T,
   know$_{U,T}$(black-V) $\vee$ know$_{U,T}$($\neg$black-V).
   (Each wise man knows the colors on the others' heads.)
3. speak-X-T $\Leftrightarrow$ [know$_{X,T}$(black-X) $\vee$ know$_{X,T}$($\neg$black-X)]. (A wise man answers the question at time T just if he knows the color on his head.)
4. For each pair of wise men U, V and each pair of times $T1, T2$ such that $T1 < T2$,
   [speak-U-T1 $\Rightarrow$ know$_{V,T2}$(speak-U-T1)] & [$\neg$speak-U-T1 $\Rightarrow$ know$_{V,T2}$($\neg$speak-U-T1)].
   (All the agents know of all past answers and failures to answer.)
5. If ϕ is any sentence in (1-5) and U is a wise man, then know$_{U,T}(\phi)$.
   (All these axioms are common knowledge among the wise men. Note that (5) is recursive; thus, the agents all know axioms 1-4; they all know that they all know them; they all know that they all know that they all know them; and so on.)

Applying the axioms of modal logic, it can now be proven that, if none of the wise men answer before the third time the question is asked, then all three will be able to answer after the third time. The proof is essentially a symbolic translation of the informal argument outlined above.

What cannot be done from these axioms is to prove that none of the wise men can answer before the third time the question is asked. This would require showing that none of them can learn their color in any other way. There is no easy way to state the "frame" axiom on ignorance needed to support this conclusion in this kind of modal languages. (It can, however, be stated in the language of possible worlds. (Scherl and Levesque, 93).)

2.1.6 Possible Worlds

The modal theory S5 is a very powerful one — often too powerful. In many applications, the axiom of positive introspection and, even more, the axiom of negative introspection, are unnecessary and implausible. For such applications the modal theory S4, which consists of axioms K.1-K.4 or T, which consists of axioms K.1-K.3, are more suitable. (Dropping the axiom of consequential closure, though often desirable, is a much harder nut to crack.) Since these theories are important, it is worthwhile trying to develop semantic models for them. Such a semantic model is provided by the theory of *possible worlds.* [Kripke, 63], [Hintikka, 69]

A possible world is, intuitively, one way that the world could be at a moment. A possible world has two aspects. First, each possible world has a *content*, a state of the external world aside from the knowledge of the agents. Secondly, the possible worlds are connected by *accessibility relations,* one for each agent, which characterize the knowledge states of the various agents.

In a propositional logic, the content of a possible world is an assignment of truth-values to all the propositional atoms. For instance, if the propositional atoms are p, q, r, and s, then one possible

Figure 2: Structure of possible worlds

world might have content { p $\rightarrow$ TRUE; q $\rightarrow$ FALSE; r $\rightarrow$ FALSE; s $\rightarrow$ TRUE }. A different possible world might have content { p $\rightarrow$ TRUE; q $\rightarrow$ TRUE ; r $\rightarrow$ TRUE ; s $\rightarrow$ FALSE }.

A possible world is thus an abstraction of the notion of an "instant of time", which we used forming the state-based definition. At any instant of time, each propositional atom is either true or false; so in each possible world. More than one possible world can have the same content.

The intuition behind the accessibility relation is this: World $W2$ is accessible from $W1$ for agent A only if $W2$ is consistent with everything that A knows in $W1$. For example, if world $W2$ is accessible from $W1$ and $W2$ has content { p $\rightarrow$ TRUE; q $\rightarrow$ FALSE; r $\rightarrow$ FALSE; s $\rightarrow$ TRUE }, then this truth assignment must be consistent with everything that A knows in $W2$; A does not know in $W1$ that this does not constitute the state of the world.

Formally, we take the accessibility relation to be primitive, and we define knowledge in terms of accessibility:

Definition 3: Agent K knows ϕ in world W iff ϕ is true in every world accessible from W.

Imbedded knowledge can be defined in terms of chains of accessibility relations. For example, the statement "In $W1$, A knows that B knows that p" is equivalent to "In every world $W2$ accessible for A from $W1$, B knows that p", which is equivalent to

> If $W2$ is accessible for A from $W1$ and $W3$ is accessible for B from $W2$, then p is true in $W3$.

Consider, for example, the structure of possible worlds shown in figure 2. The following statements are true.

In W1, A knows that p,
    because p is true in the worlds (W1 and W2) accessible for A from W1.
In W1, B does not know that p,
    because p is false in W3, which is accessible for B from W1.
In W1, B know that p $\vee$ r,
    because p $\vee$ r is true in the worlds (W1 and W3) accessible for B from W1.
In W1, B knows that q,
    because q is true in the worlds (W1 and W3) accessible for B from W1.
In W2, B knows that $\neg$q,
    because q is false in the worlds (W2 and W4) accessible for B from W2.
In W1, A knows that B knows whether q is true (that is, either B knows q or B knows

¬q) because:

The worlds accessible for A from W1 are W1 and W2;
In W1, B knows q;
In W2, B knows ¬q.

