(b) Show that for each formula θ built up from Q, if both

$$\phi \to \theta \quad , \quad \theta \to \psi$$

are tautologies, then

$$\lambda \to \theta \quad , \quad \theta \to \rho$$

are also tautologies.

These provide an interpolation result for propositional logic.

Chapter 2

The modal language

2.1 Introduction

The propositional modal language is an extension of the pure propositional language formed by adding a battery of new 1-ary connectives (known informally as box connectives). Originally there was just one new connective $\Box$, however for many purposes it is necessary to add several (possibly infinitely many) such connectives $[i]$, one for each element i of an index set I. Thus there are many possible modal languages, one for each index set I. The syntax, semantics, and proof systems associated with modal languages are designed to subsume those of the proposition language, in fact, propositional logic can be regarded as the extreme version of modal logic where $I = \emptyset$.

The element i of I are called *labels* and I itself is called the *signature* of the modal language. Thus two languages are identical precisely when they have the same signature. (We are never going to consider how one language may be be translated into another, so we need not worry about comparison of signatures.)

Unlike the propositional connectives $\neg$, $\to$, $\wedge$, $\vee$,$\top$, and $\bot$, the box connectives $[i]$ do not have a fixed interpretation. For each formula ϕ (of the modal language) we may use $[i]$ to obtain a new formula

$$[i]\phi \tag{2.1}$$

This may be read in several ways, and different readings suggest different semantics and proof systems.

In the original work on modal logic, there was just one box connective (i.e. I was a singleton), and the compound formula (2.1) was read variously as

ϕ is necessary,
ϕ is obligatory,
ϕ is known,

and later other readings such as

ϕ is provable (in some formal system of arithmetic)

were considered. A binary version of modal logic (called *tense logic*) has just two labels

$$+ \text{ (for future)} \qquad \text{and} \qquad - \text{ (for past).}$$

We may then read

$$[+]\phi \qquad\qquad [-]\phi$$

as

$$\phi \text{ is and always will be} \qquad\qquad \phi \text{ is and always was.}$$

In these two readings, it is seen quite clearly that $[+]$ and $[-]$ are intended as kinds of universal quantifiers over time.

More recently modal languages have been used to analyse the behaviour of computer programs and the state transitions of (finite) automata. It is in these applications where many different labels may be required. (In some cases these applications may also require a further enrichment of this modal language, but we won't be dealing with these applications in this book.)

Observe that for each of the above readings there is also a dual complement reading. Thus in the monomodal cases we have

$$\begin{array}{ll} \phi & \text{is possible,} \\ \phi & \text{is permissible,} \\ \neg\phi & \text{is unknown,} \end{array}$$

or

$$\phi \qquad \text{is consistent (with some formal system of arithmetic)}$$

and in the bimodal (tense) case we have

$$\begin{array}{ll} \phi \text{ will be} & \phi \text{ was} \\ \text{(at least once in the future)} & \text{(at least once in the past)} \end{array}$$

To handle these readings let

$$\langle i\rangle\phi \qquad \text{abbreviate} \qquad \neg[i]\neg\phi.$$

We may then check that each reading of $[i]\phi$ produces (via the dual complement) a reading of $\langle i\rangle\phi$. All the required properties of $\langle i\rangle\phi$ (which we read informally as 'diamond ϕ') can be deduced from those of $[i]\phi$. It is also possible to introduce $\langle i\rangle$ as a new atomic primitive. There are various arguments for and against these two approaches (abbreviation .v. primitive), and the differences are most acute when dealing with proof systems. However, these hardly matter in this book, so I have taken the approach which involves the minimum amount of work, namely to treat $\langle i\rangle$ as an abbreviation.

As suggested by the above readings, there is a strong analogy between

$$\text{the connective } [i] \qquad \text{and} \qquad \text{the quantifier } \forall$$

and between

$$\text{the connective } \langle i\rangle \qquad \text{and} \qquad \text{the quantifier } \exists.$$

The analogy will be made precise in Chapter 4 (when we describe the semantics of $[i]$ and $\langle i\rangle$). However, you should always keep the analogy in mind; it will help you understand many things.

