CHAPTER 2

THE LANGUAGE OF FIRST-ORDER LOGIC

Before any system aspiring to intelligence can even begin to reason, learn, plan, or explain its behavior, it must be able to formulate the ideas involved. You will not be able to learn something about the world around you, for example, if it is beyond you to even express what that thing is. So we need to start with a *language* of some sort, in terms of which knowledge can be formulated. In this chapter, we will examine in detail one specific language that can be used for this purpose: the language of first-order logic. FOL is not the only choice, but is a simple and convenient one to begin with.

2.1 INTRODUCTION

What does it mean to "have" a language? Once we have a set of words or a set of symbols of some sort, what more is needed? As far as we are concerned, there are three things:

1. *syntax:* we need to specify which groups of symbols, arranged in what way, are to be considered properly formed. In English, for example, the string of words "the cat my mother loves" is a well-formed noun phrase, but "the my loves mother cat" is not. For knowledge representation, we need to be especially clear about which of the well-formed strings are the *sentences* of the language, since these are what express propositions.

2. *semantics:* we need to specify what the well-formed expressions are supposed to mean. Some well-formed expressions like “the hard-nosed decimal holiday” might not mean anything. For sentences, we need to be clear about what idea about the world is being expressed. Without such an account, we cannot expect to say what believing one of them amounts to.

3. *pragmatics:* we need to specify how the meaningful expressions in the language are to be used. In English, for example, “There is someone right behind you” could be used as a warning to be careful in some contexts and a request to move in others. For knowledge representation, this involves how we use the meaningful sentences of a representation language as part of a knowledge base from which inferences will be drawn.

These three aspects apply mainly to declarative languages, the sort we use to represent knowledge. Other languages will have other aspects not discussed here, for example, what the words sound like (for spoken languages), or what actions are being called for (for imperative languages).

We now turn our attention to the specification of FOL.

2.2 THE SYNTAX

In FOL, there are two sorts of symbols: the *logical* ones and the *nonlogical* ones. Intuitively, the logical symbols are those that have a fixed meaning or use in the language. There are three sorts of logical symbols:

1. *punctuation:* “(”, “)”, and “.”.

2. *connectives:* “$\neg$,” “$\wedge$,” “$\vee$,” “$\exists$,” “$\forall$,” and “$=$.” Note the usual interpretation of these logical symbols: $\neg$ is logical negation, $\wedge$ is logical conjunction (“and”), $\vee$ is logical disjunction (“or”), $\exists$ means “there exists...,” $\forall$ means “for all...,” and $=$ is logical equality. $\forall$ and $\exists$ are called *quantifiers*.

3. *variables:* an infinite supply of symbols, which we will denote here using x, y, and z, sometimes with subscripts and superscripts.

The nonlogical symbols are those that have an application-dependent meaning or use. In FOL, there are two sorts of nonlogical symbols:

1. *function symbols:* an infinite supply of symbols, which we will write in uncapitalized mixed case, e.g., `bestFriend`, and which we will

denote more generally using a, b, c, f, g, and h, with subscripts and superscripts.

2. *predicate symbols:* an infinite supply of symbols, which we will write in capitalized mixed case, e.g., `OlderThan`, and which we will denote more generally using P, Q, and R, with subscripts and superscripts.

One distinguishing feature of nonlogical symbols is that each one is assumed to have an *arity*, that is, a nonnegative integer indicating how many "arguments" it takes. (This number is used in the syntax of the language.) It is assumed that there is an infinite supply of function and predicate symbols of each arity. By convention, a, b, and c are only used for function symbols of arity 0, which are called *constants*, while g and h are only used for function symbols of nonzero arity. Predicate symbols of arity 0 are sometimes called *propositional symbols*.

If you think of the logical symbols as the reserved keywords of a programming language, then nonlogical symbols are like its identifiers. For example, we might have "`Dog`" as a predicate symbol of arity 1, "`OlderThan`" as a predicate symbol of arity 2, "`bestFriend`" as a function symbol of arity 1, and "`johnSmith`" as a constant. Note that we are treating "=" not as a predicate symbol, but as a special logical symbol (unlike the way that it is handled in some logic textbooks).

