Chapter 2

Logic

Let us admit what all idealists admit: that the nature of the world is hallucinatory. Let us do what no idealist has done: let us look for the unrealities that confirm that nature. We shall find them, I believe, in the antinomies of Kant and in Zeno's dialectic.

"The greatest sorcerer [writes Novalis memorably] would be the one who bewitched himself to the point of taking his own phantasmagoria for autonomous apparitions. Would not this be true of us?"

I believe that it is. We (the undivided divinity that operates within us) have dreamed the world. We have dreamed it strong, mysterious, visible, ubiquitous in space and secure in time; but we have allowed tenuous, eternal interstices of injustice in its structure so we may know that it is false.

— Jorge Luis Borges, "Avatars of the Tortoise," *Other Inquisitions*

To express knowledge about particular domains, we need to develop languages, to define their meaning, and to describe what kinds of inferences they allow. It turns out that several basic issues in defining languages, semantics, and inference techniques are important in many different domains of application. It makes sense, therefore, to address these questions once and for all abstractly. That is, we will look first for a general schema for defining languages which handles these common basic issues elegantly and effectively; we can then use this schema to define a particular language for a particular domain. Such a schema is called a *logic*.

The best known logics, and the most often used, are propositional logic and classical first-order logic. We assume that the reader is familiar with these; sections 2.2 and 2.3 provide a brief review. Besides these standard logics, however, a number of non-standard logics have been proposed for use in AI. These non-standard logics typically modify the standard logics in two ways. First, they allow sentences that say something about other sentences; this is the subject of sections 2.5 through 2.8. Second, they deal with uncertain and tentative information; this is the subject of chapter 3.

2.1 Logical Systems and Languages

A *language* is a collection of meaningful strings, called *sentences.* A *logical system* (or simply *logic*) is a method for defining languages and their meanings. A logic consists of

- A set of *logical symbols.*
- A characterization of the possible *non-logical symbols* that can be defined, and the types of meanings they can be given.
- *Syntactic rules* for constructing meaningful sentences out of logical and non-logical symbols.
- A characterization of *proofs.*
- *Semantic rules* for determining the meaning of a sentence, using the syntax of the sentence, and the meanings of the constituent non-logical symbols.

The structure of a logic is roughly analogous to that of a programming language.[1] The logical symbols correspond to the reserved words or symbols of the programming language. The non-logical symbols correspond to user-defined identifiers (variable names, function names, and so on). The syntactic rules for constructing a sentence in a logic are analogous to the syntactic rules for constructing a program in the programming language. The semantic rules that define the meaning of a sentence are weakly analogous to semantic rules that define what a program does. Ordinary programming languages have no analogue of an inference rule or an axiom.

In the logics we will study, symbols are *atomic*; the particular sequence of characters that compose them are not significant. (By contrast in natural language, the form of a word often carries semantic information.) Naturally, we will use symbols that resemble English words of related meaning. When we need to name an individual object of a particular type, we will often give it a name consisting of the type followed by a number or letter; thus, a chair might be named by the symbol "chair34" or "chairb".

A *proof* is a syntactic structure; it is made of symbols organized according to specific rules. The *hypotheses* and *conclusion* of a proof are sentences that occupy a distinguished position in the proof; if there is a proof of a conclusion from a set of hypotheses, then it is legitimate to infer the conclusion given the hypotheses. Proofs may take many different forms, depending on the particular logic. For example, in an *axiomatic* logic, the logic specifies a set of logical axioms and a set of inference rules, which allow a sentence ϕ to be derived from a set of sentences Γ. A proof is then a sequence of sentences such that each sentence is either a logical axiom, or is a hypothesis, or is derivable from previous sentences via some inference rule.

An *interpretation* for a language $\mathcal{L}$ is a definition of each of the non-logical symbols of the language in terms of some domain. A *semantics* for a logic is a definition of the truth (or other characteristic) of sentences in a language in the logic in terms of the interpretation. If $\mathcal{I}$ is an interpretation of $\mathcal{L}$ and ϕ is a sentence in $\mathcal{L}$ that is true in $\mathcal{I}$, we write $\mathcal{I} \models \phi$ (read "$\mathcal{I}$ satisfies ϕ" or "$\mathcal{I}$ is a model for ϕ". The symbol $\models$ is called the "double turnstile".) Similarly, if $\mathcal{Q}$ is a set of interpretations, and ϕ is true in all the interpretations in $\mathcal{Q}$, we write $\mathcal{Q} \models \phi$. We often identify a

[1]Historically, of course, programming languages were developed later than logical notation.

set of sentences Ψ with the set of all interpretations in which all the sentences of Ψ are true. Under this identification, we can write $\Psi \models \phi$ to mean that, for any interpretation $\mathcal{I}$, if $\mathcal{I}$ is a model for all the sentences in Ψ, then $\mathcal{I}$ is also a model for ϕ.

If ϕ is true in all interpretations whatever, then we write $\models \phi$; such a ϕ is said to be *universally valid.* A universally valid sentence is thus one which is true purely by virtue of the logic, independent of what interpretation is given to the language.

The above concept of satisfying a model is a *semantic* concept; it relates to the meaning of a sentence. There is an analogous syntactic concept, based on the idea of a proof. If there is a proof of ϕ from hypotheses Ψ, then we write $\Psi \vdash \phi$ (read "ϕ is derivable from Ψ." The symbol $\vdash$ is called the single turnstile.) If ϕ is derivable without any hypotheses, then we write $\vdash \phi$. A set of hypotheses Ψ is inconsistent if, for some sentence ϕ, both ϕ and its negation[2] can be proven from Ψ.

The semantic concept of validity and the syntactic concept of provability are connected by the logical properties of soundness and completeness. A logic is sound if its proof theory preserves truth; if the sentences in Ψ are true in an interpretation $\mathcal{I}$ and ϕ can be inferred from Ψ, then ϕ is also true in $\mathcal{I}$. In other words, a logic is sound just if, for all Ψ and ϕ, $\Psi \vdash \phi$ implies that $\Psi \models \phi$. A logic is complete if any valid conclusion can be proven; that is, for any Ψ and ϕ, $\Psi \models \phi$ implies $\Psi \vdash \phi$. In a complete logic, the proof system is strong enough to derive any conclusions which are necessarily valid, given a set of axioms. Godel's completeness theorem proves that the predicate calculus is complete.

The term "complete" is also used in a different sense in logic. An axiom system is said to be complete if, for any sentence ϕ in the language, it is either possible to prove ϕ or to prove the negation of ϕ. Thus, completeness of an axiom system means that the axioms are strong enough to characterize every sentence as provably true or false. Completeness of a logic means that the proof theory can extract all the necessary consequences out of a set of axioms. Godel's incompleteness theorem proves that the standard axioms of arithmetic are incomplete.(More precisely, Godel's theorem shows that any recursively enumerable, consistent, axiomatization of arithmetic is incomplete [Nagel and Newman, 1958].)

The presentation of an axiom system may enumerate each individual axiom, or it may use an *axiom schema* to generate a class (generally infinite and recursively enumerable) of axioms. An axiom schema is specified by giving some rule which allows us to distinguish members of the class from non-members. For example, we may specify the axiom schema, "For any sentences ϕ and ψ, the sentence '$\phi \wedge \psi \Rightarrow \phi$' is an axiom." This axiom schema asserts that any sentences of the given form is an axiom: thus, "can_fly(pigs) $\wedge$ 1+1=2 $\Rightarrow$ can_fly(pigs)" is an axiom. For another example, the principle of mathematical induction on the integers is an axiom schema.[3] It has the following form: For any formula $\alpha(X)$ with one free variable X, the following is an axiom:

$$[\alpha(0) \wedge [\forall_N \, \alpha(N) \Rightarrow \alpha(N+1)]] \Rightarrow \forall_N \, \alpha(N)$$

Thus, for example, taking $\alpha(X)$ to be $X < X + 1$, we may construct the axiom

[2]This definition assumes that the concept of negation is part of the logic. An alternative definition, that avoids this assumption, is that Ψ is inconsistent if all sentences can be proven from Ψ.

[3]Strictly speaking, this is not the full principle of mathematical induction, but it is the closest one can come in a first-order schema.

$$(0 < 0+1) \wedge [\forall_N\, N < N+1 \Rightarrow (N+1) < (N+1)+1] \Rightarrow \forall_N\, N < N+1$$

Statements about a logical language, whether phrased formally or informally, are said to be *meta-linguistic* or to be in a *meta-language.* For example, any axiom schema, such as the one given above, "For any sentences, ϕ and ψ, the sentence '$\phi \wedge \psi \Rightarrow \phi$' is an axiom," is a meta-linguistic statement; it is a sentence about certain sentences in the formal language; it is is not in the language itself. The formal language itself is often called the *object language.* A *theory* is a collection of sentences in an object language, together with all the consequences of those sentences. The base sentences of the theory, from which other sentences are deduced, are called *proper axioms,* as distinguished from the *logical axioms,* which hold for all theories in all languages in this logic.

2.2 Propositional calculus

The proposition calculus (also called "propositional logic" or "sentential logic") describes how sentences can be combined using Boolean operators. The logical symbols of this theory are the Boolean operators: $\neg$ (not), $\wedge$ (and), $\vee$ (or), $\Rightarrow$ (implies), $\Leftrightarrow$ (if and only if) and $\dot\vee$ (exclusive or). Parentheses and brackets are used for grouping. The non-logical symbols, called "sentential constants," denote atomic sentences; we will here use lower-case letters. For example, p might be the proposition "Hydrogen is lighter than oxygen," q might be the proposition "All ducks are fish," and r might be the proposition "Roger Maris hit sixty-one home runs in 1961."

A sentence in the propositional calculus is defined recursively as follows:

Definition 2.1: A string ϕ is a sentence if and only if

a. ϕ is a sentential constant; or

b. ϕ has one of the following forms: $\neg\psi$; $(\psi \wedge \zeta)$; $(\psi \vee \zeta)$; $(\psi \Rightarrow \zeta)$; $(\psi \Leftrightarrow \zeta)$; $(\psi \dot\vee \zeta)$; where ψ and ζ are (recursively) sentences.

Thus, given the sentential constants p, q, and r, sentences include '$(p \vee \neg q)$', '$((p \Leftrightarrow q) \Rightarrow (\neg p \dot\vee r))$', and so on.

(Note on notation: In this book, we will often leave out some parentheses, with the convention that $\neg$ has higher priority than $\vee$, $\wedge$, and $\dot\vee$, which have higher priority than $\Rightarrow$ or $\Leftrightarrow$. Thus '$\neg p \wedge q \Rightarrow r$' is interpreted as '$((\neg p \wedge q) \Rightarrow r)$'. Also we use "running" sequences of $\Leftrightarrow$ and $\dot\vee$. Thus '$p \Leftrightarrow q \Leftrightarrow r$' means that all three sentences p, q, and r have the same truth value, rather than being equivalent to '$(p \Leftrightarrow q) \Leftrightarrow r$'. Similarly '$p \dot\vee q \dot\vee r$' means that exactly one of p, q, and r is true, rather than being equivalent to '$(p \dot\vee q) \dot\vee r$'. Such sequences of $\Leftrightarrow$ or $\dot\vee$ should be considered as constituting a single operator on the propositions connected, rather than a collection of operators. Thus, strictly speaking it would be better to write "equivalent(p, q, r)" or "exactly_one(p, q, r)", but the other notation is standard and more readable.)

The domain of the propositional calculus consists of the two truth values 'TRUE' and 'FALSE'. Given a set of sentential constants $\mathcal{S}$, an interpretation for $\mathcal{S}$ is a mapping from $\mathcal{S}$ to the values

'TRUE' and 'FALSE'. For example, if $\mathcal{S} = \{p, q, r\}$, then one interpretation $\mathcal{I}$ would be the mapping $\mathcal{I}(p)$=TRUE; $\mathcal{I}(q)$=FALSE; $\mathcal{I}(r)$=FALSE. The semantics of the propositional calculus then defines how an interpretation $\mathcal{I}$ is extended from the sentential constants to all the constants in the language. This is done according to the following recursive rule:

Definition 2.2: Let $\mathcal{S}$ be a set of sentential constants and let $\mathcal{I}$ be an interpretation over $\mathcal{S}$. Then

- $\mathcal{I}(\neg\phi)$=TRUE if $\mathcal{I}(\phi)$ = FALSE and TRUE otherwise.
- $\mathcal{I}(\phi \vee \psi)$=TRUE if either $\mathcal{I}(\phi)$=TRUE or $\mathcal{I}(\psi)$=TRUE; otherwise it is FALSE.
- $\mathcal{I}(\phi \wedge \psi)$=TRUE just if both $\mathcal{I}(\phi)$=TRUE and $\mathcal{I}(\psi)$=TRUE;
- $\mathcal{I}(\phi \Rightarrow \psi)$=TRUE just if either $\mathcal{I}(\phi)$=FALSE or $\mathcal{I}(\psi)$=TRUE;
- $\mathcal{I}(\phi \Leftrightarrow \psi)$=TRUE just if $\mathcal{I}(\phi)$ is the same as $\mathcal{I}(\psi)$; and
- $\mathcal{I}(\phi \dot{\vee} \psi)$=TRUE just if one but not both of $\mathcal{I}(\phi)$ and $\mathcal{I}(\psi)$ is TRUE.

At first glance, Definition 2.2 would seem to be no more than a tautologous translation of the formal symbols for Boolean connectives into the corresponding English words. In fact, however, it is more than that; it allows us to establish a correspondence between a string of symbols such as $\neg(p \vee \neg q)$ and the semantic objects TRUE and FALSE, given the truth of the sentential constants. The form of the definition establishes important constraints on the meaning of the Boolean connectives; the truth of a complex sentence depends *only* on the truth values of its components and on nothing else. Thus, for example, there is no way in the propositional calculus of expressing relations between sentences such as 'ϕ is the same sentence as ψ' or 'ϕ can be proven from ψ'. In particular, the material implication '$\phi \Rightarrow \psi$' means only that either ϕ is false or that ψ is true; it signifies no further connection between the two sentences. Readers who are still unconvinced that there is anything to the semantic definition 2.3 are asked to suspend judgement until they have seen the more complex semantics in sections 2.3.2 and 2.7.1.

There are several ways of defining sound and complete inference systems for the propositional calculus. The easiest method for verifying proofs by hand is to use truth tables. However, to define this formally as a proof system involves giving a syntactic characterization of a truth table as a system of symbols, which is messy though not difficult. Table 2.1 shows an axiomatic inference system for the propositional calculus, and table 2.2 shows some sample proofs.

A universally valid sentence in the propositional calculus is called a *tautology*; an inference in the propositional calculus is called a *tautological* inference.

The propositional calculus is rarely adequate for inference in AI domains. The only inferences it legitimates are the moving of Boolean operations around fixed sentences. This is sufficient to solve some combinatorial problems in fixed domains. For instance, the propositional calculus is adequate for the formulation and solution of puzzles like, "Jones, Smith and Robinson are a fireman, an engineer, and a conductor, not necessarily in that order. Jones owes the fireman thirty dollars. The conductor's wife never allows him to borrow money. Smith is a bachelor. Who has what job?" But most useful inferences involve applying a general rule to a specific case, which is beyond the power of the propositional calculus. Nonetheless, it deserves mention because it forms the basis for

Axioms: For any sentences α, β, γ, any of the following sentences is an axiom:

PROP.1. $\alpha \Rightarrow (\beta \Rightarrow \alpha)$

PROP.2. $(\alpha \Rightarrow (\beta \Rightarrow \gamma)) \Rightarrow ((\alpha \Rightarrow \beta) \Rightarrow (\alpha \Rightarrow \gamma))$

PROP.3. $(\neg\alpha \Rightarrow \neg\beta) \Rightarrow ((\neg\alpha \Rightarrow \beta) \Rightarrow \alpha)$

As an aid in reading these axioms, note that $\alpha \Rightarrow (\beta \Rightarrow \gamma)$ means the same as $(\alpha \wedge \beta) \Rightarrow \gamma$.

Definitional equivalences: (These axioms serve only to define the other Boolean operators in terms of implication and negation. They can be omitted, if we restrict our language to have only negation and implication.)

PROP.4. $(\alpha \Leftrightarrow \beta) \Rightarrow (\alpha \Rightarrow \beta)$

PROP.5. $(\alpha \Leftrightarrow \beta) \Rightarrow (\beta \Rightarrow \alpha)$

PROP.6. $(\alpha \Rightarrow \beta) \Rightarrow ((\beta \Rightarrow \alpha) \Rightarrow (\alpha \Leftrightarrow \beta))$

PROP.7. $(\alpha \vee \beta) \Leftrightarrow (\neg\alpha \Rightarrow \beta)$

PROP.8. $(\alpha \wedge \beta) \Leftrightarrow \neg(\alpha \Rightarrow \neg\beta)$

PROP.9. $(\alpha \dot{\vee} \beta) \Leftrightarrow (\alpha \Leftrightarrow \neg\beta)$

Rule of inference (Modus Ponens): For any two sentences ϕ and ψ, ψ may be inferred from the two sentences ϕ and $\phi \Rightarrow \psi$.

Proof: A proof of theorem ϕ from hypotheses Ψ is a sequence of sentences ending in ϕ such that each sentence is either

- An element of Ψ; or
- An axiom; or
- Inferrable from earlier sentences in the proof via modus ponens.

Table 2.1: Axioms for the Propositional Calculus

Note: The proof proper is just the sequence of sentences. The justifications on the side are just comments, to aid the reader.

Given { }: To prove: $p \Rightarrow p$

#	Step	Justification
1.	$p \Rightarrow ((p \Rightarrow p) \Rightarrow p)$	PROP.1: α=p; β=(p $\Rightarrow$ p).
2.	$(p \Rightarrow ((p \Rightarrow p) \Rightarrow p)) \Rightarrow ((p \Rightarrow (p \Rightarrow p)) \Rightarrow (p \Rightarrow p))$	PROP.2: α=p; β=(p $\Rightarrow$ p); γ=p.
3.	$(p \Rightarrow (p \Rightarrow p)) \Rightarrow (p \Rightarrow p)$	Modus Ponens: (1) and (2).
4.	$p \Rightarrow (p \Rightarrow p)$	PROP.1: $\alpha = \beta$ =p
5.	$p \Rightarrow p$	Modus Ponens: (4) and (3).

Given { $p \wedge q$ }: To prove: q

#	Step	Justification
1.	$p \wedge q$	Given.
2.	$(p \wedge q) \Leftrightarrow \neg(p \Rightarrow \neg q)$.	PROP.8: α =p, β =q.
3.	$((p \wedge q) \Leftrightarrow \neg(p \Rightarrow \neg q)) \Rightarrow$ $((p \wedge q) \Rightarrow \neg(p \Rightarrow \neg q))$	PROP.4: α=$(p \wedge q) \Leftrightarrow \neg(p \Rightarrow \neg q)$ β=$(p \wedge q) \Rightarrow \neg(p \Rightarrow \neg q)$
4.	$(p \wedge q) \Rightarrow \neg(p \Rightarrow \neg q)$	Modus Ponens: (2) and (3).
5.	$\neg(p \Rightarrow \neg q)$	Modus Ponens: (1) and (4).
6.	$\neg(p \Rightarrow \neg q) \Rightarrow (\neg q \Rightarrow \neg(p \Rightarrow \neg q))$	PROP.1: α=$\neg(p \Rightarrow \neg q)$; β=$\neg$q.
7.	$\neg q \Rightarrow \neg(p \Rightarrow \neg q)$	Modus Ponens: (5) and (6).
8.	$\neg q \Rightarrow (p \Rightarrow \neg q)$	PROP.1: α=$\neg$q; β=p.
9.	$(\neg q \Rightarrow \neg(p \Rightarrow \neg q)) \Rightarrow ((\neg q \Rightarrow (p \Rightarrow \neg q)) \Rightarrow q)$	PROP.3: α=q; β=(p $\Rightarrow$ $\neg$q).
10.	$(\neg q \Rightarrow (p \Rightarrow \neg q)) \Rightarrow q$	Modus Ponens: (7) and (9).
11.	q.	Modus Ponens: (8) and (10).