2.2 The language defined

Let I be some fixed signature (i.e. indexing set of labels). The primitive symbols of the modal language with this signature are:

- The elements P of a fixed countable set Var of *variables*;
- The *propositional connectives*

  $$\top, \bot, \neg, \rightarrow, \wedge, \vee$$

  of arity 0, 0, 1, 2, 2, and 2, respectively;
- The *box connectives*

  $$[i]$$

  one for each label $i \in I$;

together with the *punctuation symbols* (and). The formulas of this language are then constructed in the expected way.

2.1 DEFINITION. The formulas of the language with signature I are obtained recursively using the following clauses.

atomic Each variable $P \in Var$ and each constant $\top$ and $\bot$ is a formula.

propositional For all formulas θ, ψ, ϕ each of

$$\neg\phi \quad , \quad (\theta \rightarrow \psi) \quad , \quad (\theta \wedge \psi) \quad , \quad (\theta \vee \psi)$$

is a formula.

modal For each formula ϕ and label i,

$$[i]\phi$$

is a formula.

Let $Form$ be the set of all formulas. ■

Note that (as with the propositional language) the precise size of Var has not been specified. The most natural case is to take Var to be countably infinite, however, for some purposes (such as decidability matters) we may want Var to be finite. There is even a use for the case $Var = \emptyset$ e.g. in the analysis of concurrency. (The case where Var is uncountably infinite has some interest.)

When displaying formulas we will attempt to make them easier to read by the use of various conventions (such as the judicious omission of brackets). Also as suggested above we let

$$\langle i \rangle \phi \quad \text{abbreviate} \quad \neg [i] \neg \phi.$$

2.3 Some particular formulas

Several particular formulas (or rather shapes of formulas) play a prominent role in modal logic. In this section we gather together the majority of these shapes.

First some shapes which are concerned with just one label. In such circumstances we usually omit the label and write

$$\Box \phi \text{ for } [i]\phi \quad \text{and} \quad \Diamond \phi \text{ for } \langle i \rangle \phi.$$

This will help prevent displayed formulas from becoming cluttered with unnecessary symbols.

The first batch of shapes have names most of which are used for historical reasons (but now have rather limited significance). Thus, for an arbitrary formula ϕ, let $\mathrm{D}(\phi)$, $\mathrm{T}(\phi)$, ... , $\mathrm{M}(\phi)$ be as follows:

$$\begin{array}{ll}
\mathrm{D}(\phi) & : \ \Box\phi \to \Diamond\phi \\
\mathrm{T}(\phi) & : \ \Box\phi \to \phi \\
\mathrm{B}(\phi) & : \ \phi \to \Box\Diamond\phi \\
4(\phi) & : \ \Box\phi \to \Box\Box\phi \\
5(\phi) & : \ \Diamond\phi \to \Box\Diamond\phi \\
\mathrm{P}(\phi) & : \ \phi \to \Box\phi \\
\mathrm{Q}(\phi) & : \ \Diamond\phi \to \Box\phi \\
\mathrm{R}(\phi) & : \ \Box\Box\phi \to \Box\phi \\
\mathrm{G}(\phi) & : \ \Diamond\Box\phi \to \Box\Diamond\phi \\
\mathrm{L}(\phi) & : \ \Box\mathrm{T}(\phi) \to \Box\phi \\
\mathrm{M}(\phi) & : \ \Box\Diamond\phi \to \Diamond\Box\phi
\end{array}$$

If we use two or more labels then we get a greater variety of formulas. For instance, shapes D, B, 5 and G can be generalized as follows:

$$\begin{array}{rcl}
[i]\phi & \to & [j]\phi \\
\phi & \to & [i]\langle j \rangle \phi \\
\langle i \rangle \phi & \to & [j]\langle k \rangle \phi \\
\langle i \rangle [j] \phi & \to & [k]\langle l \rangle \phi
\end{array}$$

Various other shapes will be important at one time or another, especially when particular applications are under consideration.

2.4 Substitution

Substitution is an important aspect of formal languages which, superficially, is easy to understand, but which has many pitfalls (down which many an author has disappeared). Most problems occur when there are both free and bound variables around (such as in the predicate calculus or the λ-calculus) but even in free variable systems such as the ones considered in this book, it is not entirely straight forward.