This definition of knowledge is a generalization of the "state-based" definition of knowledge. Indeed, if we define a possible world to be an instant of time, and we define T1 and T2 to be mutually accessible for agent A if the state of A is the same in T1 and T2, then the state-based definition reduces to exactly the knowledge accessibility definition.

It may seem that our definition here is circular, or at least that it puts the cart before the horse. We informally defined knowledge accessibility in terms of knowledge — world W2 is accessible from W1 for A if everything in W2 is consistent with what A knows in W1 — and then we formally defined knowledge in terms of accessibility. Indeed, this is not a definition of knowledge in the same sense in which definitions 1 and 2 were. It does not allow us to look at an actual situation and determine what any agent knows. Rather, the function of this definition is to give a concrete model for the modal theory. We choose a desirable theory of knowledge, construct a corresponding system of possible worlds, and then use the system of possible worlds for reasoning about the knowledge.

The various axioms of modal epistemic logic turn out to be closely related to constraints on the structure of the accessibility relation.

- Axioms K.2 (knowledge of the axioms) and K.3 (consequential closure) hold in any system of possible worlds. If ϕ is a logical axiom, then ϕ must hold in every possible world, and, in particular in every accessible possible world. If ϕ and $\phi \Rightarrow \psi$ hold in every accessible world, then ψ must hold in every accessible world.
- Axiom K.1, veridicality, holds if the accessibility relation is *reflexive*; that is, every world is accessible from itself. Suppose that $W1$ is accessible from $W1$ and that ϕ is false in $W1$. Then there is a world accessible from $W1$ in which ϕ is false, so A does not know ϕ in $W1$. Conversely, if A does know ϕ in $W1$ then ϕ must be true in $W1$.
- Axiom K.4, positive introspection, holds if the accessibility relation is *transitive*; that is, for all $W1$, $W2$ and $W3$, if $W2$ is accessible from $W1$ and $W3$ is accessible from $W2$, then $W3$ is accessible from $W1$. Suppose that it is not the case that in $W1$ A knows that he knows ϕ. Translating this into the language of possible worlds, this means that there exist worlds $W2$ and $W3$ such that $W2$ is accessible from $W1$, $W3$ is accessible from $W2$, and ϕ is false in $W3$. By the transitivity property, $W3$ is accessible from $W1$. Therefore A does not know ϕ in $W1$. Conversely, if A knows ϕ in $W1$, then A knows that he knows ϕ in $W1$.
- Axiom K.5, negative introspection, holds if the accessibility relation has the *Euclidean* property: if $W2$ and $W3$ are both accessible from $W1$, then $W3$ is accessible from $W2$. Suppose that A does not know in $W1$ that he does not know ϕ. Translating this into the language of possible worlds, this means that there exists a world $W2$ accessible from $W1$ such that A knows ϕ in $W2$; that is, ϕ holds in every world $W3$ accessible from $W2$. Let WA be any world accessible from $W1$. By the Euclidean property, WA is accessible from $W2$; hence, as we have stated, ϕ is true in WA. But then, ϕ is true in every world accessible from $W1$; that is, A knows ϕ in $W1$. Conversely, if A does not know ϕ in $W1$, then he knows that he does not know ϕ.

There are also completeness results associated with the above correspondences: [Kripke, 63]

- Axioms K.2 and K.3 are complete over the class of systems of accessibility relations. That is, given any theory obeying K.2 and K.3, there is a possible worlds model in which the theory holds.

- The modal theory T, containing axioms K.1, K.2, and K.3, is complete over the class of systems of reflexive accessibility relations.
- The modal theory S4, containing axioms K.1, K.2, K.3, and K.4, is complete over the class of systems of accessibility relation that are reflexive and transitive.
- The modal theory S5, containing axioms K.1, K.2, K.3, K.4, and K.5 is complete over the class of systems of accessibility relation that are reflexive, transitive, and Euclidean. It should be noted that a relation is reflexive, transitive, and Euclidean just if it is an equivalence relation.

2.2 First-Order Theories of Knowledge: Modal and Other

It is well known that the propositional calculus, though useful in for finite combinatorial problems, is too limited and inexpressive for general knowledge representation. First-order logic, which allows one to speak about objects, their properties, and their relations, is a much more expressive language and is the most widely used language in knowledge representation. In this section we will study how to combine epistemic logic with first-order logic.

2.2.1 First-Order Logic with Equality

First, a brief review of first-order logic with equality. First-order logic uses the following types of symbols:

Constants: A constant symbol denotes a particular entity. E.g. "john", "muriel" "massachusetts", "23".