Table 2.2: Proof in the Propositional Calculus

first-order logic, and because it offers an elementary testing ground for developing logical theories. When alternative forms of logic are studied, they are generally developed first as extensions to propositional calculus, and only later as extensions to first-order logic.

2.3 Predicate Calculus

The predicate calculus, also called first-order logic, is by far the most important and commonly used logical system. We will use this logic for most of our domain theories. It is known that the predicate calculus is sufficiently powerful for classical mathematics.

The predicate calculus extends propositional calculus in two directions. First, it provides an inner structure for atomic sentences; these are viewed as expressing relations between things. Second, it gives us the means to express, and reason with, generalizations; we can say that a certain property holds of all objects, of some object, or of no object.

2.3.1 Syntax of Predicate Calculus

The logical symbols of predicate calculus are the Boolean operators of propositional logic, the quantifiers $\forall$ (for all) and $\exists$ (there exists), the comma, the open and close parentheses, and an infinite collection of variable symbols. Non-logical symbols are divided into three kinds: constant symbols, function symbols, and predicate symbols. Associated with each function symbol and predicate symbol is a positive integer, fixing the number of arguments that the symbol may take. In this book, we will use strings with italicized upper-case letters, such as "A", "$X1$", or "$THE\_DAY\_THE\_WORLD\_STOOD\_STILL$", as variables, and strings with lower-case letters, such as "john", "father_of", and "impossible_to_get_started", as non-logical symbols.

The following definitions are used to define the first-order language $\mathcal{L}$ with a given set of non-logical symbols.

Definition 3.1: The string τ is a *term* if one of the following holds:

a. τ is a constant symbol; or

b. τ is a variable symbol; or

c. τ has the form $\beta(\tau_1, \tau_2 \dots \tau_k)$ where β is a k-place function symbol, and each of the τ_i is a term.

Definition 3.2: The string ϕ is a *formula* of $\mathcal{L}$ if one of the following holds:

a. ϕ has the form $\gamma(\tau_1, \tau_2 \dots \tau_k)$, where γ is a k-place predicate symbol, and each of the τ_i are terms. (These are called atomic formulas.)

b. ϕ has one of the following forms: '$\neg\psi$'; '$(\psi \wedge \zeta)$'; '$(\psi \vee \zeta)$'; '$(\psi \Rightarrow \zeta)$'; '$(\psi \Leftrightarrow \zeta)$'; '$(\psi \dot{\vee} \zeta)$'; where ψ and ζ are (recursively) formulas.

c. ϕ has the form '$\exists\mu\psi$' or '$\forall\mu\psi$', where μ is a variable symbol and ψ is a formula.

Definition 3.3: An occurrence of a variable μ within a formula ϕ is *bound* if it is within an occurrence in ϕ of a formula of the form $\exists\mu\psi$ or $\forall\mu\psi$. An occurrence which is not bound is *free.* (We speak of free occurrences, rather than free variables, because in a formula like "(female(X) $\vee$ $\exists X$ male(X))" the first occurrence of X is free, and the second is bound.)

Definition 3.4: A formula ϕ is *closed* if every occurrence of a variable in ϕ is bound. Otherwise it is *open* in the variables that appear free. A *sentence* is a closed formula.

For example, suppose that our language contains the constant symbols "john" and "mary"; the one-place function "father_of"; the two-place function "common_ancestor_of"; the one-place relations "male" and "female", and the two-place relation "married". Then we can use the above definitions to classify strings such as the following:

Terms: "john", "father_of(john)", "common_ancestor_of(mary,$SOMEONE$)",
"common_ancestor_of(father_of(X), common_ancestor_of(Y,mary))"

Atomic formulas: "male(X)", "female(john)", "married(Y,father_of(Z))"
"male(common_ancestor_of(X,john))"

Complex formulas: "$\neg$male(john)", "(female(Y) $\Rightarrow$ female(mary))",
"$\forall PP$ (married(PP, QQ) $\vee$ male(father_of(PP)))"

Closed formulas: "female(john)", "(married(john,mary) $\vee$ $\exists X$ (male(X)))"
"$\neg\exists QQ \forall PP$ (married(PP, QQ) $\vee$ male(father_of(PP)))"

Some standard notational conventions will be followed in this book. We will use open formulas as independent sentences with the convention that free variables are considered to be universally quantified with the widest possible scope. Variables are displayed subscripted next to the quantifier that binds them. When the same quantifier occurs several times in succession, they are collapsed into one. In using mathematical symbols which are standardly written between their arguments rather than before, (infix, rather than prefix), we will adopt mathematical convention, rather than insist on forms like $+(1,X)$ or $\in (X,S)$. Thus we may write "$\exists_{X,Y}\ X+Y < P+Q$" as an alternative notation for "$\forall P \forall Q \exists X \exists Y\ < (+(X,Y), +(P,Q))$". Quantifiers are taken to have the lowest possible priority, and therefore the largest scope possible in the sentence; they apply until the end of the sentence or until the close of a bracket. For example, the sentence "$\exists_X$ p(X) $\Rightarrow$ p(a)" is read as "$\exists_X$ [p(X) $\Rightarrow$ p(a)]" and not as "[$\exists_X$ p(X)] $\Rightarrow$ p(a)".

There are many ways of defining proof systems in the predicate calculus. The most useful of these, for the purposes of actually writing down proofs, is natural deduction, illustrated in the chapter appendix. Resolution, with Skolemization, is a proof system that is relatively efficient to implement and control, but it is complex and unintuitive. Table 2.3 displays an axiomatic proof system for the predicate calculus. This system is not particularly easy to use, either in hand construction of proofs or in computation, but it have the advantage of brevity.

Definition 3.5: Let α be any formula. A formula β is a *closure* of α if (i) β consists of α preceded by some number of universal quantifiers with variables; and (ii) β is closed. Example: the formula '$\forall_A \forall_B \forall_C \exists_X$ p(X, A) $\vee$ q(C, A)' is a closure of '$\exists_X$ p(X, A) $\vee$ q(C, A)'.

2.3.2 Tarskian Semantics

The semantics for first-order logic is called Tarskian semantics, after the logician Alfred Tarski. As in propositional calculus, we begin by defining a domain $\mathcal{D}$ for the formal language; we then state how an interpretation can relate the non-logical symbols to the domain; lastly, we describe how the truth of sentences is built up out of the meanings of the non-logical symbols. The semantics of the propositional calculus uses only two semantic entities: the truth values TRUE and FALSE. The predicate calculus, by contrast, needs a richer interpretation, with a universe of objects, tuples of objects, and sets of tuples of objects. We will interpret relations holding on terms as statements that a certain tuple of objects is an element of a certain set of tuples. As with the propositional calculus, the definition of the semantics will at first look almost circular and tautologous, but in fact imposes substantial and important constraints on the meaning of the language. In particular, the use of set theory as a basis for the interpretation means that the language is sensitive only to the *extensional* properties of its symbols; that is, to the entities or sets that they describe and not to the form of the description. Moreover, we can use our understanding of set theory to analyze properties of the logic; for example, to prove that first-order logic is sound and complete.

(Note: the remainder of section 2.3.2 involves rather abstract logic. It may be omitted without loss of continuity. In the rest of this book, only the end of section 2.7.1 depends on a detailed understanding of Tarksian semantics.)

A domain $\mathcal{D}$ for a first order language is a set of entities or individuals. A constant symbol denotes an individual in $\mathcal{D}$. A k-place predicate symbol γ denotes an *extensional relation* Γ, which

For any formulas α, β, γ, any closure of any of the formulas FOL.1-FOL.5 is an axiom:

FOL.1 $\alpha\Rightarrow(\beta\Rightarrow\alpha)$

FOL.2 $(\alpha\Rightarrow(\beta\Rightarrow\gamma))\Rightarrow((\alpha\Rightarrow\beta)\Rightarrow(\alpha\Rightarrow\gamma))$

FOL.3 $(\neg\alpha\Rightarrow\neg\beta)\Rightarrow((\neg\alpha\Rightarrow\beta)\Rightarrow\alpha)$

FOL.4 $(\forall\mu(\alpha\Rightarrow\beta))\Rightarrow(\forall\mu\alpha\Rightarrow\forall\mu\beta)$

FOL.5 $\alpha\Rightarrow\forall\mu\alpha$ where μ does not appear free in α

FOL.6 Let α and β be formulas that are identical, except that each free occurrence of the variable μ in α is replaced in β by the term τ. τ is a term that is free of μ in α; that is, no free occurrences of μ in α are within the scope of any quantifier $\forall\nu$ or $\exists\nu$, where ν is a variable occurring in τ. Then any closure of the formula $\beta\Rightarrow\exists\mu\alpha$ is an axiom.

For example, the sentence "outfielder(mickey_mantle) $\Rightarrow$ $\exists X$ outfielder(X)" is an axiom, with $\mu = X$ and $\tau =$mickey_mantle. The condition that τ must be free of μ in α is needed to block invalid axioms like

$$\forall_Y\, Y < Y + 1 \Rightarrow \exists_X \forall_Y Y < X$$

which is of a similar form, with $\mu = X$ and $\tau = Y + 1$

Definitional equivalences: Any closure of the following formulas is an axiom:

FOL.7 $(\alpha\Leftrightarrow\beta)\Rightarrow(\alpha\Rightarrow\beta)$

FOL.8 $(\alpha\Leftrightarrow\beta)\Rightarrow(\beta\Rightarrow\alpha)$

FOL.9 $(\alpha\Rightarrow\beta)\Rightarrow((\beta\Rightarrow\alpha)\Rightarrow(\alpha\Leftrightarrow\beta))$

FOL.10 $(\alpha\vee\beta)\Leftrightarrow(\neg\alpha\Rightarrow\beta)$

FOL.11 $(\alpha\wedge\beta)\Leftrightarrow\neg(\alpha\Rightarrow\neg\beta)$

FOL.12 $(\alpha\dot{\vee}\beta)\Leftrightarrow(\alpha\Leftrightarrow\neg\beta)$

FOL.13 $(\forall\mu\alpha)\Leftrightarrow(\neg\exists\mu\neg\alpha)$

Rule of inference (Modus Ponens): For any two formulas ϕ and ψ, ψ may be inferred from the two formulas ϕ and $\phi\Rightarrow\psi$.

Proof: A proof of theorem ϕ from hypotheses Ψ is a sequence of sentences ending in ϕ such that each sentence is either

- An element of Ψ; or
- An axiom; or
- Inferrable from earlier sentences in the proof via Modus Ponens.

Table 2.3: Axioms for First-Order Logic

is a set of k-tuples of elements in $\mathcal{D}$ (a subset of $\mathcal{D}^k$). For example, the predicate "married," in its standard interpretation denotes the extensional relation MARRIED which is just the set of all pairs of married people: MARRIED = { < Douglas Fairbanks, Mary Pickford >, < Mary Pickford, Douglas Fairbanks >, < Queen Victoria, Prince Albert > ... } A k-place function symbol β denotes an extensional total function Θ from $\mathcal{D}^k$ to $\mathcal{D}$; a set of $k+1$-tuples, whose last element depends functionally on the first k elements. This condition of functional dependence means that any k-tuple of elements in $\mathcal{D}$ — that is, any element of $\mathcal{D}^k$ — appears as the first k elements of exactly one $k+1$-tuple in the set Θ. For example, the function symbols "father_of" denotes the function FATHER_OF, which is the set of pairs of each person with his/her father: FATHER_OF = { < Cain, Adam >, < Elizabeth I, Henry VIII > ... }

An interpretation $\mathcal{I}$ for a language $\mathcal{L}$ associates each of the constant symbols in $\mathcal{L}$ with an element of $\mathcal{D}$; each of the k-place function symbols with a function from $\mathcal{D}^k$ to $\mathcal{D}$; and each of the k-place predicate symbols with a k-place relation on $\mathcal{D}$, a subset of $\mathcal{D}^k$. For instance, let $\mathcal{L}$ contain constant symbols "john" and "mary"; function symbol "father_of"; and predicate symbols "married". One interpretation for $\mathcal{L}$ would map "john" onto some particular John Doe; "mary" onto Mary Roe; "father_of" onto the actual FATHER_OF function; "married" onto the actual MARRIED relation; and so on. Many other interpretations are possible; for instance, there is an interpretation that maps "mary" onto Thutmose III, "john" onto Lizzie Borden, "father_of" onto the function mapping each person to the oldest descendant of his paternal grandfather, and "married" onto the relation between shoe salesmen and their customers. Given a symbol α and an interpretation $\mathcal{I}$, we use the notation $\alpha^{\mathcal{I}}$ to mean the value that $\mathcal{I}$ associates with the symbol α. Thus "john$^{\mathcal{I}}$" is the person John Doe; "father_of$^{\mathcal{I}}$" is the actual FATHER_OF function; and so on.

Defining the semantics of $\mathcal{L}$ is a little tricky, because the significance of a variable in a term depends how the variable is quantified, which is specified in a context external to the term itself. We therefore cannot use a simple recursive definition, building up the interpretation from inside to outside, as we did for propositional logic. Rather, our definition proceeds in two stages. First, we define the meaning of atomic formulas with no variables or quantifiers. This is a straightforward recursive definition from inside to outside: the denotation of a complex terms is defined in terms of the denotation of the function symbol and the denotation of its arguments; and the truth of an atomic formula is defined in terms of the meanings of the predicate and its arguments. Second, we define the meaning of complex sentences. The truth of quantified sentences is defined in terms of substitutions: The formula $\forall_\mu \alpha$ is true if α holds for all potential substitutions for μ; the statement $\exists_\mu \alpha$ is true if α is true for some potential substitution for μ. The truth of a Boolean combination of sentences is defined as a Boolean combination of the truths of its components.

Definition 3.6: A *ground term* is a term containing no variables. A *ground formula* is a formula containing no variables, free or bound.

Definition 3.7: If τ is a ground term in $\mathcal{L}$ and $\mathcal{I}$ is an interpretation of $\mathcal{L}$, then there is an individual $u \in \mathcal{D}$ that is denoted by τ under $\mathcal{I}$. We write $u = \tau^{\mathcal{I}}$. We determine the denotation of τ as follows:

a. If τ is a constant symbol α, then τ denotes the individual that $\mathcal{I}$ associates with the symbol. $\tau^{\mathcal{I}} = \alpha^{\mathcal{I}}$.

b. Otherwise, τ has the form $\beta(\tau_1 \dots \tau_k)$ In this case, the denotation of τ is the result of applying the extensional function which $\mathcal{I}$ associates with β to the denotations of $\tau_1 \ldots \tau_k$.

$$< \tau_1^{\mathcal{I}} \ldots \tau_k^{\mathcal{I}}, \tau^{\mathcal{I}} > \in \ \beta^{\mathcal{I}}$$

Definition 3.8: Let $\phi = \gamma(\tau_1 \ldots \tau_k)$ be an atomic ground sentence. $\phi^{\mathcal{I}}$=TRUE under interpretation $\mathcal{I}$ just if the relation $\gamma^{\mathcal{I}}$ holds on the objects $\tau_1^{\mathcal{I}} \ldots \tau_k^{\mathcal{I}}$; that is, if the tuple $< \tau_1^{\mathcal{I}} \ldots \tau_k^{\mathcal{I}} >$ is an element of $\gamma^{\mathcal{I}}$. Otherwise, $\phi^{\mathcal{I}}$ = FALSE.

Definitions 3.7 and 3.8 just formalize the natural interpretation of ground terms and formulas. For example, let $\mathcal{I}$ maps the constant "isaac" onto Isaac (the Biblical patriarch), the function symbol "father_of" onto the real function FATHER_OF, and the predicate "male" onto the one-place relation of being male. Then the denotation of "isaac" is Isaac; the denotation of "father_of(isaac)" is the image of Isaac under the mapping FATHER_OF, namely Abraham; and the sentence "male(father_of(isaac))" is true, because the tuple $<$ Abraham $>$ is an element of the relation MALE.

We next define the truth of complex closed formulas. Boolean operators are handled just as in propositional logic:

Definition 3.9: Let $\mathcal{I}$ be an interpretation of $\mathcal{L}$, and let ϕ and ψ be closed formulas in $\mathcal{L}$. Then

a. Let $\zeta = \neg\phi$. Then $\zeta^{\mathcal{I}}$=TRUE just if $\phi^{\mathcal{I}}$=FALSE; otherwise $\zeta^{\mathcal{I}}$=FALSE.

b. Let $\zeta = \phi \vee \psi$. Then $\zeta^{\mathcal{I}}$=TRUE just if either $\phi^{\mathcal{I}}$=TRUE or $\psi^{\mathcal{I}}$=TRUE; otherwise $\zeta^{\mathcal{I}}$=FALSE.

The remaining Boolean operators are handled similarly, as in Definition 2.2.

The treatment of quantifiers is trickier, as we mentioned above. Intuitively, we would like to say that a formula $\exists \mu \alpha$ is true just if there is some value τ that makes α true when τ is substituted for μ. For example, the formula "$\exists_X X + X = X \cdot X$" is true because when '2' is substituted for 'X', we get the true sentence "$2 + 2 = 2 \cdot 2$". However, it would not be correct to demand that this substituted value be a ground term in $\mathcal{L}$, since there may be objects in the domain $\mathcal{D}$ that are not named by any term in $\mathcal{L}$. (In fact, $\mathcal{L}$ may have no constant symbols whatever, in which case there are no ground terms.) Rather, what we want to say is that $\exists \mu \alpha$ is true if there is some object u in $\mathcal{D}$ such that, if u were given the name δ, then the result of substituting δ for μ in α would be a true sentence. That is what the next two definitions do. Definition 3.10 formalizes the notion of adding a new constant symbol δ to denote object u. Definition 3.11 then uses that to define the meaning of a quantified sentence in terms of the sentences in an extended language with new constants substituted

We assume that there exists an infinite collection of symbols that are not used in the language $\mathcal{L}$, and which are therefore available for use as new constant symbols.

Definition 3.10: Let $\mathcal{I}$ be an interpretation of language $\mathcal{L}$ with domain $\mathcal{D}$. Let u be any member of $\mathcal{D}$, and let δ be a symbol not in $\mathcal{L}$. Then we define $\mathcal{L} \cup \delta$ to be the first order language containing all the symbols in $\mathcal{L}$ and also containing δ, used as a constant symbol. We define $\mathcal{I} \cup (\delta \rightarrow u)$ to be an interpretation of $\mathcal{L} \cup \delta$ with domain $\mathcal{D}$ with the following properties:

- For each symbol $\alpha \in \mathcal{L}$, $\alpha^{\mathcal{I} \cup (\delta \rightarrow u)} = \alpha^{\mathcal{I}}$
- $\delta^{\mathcal{I} \cup (\delta \rightarrow u)} = u$

EQL.1 For any term τ and variable μ, any closure of the formula $\exists_\mu \, \mu = \tau$ is an axiom.

EQL.2 Let $\alpha(\mu)$ be an open formula with free variable μ. Then any closure of the formula $\forall_{X,Y} \; X = Y \Rightarrow [\alpha(X) \Leftrightarrow \alpha(Y)]$ is an axiom.

