CHAPTER 1

INTRODUCTION

Intelligence, as exhibited by people anyway, is surely one of the most complex and mysterious phenomena that we are aware of. One striking aspect of intelligent behavior is that it is clearly conditioned by *knowledge:* for a very wide range of activities, we make decisions about what to do based on what we know (or believe) about the world, effortlessly and unconsciously. Using what we know in this way is so commonplace that we only really pay attention to it when it is not there. When we say that someone has behaved *unintelligently*, like when someone has used a lit match to see if there is any gas in a car's gas tank, what we usually mean is not that there is something that the person did not know, but rather that the person has failed to use what he or she *did* know. We might say, "You weren't thinking!" Indeed, it is *thinking* that is supposed to bring what is relevant in what we know to bear on what we are trying to do.

One definition of Artificial Intelligence (AI) is that it is the study of intelligent behavior achieved through computational means. Knowledge representation and reasoning, then, is that part of AI that is concerned with how an agent uses what it knows in deciding what to do. It is the study of thinking as a computational process. This book is an introduction to that field and in particular, to the symbolic structures it has invented for representing knowledge and to the computational processes it has devised for reasoning with those symbolic structures.

If this book is an introduction to the area, then this chapter is an introduction to the introduction. In it, we will try to address, if only briefly, some significant questions that surround the deep and challenging topics of the field: What exactly do we mean by "knowledge," by "representation," and by "reasoning," and why do we think these concepts are

useful for building AI systems? In the end, these are philosophical questions, and thorny ones at that; they bear considerable investigation by those with a more philosophical bent and can be the subject matter of whole careers. But the purpose of this chapter is not to cover in any detail what philosophers, logicians, and computer scientists have said about knowledge over the years; it is rather to glance at some of the main issues involved, and examine their bearings on Artificial Intelligence and the prospect of a machine that could think.

1.1 THE KEY CONCEPTS: KNOWLEDGE, REPRESENTATION, AND REASONING

Knowledge What is knowledge? This is a question that has been discussed by philosophers since the ancient Greeks, and it is still not totally demystified. We certainly will not attempt to be done with it here. But to get a rough sense of what knowledge is supposed to be, it is useful to look at how we talk about it informally.

First, observe that when we say something like "John knows that …," we fill in the blank with a simple declarative sentence. So we might say, "John knows that Mary will come to the party," or "John knows that Abraham Lincoln was assassinated." This suggests that, among other things, knowledge is a relation between a knower, like John, and a *proposition*, that is, the idea expressed by a simple declarative sentence, like "Mary will come to the party."

Part of the mystery surrounding knowledge is due to the nature of propositions. What can we say about them? As far as we are concerned, what matters about propositions is that they are abstract entities that can be true or false, right or wrong.[1] When we say, "John knows that p," we can just as well say, "John knows that it is true that p." Either way, to say that John knows something is to say that John has formed a judgment of some sort, and has come to realize that the world is one way and not another. In talking about this judgment, we use propositions to classify the two cases.

A similar story can be told about a sentence like "John hopes that Mary will come to the party." The same proposition is involved, but the relationship John has to it is different. Verbs like "knows," "hopes,"

[1] Strictly speaking, we might want to say that the *sentences* expressing the proposition are true or false, and that the propositions themselves are either factual or nonfactual. Further, because of linguistic features such as indexicals (that is, words whose referents change with the context in which they are uttered, such as "me" and "yesterday"), we more accurately say that it is actual tokens of sentences or their uses in specific contexts that are true or false, not the sentences themselves.

"regrets," "fears," and "doubts" all denote *propositional attitudes,* relationships between agents and propositions. In all cases, what matters about the proposition is its truth: If John hopes that Mary will come to the party, then John is hoping that the world is one way and not another, as classified by the proposition.

Of course, there are sentences involving knowledge that do not explicitly mention propositions. When we say, "John knows who Mary is taking to the party," or "John knows how to get there," we can at least imagine the implicit propositions: "John knows that Mary is taking so-and-so to the party," or "John knows that to get to the party, you go two blocks past Main Street, turn left, …," and so on. On the other hand, when we say that John has a skill, as in "John knows how to play piano," or a deep understanding of someone or something, as in "John knows Bill well," it is not so clear that any useful proposition is involved. While this is certainly challenging subject matter, we will have nothing further to say about this latter form of knowledge in this book.