Functions: A function symbol denotes a mapping from a number of entities to a single enti- ties: E.g. "father_of" is a function with one argument. "plus" is a function with two arguments. "father_of(john)" is some person. "plus(2,7)" is some number.

Predicates: A predicate denotes a relation on a number of entities. e.g. "married" is a predicate with two arguments. "odd" is a predicate with one argument. "married(john, sue)" is a sentence that is true if the relation of marriage holds between the people John and Sue. 'odd(plus(2,7))" is a true sentence.

A particular predicate is the equals sign "$X = Y$" (written infix). The equals sign denotes the identity relation: Two terms are equal if they denote the same entity.

Variables: These represent some undetermined entity. Examples: "X" "$S1$" ...

Boolean operators: $\neg$, $\vee$, &, $\Rightarrow$, $\Leftrightarrow$.

Quantifiers: The symbols $\forall$ (for all) and $\exists$ (there exists).

Grouping symbols: The open and close parentheses and the comma.

The language of the predicate calculus contains *terms*, *formulas* and *sentences*. A term denotes a particular entity. A sentence is a statement which is either true or false. A formula is an incompletely fixed sentence.

A *term* is either

1. A constant symbol; or
2. A variable symbol; or
3. A function symbol applied to terms.

Examples: "john", "X", "father_of(john)", "plus(X,plus(1,3))" are terms.

An *atomic formula* is a predicate symbol applied to terms.
Examples: "odd(X)" "odd(plus(2,2))", "married(sue,father_of(john))" are atomic formulas.

A *formula* is either

1. An atomic formula; or

2. The application of a Boolean operator to formulas; or

3. A quantifier followed by a variable followed by a formula.

Examples: "odd(X)," "odd(X) $\vee$ $\neg$odd(plus(X, X))," "$\exists_X$ odd(plus(X, Y))," "$\forall_X$ odd(X) $\Rightarrow$ $\neg$odd(plus(X,3))."

A *sentence* is a formula with no free variables. (That is, every occurrence of every variable is associated with some quantifier.)

2.2.2 First Order Modal Logic

Syntactically, we incorporate the modal operator "know(A, ϕ)" into the predicate calculus simply by adding a fourth way of constructing a formula. We revise the rule above as follows:

A *formula* is either

1. An atomic formula;

2. The application of a Boolean operator to formulas;

3. A quantifier followed by a variable followed by a formula; or

4. An expression of the form "know(μ, ϕ)" where ϕ is a formula and μ is a term denoting an agent.

Thus the following are sentences.

know(john, child(alice, charles)). — John knows that Alice is the child of Charles.
know(john, $\forall_X$ on(X,table) $\Rightarrow$ red(X)). —
    John knows that everything on the table is red.
$\exists_A$ know(A, $\exists_Y$ child(Y,charles)). —
    There is someone who knows that Charles has a child.
$\forall_A$ $\neg$know(A,P = NP) & $\neg$know(A,P $\neq$ NP). —
    No one knows whether P = NP.

2.2.3 De re and de dicto modality

There are two, closely related, tricky issues in first-order modal logic: equality substitution and quantification into modal context.

As we discussed at the beginning of section 2, ordinary first-order logic is referentially transparent; equal terms can be substituted one for another without changing the truth of a sentence. For example given "married(oedipus,jocasta)" and "jocasta=mother_of(oedipus)", it follows that "married(oedipus,mother_of(oedipus))". In modal contexts, by contrast, this does not work. Even if "know(oedipus, married(oedipus, jocasta))" and "jocasta=mother_of(oedipus)" are both true, "know(oedipus, married(oedipus, mother_of(oedipus)))" may still be false. The axiom governing the substitution of equal terms must therefore be restricted to contexts outside any modal operators.

By convention, sentence that contain a quantifier inside a modal operator are interpreted differently from sentences with the modal operator inside the quantifier. For example, the sentence "know(pete, $\exists_Y$ child(charles, Y))" is read "Pete knows that Charles is someone's child." This does not attribute to Pete any great knowledge of Charles; after all, we all know that everyone has parents. On the other hand, the sentence "$\exists_Y$ know(pete, child(charles, Y))" is read, "There is a particular person Y such that Pete knows that Charles is the child of Y"; i.e. "Pete knows the identity of one of Charles' parents. "

Note that the first statement would follow from an assertion such as "know(pete, child(charles, father_of(charles)))." If Pete knows that Charles is a child of his father, then he knows that Charles is someone's child. That degree of knowledge would certainly not justify the second sentence. On the other hand, if we said that "know(pete, child(charles, justin))", then that would (in most circumstances) justify the second statement. If Pete knows that Charles is the child of Justin, then Pete knows the identity of one of Charles' parents.