There are two types of legal syntactic expressions in FOL: *terms* and *formulas*. Intuitively, a term will be used to refer to something in the world, and a formula will be used to express a proposition. The set of terms of FOL is the least set satisfying these conditions:

- every variable is a term;
- if $t_1, \ldots, t_n$ are terms, and f is a function symbol of arity n, then $f(t_1, \ldots, t_n)$ is a term.

The set of formulas of FOL is the least set satisfying these constraints:

- if $t_1, \ldots, t_n$ are terms, and P is a predicate symbol of arity n, then $P(t_1, \ldots, t_n)$ is a formula;
- if t_1 and t_2 are terms, then $t_1 = t_2$ is a formula;
- if α and β are formulas, and x is a variable, then $\neg\alpha$, $(\alpha \wedge \beta)$, $(\alpha \vee \beta)$, $\forall x. \alpha$, and $\exists x. \alpha$ are formulas.

Formulas of the first two types (containing no other simpler formulas) are called *atomic formulas* or *atoms*.

At this point, it is useful to introduce some notational abbreviations and conventions. First of all, we will add or omit matched parentheses and periods freely, and also use square and curly brackets to improve readability. In the case of predicates or function symbols of arity 0, we will usually omit the parentheses since there are no arguments to enclose. We will also sometimes reduce the number of parentheses by assuming that $\wedge$ has higher precedence than $\vee$ (the way $\times$ has higher precedence than $+$).

By the *propositional subset* of FOL, we mean the language with no terms, no quantifiers, and where only propositional symbols are used. So, for example,

$$(P \wedge \neg(Q \vee R)),$$

where P, Q, and R are propositional symbols, would be a formula in this subset.

We also use the following abbreviations:

- $(\alpha \supset \beta)$ for $(\neg\alpha \vee \beta)$;
- $(\alpha \equiv \beta)$ for $((\alpha \supset \beta) \wedge (\beta \supset \alpha))$.

We also need to discuss the scope of quantifiers. We say that a variable occurrence is *bound* in a formula if it lies within the scope of a quantifier, and *free* otherwise. That is, x appears bound if it appears in a subformula $\forall x.\, \alpha$ or $\exists x.\, \alpha$ of the formula. So, for example, in a formula like

$$\forall y.\, P(x) \wedge \exists x[P(y) \vee Q(x)],$$

the first occurrence of the variable x is free, and the final two occurrences of x are bound; both occurrences of y are bound.[1] If x is a variable, t is a term, and α is a formula, we use the notation α^x_t to stand for the formula that results from replacing all free occurrences of x in α by t. If $\vec{x}$ is a sequence of variables, $\vec{c}$ is a sequence of constants of the same length, and α is a formula whose free variables are among those in $\vec{x}$, then $\alpha[\vec{x}]$ means α itself and $\alpha[\vec{c}]$ means α with each free x_i replaced by the corresponding c_i.

Finally, a *sentence* of FOL is any formula without free variables. The sentences of FOL are what we use to represent knowledge, and the rest is merely supporting syntactic machinery.

2.3 THE SEMANTICS

As noted in Section 2.1, the concern of semantics is to explain what the expressions of a language mean. As far as we are concerned, this involves

[1]In some textbooks, the occurrence of the variable just after the quantifier is considered neither free nor bound.

specifying what claim a sentence of FOL makes about the world, so that we can understand what believing it amounts to.

Unfortunately, there is a bit of a problem here. We cannot realistically expect to specify once and for all what a sentence of FOL means, for the simple reason that the nonlogical symbols are used in an application-dependent way. I might use the constant "john" to mean one individual, and you might use it to mean another. So there's no way we can possibly agree on what the sentence "Happy(john)" claims about the world, even if we were to agree on what "Happy" means.

Here is what we can agree to: The sentence "Happy(john)" claims that the individual named by "john" (whoever that might be) has the property named by "Happy" (whatever that might be). In other words, we can agree once and for all on how the meaning of the sentence derives from the interpretation of the nonlogical symbols involved. Of course, what we have in mind for these nonlogical symbols can be quite complex and hard to make precise. For example, our list of nonlogical symbols might include terms like

DemocraticCountry, IsABetterJudgeOfCharacterThan,
favoriteIceCreamFlavorOf, puddleOfwater27,

and the like. We should not (and cannot) expect the semantic specification of FOL to tell us precisely what terms like these mean. What we are after, then, is a clear specification of the meaning of sentences *as a function of the interpretation of the predicate and function symbols*.