A related notion that we are concerned with, however, is the concept of *belief*. The sentence "John believes that p" is clearly related to "John knows that p." We use the former when we do not wish to claim that John's judgment about the world is necessarily accurate or held for appropriate reasons. We sometimes use it when we feel that John might not be completely convinced. In fact, we have a full range of propositional attitudes, expressed by sentences like "John is absolutely certain that p," "John is confident that p," "John is of the opinion that p," "John suspects that p," and so on, that differ only in the level of conviction they attribute. For now, we will not distinguish among any of them. What matters is that they all share with knowledge a very basic idea: John takes the world to be one way and not another.

Representation The concept of *representation* is as philosophically vexing as that of knowledge. Very roughly speaking, representation is a relationship between two domains, where the first is meant to "stand for" or take the place of the second. Usually, the first domain, the representor, is more concrete, immediate, or accessible in some way than the second. For example, a drawing of a milkshake and a hamburger on a sign might stand for a less immediately visible fast food restaurant; the drawing of a circle with a plus below it might stand for the much more abstract concept of womanhood; an elected legislator might stand for his or her constituency.

The type of representor that we will be most concerned with here is the formal *symbol*, that is, a character or group of characters taken from some predetermined alphabet. The digit "7," for example, stands for the number 7, as does the group of letters "VII" and, in other

contexts, the words *sept* and *shichi*. As with all representation, it is assumed to be easier to deal with symbols (recognize them, distinguish them from each other, display them, etc.) than with what the symbols represent. In some cases, a word like "John" might stand for something quite concrete; but many words, like "love" or "truth," stand for abstractions.

Of special concern to us is when a group of formal symbols stands for a proposition: "John loves Mary" stands for the proposition that John loves Mary. Again, the symbolic English sentence is fairly concrete: It has distinguishable parts involving the three words, for example, and a recognizable syntax. The proposition, on the other hand, is abstract. It is something like a classification of all the different ways we can imagine the world to be into two groups: those where John loves Mary, and those where he does not.

Knowledge representation, then, is the field of study concerned with using formal symbols to represent a collection of propositions believed by some putative agent. As we will see, however, we do not want to insist that these symbols must represent *all* the propositions believed by the agent. There may very well be an infinite number of propositions believed, only a finite number of which are ever represented. It will be the role of *reasoning* to bridge the gap between what is represented and what is believed.

Reasoning So what is reasoning? In general, it is the formal manipulation of the symbols representing a collection of believed propositions to produce representations of new ones. It is here that we use the fact that symbols are more accessible than the propositions they represent: They must be concrete enough that we can manipulate them (move them around, take them apart, copy them, string them together) in such a way as to construct representations of new propositions.

It is useful here to draw an analogy with arithmetic. We can think of binary addition as being a certain formal manipulation: We start with symbols like "1011" and "10," for instance, and end up with "1101." The manipulation in this case is addition, because the final symbol represents the sum of the numbers represented by the initial ones. Reasoning is similar: We might start with the sentences "John loves Mary" and "Mary is coming to the party," and after a certain amount of manipulation produce the sentence, "Someone John loves is coming to the party." We would call this form of reasoning *logical inference* because the final sentence represents a logical conclusion of the propositions represented by the initial ones, as we will discuss later. According to this view (first put forward, incidentally, by the philosopher Gottfried Leibniz in the seventeenth century), reasoning is a form of calculation, not unlike arithmetic, but over symbols standing for propositions rather than numbers.

1.2 WHY KNOWLEDGE REPRESENTATION AND REASONING?

Why is knowledge even relevant at all to AI systems? The first answer that comes to mind is that it is sometimes useful to describe the behavior of sufficiently complex systems (human or otherwise) using a vocabulary involving terms like "beliefs," "desires," "goals," "intentions," "hopes," and so on.

Imagine, for example, playing a game of chess against a complex chess-playing program. In looking at one of its moves, we might say to ourselves something like this: "It moved this way because it believed its queen was vulnerable, but still wanted to attack the rook." In terms of how the chess-playing program is actually constructed, we might have said something more like, "It moved this way because evaluation procedure P using static evaluation function Q returned a value of $+7$ after an alpha-beta minimax search to depth 4." The problem is that this second description, although perhaps quite accurate, is at the wrong level of detail, and does not help us determine what chess move we should make in response. Much more useful is to understand the behavior of the program in terms of the immediate goals being pursued relative to its beliefs, long-term intentions, and so on. This is what the philosopher Daniel Dennett calls taking an *intentional stance* toward the chess-playing system.