In the philosophical literature, the first of these sentences is called "knowledge *de dicto*" (knowledge of the word, in Latin) and the second is called "knowledge *de re*" (knowledge of the thing). Terms like "justin" that allow existential abstraction outside the scope of a modal operator are called "rigid designators"; terms like "father_of(charles)" that do not are called "non-rigid designators." The distinction is admittedly a murky one and has been extensively studied in the literature of modal logic and the related philosophy. For our purposes, we will simply take this as a logical convention: We will assume that the theory specifies which terms are rigid designators, and posit that existential abstraction of large scope can be performed only on these.

For a discussion of the complete axiomatization of first-order modal logic, see [Hughes and Cresswell, 1968]

2.2.4 Applications

Using a first order modal logic, we can very much simplify the axiomatization of the examples considered in section 2.1.4.

We extend the modal operator "know" to take a temporal argument as well; thus "know(A, T, ϕ)" means "A knows ϕ at time T. Table 2 gives an axiomatization of the protocol for imposing a direction on a tree-shaped distributed system.

We can specify the topology of the network in terms of the "neighbor" relation (being sure to assert that our list of neighbors is exhaustive). We can start the protocol moving by assigning one node to the root at a particular time. We can then prove in the modal logic that all the nodes in the tree find out their father after a certain amount of time.[3]

The use of first-order modal logic for the three wise men problem is analogous.

2.2.5 Direct Use of Possible Worlds

Another approach to using first-order logic for epistemic reasoning is to talk directly about the kind of possible worlds model discussed in section 2.1.6. In this way, it is possible to represent epistemic

[3]Note, however, that this proof is for a *particular* network. To prove that this will work for *all* networks requires second-order logic, or a special-purpose induction axiom.

T.1 $\forall_{U,V,T}$ know(U,T,neighbor(U,V)) $\vee$ know(U,T,$\neg$neighbor(U,V)).
Node U always knows who his neighbors are.

T.2 $\forall_{U,T}$ know(U,T,$\forall_V$ neighbor(U,V) $\Leftrightarrow$ neighbor(V,U)). Node U knows that its neighbor relations are symmetric.

T.3 $\forall_{U,T}$ know(U,T, [root(U) $\Leftrightarrow$ $\forall_V$ neighbor(U,V) $\Rightarrow$ father(U,V)]).
Node U always knows that he is the root iff he is the father of all his neighbors.

T.4 $\forall_{U,T}$ know(U,T,$\forall_{V,W}$ [father(V,U) & father(W,U)] $\Rightarrow W = V$).
Node U always knows that he has at most one father.

T.5 $\forall_{U,T}$ know(U,T, $\forall_V$ neighbor(V,U) $\Rightarrow$ [father(U,V) $\Leftrightarrow$ $\neg$father(V,U)].
Node U always knows that, for each neighbor V, either U is the father of V, or V is the father of U, but not both.

T.6 $\forall_{U,V,T}$ know(U,T,father(U,V)) $\Rightarrow$ message(U,V,T,father(U,V)).
If U knows that he is the father of V, he should tell V.

T.7 $\forall_{U,V,T}$ know(U,T,father(V,U)) $\Rightarrow$ message(U,V,T,father(V,U)).
If U knows that he is the child of V, he should tell V.

T.8 $\forall_{U,V,T,X,Y}$ message(U,V,T,father(X,Y)) $\Rightarrow$ know($V,T+1$,father(X,Y)).
V believes the messages he gets from U.

Table 2: Axiomatization of "tree" protocol in first-order modal logic.

statements and to do epistemic reasoning without the explicit use of any modal operators or modal logic.

First, we have to extend the notion of a possible world. The intuition is the same as before: a possible world is one way that the world could be. Since we are moving to a richer theory, however, the actual structure of the possible world becomes more complete. In the propositional case, the content of a possible world was a truth assignment to the propositional atoms. In the first-order case, the content of a possible world is an *interpretation:* that is, a mapping from each constant symbol to an entity, from each function symbol to a mapping over the domain of entities, and from each predicate symbol to a relation over the space of entities. For example, suppose that John does not whether Sacramento or San Francisco is the capital of California. That means that, in some possible worlds accessible for John, Sacramento is the capital of California while in others San Francisco is the capital of California. This, in turn, means that in the first of these worlds, the interpretation of the symbol "capital" includes the pair <Sacramento, California>, while in the second world, it includes the pair <San Francisco, California>.