To get to such a specification, we take the following (simplistic) view of what the world could be like:

1. There are objects in the world.

2. For any predicate P of arity 1, some of the objects will satisfy P and some will not. An *interpretation* of P settles the question, deciding for each object whether it has or does not have the property in question. (Borderline cases are ruled in separate interpretations: In one, it has the property; in another, it does not.) Predicates of other arities are handled similarly. For example, an interpretation of a predicate of arity 3 decides on which triples of objects stand in the corresponding ternary relation. Similarly, a function symbol of arity 3 is interpreted as a mapping from triples of objects to objects.

3. No other aspects of the world matter.

The assumption made in FOL is that this is all you need to say regarding the meaning of the nonlogical symbols, and hence the meaning of all sentences.

For example, we might imagine that there are objects that include people, countries, and flavors of ice cream. The meaning of "DemocraticCountry" in some interpretation will be no more and no less than those objects that are countries that we consider to be democratic. We may disagree on which those are, of course, but then we are simply talking about different interpretations. Similarly, the meaning of "favoriteIceCreamFlavorOf" would be a specific mapping from people to flavors of ice cream (and from nonpeople to some other arbitrarily chosen object, say). Note that as far as FOL is concerned, we do not try to say what "DemocraticCountry" means the way a dictionary would, in terms of free elections, representative governments, majority rule, and so on; all we need to say is which objects are and are not democratic countries. This is clearly a simplifying assumption, and other languages would handle the terms differently.

2.3.1 Interpretations

Meanings are typically captured by specific interpretations, and we can now be precise about them. An *interpretation* $\Im$ in FOL is a pair $\langle \mathcal{D}, \mathcal{I} \rangle$, where $\mathcal{D}$ is any nonempty set of objects, called the *domain* of the interpretation, and $\mathcal{I}$ is a mapping, called the *interpretation mapping,* from the nonlogical symbols to functions and relations over $\mathcal{D}$, as described later.

It is important to stress that an interpretation need not only involve mathematical objects. $\mathcal{D}$ can be *any* set, including people, garages, numbers, sentences, fairness, unicorns, chunks of peanut butter, situations, and the universe, among other things.

The interpretation mapping $\mathcal{I}$ will assign meaning to the predicate symbols as follows: To every predicate symbol P of arity n, $\mathcal{I}[P]$ is an n-ary relation over $\mathcal{D}$, that is,

$$\mathcal{I}[P] \subseteq \underbrace{\mathcal{D} \times \cdots \times \mathcal{D}}_{n \text{ times}}.$$

So, for example, consider a unary predicate symbol Dog. Here, $\mathcal{I}$[Dog] would be some subset of $\mathcal{D}$, presumably the set of dogs in that interpretation. Similarly, $\mathcal{I}$[OlderThan] would be some subset of $\mathcal{D} \times \mathcal{D}$, presumably the set of pairs of objects in $\mathcal{D}$ where the first element of the pair is older than the second.

The interpretation mapping $\mathcal{I}$ will assign meaning to the function symbols as follows: To every function symbol f of arity n, $\mathcal{I}[f]$ is an n-ary function over $\mathcal{D}$, that is,[2]

$$\mathcal{I}[f] \in [\underbrace{\mathcal{D} \times \cdots \times \mathcal{D}}_{n \text{ times}} \rightarrow \mathcal{D}].$$

[2]Here and subsequently, mathematical functions are taken to be total.

So, for example, $\mathcal{I}[\mathsf{bestFriend}]$ would be some function $[\mathcal{D} \rightarrow \mathcal{D}]$, presumably the function that maps a person to his or her best friend (and does something reasonable with nonpersons). Similarly, $\mathcal{I}[\mathsf{johnSmith}]$ would be some element of $\mathcal{D}$, presumably somebody called John Smith.

