	1	$\forall x.\ \exists y.\ likes(x,y)$	
	2	$\forall x.\ \forall y.\ [likes(x,y) \rightarrow likes(y,x)]$	
	3	$\forall u.\ \forall v.\ [\exists w.\ [likes(u,w) \wedge likes(w,v)] \rightarrow likes(u,v)]$	
$A\forall\mathcal{I}$	4		
	5	$\exists y.\ likes(A,y)$	
$B\exists\mathcal{E}$	6	$likes(A,B)$	
	7	$likes(B,A)$	$\forall\rightarrow\mathcal{E}(2)$
	8	$likes(A,B) \wedge likes(B,A)$	$\wedge\mathcal{I}(7,6)$
	9	$\exists w.\ [likes(A,w) \wedge likes(w,A)]$	$\exists\mathcal{I}$
	10	$likes(A,A)$	$\forall\rightarrow\mathcal{E}(3)$
	11	$likes(A,A)$	$\exists\mathcal{E}(5)$
	12	$\forall x.\ likes(x,x)$	$\forall\mathcal{I}$

Figure 17.8 Proof of $\forall x.\ likes(x,x)$

	1	$\forall x,y : num.\ [(\exists z : num.\ x^z = y) \rightarrow R(x,y)]$	
$A\forall\mathcal{I}$	2	$is\text{-}num(A)$	
	3	$is\text{-}num(1)$	(arithmetic)
	4	$A^1 = A$	(arithmetic)
	5	$\exists z : num.\ A^z = A$	$\exists\mathcal{I}$
	6	$R(A,A)$	$\forall\rightarrow\mathcal{E}$
	7	$\forall w : num.\ R(w,w)$	$\forall\mathcal{I}$

Figure 17.9 Proof of $\forall w : num.\ R(w,w)$

2 that $is\text{-}num(A)$ is part of the preparation for $\forall\mathcal{I}$. Since we only want to show $R(w,w)$ for all numbers, A can be an arbitrary number. In turn, to use the sentence at line 1 requires a check that the terms substituted for x,y are both numbers. They are, for both x,y are replaced by A. At line 5 a check must be made that 1 is a number before applying $\exists\mathcal{I}$. Finally, all the rules of arithmetic apply.

Does $\forall x.\ P(x) \vdash \exists x.\ P(x)$? (Figure 17.10)
If you try to show this using natural deduction you will find that you cannot get started because you have no knowledge that any individuals exist and so cannot make any substitutions in the $\forall\mathcal{E}$ or $\exists\mathcal{I}$ rules. In order to show the conclusion you must add to the data the sentence $\exists z.\ \top$, where $\top$ is the sentence that is always true. If you think about it, it is no real surprise that

	1	$\forall x.\ P(x)$	
	2	$\exists z.\ \top$	
$I\exists\mathcal{E}$	3	$\top$	I exists
	4	$P(I)$	$\forall\mathcal{E}$
	5	$\exists y.\ P(y)$	$\exists\mathcal{I}$
	6	$\exists y.\ P(y)$	$\exists\mathcal{E}$

Figure 17.10 $\forall x.\ P(x),\ \exists z.\ \top \vdash \exists y.\ P(y)$

the proof does not work without the extra sentence. For it could be that a situation exists in which there are no individuals. In such a situation, certainly $\forall x.\ P(x)$ is true, for there is nothing to check, but, equally, $\exists y.\ P(y)$ is false.

$\exists z.\ \top$ is often taken for granted, but not in this book.

17.3 Equality

The equality relation '=' is a predicate that is very commonly used and everyone has a fairly good idea of what $a = b$ is supposed to mean — that a and b denote the same element or individual. This in turn means that whatever properties are possessed by a will also be possessed by b. So, for example, if

> Dr Jekyll = Mr Hyde
> Mr Hyde killed someone

then it can be deduced that Dr Jekyll killed someone. For, if the sentence $\exists x.\ \mathit{killed}(\mathit{Mr\ Hyde}, x)$ is satisfied by Mr Hyde, then it is also satisfied by Dr Jekyll, that is, $\exists x.\ \mathit{killed}(\mathit{Dr\ Jekyll}, x)$. The example illustrates the main rule for reasoning with equality — the rule of equality substitution — which allows one side of an equation to be substituted for the other. An equality atom such as $\mathit{Susan} = \mathit{Sue}$ is often called an *equation*.

Using equality in translation

Let us look first at how equality can be used in sentences to express sameness, uniqueness and functionhood.