Table 2.4: Axioms of equality

Definition 3.11: Let $\mathcal{I}$ be an interpretation of language $\mathcal{L}$ with domain $\mathcal{D}$. Let δ be a symbol not in $\mathcal{L}$. For any formula α, let $\alpha(\mu/\delta)$ be the formula that is just like α, except that δ has been substituted for every free occurrence of μ. (This can easily be defined formally by recursion over the form of α.)

a. Let the closed formula ζ have the form $\exists \mu \alpha$. Then $\zeta^{\mathcal{I}}$=TRUE just if there is some element $u \in \mathcal{D}$ such that
$$\alpha(\mu/\delta)^{\mathcal{I} \cup (\delta \rightarrow u)} = \text{TRUE}$$

b. Let the closed formula ζ have the form $\forall \mu \alpha$. Then $\zeta^{\mathcal{I}}$=TRUE just if, for every element $u \in \mathcal{D}$, it is the case that
$$\alpha(\mu/\delta)^{\mathcal{I} \cup (\delta \rightarrow u)} = \text{TRUE}$$

2.3.3 Other issues in first-order logic

Equality: First-order logic is often augmented by the equality relation "$X = Y$". The equals sign may be considered as just a particular non-logical predicate symbol, described by axioms EQL.1 and EQL.2 in table 2.4, or it may be considered an additional logical symbol, whose meaning is fixed in the semantics: if τ_1 and τ_2 are ground terms, then the sentence $\tau_1 = \tau_2$ is true in an interpretation just if τ_1 denotes the same thing as τ_2. Table 2.4 shows the two additional axiom schemas that are needed to handle inference on equality.

As is standard, we will use the notation $X \neq Y$ to mean that X is not equal to Y, and we will string equal signs in expressions like $X = Y = Z = W$ as an abbreviation for $(X = Y) \wedge (Y = Z) \wedge (Z = W)$. We will also use the k-place predicate "distinct$(X_1 \ldots X_k)$" to assert that the objects $X_1 \ldots X_k$ are all pairwise distinct. Thus, we have the definition

$$\text{distinct}(X_1 \ldots X_k) \Leftrightarrow \textstyle\bigwedge_{i \neq j} X_i \neq X_j.$$

Limited quantification: An often useful device in predicate calculus is to qualify a quantified variable by limiting the class of values that it can take. We would like to express "Every positive number has a positive square root," as $\forall_{X>0} \exists_{Y>0} \, Y \cdot Y = X$. Such expressions can be incorporated as simple syntactic sugar[4] for ordinary predicate calculus. In general, "Any X satisfying $\phi(X)$ also satisfies $\psi(X)$", can be translated "For any X, if $\phi(X)$ then $\psi(X)$." "There exists an X satisfying $\phi(X)$ that satisfies $\psi(X)$," can be translated "There exists an X such that both $\phi(X)$ and $\psi(X)$." Thus, the above sentence is equivalent to

$$\forall_X \, X > 0 \Rightarrow \exists_Y Y > 0 \;\wedge\; Y \cdot Y = X$$

[4]Syntactic sugar: a departure from or extension of the standard syntax of a language that increases readibility but not expressive power. "Excessive syntactic sugar leads to cancer of the semi-colons." (Alan Perlis)

Partial Functions: In our definition of the domain of an interpretation, we assumed that all functions are total; that they were defined on every individual in the domain. Frequently, we would like to use partial functions which are defined only on certain individuals. For example, in the domain of family relationships, we might like to define a function "spouse" which maps a married person to his/her spouse, and which is undefined on unmarried persons. However, this leads to complications. For example, if Anne is unmarried, we would liked to say "$\neg\exists_X X =$spouse(anne)", which contradicts axiom 6 above.

One way to handle this is to replace all function symbols by relation symbols. For example, instead of defining "spouse(X)" as a function, we can define "spouse(X, Y)" as a relation, and add an axiom that for any particular X, there is only one Y who is the spouse.

$$\forall_{X,Y,Z} \; [\; \text{spouse}(X,Y) \wedge \text{spouse}(X,Z) \;] \Rightarrow Y = Z$$

An alternative way to handle this formally is to add an additional element $\perp$ (read "undefined" or "bottom") to the universe, and to say that all terms that intuitively are undefined formally have a value of $\perp$; any function with argument $\perp$ evaluates to $\perp$; no predicate holds on $\perp$; and any quantified variable μ is implicitly understood to have the qualification $\mu \neq \perp$. Thus, the above sentence translates to "$\neg\exists_{X\neq\perp} X =$spouse(anne)."

Sorted Logics: Partial functions are particularly common when a theory must express facts about many different sorts of things. In such a theory, functions will generally be defined only on arguments of the proper sort. For example, if we had a theory with times, places, and objects, then we might have a function "midpoint(X, Y)" which mapped two places X and Y to their midpoint; a function "where(O, T)" which is the place where O is at time T; and so on. We would not wish to apply "midpoint" to an object and a time, or "where" to two places. Most of the theories that we will discuss use individuals of various sorts in this way.

Sorted logics are helpful in expressing such theories. Sorted logics are very much like typed programming languages. A fixed set of sorts is defined at the outset. Each constant symbol, and each quantified variable symbol is declared to be of a particular sort; each relational symbol is declared to take arguments of a particular sort; each function is declared to take arguments of a particular sort and to return a value of a particular sort. Formally, this can all be viewed as syntactic sugar, which can be expanded to pure predicate calculus by adding the function "sort_of(X)" mapping an entity X to its sort; adding names for the sorts as constant symbols; and adding a few new axioms, and a few new clauses to existing axioms, to reflect the sort declarations. In the example above, we would add "place" "time", and "object" as constants representing the separate sorts. We would express the declaration of the sort of the "where" function with the two following axioms:

$$\text{sort\_of}(O)=\text{object} \wedge \text{sort\_of}(T)=\text{time} \Rightarrow \text{sort\_of}(\text{where}(O,T))=\text{place}.$$

$$[\text{sort\_of}(O) \neq \text{object} \vee \text{sort\_of}(T) \neq \text{time}] \Rightarrow \text{where}(O,T) = \perp.$$

A function or predicate symbol may be *polymorphically* sorted; that is, it may take arguments of different sorts. For example, we will want the predicate $X < Y$ to be defined whenever X and Y are elements of the same quantitative sort, but not to be defined if X and Y have different sorts (comparing weights to lengths). This corresponds to type overloading of function symbols in programming languages; it is not a problem as long as the sort of any term can be determined given the sorts of the arguments.

We will use sorted logics in a fairly informal way. We will define sorts, and we will declare the sorts of our non-logical symbols. We will declare the sorts of quantified variables implicitly by the predicates and functions which take them as arguments. (A common habit in AI papers is declare the sorts of variables implicitly by the first letter of the variable symbol (shades of FORTRAN); however, we will be using too many different sorts in this book to do that.) However, we will slough over the difficult issues in developing a full theory of sorts, such as using hierarchies of sorts, and combining sorts with set theory. The bibliography gives references for systematic studies of sorted logics.

Common Errors: There are a number of errors in writing first-order formulas that often trap beginning students.

One common error is to reverse the translations of limited quantification discussed above: to represent "All crows are black" in the form '$\forall_X$ crow(X) $\wedge$ black(X)' or to represent 'Some crows are black," in the form '$\exists_X$ crow(X) $\Rightarrow$ black(X)'. One way to avoid this is to keep in mind what these incorrect forms actually mean. The first form '$\forall_X$ crow(X) $\wedge$ black(X)' is equivalent to '[$\forall_X$ crow(X)] $\wedge$ [$\forall_X$ black(X)]'; i.e. "Everything in the world is both a crow and is black." The second form, '$\exists_X$ crow(X) $\Rightarrow$ black(X)' is equivalent to '$\exists_X$ $\neg$crow(X) $\vee$ black(X)', which is equivalent to '[$\exists_X$ $\neg$crow(X)] $\vee$ [$\exists_X$ black(X)]'; i.e., "Either there is something that is not a crow, or there is something that is black," which is true but uninteresting. Actually, if you ever find yourself writing a formula of the form '$\exists_X \alpha(X) \Rightarrow \beta(X)$', you have almost certainly made a mistake; this formula will be true as long as there something in the universe satisfying $\neg\alpha(X)$.

A common error in looking for a representation for a sentence like, "If something is a crow, then it is black," is to suppose that the use of the word "something" indicates that an existential quantifier should be involved. One is thus led to try the representation '$\exists_X$ crow(X) $\Rightarrow$ black(X)', which, as we have seen, means something quite different, or, worse yet, '[$\exists_X$ crow(X)] $\Rightarrow$ black(X)', which is not even a closed formula. (If the free variable is taken to be universally quantified, as in our convention, then this means, "If there exists a crow, then everything is black.") The problem here arises from the English, which is misleading. What this sentence means is "Anything that is a crow is also black," or "For all things, if it is a crow then it is black;", the correct representation is '$\forall_X$ crow(X) $\Rightarrow$ black(X).'

Another error is to read too much into the material implication 'p $\Rightarrow$ q'. Keep in mind that *all* this means is that either p is false, or q is true. It does not mean that q can be derived from p; or that q is true as a result of p; or that q is true after p is true; or that q would be true if p were true. For example, suppose we wish to represent the rule "A sure sign of appendicitis is that, if you push on the right side of the abdomen, then there will be pain on release." The temptation is to represent this statement as a biconditional between having appendicitis and the implication, "If you push, then there will be pain on release."

$$\text{appendicitis}(X) \Leftrightarrow [\ \text{push}(\text{rightside}(\text{abdomen}(X))) \Rightarrow \text{release\_pain}(X)\ ]$$

(These primitives are bogus, of course, but the mistake we are discussing can be made even in a reasonable language that includes the temporal relations involved.) The forward implication here

$$\text{appendicitis}(X) \Rightarrow [\ \text{push}(\text{rightside}(\text{abdomen}(X))) \Rightarrow \text{release\_pain}(X)\ ]$$

is correct. If X has appendicitis, then if you push his abdomen, he will have pain. The backwards implication, however,

$$[\ \text{push}(\text{rightside}(\text{abdomen}(X))) \Rightarrow \text{release\_pain}(X)\ ] \Rightarrow \text{appendicitis}(X)$$

is not correct. The antecedent "push(rightside(abdomen(X))) $\Rightarrow$ release_pain(X)" is true whenever you don't push on the abdomen. This rule, therefore, states that anyone whose abdomen is not pushed has appendicitis. The correct form for this implication is that, if the abdomen is pushed and there is pain, then there is appendicitis.

$$[\ \text{push}(\text{rightside}(\text{abdomen}(X))) \wedge \text{release\_pain}(X)\ ] \Rightarrow \text{appendicitis}(X)$$

The two correct formulas above can be combined into a single rule as follows:

$$\text{push}(\text{rightside}(\text{abdomen}(X))) \Rightarrow [\text{release\_pain}(X) \Leftrightarrow \text{appendicitis}(X)\ ]$$

If you push on the abdomen, then pain occurs just if there is appendicitis.

The misuse of the implication sign is even more common in modal theories (see section 2.7).

2.4 Standard First-Order Notations and Theories

At this point, we may introduce a number of standard logical and mathematical notations used in first-order theories. We expect that the reader is familiar with the concepts and notations introduced below. We go through them to fix notation and to show how they fit into formal first order theories.

Unique existence: The notation $\exists^1_\mu\, \alpha(\mu)$, where μ is a variable symbol, and $\alpha(\mu)$ is a formula with the free variable μ means "There exists a unique μ for which α holds." It may translated into the form

$$\exists_\mu\, \alpha(\mu) \wedge \forall_\nu (\alpha(\nu) \Rightarrow \nu = \mu)$$

The definite descriptor: If $\alpha(\mu)$ is a first-order formula with a free variable μ which is true of exactly one individual, then the notation $\iota(\mu)\alpha(\mu)$ is a term that denotes that unique individual. For example, the tallest building in the world is denoted

$$\iota(X)\ (\text{building}(X) \wedge \forall_Y\ (\text{building}(Y) \Rightarrow \text{height}(X) \geq \text{height}(Y)))$$

We will use expressions of the form $\iota(\mu)\alpha(\mu)$ as ordinary terms in predicate calculus formulas. We can view the use of this expression in a sentence as syntactic sugar for a more complex sentence that asserts that some unique object has the property α, and that the rest of the statement is true of that object. In general, a formula of the form "$\beta(\iota(\mu)\alpha(\mu))$", where α is an open formula, and β is a predicate symbol (possibly with other arguments as well) is syntactic sugar for

$$[\exists^1_\mu \alpha(\mu)] \wedge \forall_\mu \alpha(\mu) \Rightarrow \beta(\mu)$$

For instance, the sentence

$\text{wrote}(\iota(X)\ \text{wrote}(X,\text{ivanhoe}),\text{waverley})$

(meaning "The person who wrote Ivanhoe wrote Waverley,") is syntactic sugar for the sentence

$$[\exists^1_X\ \text{wrote}(X,\text{ivanhoe})\ ] \wedge$$
$$[\forall_X\ \text{wrote}(X,\text{ivanhoe}) \Rightarrow \text{wrote}(X,\text{waverley})\ ]$$

Note that if no one or more than one person had written Ivanhoe, then the sentence would be false. In that case, the term "$\iota(X)$ wrote(X,ivanhoe)" would be considered to be undefined (equal to $\perp$).

Sets: The notations of set theory will often be useful.[5] The basic non-logical symbol here is the membership relation $X \in S$. We also use the standard set constructor notation $\{X \mid \alpha(X)\}$ where $\alpha(X)$ is a first order formula, meaning, "The set of all X such that $\alpha(X)$." For example $\{I \mid I > 1\}$ is the set of all numbers greater than 1. The notation $\{X \mid \alpha(X)\}$ may be defined using the iota notation above:

$$\{X \mid \alpha(X)\} \ = \ \iota(S)[\forall_X\ X \in S \Leftrightarrow \alpha(X)]$$

The best known axioms for set theory are the Zermelo-Frankel axioms. However, since these use a universe containing only sets, which are therefore ultimately built up purely from the null set, they are not quite suitable for describing sets of other kinds of things. We therefore use a modification, called set theory with ur-elements; an ur-element being any entity that is not a set. We assume that we start with a universe of ur-elements, and a first-order language $\mathcal{L}_0$ for describing ur-elements. We construct a language $\mathcal{L}_s$, which contains $\mathcal{L}_0$ together with the constant symbol $\emptyset$ and the two predicate symbols: $X \in S$ (X is an element of S) and set(S) (S is a set.) Table 2.5 shows the axioms that we shall use in this set theory.

The two critical axioms here are the axiom of extensionality, which asserts that all that matters to the identity of a set are the elements it contains, and the axiom of comprehension, which (roughly) states that one can define a set corresponding to any given property. Unfortunately, Russell's paradox shows that it is incorrect to state the axiom of comprehension with quite that degree of generality; rather, the comprehension axiom must be restricted in some way. The restriction chosen here is to say that given any large set B, we can construct a set containing all the elements of B with any given property. The remaining axioms SET.3 — SET.5 exist primarily in order to allow us to construct suitably large sets. Other axioms commonly given for set theory, such as the axiom of infinity, the well-foundedness axiom, and the axiom of choice, are less important for our purposes.

In order to use the comprehension axiom to construct interesting (infinite) sets of ur-elements, we must start with some large sets of ur-elements. One possible approach is to postulate that there is a set containing all ur-elements. In this book, we will assume that for each sort of entity, there is a set containing all entities of that sort.

We augment our language of sets with the standard union, intersection, and set difference functions, and with the subset predicate. Definitions are given in table 2.6.:

Sets are particularly useful for reifying properties: If it is necessary to treat a property as an entity in its own right, one can identify the entity as the set of all objects with the property. The

[5] Of course, we have already been using sets and tuples in defining Tarskian semantics. That, however, was at the meta-level, where we are describing the language. Here we are dealing with the object-level theory of sets, where we talk about sets in a first-order language.

SET.1 $[\mathrm{set}(S1) \wedge \mathrm{set}(S2) \wedge [\forall_X\, X \in S1 \Leftrightarrow X \in S2]\ ] \Rightarrow S1 = S2$.
(Extensionality: A set is determined by the elements it contains.)

SET.2 $\neg\mathrm{set}(U) \Rightarrow \neg X \in U$.
(Ur-elements contain no elements.)

SET.3 $\forall_{X,Y} \exists_S \forall_Z\, Z \in S \Leftrightarrow [Z = X \vee Z = Y]$.
(Given any two entities X and Y, there is a set $S = \{X, Y\}$.)

SET.4 $\forall_Z\, \exists_W\, \forall_Y\, [Y \in W \Leftrightarrow \exists_{X \in Z} Y \in X]$.
(Arbitrary union: For any set Z there exists a set W which is the union of all the sets in Z.)

SET.5 $\forall_Z \exists_P \forall_X\, X \in P \Leftrightarrow [\forall_{Y \in X} Y \in Z]$
(Powerset: For any set Z there exists a set P whose elements are just the subsets of Z.)

SET.6 Let $\alpha(\mu)$ be an open formula in the language $\mathcal{L}_s$. Then the following is an axiom:
$\forall_B \exists_C \forall_X\, X \in C \Leftrightarrow [X \in B \wedge \alpha(X)]$.
(Comprehension: For any property α, there is a set C containing all the elements satisfying α within some larger set B.)

SET.7 $\mathrm{set}(\emptyset) \wedge \forall_X\, \neg X \in \emptyset$.
(Definition of the empty set.)

Table 2.5: Axioms of set theory with ur-elements

$S1 \cup S2 = \{\ X \mid X \in S1 \vee X \in S2\ \}$.
$S1 \cap S2 = \{\ X \mid X \in S1 \wedge X \in S2\ \}$.
$S1 - S2 = \{\ X \mid X \in S1 \wedge \neg(X \in S2)\ \}$.
$S1 \subseteq S2 \Leftrightarrow \forall_X\, [X \in S1 \Rightarrow X \in S2]$

Table 2.6: Boolean operators on sets

comprehension axiom guarantees that such a set exists. Note that this technique does not make it possible to discriminate between two properties that hold on exactly the same objects. (Lambda abstraction is often used for this purpose instead of set theory. The expressive power is essentially the same.)

Tuples: The k-tuple of the individuals $X_1 \ldots X_k$ is written "tuple($X_1 \ldots X_k$)" or "$< X_1 \ldots X_k >$." Various functions on tuples, such as appending two tuples, will be introduced as needed.

Operators with arbitrarily many arguments: The "tuple" function just defined and the "distinct" predicate defined in section 2.3.3, technically violate the definition of the predicate calculus, which requires that every function and predicate symbol takes some fixed number of argument. Such operators may be fitted into first-order logic in either of the following two ways:

1. Redefine the syntax and semantics of the predicate calculus to allow it.
2. For each such operator O, define a collection of operators $O_1, O_2 \ldots$, each with a specific number of arguments. Consider any use of the operator O to be syntactic sugar for the appropriate specialized operator O_k; and consider any general axiom stated for the operator O with any number of arguments to be an axiom schema for each separate specialized operator. For example, we would replace the predicate "distinct($X_1 \ldots X_k$)" by the separate predicates "distinct_2(X_1, X_2)", "distinct_3(X_1, X_2, X_3)" ...

Recursive Definitions: Recursive definitions of relations and functions are common in math and computer science. For example, the predicate "ancestor(X, Y)" meaning X is an ancestor of Y might be defined as the transitive closure of the relation "parent(X, Y)" in the following rules:

ancestor(X, X).
Everyone is (in a trivial sense) his own ancestor.

ancestor(X, Y) $\wedge$ parent(Y, Z) $\Rightarrow$ ancestor(X, Z).
If X is an ancestor of Y and Y is a parent of Z then X is an ancestor of Z.

Such recursive definitions of relations do not completely characterize the relation; they permit many different possible alternative interpretations. For instance the axioms above are consistent with interpreting "ancestor(X, Y)" as a predicate which is true if X is an ancestor of Y or if X is Cary Grant and Y is Queen Elizabeth I (who had no children). It is still true of this new relationship that everyone is an ancestor of themselves, and that if X is an ancestor of Y and Y is a parent of Z then X is an ancestor of Z.