It is sometimes useful to think of the interpretation of predicates in terms of their characteristic functions. In this case, when P is a predicate of arity n, we view $\mathcal{I}[P]$ as an n-ary function to $\{0, 1\}$:

$$\mathcal{I}[P] \in [\mathcal{D} \times \cdots \times \mathcal{D} \rightarrow \{0, 1\}].$$

The relationship between the two specifications is that a tuple of objects is considered to be in the relation over $\mathcal{D}$ if and only if the characteristic function over those objects has value 1. This characteristic function also allows us to see more clearly how predicates of arity 0 (i.e., the propositional symbols) are handled. In this case, $\mathcal{I}[P]$ will be either 0 or 1. We can think of the first one as meaning "false" and the second "true." For the propositional subset of FOL, then, we can ignore $\mathcal{D}$ completely, and think of an interpretation as simply being a mapping, $\mathcal{I}$, from the propositional symbols to either 0 or 1.

2.3.2 Denotation

Given an interpretation $\Im = \langle \mathcal{D}, \mathcal{I} \rangle$, we can specify which elements of $\mathcal{D}$ are denoted by any variable-free term of FOL. For example, to find the object denoted by the term "bestFriend(johnSmith)" in $\Im$, we use $\mathcal{I}$ to get hold of the function denoted by "bestFriend," and then we apply that function to the element of $\mathcal{D}$ denoted by "johnSmith," producing some other element of $\mathcal{D}$. To deal with terms including variables, we also need to start with a *variable assignment* over $\mathcal{D}$, that is, a mapping from the variables of FOL to the elements of $\mathcal{D}$. So if μ is a variable assignment and x is a variable, $\mu[x]$ will be some element of the domain.

Formally, given an interpretation $\Im$ and a variable assignment μ, the *denotation* of term t, written $\|t\|_{\Im,\mu}$, is defined by these rules:

1. if x is a variable, then $\|x\|_{\Im,\mu} = \mu[x]$;
2. if $t_1, \ldots, t_n$ are terms, and f is a function symbol of arity n, then

$$\|f(t_1, \ldots, t_n)\|_{\Im,\mu} = \mathcal{F}(d_1, \ldots, d_n)$$

where $\mathcal{F} = \mathcal{I}[f]$, and $d_i = \|t_i\|_{\Im,\mu}$.

Observe that according to these recursive rules, $\|t\|_{\Im,\mu}$ is always an element of $\mathcal{D}$.

2.3.3 Satisfaction and Models

Given an interpretation $\Im = \langle \mathcal{D}, \mathcal{I} \rangle$ and the $\| \cdot \|_{\Im,\mu}$ relation just defined, we can now specify which sentences of FOL are true and which are false according to this interpretation. For example, "Dog(bestFriend(johnSmith))" is true in $\Im$ if and only if the following holds: If we use $\mathcal{I}$ to get hold of the subset of $\mathcal{D}$ denoted by "Dog" and the object denoted by "bestFriend(johnSmith)," then that object is in the set. To deal with formulas containing free variables, we again use a variable assignment, as shown earlier.

More formally, given an interpretation $\Im$ and variable assignment μ, we say that the formula α is *satisfied* in $\Im$, written $\Im, \mu \models \alpha$, according to these rules:

Assume that $t_1, \ldots, t_n$ are terms, P is a predicate of arity n, α and β are formulas, and x is a variable.

1. $\Im, \mu \models P(t_1, \ldots, t_n)$ iff $\langle d_1, \ldots, d_n \rangle \in \mathcal{P}$, where $\mathcal{P} = \mathcal{I}[P]$, and $d_i = \|t_i\|_{\Im,\mu}$;
2. $\Im, \mu \models t_1 = t_2$ iff $\|t_1\|_{\Im,\mu}$ and $\|t_2\|_{\Im,\mu}$ are the same element of $\mathcal{D}$;
3. $\Im, \mu \models \neg\alpha$ iff it is not the case that $\Im, \mu \models \alpha$;
4. $\Im, \mu \models (\alpha \wedge \beta)$ iff $\Im, \mu \models \alpha$ and $\Im, \mu \models \beta$;
5. $\Im, \mu \models (\alpha \vee \beta)$ iff $\Im, \mu \models \alpha$ or $\Im, \mu \models \beta$ (or both);
6. $\Im, \mu \models \exists x. \alpha$ iff $\Im, \mu' \models \alpha$, for some variable assignment μ' that differs from μ on at most x;
7. $\Im, \mu \models \forall x. \alpha$ iff $\Im, \mu' \models \alpha$, for every variable assignment μ' that differs from μ on at most x.