The key technical device here is in developing a representation along these lines is the use of *fluents.* A fluent is something that has different values in different possible worlds. A *Boolean fluent* is true in some possible worlds and false in others. For example "capital(sacramento, california)" is true in some worlds and false in others. Other fluents take values that are not truth-values. For example, if John does not know whether Alice's father is Mike or Sid, then the fluent "father_of(alice)" has the value Mike in one world and Sid in another. Any object-level statement (that is, any statement not involving knowledge) of which any agent can be uncertain must be represented by a Boolean fluent. Any term whose value is not universally known must be represented by a non-Boolean fluent.

We can now develop the first-order language of possible worlds as follows:

The domain of entities contains:

1. Agents, who have knowledge.
2. Things of the usual kind.
3. Possible worlds.
4. Fluents.

The symbols in the language include

1. Those symbols that are normally used to describe the external world. However, function and relation symbols must be interpreted as mapping their arguments onto fluents.
2. The relation "true_in(W, B)" holds on world W and Boolean fluent B if B is true in W.
3. The function "value_in(W, F)" maps world W and non-Boolean fluent F to the value of F in W.
4. The relation "k_acc($A, W1, W2$)" means that world $W2$ is accessible from world $W1$ for agent A.
5. The constant "w0" represents the real world.

Examples:

- $\forall_{W1}$ k_acc(john,w0,$W1$) $\Rightarrow$ true_in($W1$,child(alice,charles)).
  The fluent "child(alice,charles)" holds in every world accessible from w0 for John.
  John knows that Alice is the child of Charles.
- $\forall_{W1}$ k_acc(john,w0,$W1$) $\Rightarrow$
  $\forall_X$ true_in($W1$,on(X,table)) $\Rightarrow$ true_in($W1$,red(X)).
  For every world $W1$ accessible for John from w0, every object which on the table in $W1$ is red in $W1$.
  John knows that everything on the table is red.
- $\forall_{W1}$ k_acc(john,w0,$W1$) $\Rightarrow \exists_X$ true_in($W1$,capital(X,california)).
  In every world accessible for John, there is something which is the capital of California.
  John knows that California has a capital.
- $\exists_X$ $\forall_{W1}$ k_acc(john,w0,$W1$) $\Rightarrow$ true_in($W1$,capital(X,california)).
  There is a particular city which is the capital of California in every world accessible for John.
  John know which city is the capital of California.

As the last two examples above illustrate, the difference between *de re* and *de dicto* knowledge is expressed in the language of possible worlds by the relative scopes of the quantification over objects and the quantification over possible worlds. In the *de dicto* sentence

$\forall_{W1}$ k_acc(john,w0,$W1$) $\Rightarrow \exists_X$ true_in($W1$,capital(X,california)).

we assert that in each accessible world, California has a capital, but the identity of that capital may vary from one world to the next; in some it may be Sacramento and in other San Francisco. In the *de re* sentence

$\exists_X \ \forall_{W1}$ k_acc(john,w0,$W1$) $\Rightarrow$ true_in($W1$,capital(X,california)).

there must be a particular city — namely, Sacramento — which is the capital of California in all possible worlds. Thus, a rigid designator in this theory is a fluent whose value is the same in all possible worlds, while a non-rigid designator is one whose value changes from one world to the next.

2.2.6 Syntactic Representations

There is yet a third approach to the representation of epistemic facts, called *syntactic representation*. In this theory, we view knowledge as a relation between a string of characters. For example, the fact that John knows that Alice is the child of Charles is expressed in the statement "know(john, $\prec$child(alice,charles)$\succ$)", where the term $\prec$child(alice,charles)$\succ$ denotes simply the string of 20 characters, starting with 'c' and ending with a close parenthesis, between the curly angle brackets. (The curly angle brackets themselves are just string delimiters, like quotation marks.)

We can then develop a theory of knowledge by developing a first-order theory of strings of characters and expressing properties of knowledge in terms of these. For example, if we want our theory to have the consequential closure property, this can be achieved in an axiom of the following form:

$\forall_{A,S1,S2}$ know(A, $S1$) & know(A,catenate($S1$,$\prec\Rightarrow\succ$,$S2$)) $\Rightarrow$ know(A, $S2$)

(Note: This is a single axiom, not an axiom schema. $S1$ and $S2$ are first-order variables that range over strings. "catenate(SA, SB, SC)" is a function mapping three strings to the string that results from concatenating them. Here, we are catenating an implies sign '$\Rightarrow$' between the antecedent and the consequent of an implication.)

The different between syntactic theories and modal theories is roughly analogous to the difference between direct quotation ("John said 'I am hungry'") and indirect quotation ("John said that he was hungry.") Any kind of sayable or writable string may be imbedded in direct quotation, while only meaningful strings may be embedded in indirect quotation. "John said 'Arglebargle glumph'" is meaningful while "John said that arglebargle glumph" is not. Purely syntactic predicates, which relate only to the form of the utterance, may be used of the object of direct quotation; for example, one can say "John said something that began with 'T' and ended with 'y'." Analogously, in a syntactic theory, the expression, "know(john,$\prec, \Rightarrow X \forall \vee$ cat$\succ$)" is well formed (though false), while nothing of the kind is well formed in a modal theory. Similarly, in a syntactic theory we can say, "John knows a string that contains $\prec$Napoleon$\succ$ as a substring"; nothing of the kind can be expressed in a modal theory.