Some of these false interpretations can be ruled out by turning the recursive definition into a biconditional. We can say "X is the ancestor of Y if and only if $X = Y$ or X is the ancestor of some Z who is the parent of Y."

ancestor(X, Y) $\Leftrightarrow$ [$X = Y \vee$ [$\exists_Z$ ancestor(X, Z) $\wedge$ parent(Z, Y)]]

This new axiom rules out the interpretation of "ancestor(X, Y)" as ANCESTOR $\cup$ { < Cary_Grant, Elizabeth_I > }. However, it does not rule out all false interpretations. For instance, assuming that everyone has a parent, it is consistent with the interpretation that "ancestor(X, Y)"

holds between all pairs of people, or with the interpretation that "ancestor(X,Y)" holds if either X is an ancestor of Y, or X is Cary Grant and Y is an ancestor of Elizabeth I.

Intuitively, we want to impose the condition that the predicate holds only in the cases where it has to hold, by virtue of the definition.[6] This cannot be done using just first-order axioms connecting "parent" and "ancestor". It can be done using set theory. We consider sets of pairs of people. We define set S to be "ancestor-like" if it satisfies the recursive condition: S contains every pair of a person with himself, and, if S contains the pair $< A, B >$, and B is the parent of C, then S contains the pair $< A, C >$. We then define the relation ancestor(X,Y) as holding just if the pair $< X, Y >$ is an element of all ancestor-like sets.

$$\begin{array}{l}
\text{ancestor}(X,Y) \Leftrightarrow \\
[\ \forall_S\ [[\forall_A\ < A, A > \in S] \wedge \\
\qquad [\forall_{A,B,C} [< A, B > \in S\ \wedge\ \text{parent}(B,C) \Rightarrow\ < A, C > \in S]] \Rightarrow \\
< X, Y > \in S]
\end{array}$$

However, this precise characterization of recursive definitions is, in practice, too complicated to be useful in a mechanical theorem prover.

2.5 Operators on Sentences

The predicate calculus gives us great facility to make all kind of statements about individuals. It does not give us a framework in which to make statements about sentences; the only things one can do with sentences are to combine and negate them with Boolean operators, and to close a formula containing a variable by adding a quantifier. By contrast, in English there are many ways in which one sentence can contain another:

"It is doubtful whether the project will succeed."

"I believe that Ford was one of our greatest Presidents."

"If the burglar had been a stranger, the dog would have barked."

"I knocked at the door because the bell was broken."

The relations between the embedded sentences, "The project will succeed," "Ford was one of our greatest Presidents," "The burglar was a stranger," "The dog barked," "I knocked at the door," and "The bell was broken," and the complete sentences that contain them are different from those provided by the predicate calculus. An attempt to express these directly in the predicate calculus leads to trouble. If the sentence, "The ball is on the table" is expressed as "on(ball1, table1)", using "on" as a predicate, then it will be syntactically incorrect to express "I believe that the ball is on the table" as "believe(me,on(ball1,table1))" using a predicate "believe", since predicates can take as an argument only a term, not a sentence like "on(ball1,table1)".

[6] Looking ahead to the non-monotonic logics to be presented in chapter 3, we may observe that applying the closed-world assumption to the predicate "ancestor" will not give the correct results; we should like our definition to make it possible to deduce that Cary Grant was not a descendant of Elizabeth I, but to remain agnostic on the unknown question of whether Cary Grant was a descendant of Homer. Rather, we wish to circumscribe the predicate "ancestor", holding "parent" fixed.

Therefore, a formal language that expresses sentences such as these must either eliminate the embedding of sentences by using a structure substantially different from the English; or provide a system in which some or all sentences may be systematically associated with primitive individuals; or extend the predicate calculus by providing additional operators on sentences. Each of these approaches may be useful under different circumstances.

Our aim in this section is to discuss general techniques for dealing with operators on sentences. The detailed analysis of particular operators will be left to the chapters dealing with their particular domains. In particular, temporal operators will be discussed further in chapter 5 and the belief and knowledge operators will be discussed further in chapter 8. We use as illustrations operators which are important in commonsense reasoning, rather than those which have been most studied in logic and philosophy. In particular, we do not use the operators "Necessarily ϕ" and "Possibly ϕ", since these have not been much used in AI domain theories. We will restrict attention to operators which have only one sentential argument (though possibly other arguments that are not sentences). Thus we will here exclude operators, like "ϕ because ψ" or "ϕ until ψ" that take two sentences as arguments.

There are a number of important formal properties of an operator $O(\phi)$, which largely determine the general properties of the representation.

1. Is it potentially necessary to apply the operator to all types of sentences, or only to some limited type of sentences? In particular, is the operator self-embedding; that is, is it sometimes necessary to apply the operator to sentences involving the operator itself? For example, the operator "X believes that ϕ" can potentially applied to any kind of sentence; virtually any kind of sentence (except those that are necessarily false) can be believed. In particular, this operator is self-embedding; "Sue believes that Jim believes that she wants to go home," is the simplest way to express that particular fact. By contrast, it is reasonable to restrict the range of the operator, "At time t, ϕ" to sentences ϕ that express the occurrence of an event, or the state of the world. This operator is not directly self-embedding; "On January 1, 1976 it was true that on November 22, 1963 Oswald shot Kennedy," is either meaningless or equivalent to "On November 22, 1963 Oswald shot Kennedy."

2. Does the operator commute with the existential and universal quantifiers? That is, is $O(\exists_X \alpha(X))$ equivalent to $\exists_X O(\alpha(X))$ and likewise for $\forall_X$? If so, then any sentence can be transformed to one in which the operator O is applied only to quantifier-free formulas, which, as we shall see, is a substantial simplification. For example, the operator "X knows that ϕ" does not commute with the quantifiers: "John knows that some people live in Schenectady," is not the same as "There are some people who John knows lives in Schenectady." The operator "At time t, ϕ," does commute with the existential quantifier, if the world is restricted so that things do not come in and out of existence. For example, if the set of objects in a domain is fixed, then "At 5:00, some object was inside the box," is equivalent to "There exists some object which was inside the box at 5:00."

If an operator O does not commute with the existential operator, then the rule "$\alpha(\tau) \Rightarrow \exists_\mu \alpha(\mu)$" may not hold if α is a formula involving O and τ is a complex term. For example, we do not wish the statement "John knows that the oldest inhabitant of Schenectady lives in Schenectady" to imply the statement "There is some person who John knows lives in Schenectady."

3. Does the operator commute with the Boolean operators? That is, is $O(\phi \vee \psi)$ equivalent to $O(\phi) \vee O(\psi)$, and is $O(\neg\phi)$ equivalent to $\neg(O(\phi))$? For example, the operator "X knows that ϕ"

does not commute with the Boolean operators; "John knows that it is not raining" is not equivalent to "It is false that John knows that it is raining." (The first implies the second but not vice versa.) The operator "At time t, ϕ" does commute with the Boolean operators. "On January 1, 1979, Bush was not President" is equivalent to "It is false that on January 1, 1979, Bush was President." Note that any operator that commutes with the Boolean operators must obey the rules of contradiction and of excluded middle: for any sentence ϕ, either $O(\phi)$ or $O(\neg\phi)$, but not both.

4. Can equal terms be substituted for one another? That is, if $X = Y$ and $O(\alpha(X))$, is it necessarily true that $O(\alpha(Y))$? A context where such substitutions may be made is said to be "referentially transparent"; one where substitution may fail is said to be "referentially opaque." For example, "X knows that ϕ"' is referentially opaque: "Oedipus knows that he is married to Jocasta," is not equivalent to "Oedipus knows that he is married to his mother," even though Jocasta is Oedipus' mother. "It is true that ϕ" is referentially transparent: "It is true that Oedipus is married to his mother," follows necessarily from "It is true that Oedipus is married to Jocasta," and "Jocasta is Oedipus' mother."

5. Is the operator closed under the rules of inference? That is, if $O(\phi_1), O(\phi_2) \ldots O(\phi_k)$ and ψ is a consequence of $\phi_1 \ldots \phi_k$, is it necessarily true that $O(\psi)$? (This property is called "consequential closure"). For example, "At time t, ϕ", is consequentially closed; if, on September 15, all members of the Cabinet met with the President, and, on September 15, Henry Kissinger was a member of the Cabinet, then it follows that, on September 15, Kissinger met with the President. The operator "X said 'ϕ'" is not closed under inference; from the facts "John said 'All members of the Cabinet are meeting with the President,'" and "John said 'Kissinger is a member of the Cabinet,'" it does not follow that "John said 'Kissinger is meeting with the President.'" Note that, if an operator is referentially transparent and commutes with the quantifiers and the Boolean operators, then it is necessarily closed under inference.

6. How useful is it to quantify over sentences? That is, is there problem-specific information[7] which is most naturally expressed in the form $\exists_\phi\, \alpha(\phi)$ or $\forall_\phi \alpha(\phi)$, where α is a formula involving O? For example, it is often useful to quantify over sentences in the context, "John said 'ϕ'", as in "John gave a speech," or "All of John's answers were correct."

In evaluating these properties for a particular operator in a particular problem domain, it is advisable to be somewhat forgiving, and to ask, "Can the logical system give useful results despite having such and such properties," rather than, "Ideally, should the operator have such and such properties." For example, it is clear that in a complete theory, belief would not be closed under inference; it simply is not true that people believe all the logical consequences of their beliefs. However, as we shall discuss in section 8.2.1, for many purposes it is acceptable and useful to take belief as closed under inference; in many applications it leads to a simple, powerful theory with many desirable properties and only a few unnatural consequences.

[7]Problem-independent information of this form can be expressed in the meta-language as axiom schemas.

2.6 Extensional Operators

An operator O on sentences is said to be *extensional* if the answers to questions (1) through (5) all indicate a simple structure: that is, O applies only to a limited class of sentences, and, in particular, does not self-embed; it commutes with the quantifiers and the Boolean operators; it is referentially transparent; and it is closed under inference. There are a number of straightforward techniques for expressing facts involving extensional operators in first-order logic.

Probably the most important extensional operator in commonsense domains is the temporal operator "At time t, ϕ". We have discussed each of the required properties of this operator in the previous section, except referential transparency. Referential transparency, the principle that equal terms may be substituted one for another, is somewhat problematic for the temporal operator. If we are not careful applying the principle, we may legitimate such erroneous inferences as "Bush is the President; in 1965, the President was a Democrat; therefore, in 1965, Bush was a Democrat."[8] The problem here is the term "the President", which denotes different things at different times. Therefore, we will begin our discussion by considering only *time-invariant* terms like "Bush", which signify the same thing under all circumstances. Further on, we will see how *time-varying* terms, like "the President" can be handled.

Let us start with a fact like "At 12:00, either the ball was on the table, or everything was in the box." The naive translation to a logic-like notation

$$\text{true\_in(t1200, on(ball1, table1)} \vee \forall_X \text{ in}(X\text{, box1))}$$

is not correct in the syntax of predicate calculus. However, we can translate the sentence to a more tractable equivalent form using the fact that all logical operators commute with the temporal operator. Thus, the above English sentence is equivalent to "Either the ball was on the table at 12:00 or, for all X, X was in the box at 12:00", which we might write

$$\text{true\_in(t1200, on(ball1, table1))} \vee \forall_X \text{ true\_in(t1200, in}(X\text{, box1))}$$

In general, we can always move the temporal operator "inside" sentences so that it is always applied directly to atomic formulas. (Note: this translation is being done purely at the conceptual level, not at the formal level. We do not yet have any formal notation. We are massaging our concepts so that they can be easily expressed in a formal notation.)

We do not yet have first-order logic. In first-order logic, the expression "true_in(t1200, on(ball1, table1))" is not a valid sentence if "on" is a predicate symbol. There are two natural approaches. The first is to change symbols such as "on" to be predicates with three arguments: the two objects and the time. "on(X, Y, T)" will mean that X is on Y at time T. Thus we can write our initial sentence

[8] The use of English sentences here is confusing, because English can use the same term to denote either a constant or a time-varying object. The clues for disambiguation are often subtle or non-existent. For example, "In 1965, the President was a Democrat," is (in its default reading) a true sentence about Lyndon Johnson, while "The President was a Democrat in 1965" is a false statement about George Bush (as of the time of writing.) An interesting case is the difference between "the King", which may refer either to the time-varying office-holder or to a constant individual, and "His Majesty", which always refers to the individual.

on(ball1, table1, t1200) $\vee \forall_X$ in(X, box1, t1200)

This is legitimate predicate calculus.

The other approach legitimates the notation "true_in(t1200, on(ball1, table1))", by positing that "on" is a function symbol, rather than a predicate symbol, so that "on(ball1, table1)" is a term rather than a sentence. The problem here is semantics: what does the term "on (ball1, table1)" denote? To answer this, we introduce a new type of individual, a "state of affairs", into our ontology. The term "on(ball1, table1)" then denotes the state of affairs of ball1 being on table1. The predicate "true_in(T, S)" thus relate a time T to a state of affairs S, and asserts that the state of affairs S obtains at time T. We can therefore write the original sentence

true_in(t1200, on(ball1, table1)) $\vee \forall_X$ true_in(t1200, in(X, box1))

If a concrete definition is desired, we can use the device of extensionalizing, and say that a "state of affairs" is a set of times: namely, the set of times when (conceptually) the state of affairs obtains. For example, "on(ball1,table1)" denotes the set of times when ball1 is on table1. Under this reading, "true_in(T, S)" is just notation for $T \in S$.

The most obvious difference between the two approaches is aesthetic. The "extra argument" approach forces us to add a somewhat unappealing extra argument to every predicate in the language describing a state of affairs or event. The "state of affairs" approach requires a somewhat mysterious extension of the ontology.

The "state of affairs" approach has the technical advantage that it makes it possible to quantify over states, to predicate properties of states, and to construct more complex terms involving states. As we shall see in chapters 5 and 9, this expressive power can be useful for more complex temporal reasoning. For example, it allows us to express a plan like, "Hammer the nail until the head is flush with the board" as the first-order term "repeat(hammer(nail1), flush(head(nail1),board7))". This representation depends critically on the state of affairs "The head of the nail is flush with the board," being a first-order entity. (See section 5.11.)

So far, we have excluded time-varying terms, whose values change with time, such as "the President of the US". We cannot yet express the statement "In 1965, the President of the US was a Democrat," using "President of the US" as a term. There are two ways to extend our system to fix this. The first is to add a time argument to time-varying functions. Thus, we would define the function "president(C, T)" as mapping a country C and a time T onto a person. We can then write

in_party(president(usa, t1965), democrat, t1965)

using the predicate "in_party" with an extra time argument, or we can write

true_in(t1965, in_party(president (usa, t1965), democrat))

using "in_party" as a function to states of affairs. Similarly, the fact "The current (1990) President was a Republican in 1965," may be expressed

in_party(president(usa, t1990), republican, t1965) or as

true_in(t1965, in_party(president(usa, t1990), republican))

The second technique explicitly uses terms that denote a “time-varying individual” or *fluent*; that is, a function from time to individuals. In this system, “president(usa)” denotes the conceptual function that maps points of time into the person who was President at that time. A general function “value_in(T, F)” takes a instant of time T and a fluent F and denotes the value of F at time T. In this approach, the fact “In 1965, the President was a Democrat,” could be written

in_party (value_in(t1965, president(usa)), democrat)

or as

true_in(t1965, in_party(value_in(t1965, president(usa)), democrat))

Note that, in any of the above notations, once we have defined a concept like “on” or “in_party” or “president” to designate a time-dependent relationship or thing, we cannot ever use the same concept in a time-independent way. The connection to time is built into the semantics. Thus, for instance, we cannot express “The ball is on the table” timelessly as “on(ball1, table1)”; we must include an explicit mention of the time or times referred to. Similarly, “Bush is the President” is not correctly represented as “bush = president(usa)” but as “bush = president(usa, t1990)” or “bush = value_in(t1990, president(usa)).” This explicit reference to time blocks erroneous inferences like, “Bush is the President; in 1965, the President was a Democrat; hence, in 1965, Bush was a Democrat.” “The President” in the first clause is represented “value_in(t1990, president(usa)),” while “the President” in the second clause is represented “value_in(t1965, president(usa)).”

These techniques can be used with any extensional operator, not just with the temporal operator. Let $O(X_1 \ldots X_k, \phi)$ be an extensional operator with sentential argument ϕ and non-sentential arguments $X_1 \ldots X_k$. The arguments $X_1 \ldots X_k$ enter into representations of O in exactly the same way as the time variable enters into the representations discussed above. Let ϕ have the form $\alpha(\tau_1 \ldots \tau_m)$; for example, if ϕ were “on(ball1, table1)”, α would be “on”, τ_1 would be “ball1”, and τ_2 would be “table1”. A sentence involving O can be represented in a first-order language in two ways:

i. Change α so that it takes $X_1 \ldots X_k$ as extra arguments in addition to $\tau_1 \ldots \tau_m$.

ii. Construe $\alpha(\tau_1 ... \tau_m)$ as a term A whose value is a “state of affairs” over the X_i. Construe the formula $O(X_1 \ldots X_k, A)$ as asserting that the state A holds on the tuple $< X_1 \ldots X_k >$. In the temporal example, X_1 would be the time instant, and A would be a temporal state, such as “on(ball1,table1)”.

The definition of “fluents” over the parameter $X_1 \ldots X_k$ is analogous.

2.7 Modal Logic

In some commonsense domains, virtually all facts can be expressed in terms of extensional operators. When this is possible, as it is in physical domains, then the “first-orderizing” techniques of the previous section yield straightforward and tractable representations. Unfortunately, it seems that some types of commonsense knowledge, particularly commonsense theories of mind, unavoidably require operators that are not extensional: operators that do not commute with quantifiers, or are referentially opaque. Incorporating these in a logic requires more powerful tools; how much more powerful depends on whether quantification over sentences is allowed. Information that does not require quantification over sentences can be expressed using modal logic or structures of possible worlds, the subjects of this section. Quantification over sentences requires the use of syntactic operators, the subject of section 2.8.

A modal logic augments predicate calculus (sometimes propositional calculus) with a number of operators, called modal operators, that take sentential arguments. As usual, the logic defines the syntax of sentences using these operators, a set of logical axioms, a set of inference rules, and a semantics. We will first discuss the syntactic aspects of typical modal logics, and then discuss their semantics.

We will confine our discussion in this section to logics that contain a single modal operator L(ϕ) and its dual M(ϕ) $\equiv \neg$L($\neg\phi$). L and M have only the one sentential argument ϕ and no other arguments. (Later in this book, we will look at more complicated modal operators and at logics which combine several modal operators.) In the most extensively studied modal logics, L(ϕ) is the operator "ϕ is necessarily true" and M(ϕ) is the operator "ϕ is not necessarily false" or, equivalently, "ϕ is possible". However, necessity and possibility have not been extensively applied to commonsense reasoning. Instead, we will use some less abstract operators as examples of L, particularly "I now know that ϕ", "I now believe that ϕ", and "ϕ is true at all times." The duals of these may easily be seen to be "I do not now know that ϕ is false," "I do not doubt ϕ", and "There is some time when ϕ is true." (In the context of the operator "ϕ is true at all times", we will here interpret the simple sentence ϕ as meaning "ϕ is true now." In section 5.12, we will study a temporal modal logic which gives a different interpretation to sentences without temporal operators.) Our aim here, as throughout this chapter, is to study logical techniques rather than to analyze specific domains; we will study theories of knowledge and belief in greater depth in chapter 8.

The syntax of modal logic is the same as the syntax of ordinary predicate calculus, except that modal operators may be applied to any formula.

Definition 7.1: A formula in a language with modal operators L and M is one of the following:

i. A predicate calculus atomic formula;

ii. Either $\neg\phi$, $\phi \vee \psi$, $\phi \wedge \psi$, $\phi \Rightarrow \psi$, $\phi \Leftrightarrow \psi$, $\phi \dot{\vee} \psi$ where ϕ and ψ are formulas.

iii. Either $\exists_\mu \phi$ or $\forall_\mu \phi$ where μ is a variable and ϕ is a formula.

iv. Either L(ϕ) or M(ϕ) where ϕ is a formula.

Definition 7.2: A sentence is a formula with no free variables.

The following are sample sentences:

$\forall_X \exists_Y$ loves(X, Y).