When the formula α is a sentence, it is easy to see that satisfaction does not depend on the given variable assignment (recall that sentences do not have free variables). In this case, we write $\Im \models \alpha$ and say that α *is true* in the interpretation $\Im$, or that α *is false* otherwise. In the case of the propositional subset of FOL, it is sometimes convenient to write $\mathcal{I}[\alpha] = 1$ or $\mathcal{I}[\alpha] = 0$ according to whether $\mathcal{I} \models \alpha$ or not. We will also use the notation $\Im \models S$, where S is a set of sentences, to mean that all of the sentences in S are true in $\Im$. We say in this case that $\Im$ is a *logical model* of S.

2.4 THE PRAGMATICS

The semantic rules of interpretation tell us how to understand precisely the meaning of any term or formula of FOL in terms of a domain and an interpretation for the nonlogical symbols over that domain. What is less clear, perhaps, is why anyone interested in knowledge representation should care about this. How are we supposed to use this language to represent knowledge? How is a knowledge-based system supposed to reason about concepts like "DemocraticCountry" or even "Dog" unless it is somehow given the intended interpretation to start with? How could we possibly "give" a system an interpretation, which could involve (perhaps infinite) sets of honest-to-goodness objects like countries or animals?

2.4.1 Logical Consequence

To answer these questions, we first turn to the notion of logical consequence. Observe that although the semantic rules of interpretation depend on the interpretation of the nonlogical symbols, there are connections among sentences of FOL that do not depend on the meaning of those symbols.

For example, let α and β be any two sentences of FOL, and let γ be the sentence $\neg(\beta \wedge \neg\alpha)$. Now suppose that $\Im$ is any interpretation where α is true. Then, by using the earlier rules we can see that γ must be also true under this interpretation. This does not depend on how we understand any of the nonlogical symbols in α or β. As long as α comes out true, γ will as well. In a sense, the truth of γ is implicit in the truth of α. We say in this case that γ is a logical consequence of α.

More precisely, let S be a set of sentences, and α any sentence. We say that α is a *logical consequence* of S, or that S *logically entails* α, which we write $S \models \alpha$, if and only if, for *every* interpretation $\Im$, if $\Im \models S$ then $\Im \models \alpha$. In other words, every model of S satisfies α. Yet another way of saying this is that there is no interpretation $\Im$ where $\Im \models S \cup \{\neg\alpha\}$. We say in this case that the set $S \cup \{\neg\alpha\}$ is *unsatisfiable*.

As a special case of this definition, we say that a sentence α is logically *valid*, which we write $\models \alpha$, when it is a logical consequence of the empty set. In other words, α is valid if and only if, for every interpretation $\Im$, it is the case that $\Im \models \alpha$ or, in still other words, if and only if the set $\{\neg\alpha\}$ is unsatisfiable.

It is not too hard to see that not only is validity a special case of entailment but entailment when the set is finite also reduces to validity: If $S = \{\alpha_1, \ldots, \alpha_n\}$, then $S \models \alpha$ if and only if the sentence $[(\alpha_1 \wedge \cdots \wedge \alpha_n) \supset \alpha]$ is valid.

2.4.2 Why We Care

Now let us reexamine the connection between knowledge-based systems and logical entailment, since this is at the heart of the knowledge representation enterprise.

What we are after is a system that can reason. Given something like the fact that Fido is a dog, it should be able to conclude that Fido is also a mammal, a carnivore, and so on. In other words, we are imagining a system that can be told or learn a sentence like "Dog(fido)" that is true in some user-intended interpretation, and that can then come to believe other sentences true in that interpretation.

A knowledge-based system will not and cannot have access to the interpretation of the nonlogical symbols itself. As we noted, this could involve infinite sets of real objects quite outside the reach of any computer system. So a knowledge-based system will not be able to decide what to believe by using the rules of Section 2.3.3 to evaluate the truth or falsity of sentences in this intended interpretation. Nor can it simply be "given" the set of sentences true in that interpretation as beliefs, because, among other things, there will be an infinite number of such sentences.