Syntactic representations have two major strengths. First, they are an extremely expressive language. Statements like "John knows something that Bill doesn't;" "Vladimir knows the first 26 digits of π;" "Kathy knows a rule which gives sufficient conditions for appendicitis in terms of observable symptoms," "The database keeps a log of the exact text of all transaction entries" can all be expressed quite directly in a syntactic representation, while they can be expressed only clumsily if at all in a modal or a possible worlds representation.

Second, the foundations of first-order modal and possible worlds theories rest on concepts that are obscure and ill-defined: the *de re / de dicto* distinction; the notion of a rigid designator; the notion of a possible world and of accessibility between possible worlds; and so on. By contrast, in developing a syntactic representation it is at least clear what category we are talking about; namely, strings of characters and their properties.

However, there are also significant disadvantages to syntactic representations:

- The representation is clumsy. Typically, statements of any complexity involve putting together strings by long catenations of terms of various kinds representing strings. Imbedded quotations lead to even greater complexity. This can be visually ameliorated by adopting a variety of syntactic conventions, but these are themselves complicated. [Morgenstern, 88], [Davis, 90].

- Using a syntactic representation eliminates much of the advantages of having a semantic model. One of the main purposes of a semantic model is that one can reason directly about the model and thus avoid doing the syntactic manipulations involved in creating explicit proofs. In a syntactic theory, one ends up doing syntactic manipulations anyway.

- Sufficiently powerful syntactic theories lead to paradox. In a syntactic theory of knowledge, it is possible to construct a sentence Φ that essentially states "A does not know Φ." Now is Φ true or false? Suppose Φ is false; then A does know Φ; then Φ must be true, by the veridicality property. Thus the assumption that Φ was false is contradictory, so Φ must be true. Thus, we have proven that logically, Φ must be true. But by axiom K.2, A knows all truths that are logically necessary. Therefore A knows that Φ. But to say that A knows Φ is to deny Φ; so Φ is false.

  This is known as the Knower's Paradox; it is exactly analogous to the better known Liar Paradox, which involves the sentence, "This sentence is false." A variety of solutions to these paradoxes have been developed. Mostly, they either involve restricting the language so that such sentences are ill-formed or using a multi-valued logic, where sentences can be considered true, false, or unfounded.

- No effective method is known for automating reasoning in a syntactic representation.

2.3 Application: Knowledge Preconditions for Plans

The question of whether an agent is able to carry out a plan may be divided into two parts. First, is the plan *physically feasible* for the agent; that is, is it physically possible to carry out the actions specified by the plan? Second, is the plan *epistemically feasible*; that is, does the agent know enough to perform the plan? A plan like "Make money at the roulette wheel by betting on the numbers that win," is not a useful one; though the physical actions involved are feasible, there is no way to find out in time what they are.

The epistemic feasibility of a plan depends both on the knowledge the agent has at the beginning of execution, and on the knowledge he gains during the course of execution. For example, suppose that Edith does not know Frank's phone number, but she has a directory in which she knows that his name is listed. In that case, she is not immediately able to carry out the plan "dial Frank's number," but she is immediately able to carry out the plan "sequence(look up Frank's number; dial Frank's number)".

Reasoning about the epistemic feasibility of plans requires a theory that integrates temporal reasoning with reasoning about knowledge. In order to determine whether an agent knows enough to perform a plan, we must be able to characterize what knows at the beginning of the plan, and how the state of the world and the knowledge of the agent change as a result of the execution of the plan. Like the relation, "At time T agent A knows fact ϕ", the relation, "At time T, agent A knows enough to perform plan P" is referentially opaque (intensional) in its final argument. Edith does not at this moment have enough information to perform the action "Dial Frank's phone number", but she does have enough information to perform the action "Dial 998-3123" which is extensionally the same action.

The problem of epistemic feasibility may be divided into two parts:

1. The knowledge preconditions (KP) problem for *actions.* Characterizing whether an agent knows enough to perform a single specified action in a given situation. For example, expressing the fact that Edith does not now have enough information to execute "Dial Frank's office number"; however, if she finds out that "Frank's office number is 998-3123" then she will have enough information. (We are here viewing dialing a seven-digit number as an single atomic action, rather than as a series of separate finger movements.)

2. The KP problem for *plans.* For example, determining that the plan "sequence(look up Frank's number; dial Frank's number)" is epistemically feasible for Edith, given that she knows that, after looking up Frank's number, she will know what his number is.