This is not to say that an intentional stance is always appropriate. We might think of a thermostat, to take a classic example, as "knowing" that the room is too cold and "wanting" to warm it up. But this type of anthropomorphization is typically inappropriate—there is a perfectly workable electrical account of what is going on. Moreover, it can often be quite misleading to describe a system in intentional terms: Using this kind of vocabulary, we could end up fooling ourselves into thinking we are dealing with something much more sophisticated than it actually is.

But there's a more basic question: Is *this* what knowledge representation is all about? Is all the talk about knowledge just that—talk—a stance one may or may not choose to take toward a complex system?

To understand the answer, first observe that the intentional stance says nothing about what is or is not represented symbolically within a system. In the chess-playing program, the board position might be represented symbolically, say, but the goal of getting a knight out early, for instance, may not be. Such a goal might only emerge out of a complex interplay of many different aspects of the program, its evaluation functions, book move library, and so on. Yet we may still choose to describe the system as "having" this goal if this properly explains its behavior.

So what role is played by a symbolic representation? The hypothesis underlying work in knowledge representation is that we will want to construct systems that contain symbolic representations with two important properties. First is that we (from the outside) can understand

them as standing for propositions. Second is that the system is designed to behave the way that it does *because* of these symbolic representations. This is what the philosopher Brian Smith calls the *Knowledge Representation Hypothesis:*

> Any mechanically embodied intelligent process will be comprised of structural ingredients that a) we as external observers naturally take to represent a propositional account of the knowledge that the overall process exhibits, and b) independent of such external semantic attribution, play a formal but causal and essential role in engendering the behaviour that manifests that knowledge.

In other words, the Knowledge Representation Hypothesis implies that we will want to construct systems for which the intentional stance is grounded by design in symbolic representations. We will call such systems *knowledge-based systems* and the symbolic representations involved their *knowledge bases* (KBs).

1.2.1 Knowledge-Based Systems

To see what a knowledge-based system amounts to, it is helpful to look at two very simple PROLOG programs with identical behavior. Consider the first:

```
printColor(snow) :- !, write("It's white.").
printColor(grass) :- !, write("It's green.").
printColor(sky) :- !, write("It's yellow.").
printColor(X) :- write("Beats me.").
```

Here is an alternate:

```
printColor(X) :- color(X,Y), !,
       write("It's "), write(Y), write(".").
printColor(X) :- write("Beats me.").
color(snow,white).
color(sky,yellow).
color(X,Y) :- madeof(X,Z), color(Z,Y).
madeof(grass,vegetation).
color(vegetation,green).
```

Observe that both programs are able to print out the color of various items (getting the sky wrong, as it turns out). Taking an intentional stance, both might be said to "know" that the color of snow is white. The crucial point, as we will see, however, is that only the second program is designed according to the Knowledge Representation Hypothesis.

Consider the clause `color(snow,white)`, for example. This is a symbolic structure that we can understand as representing the proposition that snow is white, and moreover, we know, by virtue of knowing how the PROLOG interpreter works, that the system prints out the appropriate color of snow precisely *because* it bumps into this clause at just the right time. Remove the clause and the system would no longer do so.

There is no such clause in the first program. The one that comes closest is the first clause of the program, which says what to print when asked about snow. But we would be hard-pressed to say that this clause literally represents a belief, except perhaps a belief about what ought to be printed.

So what makes a system knowledge-based, as far as we are concerned, is not the use of a logical formalism (like PROLOG), or the fact that it is complex enough to merit an intentional description involving knowledge, or the fact that what it believes is true; rather, it is the presence of a knowledge base, a collection of symbolic structures representing what it believes and reasons with during the operation of the system.

Much (though not all) of AI involves building systems that are knowledge-based in this way, that is, systems whose ability derives in part from reasoning over explicitly represented knowledge. So-called expert systems are a very clear case, but we also find KBs in the areas of language understanding, planning, diagnosis, and learning. Many AI systems are also knowledge-based to a somewhat lesser extent—some game-playing and high-level vision systems, for instance. Finally, some AI systems are not knowledge-based at all: Low-level speech, vision, and motor-control systems typically encode what they need to know directly in the programs themselves.

How much of intelligent behavior needs to be knowledge-based in this sense? This remains an open research question. Perhaps the most serious challenge to the Knowledge Representation Hypothesis is the "connectionist" methodology, which attempts to avoid any kind of symbolic representation and reasoning, and instead advocates computing with networks of weighted links between artificial "neurons."