L [on(ball1, table1) $\wedge$ $\forall_X$ (in(X,box1) $\Leftrightarrow$ X =top1)]

$L(\neg\exists_X L(\text{spy}(X)))$

The first is simply predicate calculus; our language includes all predicate calculus sentences. If $L(\phi)$ is taken to be the operator "I know that ϕ", then the second means, "I know both that the ball is on the table and that the top is the only thing in the box." The third means, "I know that there is no one whom I know to be a spy."

Different modal operators satisfy different sets of axioms. However, most modal logics which have been studied draw their axioms from a fairly small set of standard axioms. Table 2.7 lists some of these axioms.

We will discuss each of these axioms and inference rules in turn.

Axioms MODAL.1, MODAL.2, and MODAL.3 together with modus ponens bring all of predicate calculus with equality into modal logic. Axiom MODAL.1 also ensures that tautologies of the propositional calculus still holds, even when the propositions contain modal operators. These axioms always hold, whatever the modal operators.

Axiom MODAL.4 states that existential abstraction can be performed in modal contexts if the term being abstracted is a constant symbol. For example, we can infer "There is some particular person who John knows lives in Schenectady," from "John knows that Clyde lives in Schenectady."

Axioms MODAL.3 and MODAL.4 thus do not allow the application of existential abstraction to complex terms in modal contexts. For instance, we cannot infer "There is someone that John knows lives in Schenectady" from "John knows that the mayor of Schenectady lives in Schenectady" (see exercise 10).

Axioms MODAL.5 and MODAL.6 allow all standard inferences (all inferences not involving the rule of Necessitation) to be carried out within the scope of the modal operator. MODAL.5 asserts that L applies to all the logical axioms, and MODAL.6 asserts that the inference rule modus ponens can be performed within the scope of the modal operator. From these two axioms follows the general principle of consequential closure; if $L(\phi_1)$, $L(\phi_2) \ldots L(\phi_k)$ and ψ is a logical consequence of $\phi_1 \ldots \phi_k$ then $L(\psi)$ must hold. In particular, L applies to all logical and mathematical theorems. This principle is plausible for operators like "Necessarily ϕ" or "At all times, ϕ"; all logical truths are necessarily true, and true at all times. It is not plausible for operators "I know that ϕ" or "I believe that ϕ"; people do not know all the logical consequences of their knowledge and they do not know all mathematical theorems. (More on this point in section 8.2.) Despite this implausibility, axioms MODAL.5 and MODAL.6 are part of virtually every modal logic, since it seems to be impossible to get either interesting logical conclusions or a coherent semantics without them (see section 2.7.1.)

Axioms MODAL.7 and MODAL.8 relate the strengths of $L(\phi)$, ϕ, and $M(\phi)$. Axiom MODAL.7 requires that $L(\phi)$ implies ϕ, from which it follows logically that ϕ implies $M(\phi)$. Axiom MODAL.8 is weaker, requiring only that $L(\phi)$ implies $M(\phi)$, with no connection to the truth value of ϕ. Axiom MODAL.7 is appropriate to operators like "I know that ϕ", "Necessarily ϕ", "ϕ is true at all times"; it is reasonable to posit that anything that is known is true, that anything that is necessarily true is in fact true, and that anything that is always true is true at the current moment. The equivalent form $\phi \Rightarrow M(\phi)$ gives the assertions that anything that is true cannot be known to be false, that anything that is true must be possible, and that anything true at the current moment is true at

MODAL.1 (Predicate Calculus) Axiom schemas FOL.1 through FOL.5, FOL.7 through FOL.13, and EQL.1 of first-order logic with equality are axiom schemas of modal logic. Where these schemas refer to "sentences" or "formulas", all the sentences or formulas of the modal language are included.

MODAL.2 (Existential abstraction, in non-modal contexts): Any instance of axiom schema FOL.6, where the sentences have no modal operators, is an axiom of modal logic.

MODAL.3 (Substitution of equals): Any instance of axiom schema EQL.2 is an axiom of modal logic.

MODAL.4 (Existential abstraction of constants in modal contexts: Let a be a constant or variable symbol, let μ be a variable, let $\alpha(\mu)$ be a formula, and let $\alpha(\mu/a)$ be a formula identical to $\alpha(\mu)$, except that a is substituted for every free occurrent of μ. Any closure of the formula $\alpha(\mu/a) \Rightarrow \exists_\mu \alpha(\mu)$ is an axiom. Note that this axiom applies when the formula α contains modal operators, but it does not apply to substitution of complex terms, other than constant symbols.

(In theories that allow constant symbols to be non-rigid designators, this axiom must be restricted to constants that represent rigid designators. See section 2.7.1.)

MODAL.5 If ϕ is a logical axiom, then $\mathrm{L}(\phi)$ is an axiom.

In the remaining axioms, let ϕ and ψ be formulas, and let μ be a variable. Then any closure of the following formulas may be an axiom:

MODAL.6 (Consequential Closure) $(\mathrm{L}(\phi) \wedge \mathrm{L}(\phi \Rightarrow \psi)) \Rightarrow \mathrm{L}(\psi)$.

MODAL.7 (Veridicality) $\mathrm{L}(\phi) \Rightarrow \phi$.

MODAL.8 $\mathrm{L}(\phi) \Rightarrow \mathrm{M}(\phi)$.

MODAL.9 $\mathrm{L}(\phi) \Rightarrow \mathrm{L}(\mathrm{L}(\phi))$.

MODAL.10 $\mathrm{M}(\phi) \Rightarrow \mathrm{L}(\mathrm{M}(\phi))$.

MODAL.11 (Barcan axiom). $\forall_\mu \mathrm{L}(\alpha) \Rightarrow \mathrm{L}(\forall_\mu \alpha)$

MODAL.12 $\mathrm{L}(\forall_\mu \alpha) \Rightarrow \forall_\mu \mathrm{L}(\alpha)$

MODAL.13 (Definitional equivalence) $\mathrm{M}(\alpha) \Leftrightarrow \neg \mathrm{L}(\neg \alpha)$

There are two rules of inference generally used:
Modus Ponens: From ϕ and $\phi \Rightarrow \psi$, infer ψ.
Necessitation: From ϕ infer $\mathrm{L}(\phi)$.

Table 2.7: Axioms of Modal Logic

some time. Axiom MODAL.8 is appropriate to operators like "I believe that ϕ", "It is obligatory that ϕ" or "ϕ will be true at all future times". It is reasonable to posit that, if ϕ is believed true, it is not believed false; that if it is obligatory that ϕ be true, then it cannot also be obligatory that ϕ be false; and that, if ϕ is true at all future times, then it cannot be false at all future times.

MODAL.9 and MODAL.10 relate to iterated modalities. MODAL.9 states that L(ϕ) implies L(L(ϕ)), or, equivalently, that M(M(ϕ)) implies M(ϕ). With knowledge, this is the principle, "If I know ϕ then I know that I know ϕ." MODAL.9 is plausible for knowledge and for most other modal operators. In combination with MODAL.6, MODAL.9 implies that iterated L's are equivalent to a single L and that iterated M's are equivalent to a single M. MODAL.10 together with MODAL.6 implies that any string of iterated modal operators is equivalent to the innermost; for example, L(M(L(L(M(ϕ))))) is equivalent to M(ϕ). This principle is plausible like an operator like "It is always true that ϕ"; if the statement "At all times, it is true that at some times it is true that ϕ," means anything at all, it can only mean the same thing as "At some times it is true that ϕ." The rule "If I do not believe ϕ then I believe that I do not believe it," is often plausible (though note that, combined with the rule of consequential closure, it leads to very strong results); the rule, "If I do not know ϕ, then I know that I don't know it" is much less plausible, but occasionally useful. MODAL.10 is demonstrably false for the operator "Provably ϕ"; it is known that there are statements that are unprovable, but which cannot be proven to be unprovable.

Axiom MODAL.11 (known as the "Barcan" axiom, after the philosopher Ruth Barcan Marcus) asserts that, if a modality applies to every instance of a proposition, then it applies to the universal generalization. For example, let L be the operator "John knows that ϕ" and let α be the formula "If X is a rhinoceros then X has a horn." Then the Barcan formula $\forall_X \mathrm{L}(\alpha) \Rightarrow \mathrm{L}(\forall_X \alpha)$ means that, if you know about each X in the world that either it is not a rhinoceros or it has a horn, then you know the proposition "For all X, if X is a rhinoceros, then it has a horn." For psychological operators, such as "know" or "believe" this seems to be a safe inference for most α, since it is very rare to know something about every individual in the world without knowing the general rule. (This argument may not be valid in a sorted logic.) However, there are a few exceptions. Let $\alpha(N)$ be the formula, "If N is an integer, then T does not halt after N steps" where T is a Turing machine which never halts. Then it follows from the law of consequential closure that one knows every instance of $\alpha(N)$; however, it may not be true that one knows the general rule.[9]

MODAL.12 is the converse of MODAL.11. It states that if L holds on a general rule, then it holds on every instance. It follows from consequential closure that $\mathrm{L}(\forall_\mu \alpha(\mu)) \Rightarrow \mathrm{L}(\alpha(\tau))$ for any ground term τ. Axiom MODAL.12 is a slightly stronger statement. MODAL.12 is taken as axiomatic in any system that accepts consequential closure. (Many modal logics allow formulas with free variables to be axioms in their own right, and adopt the inference rule of universal generalization, "Infer $\forall_\mu \phi$ from ϕ. In such a logic, MODAL.12 follows directly from consequential closure.)

MODAL.13 is just the definition of the modal operator M in terms of L. It is an axiom in all modal logics.

The inference rule Necessitation, "From ϕ infer L(ϕ)" is rather curious. The intention is, essentially, to replace axiom MODAL.5 with a slightly stronger statement; all logical theorems are necessarily true. In a theory without proper axioms, ϕ can be inferred as true only if it is a logical

[9]Thanks to Larry Manevitz for pointing out this example to me.

theorem, so this rule will be legitimate. It is not, of course, legitimate in a theory with proper axioms; we do not want to infer "Necessarily, John is bald," from the proper axiom, "John is bald." Modal logics are often formulated without considering proper axioms; AI, however, is primarily concerned with theories that do have proper axioms.

Thus, the original motivation behind this inference rule disappears in the AI context. Nonetheless, the inference rule is still useful for some operators even in theories with proper axioms. For example, it is reasonable to infer either "I know that ϕ" or "I believe that ϕ" from ϕ. That is, if you have somehow gotten ϕ into your knowledge base, then you can infer that you know ϕ or that you believe it. On the other hand, there are many operators where the inference rule is obviously false, for proper axioms; for instance, there is no inference from "ϕ is true now" to "ϕ is always true."

It should be noted that the logical literature on modal logic generally accepts the necessitation rule without question. This does not invalidate the use of modal logics for operators where the rule does not apply, but it does mean that standard theorems must be used with some caution.

The necessitation rule $\phi \vdash \mathrm{L}(\phi)$ is by no means equivalent to the implication "$\phi \Rightarrow \mathrm{L}(\phi)$", which is never true for a useful operator L. "$\phi \Rightarrow \mathrm{L}(\phi)$" is the truth-value relation "Either ϕ is false or $\mathrm{L}(\phi)$ is true"; applied to the "Know" operator, for example, it would state "I know all true sentences." In general, if $\phi \Rightarrow \mathrm{L}(\phi)$, then, since $\mathrm{L}(\phi) \Rightarrow \phi$, by MODAL.7, it follows that ϕ is equivalent to $\mathrm{L}(\phi)$; i.e. the L operator is useless. The difference between the implication and the inference rule is that the implication $\phi \Rightarrow \psi$ means that ψ is true whenever ϕ is true, while the inference $\phi \vdash \psi$ means only that ψ is true whenever ϕ may be inferred, a much stronger condition.

The use of the necessitation rule and the restrictions placed on existential abstraction (axiom MODAL.4) make the construction of proofs in modal logic substantially different than in first-order logic. In general, one has to be careful using intuitions built up in ordinary logic; they may lead to invalid results. In particular the following standard proof techniques of FOL are not valid in modal logic:

- Discharging: If $\phi \vdash \psi$ — that is, ψ may be proven from the hypothesis ϕ — then infer $\phi \Rightarrow \psi$.
- Splitting: If $\phi \vdash \psi$ and $\zeta \vdash \psi$, infer that $\phi \vee \zeta \vdash \psi$. An example where this fails in modal logic: For any sentence "p", $\mathrm{p} \vdash \mathrm{L}(\mathrm{p})$ (Necessitation), so $\mathrm{p} \vdash (\mathrm{L}(\mathrm{p}) \vee \mathrm{L}(\neg\mathrm{p}))$ (MODAL.1). Similarly $\neg\mathrm{p} \vdash \mathrm{L}(\neg\mathrm{p})$ and so $\neg\mathrm{p} \vdash \mathrm{L}(\mathrm{p}) \vee \mathrm{L}(\neg\mathrm{p})$. But it is not the case that $(\mathrm{p} \vee \neg\mathrm{p}) \vdash \mathrm{L}(\mathrm{p}) \vee \mathrm{L}(\neg\mathrm{p})$.

As mentioned above (section 2.3.3), in modal contexts it is particularly easy to over-interpret material implication $p \Rightarrow q$ as meaning more than just "p is false or q is true"; such a mistake can lead to bad trouble. For example, let $\mathrm{L}(\phi)$ be the modal operator "ϕ is provable," and suppose we wish to express the statement, "If ψ follows from ϕ and ϕ is provable, then ψ is provable." The temptation is to express the first clause of this rule "ψ follows from ϕ" as the implication "$\phi \Rightarrow \psi$", and so to express the rule as the axiom schema

$$[[\ \phi \Rightarrow \psi] \wedge \mathrm{L}(\phi)\ ] \Rightarrow \mathrm{L}(\psi)$$

But this rule is wrong. The sentence "$\phi \Rightarrow \psi$" does not mean "ψ follows from ϕ"; it means only that ψ is true or ϕ is false.

Various sets of the above axioms on modal logics have been singled out for study by logicians. The best known are the systems "T", which contains MODAL.1-8 and the rule of necessitation; "S4", which adds MODAL.9 to T; and "S5", which adds MODAL.10 to S4.[10]

A sample proof in modal logic is given in Table 8.3, at the end of section 8.2.2.

2.7.1 Possible Worlds Semantics

The meaning of a first-order language is defined in terms of a Tarskian semantics. An interpretation for the language is an association of each constant, function, and predicate symbol in the language with an individual, mapping, or set in the world. The semantics then specifies the meaning of every term in the language, and the truth conditions for every sentence.

There is no simple way to extend a Tarskian semantics to define the meaning of modal operators. A Tarskian semantics is inherently incapable of distinguishing between two terms that refer to the same object, or between two sentences with the same truth value. For example, there is no way of defining a Tarskian semantics so that "He knows that the shortest spy is the shortest spy," and "He does not know that Ralph Ortcutt is the shortest spy," are both true, given Ralph Ortcutt is indeed the shortest spy. "Ralph Ortcutt" and "The shortest spy" will both map to the same individual, and either term can be substituted for the other in any sentence. It is not even possible to fix things so that "He knows that snow is white," and "He does not know that President Harding's middle name was Gamaliel," are both true, given that both embedded sentences are true.

The semantics for modal languages requires a more complex type of model, known as a Kripke structure (after Saul Kripke). A Kripke structure consists of a collection of *possible worlds,* connected by *accessibility* relations. Each possible world represents one way that the world could possibly be; it is one complete model of the language without modal operators. Thus, each world specifies the truth or falsehood of every non-modal sentence in a consistent way. The real world is one particular possible world.

The modalities are incorporated in the accessibility relations. There is a separate accessibility relation for each modal operator with non-sentential arguments. Thus, there is an accessibility relation corresponding to the modal operator "John believes ϕ"; a relation corresponding to the operator "Mary believes ϕ"; a relation corresponding to the operator "John knows ϕ"; and so on. The semantics for the modal operators L and M are given by the following rules:

- L(ϕ) is true in a world $\mathcal{W}$ iff ϕ is true in all worlds accessible from $\mathcal{W}$.
- M(ϕ) is true in a world $\mathcal{W}$ iff ϕ is true in some world accessible from $\mathcal{W}$.

For example, let L(ϕ) be the operator "John believes ϕ." We define the following accessibility relation $\mathcal{A}$ between possible worlds: $\mathcal{W}_1$ is accessible from $\mathcal{W}$ just if $\mathcal{W}_1$ is consistent with all the beliefs that John holds in $\mathcal{W}$. One can imagine taking John, as he is in the world $\mathcal{W}$, and showing him other worlds one by one. If he finds something in a world that violates his beliefs, then that

[10]This nomenclature is purely historical. "T" was introduced in [Feys, 37], and given its name in [Sobocinski, 53]. "S4" and "S5" were introduced and named in [Lewis and Langford, 32]. Lewis and Langford also introduced systems named S1, S2, and S3, and subsequent papers have defined and named slews of other systems; but none of these has become as popular.

world is not accessible; if everything in the world looks reasonable, then the world is accessible. When we are done, any statement which John believes true will be true in all accessible worlds; any statement which he believes false will be false in all accessible worlds; any statement about which he has no opinion will be true in some worlds and false in others. Thus, the statements that are true in all worlds are just those that he believes, as stated in the rule above. (Figure 2.1)

Using the above rule, we can translate iterated modal operators into chains of accessibility relations. For example, "Mary knows that John believes that taking vitamin C prevents cancer," translates to, "If a world $\mathcal{W}_1$ is accessible from the real world via Mary's knowledge, then in $\mathcal{W}_1$ it is true that John believes that taking vitamin C prevents cancer." This, in turn, translates to, "If $\mathcal{W}_1$ is accessible from the real world via Mary's knowledge, and $\mathcal{W}_2$ is accessible from $\mathcal{W}_1$ via John's belief, then it is true in $\mathcal{W}_2$ that taking vitamin C prevents cancer."