However, suppose a set of sentences S entails a sentence α. Then we do know that whatever the intended interpretation is, if S happens to be true in that interpretation, then so must be α. If the user imagines the world satisfying S according to his or her understanding of the nonlogical symbols, then it satisfies α as well. Other nonentailed sentences may or may not be true, but a knowledge-based system can safely conclude that the entailed ones are. If we tell our system that "Dog(fido)" is true in the intended interpretation, it can safely conclude any other sentence that is logically entailed, such as "$\neg\neg$Dog(fido)" and "(Dog(fido) $\vee$ Happy(john))," without knowing anything else about that interpretation.

But who cares? These conclusions are logically unassailable of course, but not the sort of reasoning we would likely be interested in. In a sense, logical entailment gets us nowhere, since all we are doing is finding sentences that are already implicit in what we were told.

As we said, what we really want is a system that can go from "Dog(fido)" to conclusions like "Mammal(fido)," and on from there to other interesting animal properties. This is no longer logical entailment, however: There are interpretations where "Dog(fido)" is true and "Mammal(fido)" is false. For example, let $\Im = \langle \mathcal{D}, \mathcal{I} \rangle$ be an interpretation where for some dog d, $\mathcal{D} = \{d\}$, for every predicate P other than "Dog," $\mathcal{I}[P] = \{\}$, where $\mathcal{I}[\mathsf{Dog}] = \{d\}$, and where for every function symbol f, $\mathcal{I}[f](d, \ldots, d) = d$. This is an interpretation where the one and only dog is not a mammal. So the connection between the two sentences is not a strictly logical one.

To get the desired connection between dogs and mammals, we need to include within the set of sentences S a statement connecting the nonlogical symbols involved. In this case, the sentence

$$\forall x.\, \mathsf{Dog}(x) \supset \mathsf{Mammal}(x)$$

should be an element of *S*. With this universal and "Dog(fido)" in *S*, we do get "Mammal(fido)" as a logical consequence. We will examine claims of logical consequence like this in more detail later, but for now note that by including this universal as one of the premises in *S*, we rule out interpretations like the one where the set of dogs is not a subset of the set of mammals. If we then continue to add more and more sentences like this to *S*, we will rule out more and more unintended interpretations, and in the end, logical consequence itself will start to behave much more like "truth in the intended interpretation."

This, then, is the fundamental tenet of knowledge representation:

> Reasoning based on logical consequence only allows safe, logically guaranteed conclusions to be drawn. However, by starting with a rich collection of sentences as given premises, including not only facts about particulars of the intended application but also those expressing connections among the nonlogical symbols involved, the set of entailed conclusions becomes a much richer set, closer to the set of sentences true in the intended interpretation. Calculating these entailments thus becomes more like the form of reasoning we would expect of someone who understood the meaning of the terms involved.

In a sense, this is all there is to knowledge representation and reasoning; the rest is just details.

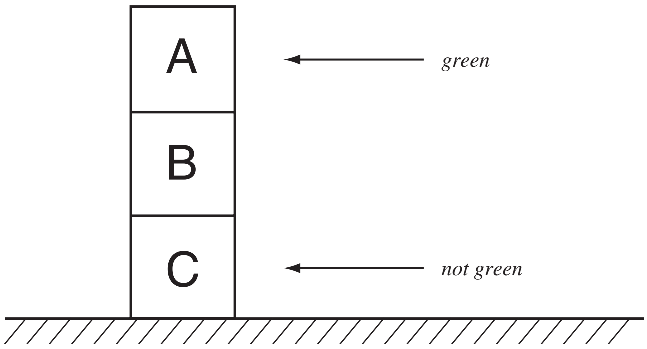
2.5 EXPLICIT AND IMPLICIT BELIEF

The collection of sentences given as premises to be used as the basis for calculating entailments is what we called a *knowledge base* in Chapter 1—in our case, a finite set of sentences in the language of FOL. The role of a knowledge representation system, as discussed before, is to calculate entailments of this KB. We can think of the KB itself as the beliefs of the system that are *explicitly* given, and the entailments of that KB as the beliefs that are only *implicitly* given.