Consider a formula ϕ built up from the variables $P_1, \ldots, P_n$, that is, the only variables occurring in ϕ are amongst the ones listed. Suppose also that $\pi_1, \ldots, \pi_n$ is a list of formulas matching the list of variables. It is intuitively clear what we mean by the formula obtained from ϕ by simultaneously replacing P_1 by π_1, P_2 by π_2, ..., P_n by π_n. The resulting formula we may write as

$$\phi[P_1 := \pi_1, P_2 := \pi_2, \ldots, P_n := \pi_n].$$

(There are also several other equally cumbersome and uninformative notations in common use.) For example, let P and Q be distinct variables and let ψ be

$$(P \to Q).$$

Then, for formulas λ and μ,

$$\psi[P := \lambda, Q := \mu] \quad \text{is} \quad (\lambda \to \mu).$$

Even such simple examples can cause trouble. For instance, what happens if λ and μ also contain P and Q, and how do we handle iterated substitutions? Before you continue you should make sure you know why the two formulas

$$\begin{array}{l}
\psi[P := (Q \to P), Q := P] \\
\psi[P := (Q \to P), Q := Q][P := P, Q := P]
\end{array}$$

are

$$((Q \to P) \to P) \quad \text{and} \quad ((P \to P) \to P)$$

respectively.

The great majority of textbooks leave the notion of substitutions as an 'intuitively obvious' operation (and many don't consider it at all). Unfortunately there are one or two places where a more precise knowledge is required, and for these the only safe way is to give a formal definition. (This, of course, requires also that the definition is correct.)

There are several slightly different ways of handling substitution. The one described below is chosen because of its similarity with the application of a valuation (which we look at in Chapter 4).

2.2 DEFINITION. A *substitution* is a function

$$\sigma : Var \longrightarrow Form$$

Let Sub be the set of all such substitutions. ■

Thus a particular substitution σ assigns to each variable P a formula $\sigma(P)$. We may then use σ to modify any formula ϕ by replacing each variable P occurring in ϕ by $\sigma(P)$. This replacement must be done simultaneously for all variables. Let us write ϕ^σ for the result of applying the substitution σ to ϕ. For example, if

$$\sigma(P) = \lambda \quad , \quad \sigma(Q) = \mu$$

then

$$(P \to Q)^\sigma \quad , \quad (\lambda \to \mu).$$

This construction produces an operation

$$\begin{array}{c} Form \times \ \mathsf{Sub} \longrightarrow Form \\ \phi \quad , \quad \sigma \longmapsto \phi^\sigma. \end{array}$$

with the following formal definition.

2.3 DEFINITION. Let $\sigma \in$ Sub. For each formula ϕ the substitution instance ϕ^σ is defined by recursion on ϕ using the following clauses.

(Const) For the constants

$$\top^\sigma = \top \quad , \quad \bot^\sigma = \bot.$$

(Var) For each variable P

$$\sigma(P) \quad , \quad \sigma(P).$$

($\neg$) For each formula $\phi = \neg\theta$

$$(\neg\phi)^\sigma = \neg\phi^\sigma.$$

($\wedge, \vee, \to$) For each formula θ, ψ

$$\begin{array}{rcl} (\theta \wedge \psi)^\sigma & = & (\theta^\sigma \wedge \psi^\sigma) \\ (\theta \vee \psi)^\sigma & = & (\theta^\sigma \vee \psi^\sigma) \\ (\theta \to \psi)^\sigma & = & (\theta^\sigma \to \psi^\sigma). \end{array}$$

(i) For each label i and formula ϕ

$$([i]\phi)^\sigma = [i]\phi^\sigma.$$

where $[i]$ is the appropriate box connective. ■

2.5 Two remarks

Every modal language has a modal-free part consisting of all the formulas which may be constructed without the use of a box (or diamond) symbol. These formulas are precisely the formulas of the (modal-free) propositional language of Chapter 1. Amongst these modal-free formulas we find the tautologies (i.e. those formulas which are true under all 2-valuations).

Given a modal-free formula ϕ and a substitution σ, we may apply σ to ϕ to obtain a modal formula ϕ^σ which, of course, may no longer be modal-free. When ϕ is a tautology we say ϕ^σ is *an instance of a tautology* or sometimes, rather loosely, simply a tautology. When we come to deal with the semantics of modal formulas (in Chapter 4) we will see that such generalized tautologies hold in all modal semantic situations.