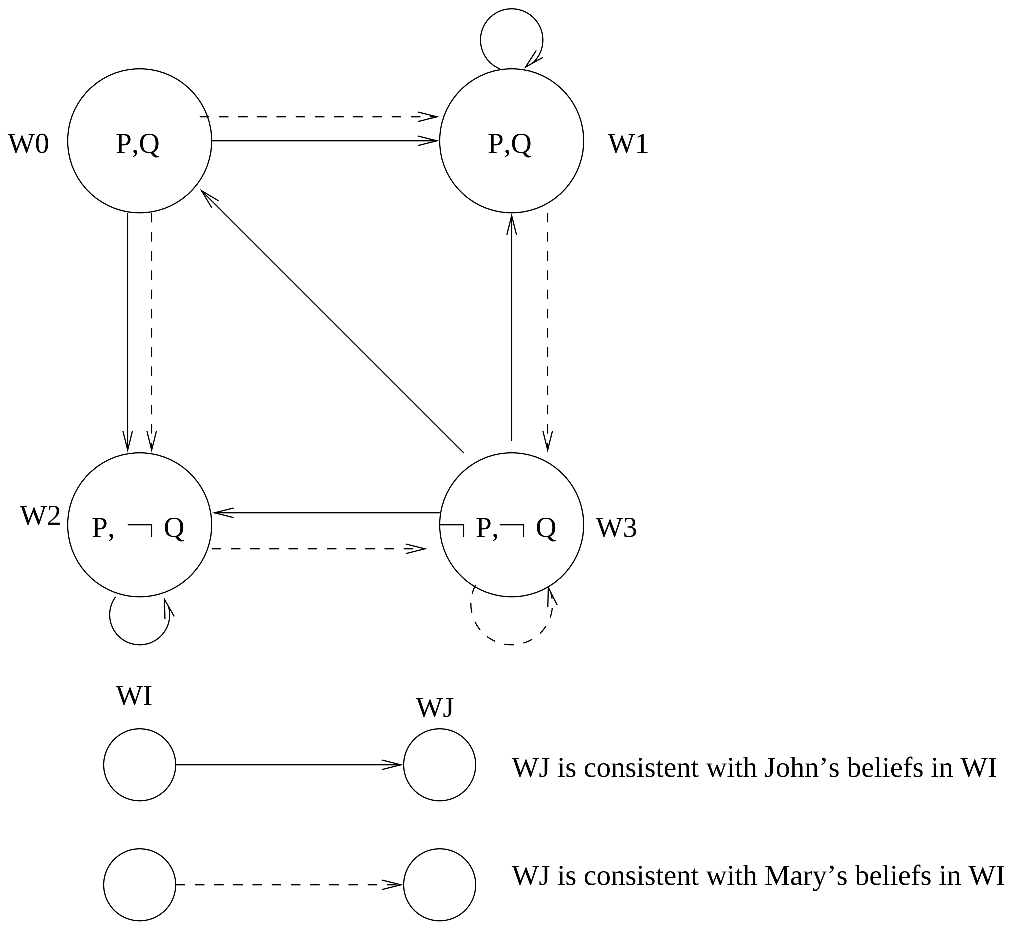
This kind of semantic definition helps clarify where the formal properties of the modal logic "come from". In this semantics, the operator L(ϕ) is essentially a universal quantifier over accessible worlds, and M(ϕ) is an existential quantifier; and their formal properties in the logic are similar to the properties of the quantifiers. Thus, for example, the facts that L(ϕ) does not commute with negation, disjunction, or existential quantification — "John knows that it is not raining" is not equivalent to "It is false that John knows that it is raining;" "John knows that it is either raining or sunny" is not equivalent to "Either John knows it is raining or he knows that it is sunny;" "John knows that there are citizens of Mozambique," is not equivalent to "There are people who John knows are citizens of Mozambique;" — are explained by the corresponding facts about the universal quantifier: $\forall_\mu \neg\alpha(\mu)$ is not equivalent to $\neg\forall_\mu \alpha(\mu)$; $\forall_\mu (\alpha(\mu) \vee \beta(\mu))$ is not equivalent to $(\forall_\mu \alpha(\mu)) \vee (\forall_\mu \beta(\mu))$; $\forall_\mu \exists_\nu \alpha(\mu, \nu)$ is not equivalent to $\exists_\nu \forall_\mu \alpha(\mu, \nu)$.

(Note: the remainder of section 2.7.1 involves rather abstract logic. It may be skipped without loss of continuity. It is necessary to read section 2.3.2 before reading this section.)

To express this semantics formally, we change the interpretation function to take two parameters: the sentence or term being interpreted, and the possible world referred to. A constant symbol always refers to the same entity in all possible worlds (see section 2.7.3), but function and relation symbols may change their extension. For example, if Rosamond is married to Lenny in one possible world but not in another, then the extension of the relation "married_to" includes the pair < rosamond, lenny > in the first, but not in the second. We will write $\mathcal{I}(\gamma, \mathcal{W}_i)$ to mean the meaning of the non-logical symbol γ in the world $\mathcal{W}_i$ under interpretation $\mathcal{I}$; this is an individual, function, or relation, depending on whether γ is a constant symbol, function symbol, or relation symbol. We then define the concept "ϕ is true in world $\mathcal{W}_i$ under interpretation $\mathcal{I}$." If ϕ is a sentence without a modal operator, ϕ is true in $\mathcal{W}_i$ just if it satisfies the usual Tarskian definition, relative to the interpretations of its symbols in $\mathcal{W}_i$. If ϕ is a sentence of the form L(ψ), then ϕ is true in a world just if ψ is true in all accessible worlds. Correspondingly, a sentence M(ϕ) is true in world $\mathcal{W}_i$ just if ϕ is true in some accessible world $\mathcal{W}_j$. The truth value of a compound sentence is defined in the usual way.

We will now give a formal definition of the semantics of modal logic with dual modal operators L(ϕ) and M(ϕ).

Definition 7.3: A *Kripke structure* consists of four elements:


W0
P,Q
P,Q
W1
W2
P, ¬ Q
¬ P,¬ Q
W3
WI
WJ
WJ is consistent with John's beliefs in WI
WJ is consistent with Mary's beliefs in WI

In W0: John believes P.
John neither believes Q nor believe ¬Q.
Mary believes Q $\Rightarrow$ P.
John believes that Mary believe ¬Q.
Mary believes that John believes P.

Figure 2.1: Possible worlds

a. A set of possible worlds, $\mathcal{W}$.

b. A binary relationship on the worlds $\mathcal{A}(\mathcal{W}_i, \mathcal{W}_j)$, read "$\mathcal{W}_j$ is accessible from $\mathcal{W}_i$". $\mathcal{A}$ has the property that, for any world $\mathcal{W}_i$, there exists at least one world $\mathcal{W}_j$ such that $\mathcal{A}(\mathcal{W}_i, \mathcal{W}_j)$. This condition is called "seriality."

c. A distinguished world $\mathcal{W}_0 \in \mathcal{W}$ (the real world).

d. A domain of individuals $\mathcal{D}$.

Definition 7.4: Let $\mathcal{L}$ be a modal language, let Γ be the set of non-logical symbols in $\mathcal{L}$, and let $\mathcal{K} = < \mathcal{W}, \mathcal{A}, \mathcal{W}_0, \mathcal{D} >$ be a Kripke structure. An interpretation $\mathcal{I}$ of $\mathcal{L}$ over $\mathcal{K}$ is a function with two arguments: a symbol $\gamma \in \Gamma$ and a world $\mathcal{W}_i \in \mathcal{W}$. The function $\mathcal{I}$ has the following properties:

a. If γ is a constant symbol, then there exists some individual $d \in \mathcal{D}$ such that, for all worlds $\mathcal{W}_i \in \mathcal{W}$, $\mathcal{I}(\gamma, \mathcal{W}_i) = d$.

b. If γ is a k-place function symbol, then for each world $\mathcal{W}_i \in \mathcal{W}$, $\mathcal{I}(\gamma, \mathcal{W}_i)$ is an extensional function from $\mathcal{D}^k$ to $\mathcal{D}$.

c. If γ is a k-place predicate symbol, then for each world $\mathcal{W}_i \in \mathcal{W}$, $\mathcal{I}(\gamma, \mathcal{W}_i)$ is an extensional relation from $\mathcal{D}^k$ to $\mathcal{D}$.

Note that this definition creates a strong distinction between constant symbols, which always represent the same individual, and 0-place function symbols, which may denote different individuals in different possible worlds.[11] What exactly is meant by "the same individual in different possible worlds" is a subtle issue, that we will address in section 2.7.3.

We extend the interpretation $\mathcal{I}$ to complex terms and sentences using the following definitions.

Definition 7.5: Let τ be a complex ground term in $\mathcal{L}$ of the form $\beta(\tau_1 \ldots \tau_k)$. Let $\mathcal{I}$ be an interpretation of $\mathcal{L}$, and let $\mathcal{W}_i$ be a possible world. $\mathcal{I}(\tau, \mathcal{W}_i)$, read "the denotation of τ in world $\mathcal{W}_i$" is the image of the denotations of $\tau_1 \ldots \tau_k$ under the function $\mathcal{I}(\beta, \mathcal{W}_i)$.

$$< \mathcal{I}(\tau_1, \mathcal{W}_i) \ldots \mathcal{I}(\tau_k, \mathcal{W}_i), \mathcal{I}(\beta(\tau_1 \ldots \tau_k), \mathcal{W}_i) > \in \mathcal{I}(\beta, \mathcal{W}_i)$$

Definition 7.6: Let ϕ be a ground atomic formula $\gamma(\tau_1 \ldots \tau_k)$ in $\mathcal{L}$, where γ is a predicate symbol. $\mathcal{I}(\phi, \mathcal{W}_i)$=TRUE just if the relation $\mathcal{I}(\gamma, \mathcal{W}_i)$ holds on the objects $\mathcal{I}(\tau_1, \mathcal{W}_i) \ldots \mathcal{I}(\tau_k, \mathcal{W}_i)$; that is, if

$$< \mathcal{I}(\tau_1, \mathcal{W}_i) \ldots \mathcal{I}(\tau_k, \mathcal{W}_i) > \in \mathcal{I}(\gamma, \mathcal{W}_i)$$

Definition 7.7: Let ϕ be a sentence in $\mathcal{L}$ of the form "$\neg\psi$" or "ψ op ζ", where "op" is a Boolean operator. $\mathcal{I}(\phi, \mathcal{W}_i)$=TRUE if the truth-value conditions associated with the Boolean operator in definition 2.2 hold for the two truth values $\mathcal{I}(\psi, \mathcal{W}_i)$ and $\mathcal{I}(\zeta, \mathcal{W}_i)$.

Definition 7.8: For any interpretation $\mathcal{I}$, object $u \in \mathcal{D}$, and symbol $\delta \notin \mathcal{L}$, define $\mathcal{I}^{\delta \rightarrow u}$ as in definition 3.10:

[11] Some formulations of possible-worlds semantics (e.g. [Moore, 1980], [Genesereth and Nilsson, 1987]) allow constant symbols to be specified as either rigid designators, which denote the same individual in all possible worlds, or as non-rigid designators, which may denote different things in different possible worlds. I am not aware of any advantages of this approach.

a. Let ϕ be a sentence in $\mathcal{L}$ of the form $\forall_\mu \alpha(\mu)$.
Then $\mathcal{I}(\phi, \mathcal{W}_i)$=TRUE just if $\mathcal{I}^{\delta \to u}(\alpha(\mu/\delta), \mathcal{W}_i)$=TRUE for every $u \in \mathcal{D}$.

b. Let ϕ be a sentence in $\mathcal{L}$ of the form $\exists_\mu \alpha(\mu)$.
Then $\mathcal{I}(\phi, \mathcal{W}_i)$=TRUE just if $\mathcal{I}^{\delta \to u}(\alpha(\mu/\delta), \mathcal{W}_i)$=TRUE for some $u \in \mathcal{D}$.

Definition 7.9:

a. Let ϕ be a sentence in $\mathcal{L}$ of the form L(ψ).
Then $\mathcal{I}(\phi, \mathcal{W}_i)$=TRUE just if $\mathcal{I}(\psi, \mathcal{W}_j)$=TRUE for every $\mathcal{W}_j$ such that $\mathcal{A}(\mathcal{W}_i, \mathcal{W}_j)$.

b. Let ϕ be a sentence in $\mathcal{L}$ of the form M(ψ).
Then $\mathcal{I}(\phi, \mathcal{W}_i)$=TRUE just if $\mathcal{I}(\psi, \mathcal{W}_j)$=TRUE for some $\mathcal{W}_j$ such that $\mathcal{A}(\mathcal{W}_i, \mathcal{W}_j)$.

Definition 7.10: A sentence ϕ in $\mathcal{L}$ is true (simply) if it is true in $\mathcal{W}_0$.

All of the modal axioms MODAL.1 - MODAL.13, except MODAL.7, MODAL.9, and MODAL.10 follow necessarily from this possible worlds semantics. Consider, for example, MODAL.6, the law of consequential closure, L(ϕ) $\wedge$ L($\phi \Rightarrow \psi$) $\Rightarrow$ L(ψ). In a possible-worlds semantics, this amounts to saying, "If ϕ is true in all accessible worlds, and $\phi \Rightarrow \psi$ is true in all accessible worlds, then ψ is true in all accessible worlds." $\phi \Rightarrow \psi$ is true in a world just if either ϕ is false or ψ is true. Thus, the axiom is clearly true. Similar simple arguments can be made to justify the other axioms. Moreover, this collection of axioms, together with the inference rules of modus ponens and necessitation, is complete for this possible worlds semantics; that is, if a sentence is true in all structures of possible worlds then it is provable from the axioms [Kripke, 1963a].

The truth of MODAL.7, MODAL.9, and MODAL.10 depends on particular constraints placed on the accessibility between worlds. MODAL.7, the axiom L(ϕ)$\Rightarrow\phi$, states that if ϕ is true in all worlds accessible from a world $\mathcal{W}_i$, then it is true in $\mathcal{W}_i$ itself. This is justified if every world is accessible from itself; i.e. if the accessibility relation is reflexive. To interpret MODAL.9, the axiom L(ϕ) $\Rightarrow$ L(L(ϕ)), we observe that L(ϕ) means that ϕ is true in all worlds accessible from $\mathcal{W}_i$, and that L(L(ϕ)) means that L(ϕ) is true in all worlds $\mathcal{W}_j$ accessible from $\mathcal{W}_i$, and hence that ϕ is true in any world $\mathcal{W}_k$ such that $\mathcal{W}_k$ is accessible from $\mathcal{W}_j$ and $\mathcal{W}_j$ is accessible from $\mathcal{W}_i$. Hence, the axiom L(ϕ) $\Rightarrow$ L(L(ϕ)) will be true if $\mathcal{A}(\mathcal{W}_i, \mathcal{W}_j)$ and $\mathcal{A}(\mathcal{W}_j, \mathcal{W}_k)$ imply $\mathcal{A}(\mathcal{W}_i, \mathcal{W}_k)$; that is, the accessibility relation is transitive. In a similar way, axioms MODAL.10 and MODAL.7 are together justified if the accessibility relationship is an equivalence relation.

2.7.2 Direct Use of Possible Worlds

Thus far, we have used possible worlds as a meta-level theory, to give a coherent semantics to a modal language. However, it is often possible to eliminate the modal language altogether, and to use a object-level language that directly refers to possible worlds and their accessibility relations. This allows us to capture the content of modal sentences in a first-order language, to which we can apply standard computational techniques for first-order inference.[12] The resultant language

[12]The construction described in this section was first developed in [Moore, 1980]. Moore used an "extra argument" notation, while we shall use a "state of affairs" notation.

looks very much like the extensional languages discussed in section 2.6, with the addition of the "accessibility" relation. However, the relation of the representation to the original sentence operator is more complex here. In the theories discussed in section 2.6, the "possible worlds" entered as a simple argument. "The ball was on the table at noon" was represented as "on(ball1, table1, t1200)" or "true_in(t1200, on(ball1, table1))." In the theories we will discuss here, we capture modal operators such as "believe" using quantification over accessible worlds. "John believes that the ball is on the table," becomes "The ball is on the table in all worlds accessible via John's belief." Introducing a first-order predicate "bel_acc($A, W0, W1$)", meaning that world $W1$ is accessible from world $W0$ relative to the beliefs of A, we can represent this either as,

$$\forall_{W1} \text{ bel\_acc(john,w0,}W1) \Rightarrow \text{on(}W1\text{, ball1, table1)}$$

or

$$\forall_{W1} \text{ bel\_acc(john,w0,}W1) \Rightarrow \text{true\_in(}W1\text{, on(ball1, table1))}$$

where w0 is the real world.

For example, consider the sentential operator "X believes that ϕ". Let $\mathcal{L}$ be a modal language with some non-logical symbols and the modal operator "X believes that ϕ". Let $\mathcal{L}_0$ be the first-order language over domain $\mathcal{D}_0$ with all the non-logical symbols of $\mathcal{L}$. The equivalent language of possible worlds $\mathcal{L}_p$ will involve the following:

i. The individuals in the universe must include

   a. People (the first argument of the "believes" operator).
   b. All individuals in $\mathcal{D}_0$
   c. Possible worlds. The symbol w0 represents the real world.
   d. States of affairs over possible worlds. Extensionally, we can view a state of affairs as just a set of possible worlds; those in which the state obtains.
   e. Fluents over possible worlds. Extensionally, we can view a fluent as a function from possible worlds to individuals in $\mathcal{D}_0$.

ii. The non-logical symbols of $\mathcal{L}_p$ include the following:

   a. All constant symbols of $\mathcal{L}_0$ are constant symbols of $\mathcal{L}_p$.
   b. Any function symbol of $\mathcal{L}_0$ is a function symbol of $\mathcal{L}_p$. Its meaning, however, is changed. The value of the term "father_of(john)" is no longer a person; it is the fluent that denotes, in each world, the father of John in that world. The father of John in a particular world W is value_in(W,father_of(john)). In particular, John's real father is value_in(w0,(father_of(john)). The function "father_of(X)" thus represents a function whose argument X is either a person (such as "john") or a fluent from possible worlds to persons (such as "mother_of(bob)"), and whose value is a fluent from possible worlds to persons.

c. Each relation symbol of $\mathcal{L}_0$ becomes a function symbol of $\mathcal{L}_p$. An atomic formula of $\mathcal{L}_0$, like "on(block1,table)", is a term of $\mathcal{L}_p$. It denotes the state of affairs in which the block is on the table. To say that the block is on the table in some particular world W, we write "true_in(W,on(block1,table))." In particular, to say that the block is really on the table, we write "true_in(w0,on(block1,table))".

d. The function "value_in(W, F)" takes a fluent F from possible worlds to entities, and a possible world W, and returns the value of F in W. For instance, "value_in(W,father_of(john))" is John's father in world W.

e. The relation "true_in(W, A)" means that the state of affairs obtains in possible world W. For instance, "true_in(W,on(block1,table))" means that the block is on the table in world W.

f. The relation bel_acc(P, W_1, W_2) is the relation, "W_2 is accessible from W_1 in the beliefs of P."

g. The constant symbol "w0" represents the real world.

Any statement in the modal logic can be translated into this new language, using the possible worlds semantics for the modal logic. For example, the statement, "John believes that Mary believes that all tall people have tall fathers," becomes

$$\forall_{W1,W2}\ \text{bel\_acc(john,w0,}W1) \wedge \text{bel\_acc(mary,}W1, W2) \Rightarrow$$
$$\forall_X\ [\ \text{true\_in(}W2\text{,tall(}X)) \Rightarrow \text{true\_in(}W2\text{,tall(value\_in`(}W2\text{,father\_of(}X)))\ ]$$

2.7.3 Individuals and Modality

The difficult and dubious parts of modal logic lie in the treatment of constant symbols and quantified variables inside the scope of modal operators. A thorough treatment of this topic involves many difficult technical and philosophical issues. Within the scope of this chapter, we can only point out some of the important issues involved.

Cross World Identification: One strength of a possible worlds semantics is that it greatly clarifies the interrelation of quantifiers and modal operators. "John believes that someone wrote Waverley," means "In each accessible world, there exists a person who wrote Waverley." "There is someone whom John believes to have written Waverley," means "There exists an individual who, in every accessible world, wrote Waverley." The difference between the two reduces to a difference in the ordering of quantifiers. In the first form, often called *"de dicto"* modality, the author of Waverley may vary from one world the next; in the second, called *"de re"* modality, the same person wrote Waverley in all worlds.

This notion of "the same entity in all possible worlds" is a fundamental part of the possible worlds semantics. In particular, our definition of an interpretation required that any constant symbol represent a fixed entity in all possible worlds. This is necessary if we are to perform existential abstraction over constant symbols, to deduce "There is someone whom John believes to have written Waverley," from "John believes that George IV wrote Waverley."

But, though the formal properties of the idea are clear, it is not at all clear what it means. It can be hard enough, even in principle, to identify "the same thing" over time and circumstance; how are we to do so over the range of imaginable universes? Indeed, one can easily dream up situations in which the question of identity becomes obviously unanswerable or meaningless. If Mrs. Bonaparte's oldest child had been a girl, would she have been Napoleon? If France had invaded Germany in 1938 in response to the Anschluss, would that have been World War II? If the Founding Fathers had decided to place the national capital on the banks of the Connecticut River, would that city be Washington, D.C.? These problems seem silly, but they are hard to avoid in developing this kind of logic.

There is another, more subtle kind of problem with this notion. If objects are to be identified across possible world, then we must distinguish between *rigid designators*, terms whose denotation does not change from one world to the next, and *non-rigid designators,* terms which denote different objects in different possible worlds. For example, in the sentence "John believes that Scott wrote Waverley," the term "Scott" is a rigid designator. Therefore this sentence implies the *de re* modal statement, "There is someone whom John believes wrote Waverley," which we are glossing as "Waverley was written by a single person in every world consistent with John's belief," that single person being Scott. By contrast, in the sentence "John believes that the author of Waverley wrote Waverley," the term "the author of Waverley" is a non-rigid designator. Therefore, this sentence does not imply the above *de re* sentence; John's beliefs (as far as we have specified them) are consistent with any of a number of people having written Waverley. The sentence "John believes that the author of Waverley wrote Waverley," implies only the *de dicto* sentence, "John believes that someone wrote Waverley."