Just because we are imagining a "rich" collection of sentences in the KB, including the intended connections among the nonlogical symbols, we should not be misled into thinking that we have done all the work and there is no real reasoning left to do. As we will see in the following example, it is often nontrivial to move from explicit to implicit beliefs.

2.5.1 An Example

Consider the "blocks-world" example illustrated in Figure 2.1. Suppose we have three colored blocks stacked on a table, where the top one is green, the bottom one is not green, and the color of the middle block



▪ **FIGURE 2.1**

A Stack of Three Blocks

is not known. The question to consider is whether there is a green block directly on top of a nongreen one. The thing to observe about this question is that the answer (which happens to be *yes*) is not immediately obvious without some thinking.

We can formalize this problem in FOL, using a, b, and c as the names of the blocks and predicate symbols G and O to stand for "green" and "on." The facts we have in S are

$$\{O(a,b), O(b,c), G(a), \neg G(c)\}$$

and this is all we need. The claim we make here is that these four facts *entail* that there is indeed a green block on top of a nongreen one, that is, that $S \models \alpha$, where α is

$$\exists x \exists y.\, G(x) \wedge \neg G(y) \wedge O(x,y).$$

To see this, we need to show that any interpretation that satisfies S also satisfies α. So let $\Im$ be any interpretation, and assume that $\Im \models S$. There are two cases to consider:

1. Suppose $\Im \models G(b)$. Then because $\neg G(c)$ and $O(b,c)$ are in S,

$$\Im \models G(b) \wedge \neg G(c) \wedge O(b,c).$$

It follows from this that

$$\Im \models \exists x \exists y.\, G(x) \wedge \neg G(y) \wedge O(x,y).$$

2. Suppose on the other hand that it is not the case that $\Im \models G(b)$. Then it is the case that $\Im \models \neg G(b)$, and because $G(a)$ and $O(a,b)$ are in S,

$$\Im \models G(a) \wedge \neg G(b) \wedge O(a,b).$$

It follows from this that

$$\Im \models \exists x \exists y.\, G(x) \wedge \neg G(y) \wedge O(x,y).$$

Either way, it is the case that $\Im \models \alpha$. Thus, α is a logical consequence of S.

Even though this is a very simple example, we can see that calculating what is implicit in a given collection of facts will sometimes involve subtle forms of reasoning. Indeed, it is well known that for FOL the problem of determining whether one sentence is a logical consequence of others is in general *unsolvable:* No automated procedure can decide validity, and so no automated procedure can tell us in all cases whether or not a sentence is entailed.

2.5.2 Knowledge-Based Systems

To recap, we imagine that for knowledge representation we will start with a (large) KB representing what is explicitly known by a knowledge-based system. This KB could be the result of what the system is told, or perhaps what the system found out for itself through perception or learning. Our goal is to influence the behavior of the overall system based on what is *implicit* in this KB, or as close as possible.

In general, this will require reasoning. By *deductive inference*, we mean the process of calculating the entailments of a KB, that is, given the KB, and any sentence α, determining whether or not KB $\models \alpha$.

We consider a reasoning process to be *logically sound* if whenever it produces α, then α is guaranteed to be a logical consequence. This rules out the possibility of producing plausible assumptions that may very well be true in the intended interpretation but are not strictly entailed.

We consider a reasoning process to be *logically complete* if it is guaranteed to produce α whenever α is entailed. This rules out the possibility of missing some entailments, for example, when their status is too difficult to determine.

As noted, no automated reasoning process for FOL can be both sound and complete in general. However, the relative simplicity of FOL makes it a natural first step in the study of reasoning. The computational difficulty of FOL is one of the factors that will lead us to consider various other options in subsequent chapters.

2.6 BIBLIOGRAPHIC NOTES

For a history of the development of logic beginning in ancient Greece, refer to Kneale and Kneale [218]. Books on first-order logic tend to focus on three broad categories: philosophical logic [172, 187, 234], mathematical logic [26, 34, 119, 288], and computer science [194, 265]. The role of logic in Artificial Intelligence is treated by Genesereth and Nilsson [153]. The importance of first-order logic in knowledge representation is argued by Hayes [181], Israel [195] and Moore [294, 297]. John McCarthy, probably the first to advocate the use of formal logic as a basis for automated reasoning in Artificial Intelligence [275], also argues that first-order logic is sufficient for knowledge representation [281]. Interesting related works in the philosophy of language include those by Ayer [16] and Wittgenstein [429].