1.2.2 Why Knowledge Representation?

An obvious question arises when we start thinking about the two PROLOG programs of the previous section: What advantage, if any, does the knowledge-based one have? Wouldn't it be better to "compile out" the KB and distribute this knowledge to the procedures that need it, as we did in the first program? The performance of the system would certainly be better. It can only slow a system down to have to look up facts in a KB and reason with them at runtime in order to decide what actions to take. Indeed, advocates within AI of what has been called procedural knowledge take pretty much this point of view.

When we think about the various skills we have, such as riding a bicycle or playing a piano, it certainly *feels* like we do not reason about the various actions to take (shifting our weight or moving our fingers); it seems much more like we just know what to do, and do it. In fact, if we try to think about what we are doing, we end up making a mess of it. Perhaps (the argument goes), this applies to most of our activities: making a meal, getting a job, staying alive, and so on.

Of course, when we first learn these skills, the case is not so clear: It seems like we need to think deliberately about what we are doing, even riding a bicycle. The philosopher Hubert Dreyfus first observed this paradox of "expert systems." These systems are claimed to be superior precisely because they are knowledge-based, that is, they reason over explicitly represented knowledge. But novices are the ones who think and reason, claims Dreyfus. Experts do not; they learn to recognize and to react. The difference between a chess master and a chess novice is that the novice needs to figure out what is happening and what to do, but the master just "sees" it. For this reason (among others), Dreyfus believes that the development of knowledge-based systems is completely wrongheaded if it is attempting to duplicate human-level intelligent behavior.

So why even consider knowledge-based systems? Unfortunately, no definitive answer can yet be given. We suspect, however, that the answer will emerge in our desire to build a system that can deal with a set of tasks that is *open-ended*. For any fixed set of tasks it might work to "compile out" what the system needs to know, but if the set of tasks is not determined in advance, the strategy will not work. The ability to make behavior depend on explicitly represented knowledge seems to pay off when we cannot specify in advance how that knowledge will be used.

A good example of this is what happens when we read a book. Suppose we are reading about South American geography. When we find out for the first time that approximately half of the population of Peru lives in the Andes, we are in no position to distribute this piece of knowledge to the various routines that might eventually require it. Instead, it seems pretty clear that we are able to assimilate the fact in declarative form for a very wide variety of potential uses. This is a prototypical case of a knowledge-based approach.

Further, from a system-design point of view, the knowledge-based approach exhibited by the second PROLOG program seems to have a number of desirable features:

- We can add new tasks and easily make them depend on previous knowledge. In our PROLOG program example, we can add the task of enumerating all objects of a given color, or even of painting a picture, by making use of the already specified KB to determine the colors.

- We can extend the existing behavior by adding new beliefs. For example, by adding a clause saying that canaries are yellow, we automatically propagate this information to any routine that needs it.
- We can debug faulty behavior by locating the erroneous beliefs of the system. In the PROLOG example, by changing the clause for the color of the sky, we automatically correct any routine that uses color information.
- We can concisely explain and justify the behavior of the system. Why did the program say that grass was green? It was because it believed that grass is a form of vegetation and that vegetation is green. We are justified in saying "because" here, since if we removed either of the two relevant clauses the behavior would indeed change.

Overall, then, the hallmark of a knowledge-based system is that by design it has the ability to be *told* facts about its world and adjust its behavior correspondingly.

This ability to have some of our actions depend on what we believe is what the cognitive scientist Zenon Pylyshyn has called *cognitive penetrability*. Consider, for example, responding to a fire alarm. The normal response is to get up and leave the building, but we would not do so if we happened to believe that the alarm was being tested. There are any number of ways we might come to this belief, but they all lead to the same effect. Our response to a fire alarm is cognitively penetrable because it is conditioned on what we can be made to believe. On the other hand, something like a blinking reflex as an object approaches your eye does not appear to be cognitively penetrable: Even if you strongly believe the object will not touch you, you still blink.

1.2.3 Why Reasoning?

To see the motivation behind reasoning in a knowledge-based system, it suffices to observe that we would like action to depend on what the system *believes* about the world, as opposed to just what the system has *explicitly represented*. In the second PROLOG example, there was no clause representing the belief that the color of grass was green, but we still wanted the system to know this. In general, much of what we expect to put in a KB will involve quite general facts, which will then need to be applied to particular situations.