However, in practice, this distinction between rigid and non-rigid designators seems to be very hard to make, and, often, not worth making. Consider the following scenario: One morning you get an anonymous letter, threatening to publish compromising photographs. The letter is followed by a phone call. You meet with the blackmailer face to face, and make appropriate arrangements. Some months later, you see a photograph of the blackmailer, identified as Ralph Norbertson, in the paper. The question is, at what point can you claim to have a rigid designator that denotes this man? That is, at what point can you make statement of *de re* knowledge about him, like, "There is someone who I know sent me a blackmailing letter?" It seems clear that there is no dividing line, no clear criterion for distinguishing how much and what kind of contact is needed for *de re* knowledge. But, unfortunately, it makes a great deal of difference in terms of the logic. If we grant that you have *de re* knowledge of the blackmailer after seeing him face to face, and that you have *de re* knowledge of your long lost brother, and it happens that the blackmailer is your brother, then it follows logically that you know that the blackmailer is your brother.

Changing Domains: It is often desirable to make the very existence of objects subject to modal operators. It seems natural to represent, "I don't know whether the United States has a Prime Minister," or "The Eiffel Tower has not always existed," as "$\neg$ L($\exists_X X =$prime_minister(usa))," and "M($\neg\exists_X X =$eiffel_tower)."

In other words, we would like to allow different worlds to contain different objects. The Prime Minister of the USA does not exist in the real world, but she exists in worlds consistent with my knowledge. The Eiffel Tower exists in the possible worlds of the 1980's but not in the possible worlds of the 1780's. However, in our semantics, we required that all possible worlds have the same domains. The formula M($\neg\exists_X X = \tau$) is provably false for any term τ.

It is fairly easy to change the semantics of modal logic to allow varying domains. In the new semantics, each world has associated with it a particular domain of individuals, which is a subset of the universe of individuals. An interpretation maps a constant symbol a onto an individual u. In any particular world $\mathcal{W}_i$, a denotes u if u is an element of the domain of the world; otherwise, a has no denotation in the world. In each world a function symbol represents a partial function over the domain of that world. Terms involving non-existent objects are non-existent; atomic formulas involving non-existent objects are false. Quantified formulas are true in a world if they are true of each individual in that world. However, the axiomatization needed for this logic is more complex than that of table 2.7 above.

2.8 Syntactic Theories

There are facts in commonsense domains that seem to involve quantifying over sentences. For example, one would like to express the fact "John knows something that Bill doesn't," in the form "$\exists_P$ know(john, P) $\wedge \neg$ know(bill, P)," or the fact "George said something about taxes" as "$\exists_P$ said(george, P) $\wedge$ about(P, taxes)." Quantification of this kind is beyond the expressive power of first order modal logic and possible worlds semantics. Certain kinds of quantification can be expressed within "higher order" logics, which we will not discuss. (See, for example, [Andrews, 86].) The most general and powerful technique for dealing with facts of this kind is the use of a *syntactic* theory, a first-order theory incorporating strings that represent sentences and other meta-level entities.[13] Syntactic theories, however, have a severe drawback: it can be hard to give them a consistent axiomatization. Section 2.8.2 will discuss this difficulty.

Besides allowing quantification over sentences, syntactic theories are also useful for expressing meta-level properties of descriptions of entities, such as computational properties. For example, we must use a syntactic theory to express facts such as, "Archie knows the primes up to 200," where what is meant is that he knows some explicit enumeration of the primes, expressed as sequences of digits.

The difference 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 embedded in direct quotation, while only meaningful sentences 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 a direct quotation; one can say "John said something that that began with an 'I' and ended with a 'y'." As we shall see, similar things can be done in syntactic theories. However, there is one important difference between quotation and other operators on sentences which weakens this analogy. Any fact about an indirect quotation acquires its truth by virtue of some fact about a direct quotation. If "John said that he was hungry" is true, then there is some particular string that John said; there is some sentence ϕ such that "John said 'ϕ' " is true. There is no reason to believe that this property holds of other

[13] In logic texts, it is common to use numbers to represent sentences rather than strings, so that meta-theoretic statements can be interpreted as statements of integer arithmetic. The mapping of sentences to numbers was introduced by Kurt Godel, in proving the incompleteness of arithmetic. Where the meta-theory of arithmetic is not involved, the string representation of sentences is considerably more readable than the numeric representation.

operators on sentences such as "believes." (See sections 8.2 and 9.4.)

2.8.1 Strings

We start with some finite alphabet of characters. We will denote a single character by prefixing a colon; thus :a, :b, :X, :∃, and :: are constant symbols which denote the particular character. The alphabet includes all the characters we use in constructing first-order sentences: the upper and lower case letters, the digits, the logical symbols, the parentheses, and also the colon itself. Other standard characters can be added to the alphabet as desired. A *string* is a finite tuple of characters; for example tuple(:C, :a, :t) is the string "Cat". A *syntactic* theory (also called a theory of quoted strings) is a first-order theory that allows strings as a sort of individual, provides certain standard functions and relations on strings characterized by standard axioms, and provides some axioms or axiom schemas that relate the strings that represent sentences to the sentences they represent.

We abbreviate tuples of characters using the symbols ≺ ≻ as string delimiters. For example, ≺Cat≻ is an alternate notation for tuple(:C, :a, :t). It should be emphasized that this is merely a notational convenience (syntactic sugar) and that it can always be expanded to the "tuple" notation. Quotation marks can be embedded; in this case, the meaning of the inner quotation is derived by expanding it into "tuple" notation. For example, the string ≺length(≺Cat≻)=3≻ is equivalent to ≺length(tuple(:C, :a, :t))=3≻ which is equivalent to the tuple of 25 characters

tuple(:l, :e, :n, :g, :t, :h, :(, :t, :u, :p, :l, :e, :(, ::, :C, :,, ::, :a, :,, ::, :t, :), :), :=, :3)

Quotation marks create a context which is completely opaque as regards substitution under equality. For example, from the statements length(≺bush≻)=4 and bush=president(usa), we are certainly not entitled to deduce length(≺president(usa)≻) = 4; the length of the string ≺president(usa)≻ is 14. Substitutivity fails because the sentence "length(≺bush≻) = 4" contains no reference to Bush, and, in fact, no use of the symbol "bush", merely uses of the characters :b, :u, :s, and :h, as can be seen if we rewrite it

length(tuple(:b, :u, :s, :h)) = 4

Our interest is in strings that spell out first-order terms or formulas; these we will call *meaningful* strings. Meaningful strings are built up by combining meaningful symbols. We define a string as *symbolic* if it spells out a non-logical symbol (constant function, or predicate symbol), a variable symbol, a Boolean operator, or a quantifier. Thus, ≺mary≻, $\prec X2 \succ$, $\prec \wedge \succ$, and $\prec \exists \succ$ are symbolic; the strings $\prec X \wedge \exists \succ$, ≺)≻, and ≺f($X$)≻ are not symbolic. We introduce the function "apply($O, A1, \ldots, Ak$)", which constructs a meaningful string expressing the application of operator symbol O to meaningful strings $A1, \ldots, Ak$. It is possible for O to be a function or predicate symbol and $A1 \ldots Ak$ to spell out terms; or for O to be a Boolean operator and for $A1$ and $A2$ to spell out sentences; or for O to be a quantifier, $A1$ to be a variable symbol, and $A2$ to spell out a formula.

Table 2.8 gives some examples of the use of the "apply" function. Table 2.9 enumerates a number of non-logical symbols useful in a syntactic theory.

The operators "is_symbol," "is_meaningful," "is_constant," "is_term," "is_sentence," "apply," and "subst" are called *syntactic* operators. Their truth can be computed merely from knowing the forms

apply(≺parent≻, ≺marion≻, ≺$X2$≻) = ≺parent(marion,$X2$)≻.
apply(≺ $>$ ≻, ≺1000≻, apply(≺ $+$ ≻, ≺200≻, ≺34≻)) = ≺$1000 > 200 + 34$≻.
apply(≺ $\wedge$ ≻, ≺p(A)≻, ≺$\neg$q(b)≻) = ≺p(A) $\wedge$ $\neg$q(b))≻.
apply(≺$\exists$≻, ≺X≻, ≺ father(marion, X)≻) = ≺$\exists X$ father(marion,X)≻.

Table 2.8: Use of the "apply" function

of the arguments, and the non-logical symbols of the language. They can be fully axiomatized in a standard way. The operators "denotes," "true," and "name_of" are *semantic* operators. Determining their truth requires knowing the meaning of the language and the state of the world.

In addition to these object level relations, a central concept is that of an object-level string *spelling out* a meta-level construct. We will not give a formal definition, as that would require formalizing the meta-language, but the meaning should be obvious: The object level string ≺father_of(john)≻ (= tuple(:f, :a, :t ...)) spells out the meta-level term "father_of(john)"; the object level string ≺X=f(X)≻ spells out the meta-level formula "X=f(X)", and so on.

We can now add any operator on sentences that we like by treating it as an operator on strings. The properties of the operator are specified in terms of particular axioms for that operator; no particular properties are imposed *a priori.* The result is a very flexible language for talking about sentences.

For example, we can introduce the operator "know(A, S)" meaning that person A knows sentence S. We can then express "John knows something that Bill doesn't" as

$\exists_S$ know(john,S) $\wedge$ $\neg$know(bill,S)

This is now legitimate, since S ranges over strings.

We can express the fact, "John knows the name of the capital of Massachusetts" as

$\exists_S$ is_constant(S) $\wedge$ know(john,apply(≺ $=$ ≻, ≺capital(massachusetts)≻, S))

This is true, because for the value S=≺boston≻, which is a constant symbol, the value of the "apply" term becomes ≺capital(massachusetts)=boston≻, which John knows to be true. The requirement that the string be a constant means that the statement would not be true if John only knew the statement ≺capital(massachusetts) = capital(massachusetts).≻ (A more readable notation will be introduced in section 8.2.3.) Constant strings thus play a role in syntactic theories similar to the role of rigid designators in modal theories (section 2.7.3). The problems connected with rigid designators can be directly addressed in syntactic theories by using a variety of syntactic predicates, depending on the circumstance.

In this way, we can express operators on sentences, where the sentences may be quantified over with any definable criterion. In some ways, this often feels like too much power; the language is so expressive that it gives no constraints. In possible world semantics, the interpretation of an operator on a sentence is related to the meaning of the sentence; the operator is true if the sentence is true in certain accessible worlds. In a syntactic theory, there is not any necessary relation between the interpretation of operators and their meaning. We can perfectly well define operators on strings that,

- is_symbol(S) is a predicate meaning that string S is a symbolic string.
- is_meaningful(S) is a predicate meaning that string S is meaningful (a term or sentence).
- is_constant(S) is a predicate meaning that the string S is a constant symbol in the language. For example, is_constant(≺john≻) is true; is_constant(≺father_of(john)≻) is false.
- is_term(S) is a predicate meaning that the string S is a term in the language. For example, is_term(≺father(X)≻) is true, while is_term(≺)∃cat≻) is false.
- is_sentence(S) is a predicate meaning that the string S is a sentence in the language. For example, is_sentence(≺elephant(clyde)≻) is true while is_sentence(≺father_of(John)=≻) is false.

- apply($O, A1, \ldots Ak$) is a function that gives the string consisting of the operator O applied to arguments $A1 \ldots Ak$.
- subst($SNEW, SVAR, SOLD$) is a function that gives the result of substituting the term string $SNEW$ for every genuine occurrence of the variable string $SVAR$ (that is, every occurrence outside embedded quotation marks) in the formula string $SOLD$. For example,

  subst(≺john≻, ≺X≻, ≺loves(X,father_of(X))≻) = ≺loves(john, father_of(john))≻

  It is straightforward to define "subst" in terms of "symbolic" and "apply".
- dbl_quote(S) is a function that adds a level of quotation to a string. For example,

  dbl_quote(≺cat≻) = dbl_quote(tuple(:c, :a, :t)) = ≺≺cat≻≻ = ≺tuple(:c, :a, :t)≻
  =
  tuple(:t, :u, :p, :l, :e, :(, ::, :c, :, ::, :a, :, ::, :t, :))

  Note that the argument to dbl_quote must be a string. If "bill" is a constant symbol denoting the person Bill, then "dbl_quote(bill)" is a meaningless expression. We cannot have dbl_quote(bill) be the string ≺bill≻, because, since "bill" is an ordinary constant symbol, it can always be replaced by equal terms. That is, if "dbl_quote" were a function that applied to bill and bill=father_of(john) were true, then dbl_quote(bill) would have to be equal to dbl_quote(father_of(john)).
- denotation(S) is a function mapping a string expressing a ground term S into the object that S denotes. Thus, if John's father is Bill, then all of the following are true:

  denotation(≺bill≻) = bill
  denotation(≺father_of(john)≻) = bill
  denotation(≺bill≻) = father_of(john)
  denotation(≺father_of(john)≻) = father_of(john)
- true(S) means that the string S is a true sentence. For example, true(≺1+1=2≻) is true.
- name_of(X) maps an object X onto a constant denoting X.

  constant(name_of(X)) $\wedge$ denotation(name_of(X)) = X

Table 2.9: Non-logical symbols for a syntactic theory

by themselves, are completely meaningless. For instance, we can define an operator "shmow(X, P)" which is true just if X knows P written backwards, and, moreover, the letters occurring in P are in alphabetical order. Moreover, it can be argued that using a logic grounded in the theory of strings eliminates one of the major advantages of defining an extensional semantics. A Tarskian or possible worlds semantics for a domain without quotation allows us to forget about the syntactic structure of proofs, and reason about the objects, functions, and relations of the domain directly. By using a theory of quotation, we must return to worrying about the details of string manipulation.

Others find syntactic theories more intelligible than modal theories. The concepts of possible worlds and of accessibility are rather metaphysical, in the pejorative sense; and the reduction of modality to possible worlds often seems strained. It seems strange (to me) to say that John knows that Clyde is an elephant by virtue of the nature of Clyde in other possible worlds, rather than by virtue of something about John in this world. (By contrast, it seems reasonable to say that "Yesterday, it rained" is true by virtue of the state of the world at a different time, or that "If Barney had known French, he would have understood the lecture," is true by virtue of a hypothetical world (or worlds) in which Barney did knew French.) The idea that John knows that Clyde is an elephant by virtue of a relation between John and the string $\prec$elephant(clyde)$\succ$ seems comparatively clear: the string is somehow encoded in his brain. (See section 8.2.)

2.8.2 Paradoxes of Self-Reference

Syntactic theories suffer from a very serious problem. It is easy to construct contradictions in them. These contradictions arise from the combination of self-referential sentences and terms — sentences and terms that refer to strings that embed them — with axioms that relate strings to their meanings.

The axioms are, in themselves, very plausible. We would like to say that the predicate "true" applies to the quoted form of a sentence just if the sentence is true. For example the sentence "true($\prec$parent(john,bill)$\succ$)" is true just if "parent(john,bill)" is true. This is, after all, what we mean by "true". We can express this as follows:

Axiom Schema of Truth: Let P be a string that spells out the sentence ϕ. Then

$$\text{true}(P) \Leftrightarrow \phi$$

is an axiom. Thus, for example,

$$\text{true}(\prec\text{parent(john,bill)}\succ) \Leftrightarrow \text{parent(john,bill)}$$

is an axiom.

In the same way we would like to assert that the quoted forms of terms denote the object that the terms mean. That is, if T is a string that spells out term τ, then "denotation(T)=τ" is an axiom. For example,

$$\text{denotation}(\prec\text{father\_of(john)}\succ) = \text{father\_of(john)}$$

is an axiom.

The problem is that we can construct sentences that assert that their own quoted form is false, and terms that are described in terms of their own failure to denote. Standard examples of these in English are "This sentence is false," and "The smallest number not describable in fewer than twelve English words." The first of these is called the Liar sentence. It is not consistent, either to suppose that the sentence is true or to suppose that it is false.

We cannot directly translate the above liar sentence into first-order logic, since logic has no equivalent of the demonstrative "this", used self-referentially. However, a syntactic theory does allow us to construct a term τ that denotes a string that spells out a sentence containing τ. In this way, we can construct a sentence ϕ that can be shown to be equivalent to $\neg$true(ϕ). A syntactic notation allows us to create such a sentence; a sentence ϕ that asserts $\neg$true(P), where P spells out ϕ. The sentence below is an example.

$$\begin{aligned}\neg\ \text{true}(\text{subst}(&\prec\prec\neg\ \text{true}(\text{subst}(\text{dbl\_quote}(X1),\prec X1\succ,\ X1))\succ\succ,\\ &\prec X1\succ,\\ &\prec\neg(\text{true}(\text{subst}(\text{dbl\_quote}(X1),\prec X1\succ,X1))\succ))\end{aligned}$$

Using the axiom schema of truth and the definitions of subst and quote, one can easily derive a contradiction. If the Liar sentence above is true, then it must be false; if it is false, it must be true. (The casual reader may take this on faith. The more intense reader may enjoy working out how this works (exercise 2.9).)

Sentences of similar structure can be created for any operator on sentences. We can construct a sentence that says, in effect, "John believes that he does not believe this," "John knows that he does not know this", "John says that he does not say this", and so on. If sufficiently strong axioms are asserted about the operator, it may be possible to use these to derive a contradiction. It should be noted that the paradoxes rely as much on the axioms governing the operator as on the self-referential sentences. There is, for instance, nothing paradoxical about someone standing up and saying, "I am not speaking this sentence;" he is simply obviously lying. No contradiction can be derived because there are practically no axioms whatever that govern what a person can speak.

(Digression: A more interesting example is the predicate 'ϕ is provable'. Let $\mathcal{T}$ be a first-order theory of quoted strings containing syntactic operators such as concat and subst, but no semantic operators. Then, since proof is a purely syntactic notion, it is possible to define the predicates "proof(P, A)", meaning "String P is a proof of string A in $\mathcal{T}$," and the predicate "provable(A)" meaning "String A can be proven in $\mathcal{T}$". Using a construction like that above, it is possible to construct a string P such that

a. P spells out a sentence ϕ

b. The sentence '$\phi\Leftrightarrow\neg$provable(P)' is provable in $\mathcal{T}$.

Furthermore, we can prove in the meta-theory that if string A spells out sentence α and A is provable, then α is true. However, the sentence, "provable(A) $\Rightarrow \alpha$," need not be provable in $\mathcal{T}$. Thus, since in the meta-theory we have shown that $\phi\Leftrightarrow\neg$provable(P) and that provable(P) $\Rightarrow \phi$, we can conclude that ϕ is true but not provable in $\mathcal{T}$. There is no contradiction, since this argument is a proof in the meta-theory, not in $\mathcal{T}$.

The argument above is the second part of Godel's incompleteness theorem., The first, and more difficult, part is the number theory necessary to show that the syntactic definitions we need to define provability and to construct the self-referential sentence can all be mirrored in the language of arithmetic. End of digression.)

These contradictions can only be resolved either by dropping the axiom of truth or, more extremely, by abandoning two-valued logic. Various ways of doing these have been proposed. (See References). A natural proposal is to say that the axiom schema only applies to ordinary sentences, not to paradoxical sentences. The problem is that it is not, in general, possible to determine which sentences are paradoxical; it may, in fact, depend on external facts about the world. For example, the sentence "Either this sentence is false or Paris is the capital of France" is not paradoxical, but true; while the sentence "Either this sentence is false, or Rome is the capital of France" is paradoxical. For another example, "THE FIRST QUOTED SENTENCE PRINTED IN BLOCK CAPITALS IN *REPRESENTATIONS OF COMMONSENSE KNOWLEDGE* IS FALSE," is paradoxical, but it would not be if we had inserted an innocuous sentence in block capitals earlier. Thus, we cannot, in general, determine whether a sentence is an instance of the axiom of truth, or whether it is an invalid attempt to apply this schema to a paradoxical sentence. We could restrict the axiom schema so that it applies only to sentences that can be easily determined syntactically not to be paradoxical, but this results in a very limited theory.