The distinction between explicit and implicit belief was first discussed in AI by Levesque [239] and further developed by Levesque and Lakemeyer [244]. Other approaches to the same issue were taken by Fagin and Halpern [123], Delgrande [100], and others. The topic is also discussed in the psychological literature (see [103], for example).

The semantics of first-order logic is largely due to Tarski [405]. For a proof of the undecidability of first-order logic, see Büchi [61] (the undecidability of FOL was first shown independently by Church and Turing in 1936). Completeness of first-order logic for a given set of axioms was first proven by Gödel [161]. Gödel's famous incompleteness theorem regarding number theory [163] is discussed in [301].

The *Handbook of Logic in Artificial Intelligence and Logic Programming* [139] and the *Handbook of Philosophical Logic* [138] are two series of volumes that are excellent sources of articles on a large variety of logics and their uses.

2.7 EXERCISES

1. For each of the following sentences, give a logical interpretation that makes that sentence false and the other two sentences true:

(a) $\forall x \forall y \forall z[(P(x,y) \wedge P(y,z)) \supset P(x,z)]$;

(b) $\forall x \forall y[(P(x,y) \wedge P(y,x)) \supset (x = y)]$;

(c) $\forall x \forall y[P(a,y) \supset P(x,b)]$.

2. This question involves formalizing the properties of mathematical *groups* in FOL. Recall that a set is considered to be a group relative to a binary function f and an object e if and only if (1) f is associative;

(2) e is an identity element for f, that is, for any x, $f(e,x) = f(x,e) = x$; and (3) every element has an inverse, that is, for any x, there is an i such that $f(x,i) = f(i,x) = e$. Formalize these as sentences of FOL with two nonlogical symbols, a function symbol f, and a constant symbol e, and show using interpretations that the sentences logically entail the following property of groups:

For every x and y, there is a z such that $f(x,z) = y$.

Explain how your answer shows the value of z as a function of x and y.

3. This question involves formalizing some simple properties of *sets* in FOL. Consider the following three facts:

- *No set is an element of itself.*
- *A set x is a subset of a set y iff every element of x is an element of y.*
- *Something is an element of the union of two sets x and y iff it is an element of x or an element of y.*

(a) Represent the facts as sentences of FOL. As nonlogical symbols, use $\mathsf{Sub}(x,y)$ to mean "x is a subset of y," $\mathsf{Elt}(e,x)$ to mean "e is an element of x," and $\mathsf{u}(x,y)$ to mean "the union of x and y." Instead of using a special predicate to assert that something is a set, you may simply assume that in the domain of discourse (assumed to be nonempty) everything is a set. Call the resulting set of sentences $\mathcal{T}$.

(b) Show using logical interpretations that $\mathcal{T}$ entails that x is a subset of the union of x and y.

(c) Show using logical interpretations that $\mathcal{T}$ does not entail that the union of x and y is equal to the union of y and x.

(d) Let A be any set. Show using logical interpretations that $\mathcal{T}$ entails that there is a set z such that the union of A and z is a subset of A.

(e) Does $\mathcal{T}$ entail that there is a set z such that for any set x the union of x and z is a subset of x? Explain.

(f) Write a sentence that asserts the existence of singleton sets, that is, for any x, the set whose only element is x. $\mathcal{T}_1$ is $\mathcal{T}$ with this sentence added.

(g) Prove that $\mathcal{T}_1$ is not finitely satisfiable (again, assuming the domain is nonempty). *Hint:* In a finite domain, consider u, the object interpreted as the union of all the elements in the domain.

(h) Prove or disprove that $\mathcal{T}$ entails the existence of an empty set.

4. In a certain town, there are the following regulations concerning the town barber:

- *Anyone who does not shave himself must be shaved by the barber.*
- *Whomever the barber shaves, must not shave himself.*

Show that no barber can fulfill these requirements. That is, formulate the requirements as sentences of FOL and show that in any interpretation where the first regulation is true, the second one must be false. (This is called the *barber's paradox* and was formulated by Bertrand Russell.)