Since formulas are constructed recursively on their complexity, many proofs about them proceed by induction on this complexity. For some of these proofs the notion of the *set* $\Gamma(\phi)$ *of subformulas of* ϕ is useful. This set is defined by:

(Const) For the constants

$$\Gamma(\top) = \{\top\} \quad , \quad \Gamma(\bot) = \{\bot\}.$$

(Var) For each variable P

$$\Gamma(P) = \{P\}.$$

($\neg$) For each formula $\phi = \neg\theta$

$$\Gamma(\phi) = \Gamma(\theta) \cup \{\phi\}.$$

($\wedge, \vee, \to$) For each formula $\phi = \theta * \psi$ where $*$ is a binary connective

$$\Gamma(\phi) = \Gamma(\theta) \cup \Gamma(\psi) \cup \{\phi\}.$$

(i) For each label i and formula $\phi = [i]\theta$

$$\Gamma(\phi) = \Gamma(\theta) \cup \{\phi\}.$$

In particular, note that $\phi \in \Gamma(\phi)$ for all ϕ.

2.6 Exercises

2.1 Many properties of formulas ϕ are defined by recursion on the complexity of ϕ, and then proof about these properties are achieved by induction on this complexity. For instance, with each formula ϕ we may associate three sets

$$Var(\phi) \quad , \quad Pos(\phi) \quad , \quad Neg(\phi)$$

of variables using the following clauses.

ϕ	$Var(\phi)$	$Pos(\phi)$	$Neg(\phi)$
$\bot, \top$	$\emptyset$	$\emptyset$	$\emptyset$
P	$\{P\}$	$\{P\}$	$\emptyset$
$\neg\theta$	$Var(\theta)$	$Neg(\theta)$	$Pos(\theta)$
$\theta \wedge \psi$	$Var(\theta) \cup Var(\psi)$	$Pos(\theta) \cup Pos(\psi)$	$Neg(\theta) \cup Neg(\psi)$
$\theta \vee \psi$	$Var(\theta) \cup Var(\psi)$	$Pos(\theta) \cup Pos(\psi)$	$Neg(\theta) \cup Neg(\psi)$
$\theta \to \psi$	$Var(\theta) \cup Var(\psi)$	$Neg(\theta) \cup Pos(\psi)$	$Pos(\theta) \cup Neg(\psi)$
$[i]\theta$	$Var(\theta)$	$Pos(\theta)$	$Neg(\theta)$

(a) Show that for each formula ϕ we have

$$Var(\phi) \;=\; Pos(\phi) \cup Neg(\phi)$$

and find an example with $Pos(\phi) \cap Neg(\phi) \neq \emptyset$.

(b) Give recursive definitions of the two set

$$Pos^+(\phi) \;=\; Var - Neg(\phi) \quad , \quad Neg^+(\phi) \;=\; Var - Pos(\phi)$$

where Var is the set of all variables.

2.2 For a given variable P, consider the four formulas

$$\phi_1 := P \quad , \quad \phi_2 := \neg P \quad , \quad \phi_3 := \neg P \to P \quad , \quad \phi_4 := P \to \neg P.$$

Compute $\phi_i[P := \phi_j]$ for all $1 \leq i, j \leq 4$.

2.3 For distinct variables P and Q let

$$\theta := P \quad , \quad \psi := Q \quad , \quad \phi := P \to Q.$$

For arbitrary formulas ρ and σ compute

(a) $\phi[P := \psi, Q := \theta][Q := \rho, P := \sigma]$

(b) $\phi[P := \psi[Q := \rho, P := \sigma], Q := \theta[Q := \rho, P := \sigma]]$

showing that the resulting formulas are the same.

2.4 Let σ and τ be a pair of substitutions which agree on the variables occurring in a given formula ϕ. Show that $\phi^\sigma = \phi^\tau$.

2.5 For substitutions σ and τ let $\tau \bullet \sigma$ be the substitution given by

$$(\tau \bullet \sigma)(P) \;=\; (\sigma(P))^\tau$$

for all variables P.

(a) Show that

$$(\phi^\sigma)^\tau \;=\; \phi^{\tau \bullet \sigma}$$

holds for all formulas ϕ.

(b) Show that

$$(\tau \bullet \sigma) \bullet \rho \;=\; \tau \bullet (\sigma \bullet \rho)$$

hold for all substitutions ρ, σ, τ.

2.6 Make sure you understand the notion of 'subformula'.

(a) For a variable P, write out $\Gamma(\mathrm{L}(P))$.

(b) Show that

$$\psi \in \Gamma(\phi) \;\Rightarrow\; \Gamma(\psi) \subseteq \Gamma(\phi)$$

(for arbitrary formulas ψ and ϕ).