In practice, this may not make very much difference to AI programs. An AI program might plausibly run for a long time applying the axiom of truth without worrying and never run into self-referential sentences. However, this is not very satisfying. There is no useful way to define logical consequence in an inconsistent logic; hence, there is no way to use this logic to verify that an inference engine is behaving reasonably. Having a fundamental flaw like this in a logic is worrisome, like carrying a loaded grenade; you never know when it might go off. Moreover, sentences that, taken literally, are self-referential do come up, from time to time, even in the most innocuous contexts. An magazine article on pasta contained the following sentence: "Once you have made pasta that is neither mushy nor rubbery and you have experimented with the ways different shapes and thicknesses combine with different sauces . . . the end of this sentence is not 'you'll never accept substitutes.'" Or they may be brought up with malice aforethought. There was an episode of Star Trek in which a malignant computer suffered a nervous breakdown when it was presented with the Liar Sentence. We would wish our programs to be immune to this sad fate.

2.9 Appendix A: Natural Deduction

Natural deduction is a proof system for first-order logic that generates proofs that are, in their structure, relatively close to the kinds of proofs written by human theorem provers, and therefore relatively readable. Numerous minor variants of natural deduction exist. The one presented below is adapted from [Mates, 1972]; it contains nine rules of inference and no logical axioms.

Two features of natural deduction are particularly notable. First, in a natural deduction proof it is possible to assume one fact ϕ, deduce a new fact ψ from ϕ, and then conclude the material implication $\phi \Rightarrow \psi$ from the fact that that $\phi \vdash \psi$. This inference is formalized in the rule of discharging. Such an argument can be used, for example, in proofs by contradiction: To show that ϕ is true, assume $\neg\phi$, show that that leads to a contradiction, and conclude that ϕ must be true. Second, it is possible in natural deduction to use a new constant symbol "locally" to represent an object with a given property. This reflects such forms in informal proofs as "Let p be a prime number." In natural deduction, we can make assumptions such as "p is a prime number" and see where it leads us. If we can conclude some other property of p — for example, that $X^p \equiv X \bmod p$ for all X, — then we can conclude that this property holds for all prime numbers. Moreover, if we know that there exist prime numbers, then we can conclude that there are numbers with the above property. To do this, we must, of course, use constants that are not used elsewhere in the proof. Moreover, we may need arbitrarily many constants. We therefore assume that the language has an infinite collection of constants that we can draw on. These types of inference are formalized in the rules of universal generalization and existential specification.

The structure of a proof in natural deduction is more complicated than in the axiomatic proof theory considered earlier. In order to carry out the discharging inference, in which we assume ϕ, derive ψ, and infer $\phi \Rightarrow \psi$, we must keep track of all the assumptions that underlie any given step of the proof. We therefore define a proof step as follows:

Definition A.1: Let $\mathcal{L}$ be a first-order language. A *proof step* $\mathcal{S}$ over $\mathcal{L}$ is a triple consisting of

i. A sentence in $\mathcal{L}$, denoted "content($\mathcal{S}$)".

ii. A label of the step, denoted "label($\mathcal{S}$)". We will use integers as labels, but any type of symbol may be used.

iii. The set of the labels of the assumptions underlying $\mathcal{S}$, denoted "assumptions($\mathcal{S}$)".

Definition A.2: A *proof structure* is a sequence of proof steps, no two of which have the same label.

We now define the various types of inference. In all the definitions below, $\mathcal{S}, \mathcal{I}, \mathcal{J}, \mathcal{K}$ are proof steps and $\mathcal{P}$ is a proof structure.

Definition A.3: Axiom. $\mathcal{S}$ can be inferred in proof structure $\mathcal{P}$ from hypotheses Γ as an *axiom* if

i. content($\mathcal{S}$) $\in \Gamma$.

ii. assumptions($\mathcal{S}$)$=\emptyset$.

Definition A.4: Assumption. $\mathcal{S}$ can be inferred in $\mathcal{P}$ as an *assumption* if assumptions($\mathcal{S}$) $=$ $\{$ label($\mathcal{S}$) $\}$. Note that there is no constraint on the content of $\mathcal{S}$; any sentence can be taken as an assumption.

Definition A.5: Tautology. $\mathcal{S}$ can be inferred from steps $\mathcal{I}_1 \ldots \mathcal{I}_k$ in $\mathcal{P}$ by the rule of tautology if

i. Steps $\mathcal{I}_1 \ldots \mathcal{I}_k$ precede $\mathcal{S}$ in $\mathcal{P}$.

ii. content($\mathcal{S}$) is a tautological (propositional calculus) consequence of content($\mathcal{I}_1$) $\ldots$ content($\mathcal{I}_k$);

iii. assumptions($\mathcal{S}$) $= \cup_{i=1 \ldots k}$ assumptions($\mathcal{I}_i$).

Definition A.6: Discharge. $\mathcal{S}$ can be inferred from $\mathcal{I}$ and $\mathcal{J}$ in $\mathcal{P}$ through the rule of discharge (conditionalizing) if the following hold:

i. $\mathcal{P}$ contains $\mathcal{I}$, $\mathcal{J}$, and $\mathcal{S}$ in that order (though not necessarily consecutively).

ii. Let content($\mathcal{I}$)=ϕ and content($\mathcal{J}$)=ψ. Then content($\mathcal{S}$)=$\phi \Rightarrow \psi$.

iii. assumptions($\mathcal{S}$) = assumptions($\mathcal{J}$) $-$ { label($\mathcal{I}$) }.

Definition A.7: Universal Specification. $\mathcal{S}$ can be inferred from $\mathcal{I}$ in $\mathcal{P}$ by the rule of universal specification if the following hold:

i. $\mathcal{I}$ precedes $\mathcal{S}$ in $\mathcal{P}$.

ii. There is an open formula α, a variable μ, and a ground term τ such that content($\mathcal{I}$) = $\forall_\mu \, \alpha(\mu)$ and content($\mathcal{S}$) = $\alpha(\mu/\tau)$.

iii. assumptions($\mathcal{S}$) = assumptions($\mathcal{I}$).

Definition A.8: Universal Generalization. $\mathcal{S}$ can be inferred from $\mathcal{I}$ in $\mathcal{P}$ by the rule of universal generalization if there is an open formula α, a variable μ, and a constant symbol β such that the following hold:

i. $\mathcal{I}$ precedes $\mathcal{S}$ in $\mathcal{P}$.

ii. content($\mathcal{I}$) = $\alpha(\mu/\beta)$; content($\mathcal{S}$)=$\forall_\mu \, \alpha(\mu)$.

iii. assumptions($\mathcal{S}$)=assumptions($\mathcal{I}$).

iv. The constant β does not appear in $\alpha(\mu)$ or in the content of any assumption of $\mathcal{I}$.

v. The constant β does not appear in the content of any axiom in $\mathcal{P}$.

Definition A.9: Existential Generalization. $\mathcal{S}$ can be inferred from $\mathcal{I}$ in $\mathcal{P}$ by existential generalization if

i. $\mathcal{I}$ precedes $\mathcal{S}$ in $\mathcal{P}$.

ii. content($\mathcal{I}$) = $\alpha(\mu/\tau)$ and content($\mathcal{S}$) = $\exists_\mu \, \alpha(\mu)$, for some formula α, variable μ, and term τ.

iii. assumptions($\mathcal{S}$) = assumptions($\mathcal{I}$).

Definition A.10: Existential Specification. $\mathcal{S}$ can be inferred from $\mathcal{I}, \mathcal{J}, \mathcal{K}$ in $\mathcal{P}$ by existential specification if there is a formula α, a constant symbol β, and a variable symbol μ such that

i. $\mathcal{I}, \mathcal{J}, \mathcal{K}, \mathcal{S}$ occur in that order (not necessarily consecutively) in $\mathcal{P}$.

ii. content($\mathcal{I}$) = $\exists_\mu \, \alpha(\mu)$; content($\mathcal{J}$) = $\alpha(\mu/\beta)$.

iii. content($\mathcal{S}$) = content($\mathcal{K}$)

iv. β does not appear in content($\mathcal{I}$), in content($\mathcal{K}$), or in the content of any assumption of $\mathcal{K}$ other than $\mathcal{J}$.

v. β does not appear in the content of any axiom in $\mathcal{P}$.

vi. assumptions($\mathcal{S}$) = assumptions($\mathcal{I}$) $\cup$ assumptions($\mathcal{K}$) $-$ { label($\mathcal{J}$) }

Definition A.11: Quantifier exchange: $\mathcal{S}$ can be inferred from $\mathcal{I}$ in $\mathcal{P}$ by quantifier exchange if

i. $\mathcal{I}$ precedes $\mathcal{S}$ in $\mathcal{P}$.

ii. There is a formula α and variable μ such that either

   a. content($\mathcal{S}$) = $\forall_\mu \, \alpha$; content($\mathcal{I}$) = $\neg\exists_\mu \, \neg\alpha$;
   b. content($\mathcal{S}$) = $\forall_\mu \, \neg\alpha$; content($\mathcal{I}$) = $\neg\exists_\mu \, \alpha$;
   c. content($\mathcal{S}$) = $\exists_\mu \, \alpha$; content($\mathcal{I}$) = $\neg\forall_\mu \, \neg\alpha$; or
   d. content($\mathcal{S}$) = $\exists_\mu \, \neg\alpha$; content($\mathcal{I}$) = $\neg\forall_\mu \, \alpha$.

iii. assumptions($\mathcal{S}$) = assumptions($\mathcal{I}$)

Finally we can define a proof.

Definition A.12: A *proof* of conclusion ϕ from hypotheses Γ is a proof structure in which every step can be inferred either as an axiom of Γ, as an assumption, or via one of the rules of tautology, discharge, universal specification, universal generalization, existential specification, existential generalization, or quantifier exchange.

Strictly speaking, the rules of existential specification and existential generalization are redundant; any theorem can be proven without recourse to these rules.

Table 2.10 shows an example of a proof.

2.10 References

[Turner, 1984] surveys many of the logics used in AI.

[Genesereth and Nilsson, 1987] contains an excellent introduction to first-order logic addressed to the student of AI. As a textbook for further study of first-order logic I recommend [Mates, 1972]. (In particular, [Mates, 1972] has an extensive discussion of the translation of English sentences to first-order logic.) The axioms given here for propositional and predicate calculus are slightly modified from [Genesereth and Nilsson, 1987]. The definition of Tarskian semantics and the natural deduction system described in appendix A are adapted from [Mates, 1972].

Sorted logics are discussed in [Cohn, 1985] and [Walther, 1985].

[Halmos, 1960] is a readable introduction to set theory. The only discussion of set theory with ur-elements that I have found is [Barwise, 1975]; this, however, is an advanced text requiring a great deal of background. [Zadrozny, 1989] presents a new, less constraining set theory that he argues is more suitable to commonsense reasoning.

Givens:

$\forall_P$ vegetarian(P) $\Leftrightarrow$ [$\forall_X$ eats(P, X) $\Rightarrow \neg$meat(X)]
(A vegetarian is someone who does not eat any meat.)

$\exists_P$ $\forall_X$ eats(P, X) $\Rightarrow$ [tomato(X) $\vee$ carrot(X)]
(There is someone who eats only tomatoes and carrots.)

$\forall_X$ tomato(X) $\Rightarrow \neg$meat(X).
(Tomatoes are not meat.)

$\forall_X$ carrot(X) $\Rightarrow \neg$meat(X).
(Carrots are not meat.)

To prove: $\exists_X$ vegetarian(X). (There exists a vegetarian.)

Label	Content	Assumptions	Justification
1.	$\exists_P$ $\forall_X$ eats(P, X) $\Rightarrow$ [tomato(X) $\vee$ carrot(X)]	{ }	Axiom.
2.	$\forall_X$ eats(p1,X) $\Rightarrow$ [tomato(X) $\vee$ carrot(X)]	{ 2 }	Assumption.
3.	eats(p1,x1) $\Rightarrow$ [tomato(x1) $\vee$ carrot(x1)]	{ 2 }	Universal Spec. (2)
4.	eats(p1,x1)	{ 4 }	Assumption.
5.	tomato(x1) $\vee$ carrot(x1)	{ 2,4 }	Tautology (3,4).
6.	$\forall_X$ tomato(X) $\Rightarrow \neg$meat(X)	{ }	Axiom.
7.	tomato(x1) $\Rightarrow \neg$meat(x1)	{ }	Universal Spec. (6).
8.	$\forall_X$ carrot(X) $\Rightarrow \neg$meat(X)	{ }	Axiom.
9.	carrot(x1) $\Rightarrow \neg$meat(x1)	{ }	Universal Spec. (8).
10.	[tomato(x1) $\vee$ carrot(x1)] $\Rightarrow \neg$meat(x1)	{ }	Tautology (7,9)
11.	$\neg$meat(x1)	{ 2,4 }	Tautology (5,10)
12.	eats(p1,x1) $\Rightarrow \neg$meat(x1)	{ 2 }	Discharge (4,11)
13.	$\forall_X$ eats(p1,X) $\Rightarrow \neg$meat(X)	{ 2 }	Universal Gen. (12).
14.	$\forall_P$ vegetarian(P) $\Leftrightarrow$ [$\forall_X$ eats(P, X) $\Rightarrow \neg$meat(X)]	{ }	Axiom.
15.	vegetarian(p1) $\Leftrightarrow$ [$\forall_X$ eats(p1,X) $\Rightarrow \neg$meat(X)]	{ }	Universal Spec. (14)
16.	vegetarian(p1)	{ 2 }	Tautology (13,15).
17.	$\exists_P$ vegetarian(P).	{ 2 }	Existential Gen. (16)
18.	$\exists_P$ vegetarian(P).	{ }	Existential Spec. (1,2,17)

Table 2.10: Proof in Natural Deduction

Extensional operators have been used in AI logics of time since [McCarthy, 1959]. [Hobbs, 1985c] is a general discussion of extensional operators; Hobbs proposes that it should be possible to create an entity corresponding to any atomic sentence.

[Hughes and Cresswell, 1968] is a standard text on modal logic. Unfortunately, the logic presented there uses neither constant symbols nor proper axioms, so it required some modification for its presentation here. (The uninterest of modal logicians in theories with proper axioms is indicated by the fact that modal logic with proper axioms was first proven complete in [McDermott, 1982b]. I do not know whether the axioms MODAL.1 through MODAL.12 are complete for the modal logic with equality.) Possible worlds semantics for modal logic was introduced in [Kripke, 1963a and 1963b]; these used a possible worlds semantics in which different worlds could have different domains of individuals. Higher order modal logics are discussed in [Gallin, 1975]. Modal logics of knowledge and belief were developed in [Hintikka, 1962]. Hintikka's logic of knowledge was extended in [Moore, 1980] to incorporate a representation for action. Moore also introduced the idea of translating the modal logic into a first-order logic over possible worlds. Other AI applications of modal logic include [Appelt, 1982] and [Shoham, 1988].

For higher-order logics, see [Church, 1956] and [Andrews, 1986], chapter 5.

There is a large philosophical literature dealing with the status of *de re* modalities and rigid designators. The best known work is probably [Kripke, 1972]; see also [Burge, 1977] [Dennett, 1981], [Goodman, 1961], [Kaplan, 1968], [Moore, 1980], and [Quine, 1969]. Possible worlds interpretations for counterfactual sentences are discussed in [Lewis, 1973]. [Ginsberg, 1986] discusses applications of counterfactuals to AI.

The theory of quoted strings is discussed in [Genesereth and Nilsson, 1987 chap. 10], though in the context of the meta-language rather than in the object language. AI theories that have used syntactic operators include [Perlis, 1985], [Konolige, 1982], [Haas, 1983], and [Morgenstern, 1988]. [Tarski, 1956], [Kripke, 1975], [Gupta, 1982], and [Barwise and Etchemendy, 1987] are analyses of the Liar Paradox. [Hofstadter, 1979] and [Hofstadter, 1985] are entertaining popular books which explore problems of self-reference and related issues. [Smullyan, 1978] is one of a number of collection of puzzles on the same theme. [Kaplan and Montague, 1960] discuss analogues of the liar paradox with the operators "know" and "believe".

Situation logic, presented in [Barwise and Perry, 1982], is an alternative approach to the problems discussed in this chapter.

2.11 Exercises

2.1. Express the "Jones, Smith, and Robinson" puzzle on section 2.2 in the propositional calculus.

2.2. a. Given the hypothesis $\neg\neg p$, prove p, using axioms PROP.1 — PROP.7 of the propositional calculus.

b.* Given the hypothesis p, prove $p \lor q$, using axioms PROP.1 — PROP.7.

2.3. Consider a domain consisting of people, books, and copies of books (volumes). Let $\mathcal{L}$ be a sorted first-order language with "person", "book", and "volume" as sorts, and with the following non-logical symbols:

Constants: sam, barbara, tolstoy, joyce.
Predicates: owns(P, V) — Person P owns volume V.
author(P, B) — Person P wrote book B.
copy(V, B) — Volume V is a copy of book B.

Express each of the following statements as a sentence in $\mathcal{L}$: (You need not include the conditions to enforce correct sorting.)

i. Sam owns a copy of every book that is either by Tolstoy or Joyce.

ii. All the volumes that Barbara owns are copies of books by Tolstoy.

iii. If Barbara owns a copy of a book, then Sam owns a copy of the same book.

iv. There is some book that Sam owns but Barbara doesn't.

v. Every author owns a copy of each of his own books.

2.4. Consider the following "proof" of sentence (iv) in problem 2.3 from (i) and (ii): Sam owns all books by Joyce (from (i)) and Barbara owns only books by Tolstoy (from (ii)). Therefore any book by Joyce is owned by Sam but not by Barbara. Therefore Sam own some book that Barbara doesn't.

a. What background assumptions does this proof rely on? Express these as sentences in $\mathcal{L}$.

b. Using the natural deduction system described in appendix A, give a proof of (iv) from (i), (ii), and the additional assumptions in (a).

2.5. Define the modal operator, "knows(X, ϕ)", to mean "X knows that ϕ." In a modal language containing this operator, the language defined in ex. 2.3, and the constant "ulysses" (the name of a book), express the following sentences.

a. Sam knows that Joyce is the author of Ulysses.

b. Barbara knows that Sam owns a copy of every book by Joyce.

c. There is a book B by Tolstoy such that Barbara knows that she owns a copy of B.

d. Everyone knows that there is no book by Tolstoy with no copies owned by anyone.

e. Sam knows that everyone knows that Joyce is the author of Ulysses.

f. Someone knows that Sam owns a copy of a book that he himself (the someone) wrote.

2.6. Express each of the sentences in ex. 2.5 in terms of a possible worlds semantics

2.7.* Express each of the sentences in ex. 2.5, treating "know(X, ϕ)" as a syntactic operator. (Note: This exercise will be much easier after chapter 8 has been covered.)

2.8. Show that axiom MODAL.10 (section 2.7) is true in any Kripke structure in which accessibility of possible worlds is an equivalence relation.

2.9.* Show that the Liar sentence in section 2.8.2 asserts its own falsehood. Note the difference between the term dbl_quote($X1$) which contains the symbol $X1$, and is therefore changed by the subst, and the term $\prec X1 \succ$, which does not contain the symbol X1, only the characters :X and :1, and is therefore unchanged by the subst.

2.10 a. Show that $\forall_{X,Y}$ X=Y $\Rightarrow$ L($X = Y$) is a consequence of axiom MODAL.3.

b.* Explain why the false conclusion "sid=mayor(schenectady) $\Rightarrow$ L(sid=mayor(schenectady))" does not follow from the formula in (a).