For example, we might represent the following two facts explicitly:

1. Patient x is allergic to medication m.
2. Anyone allergic to medication m is also allergic to medication m'.

In trying to decide if it is appropriate to prescribe medication m' for patient x, neither represented fact answers the question. Together, however, they paint a picture of a world where x is allergic to m', and this, together with other represented facts about allergies, might be sufficient to rule out the medication. We do not want to condition behavior only on the represented facts that we are able to *retrieve*, like in a database system. The beliefs of the system must go beyond these.

But beyond them to where? There is, as it turns out, a simple answer to this question, but one which, as we will discuss many times in subsequent chapters, is not always practical. The simple answer is that the system should believe p if, according to the beliefs it has represented, the world it is imagining is one where p is true. In the example, facts (1) and (2) are both represented. If we now imagine what the world would be like if (1) and (2) were both true, then this is a world where

3. Patient x is allergic to medication m'

is also true, even though this fact is only implicitly represented.

This is the concept of *logical entailment:* We say that the propositions represented by a set of sentences S entail the proposition represented by a sentence p when the truth of p is implicit in the truth of the sentences in S. In other words, if the world is such that every element of S comes out true, then p does as well. All that we require to get some notion of entailment is a language with an account of what it means for a sentence to be true or false. As we argued, if our representation language is to represent knowledge at all, it must come with such an account (again, to know p is to take p to be true). So any knowledge representation language, whatever other features it may have, whatever syntactic form it may take, whatever reasoning procedures we may define over it, ought to have a well-defined notion of entailment.

A simple answer to what beliefs a knowledge-based system should exhibit, then, is that it should believe all and only the entailments of what it has explicitly represented. The job of reasoning, then, according to this account, is to compute the entailments of a KB.

What makes this account simplistic is that there are often quite good reasons not to calculate entailments. For one thing, it can be too *difficult* computationally to decide which sentences are entailed by the kind of KB we will want to use. Any procedure that always gives us answers in a reasonable amount of time will occasionally either miss some entailments or return some incorrect answers. In the former case, the reasoning process is said to be *logically incomplete;* in the latter case, the reasoning is said to be *logically unsound*.

But there are also conceptual reasons why we might consider unsound or incomplete reasoning. For example, suppose p is not entailed by a KB, but is a reasonable guess, given what is represented. We might still want

to believe that p is true. To use a classic example, suppose all I know about an individual Tweety is that she is a bird. I might have a number of facts about birds in the KB, but likely none of them would *entail* that Tweety flies. After all, Tweety might turn out to be an ostrich. Nonetheless, it is a reasonable assumption that Tweety flies. This is logically unsound reasoning since we can imagine a world where everything in the KB is true but where Tweety does not fly.

Alternately, a knowledge-based system might come to believe a collection of facts from various sources that, taken together, cannot all be true. In this case, it would be inappropriate to do logically complete reasoning, because then *every* sentence would be believed: Since there are no worlds where the KB is true, every sentence p will be trivially true in all worlds where the KB is true. An incomplete form of reasoning would clearly be more useful here until the contradictions were dealt with, if ever.

Despite all this, it remains the case that the simplistic answer is by far the best starting point for thinking about reasoning, even if we intend to diverge from it. So while it would be a mistake to *identify* reasoning in a knowledge-based system with logically sound and complete inference, it is the right place to begin.

1.3 THE ROLE OF LOGIC

The reason *logic* is relevant to knowledge representation and reasoning is simply that, at least according to one view, logic *is* the study of entailment relations—languages, truth conditions, and rules of inference. Not surprisingly, we will borrow heavily from the tools and techniques of formal symbolic logic. Specifically, we will use as our first knowledge representation language a very popular logical language, that of the predicate calculus, or as it is sometimes called, the language of first-order logic (FOL). This language was invented by the philosopher Gottlob Frege at the turn of the twentieth century for the formalization of mathematical inference, but has been co-opted by the AI community for knowledge representation purposes.

It must be stressed, however, that FOL itself is also just a starting point. We will have good reason in what follows to consider subsets and supersets of FOL, as well as knowledge representation languages quite different in form and meaning. Just as we are not committed to understanding reasoning as the computation of entailments, even when we do so we are not committed to any particular language. Indeed, as we shall see, certain representation languages suggest forms of reasoning that go well beyond whatever connections they may have ever had with logic.