Clearly, a solution of the KP problem for plans must rest on a solution of the KP problem for actions.

The KP problems for actions is generally addressed as follows: An action E is epistemically feasible for agent A at time T iff A knows at T a specific behavior that constitutes that action in T. Edith cannot execute dial Frank's number because she does not know what behavior would constitute dialing Frank's number. If she finds out that dialing 998-3123 constitutes dialing Frank's number, then she will know how to dial Frank's number. In other words, in order to perform the action "Dial Frank's number," she must know what that action is. As we have discussed above (section 2.23, 2.25) the notion of "knowing what Q is" is represented in formal theories by using a existential quantifier of larger scope than the modal operator. In addressing the KP problem for plans, we will here consider only plans consisting of a fixed sequence of primitive actions by the agent himself. Richer categories of plans are considered in the literature: for example, [Moore, 85] considers plans with conditionals and loops; [Morgenstern, 87] considers plans involving several agents; [Davis, 93] considers indeterminate plans.

For sequences of primitive actions, we posit the following rule:

> The plan "sequence($E_1, E_2 \dots E_k$)" is epistemically feasible for A starting in S if A knows in S that
>
> - E_1 is epistemically feasible in S;
> - After E_1 has completed, E_2 will be epistemically feasible;
> - After E_1 and E_2 have completed, E_3 will be epistemically feasible;
>
>   ...
> - After $E_1, E_2 \dots E_{k-1}$ have completed, E_k will be epistemically feasible.

For example, the plan "sequence(look up Frank's number; dial Frank's number)" is epistemically feasible because Edith now knows how to look up Frank's number, and she knows that, after she looks up his number, she will know how to dial it. The plan "sequence(look up Harvey's number; dial Frank's number)" is not feasible because after executing the first step, Edith will not know how to execute the second; that is, she will not know what dialing action constitutes dialing Frank's number.

3 Automated Reasoning

In section 2, we have discussed at length what can be inferred in various epistemic representations. We now discuss how an automated reasoner can actually infer it.

The study of automated epistemic reasoning is still in a comparatively primitive state. In the study of automated reasoning in ordinary first-order logic, there is an extensive literature of many different techniques and representation of varying degrees of expressivity, completeness, efficiency and domains of application. Nothing of the kind exists for epistemic reasoning.

In addition to the specific techniques that we will survey here, there is a small body of work on worst-case complexity results for inference in various epistemic theories. See [Halpern and Moses, 85], [Shoham, 88].

3.1 Multiple Contexts

The most natural implementation of epistemic reasoning and probably the most frequently used in practice[4] is to multiple contexts. The basic idea here is that for each state of knowledge, or state of imbedded knowledge, one creates a separate context of object-level statements. Inference within each single context uses "ordinary" first-order inference techniques. Inference from one context to another uses special purpose inference rules.

Example: Suppose we are given the following facts:

In T1, A knows p and r.
In T1, B knows p $\Rightarrow$ q and knows r.
In T1, A knows that B knows r.
A tells p to B during [T1, T2].

We begin by initializing a number of contexts corresponding to time T1,

Context T1A (what A knows in T1) : { p, r }
Context T1B (what B knows in T1) : { p $\Rightarrow$ q, r}
Context T1AB (what A knows that B knows in T1): { r }

We also create the corresponding contexts T2A, T2B, and T2AB for time T2. These are initialized to be null and are filled in through inference rule.

First, we can apply a frame inference, that agents do not forget what they know. This gives us

Context T2A: { p, r }
Context T2B: { p $\Rightarrow$ q, r }
Context T2AB: { r }

Next we can apply an inference rule associated with "tell": If A tells ϕ to B during $[T1, T2]$, then in $T2$, B knows ϕ and A knows that B knows ϕ. Thus we update contexts T2B and T2AB to contain p.

Context T2B: { p $\Rightarrow$ q, r, p }.
Context T2AB: { p, r }

Finally we can apply modus ponens within context T2B.

Context T2B: { p $\Rightarrow$ q, r, p, q }

As this example illustrates, frequently much of the knowledge will be the same in several contexts. Knowledge base management can therefore be made more efficient if facts are labelled with contexts,

[4] For example, CYC [Lenat and Guha, 90] uses a variant of this method.

as in CONNIVER [McDermott and Sussman, 73] or ATMS [de Kleer, 85], rather than copying the knowledge base as a whole.

The major limitation of this approach is its limited expressivity. The only type of partial information that is directly represented is partial knowledge within the context. Therefore, facts such as the following are clumsy or impossible to express.

A knows p or A knows q.
A knows who the President of the Congo is.
The man with the white hat knows p.
A will know p when the bell rings. (partial specification of time.)
A knows that B does not know p.