Where logic really does pay off from a knowledge representation perspective is at what Allen Newell has called the *knowledge level*.

The idea is that we can understand a knowledge-based system at two different levels (at least). At the knowledge level, we ask questions concerning the representation language and its semantics. At the *symbol level*, on the other hand, we ask questions concerning the computational aspects. There are clearly issues of adequacy at each level. At the knowledge level, we deal with the expressive adequacy of a representation language and the characteristics of its entailment relation, including its intrinsic computational complexity; at the symbol level, we ask questions about the computational architecture and the properties of the data structures and reasoning procedures, including their algorithmic complexity.

The tools of formal symbolic logic seem ideally suited for a knowledge-level analysis of a knowledge-based system. In the next chapter we begin such an analysis using the language of first-order logic, putting aside for now all computational concerns.

1.4 BIBLIOGRAPHIC NOTES

Much of the material in this chapter is shared with the first chapter of Levesque and Lakemeyer [244].

The area of knowledge representation and reasoning is one of the most active areas of AI research, dominating many of the general AI conferences, as well as having a conference of its own. Some of the more influential early papers (that is, before 1985) may be found in a collection of readings by Brachman and Levesque [47]. An overview of the field as a whole at about that time is presented by Levesque [240]. For an even earlier view of the state of the art during a time of great progress and intellectual debate, see the *SIGART Newsletter, Special Issue on Knowledge Representation* [50].

Sowa [394] also discusses general aspects of knowledge representation and reasoning and, in particular, how various proposals relate to formal logic and to a specific representation formalism he invented called *conceptual graphs* [392]. Other books devoted to the general topic include those by Davis [86], Ringland and Duce [354], and Reichgelt [338].

Most textbooks on Artificial Intelligence contain material on knowledge representation and reasoning. A general AI textbook by Genesereth and Nilsson [153] emphasizes (first-order) logic and its use in Artificial Intelligence. Some popular textbooks that spend considerable time on knowledge representation include those by Dean et al. [98], Ginsberg [160], Luger [260], Nilsson [311], Poole et al. [332], Rich and Knight [352], Russell and Norvig [359], and Winston [428].

Leibniz was one of the first to see logic as providing a unifying basis for all mathematics and science. For a summary of his views about thinking

as a form of calculation, see [118], vol. 3, p. 422. Frege [135] developed propositional logic and introduced the idea of quantification and a notation for expressing logical concepts.

The distinction between knowledge and belief is discussed at length in the philosophical literature. Gettier's article [156] represents one of the landmarks in this area, presenting arguments against the common view of knowledge as true, justified belief. A collection of papers by Pappas and Swain [314] contains responses to Gettier's work.

Dennett's intentional stance is discussed in [102]. The notion of cognitive penetrability is addressed by Pylyshyn [334]. The knowledge representation hypothesis is due to Smith [390]. An alternate opinion on the general question of what constitutes a knowledge representation is expressed by Davis et al. [89]. The original conception of the knowledge level is due to Newell [305]. The knowledge level as applied directly to the knowledge representation and reasoning enterprise was addressed by Levesque [238] and by Brachman et al. [52].

Despite the centrality of knowledge representation and reasoning to AI, there are alternate views. Some authors have claimed that human-level reasoning is not achievable via purely computational means. These include Dreyfus [113], Searle [373, 374], and Penrose [327] (see also the collection by Boden [33]). Others suggest that intelligence derives from computational mechanisms that are not as directly representational as those discussed in this book. Among these are the so-called connectionists, such as Kohonen [220] and Smolensky [391].

1.5 EXERCISES

These exercises are all taken from [244].

1. Consider a task requiring knowledge, like baking a cake. Examine a recipe and state what needs to be known to follow the recipe.

2. In considering the distinction between knowledge and belief in this book, we take the view that belief is fundamental and knowledge is simply belief where the outside world happens to be cooperating (the belief is true, is arrived at by appropriate means, is held for the right reasons, and so on). Describe an interpretation of the terms where knowledge is taken to be basic and belief is understood in terms of it.

3. Explain in what sense reacting to a loud noise is and is not cognitively penetrable.

4. It has become fashionable to attempt to achieve intelligent behavior in AI systems without using propositional representations. Speculate on what such a system should do when reading a book on South American geography.

5. Describe some ways in which the firsthand knowledge we have of some topic goes beyond what we are able to write down in a language. What accounts for our inability to express this knowledge?