Other issues that arise include

- How are new contexts generated?
- What are the cross-context inference rules?
- What closed world assumptions should be made? If T1A does not contain fact w, should we conclude that, in T1, A knows that w is false? or that he does not know whether it is true or false? If T1AB does not contain fact w, should we conclude that A knows that B knows that w is false? or that A knows that B does not know whether w? or that A does not know whether B knows whether w?

3.2 Possible worlds representation

The idea here is simple. Translate all epistemic statements to a first-order language of possible worlds and use a standard proof technique for first-order logic.

Example: Given:

1. Joe knows that a person always knows whether he is hungry.
$\forall_{W1}$ k_acc(joe,w0,$W1$) $\Rightarrow$
$\forall_X$ [[$\forall_{W2}$ k_acc($X, W1, W2$) $\Rightarrow$ true_in($W2$,hungry(X))] $\vee$
[[$\forall_{W3}$ k_acc($X, W1, W3$) $\Rightarrow$ $\neg$true_in($W3$,hungry(X)]].

2. Joe knows that Fred is hungry.
$\forall_{W4}$ k_acc(joe,w0,$W4$) $\Rightarrow$ true_in($W4$, hungry(fred)).

To prove:

3. Joe knows that Fred knows that he is hungry.
$\forall_{W1}$ k_acc(joe,w0,$W1$) $\Rightarrow$
$\forall_{W2}$ k_acc(fred,$W1, W2$) $\Rightarrow$ true_in($W2$, hungry(fred)).

Using the standard resolution procedure, we begin by Skolemizing 1 and 2 and the negation of 3.

1. $\neg$k_acc(joe,w0,$W1$) $\vee$ $\neg$k_acc($X, W1, W2$) $\vee$ true_in($W2$,hungry(X)) $\vee$
$\neg$k_acc($X, W1, W3$) $\vee$ $\neg$true_in($W3$,hungry(X)).

2. $\neg$k_acc(joe,w0,$W4$) $\vee$ true_in($W4$,hungry(fred)).

Negation of 3:

3.A. k_acc(joe,w0,w5).
3.B. k_acc(fred,w5,w6).
3.C. ¬true_in(w6,hungry(fred)).

The axiom of veridicality is equivalent to reflexivity of the accessibility relation (section 2.1.6).
4. k_acc(X, W, W).

The resolution proof is then straightforward.

The major disadvantage of this approach is that the proofs generated are not easy to understand, both since they are couched in the unnatural language of possible worlds, and since they are characteristically proofs by contradiction.

3.3 Proof techniques within modal logic

A number of techniques have been developed extending standard proof techniques to modal logics (e.g. [Geissler and Konolige, 86], [Frish and Scherl, 91].) These tend to be complicated. They are, so to speak, halfway between the method of contexts and the method of possible worlds. One generate contexts in the course of theorem proving. These may correspond to a partially specified agent, or a variable ranging over a class of agents, or one disjunctive possibility of what an agent might know, thus extending the expressivity of the language of contexts.

3.4 Proof in syntactic theories

A syntactic theory is first-order, so in principle first-order proof techniques should work. The problems are:

- The theory of quoted strings is a complex and rich one, comparable to the theory of arithmetic. Finding good proofs in such a theory is difficult and requires specialized techniques.
- Proof techniques must know to avoid paradoxical sentences.

There is very little on this problem in the literature.

3.5 Vivid Reasoning

Halpern and Vardi (1991) argue for the use of *vivid* reasoning (Levesque, 86) — that is, reasoning based on the inspection of fully elaborated, constructive models — in epistemic reasoning. Their suggestion is that, given an epistemic theory, one construct a specific model of possible worlds instantiating the theory. One can then check whether a sentence Φ is true by checking whether it holds in the model.

The problem with this approach is that with vivid reasoning generally (Davis, 91); it is very difficult to know whether Φ is true because it is a consequence of the original theory or whether it just happens to be true because the way the model was chosen.

Peripheral References

Most of the references in this paper are to papers in the larger bibliography attached. A few, however, are to AI papers not specifically dealing with epistemic reasoning. The citation for these are given here.

Ernest Davis (1991). "Lucid Representations" NYU Tech. Rep. #565.

Johan de Kleer. (1986). "An assumption-based TMS." *Artificial Intelligence,* vol. 28, pp. 127-162.

Douglas Lenat and R.V. Guha (1990). *Building Large Knowledge-Based Systems.* Addison Wesley.

Hector Levesque. (1986). "Making Believers out of Computers." *Artificial Intelligence,* vol. 30, pp. 81-108.

Drew McDermott and Gerry Sussman (1972). "The CONNIVER Reference Manual." AI Memo 259, MIT.