Consider the following short propositions:

1. Tig eats vegetables
2. Tig only eats vegetables
3. Tig dances with Jig
4. Tig only dances with Jig

The straightforward translations of the first two into logic are

1. $\forall x.\ [\mathit{vegetable}(x) \rightarrow \mathit{eats}(\mathit{Tig}, x)]$
2. $\forall x.\ [\mathit{eats}(\mathit{Tig}, x) \rightarrow \mathit{vegetable}(x)]$

If the third and fourth sentences are paraphrased in a similar way then they become

$$\forall x.\ [x = \mathit{Jig} \rightarrow \mathit{dances\text{-}with}(\mathit{Tig}, x)]$$

and

$$\forall x.\ [\mathit{dances\text{-}with}(\mathit{Tig}, x) \rightarrow x = \mathit{Jig}]$$

An equation is used to express the proposition that 'x is *Jig*', that is, $x = \mathit{Jig}$. The third sentence can be rewritten equivalently and more naturally as $\mathit{dances\text{-}with}(\mathit{Tig}, \mathit{Jig})$.

Equality is also used to express uniqueness. For example, suppose we wanted to express in logic the sentence

There is exactly one green bottle.

This sentence says the following:

1. There is at least one green bottle.
2. There is at most one green bottle.

And in logic we have

$$\exists x.\ \mathit{greenbottle}(x) \wedge \neg\exists u.\ \exists v.\ [\mathit{greenbottle}(u) \wedge \mathit{greenbottle}(v) \wedge u \neq v]$$

An alternative and equivalent expression is obtained by paraphrasing the sentence as

There is a greenbottle x and all greenbottles are the same as x

which in logic is

$$\exists x.\ [\mathit{greenbottle}(x) \wedge \forall u.\ [\mathit{greenbottle}(u) \rightarrow u = x]\ ]$$

The first approach can be generalized for $n \geq 1$ greenbottles:

$$\exists x_1 \ldots x_n \left[\begin{array}{c} \mathit{greenbottle}(x_1) \wedge \ldots \wedge \mathit{greenbottle}(x_n) \\ \wedge x_1 \neq x_2 \wedge \ldots \wedge x_n \neq x_{n-1} \end{array} \right] \wedge$$

$$\neg\exists u_0, \ldots, u_n \left[\begin{array}{c} \mathit{greenbottle}(u_0) \wedge \ldots \wedge \mathit{greenbottle}(u_n) \wedge \\ u_0 \neq u_1 \wedge u_0 \neq u_2 \wedge \ldots \wedge u_1 \neq u_2 \wedge \ldots \\ \wedge \ldots \wedge u_n \neq u_{n-1} \end{array} \right]$$

The second approach can also be generalized:

$$\exists x_1 \ldots x_n \left[\begin{array}{l} greenbottle(x_1) \wedge \ldots \wedge greenbottle(x_n) \wedge \\ x_1 \neq x_2 \wedge \ldots \wedge x_n \neq x_{n-1} \wedge \\ \forall u.\ [greenbottle(u) \rightarrow u = x_1 \vee \ldots \vee u = x_n] \end{array} \right]$$

It is not always necessary to use equality to express 'sameness'. For example, 'a and b have the same parents' might be written as

$$\forall x.\ [parent\text{-}of(x, a) \leftrightarrow parent\text{-}of(x, b)]$$

Actually, the logic only says that 'if a and b have any parents then they have the same ones', and to express that a and b have *some* parents (as implied by the English) we must add

$$\wedge \exists x.\ [parent\text{-}of(x, a)]$$

Equality is also used in expressing that a particular relation is a function. For example, the relation $mother\text{-}of(x, y)$ is a function of y — for each y there is just one x that is related to it. This is expressed as

$$\forall y.\ \forall x.\ \forall z.\ [mother\text{-}of(x, y) \wedge mother\text{-}of(z, y) \rightarrow z = y]$$

If, in addition, we state that 'everyone has a mother'

$$\forall y.\ \exists x.\ mother\text{-}of(x, y)$$

then it is possible to simplify sentences such as

$$\forall u.\ [mother\text{-}of(u, Ann) \leftrightarrow mother\text{-}of(u, Jeremy)]$$

to

$$\exists u.\ [mother\text{-}of(u, Ann) \wedge mother\text{-}of(u, Jeremy)]$$

See Exercise 9.

17.4 Substitution of equality

Equality is such a frequently used predicate that there are built-in natural deduction rules to deal with it. The main natural deduction rule for making use of equations is the *rule of substitution*:

$$\frac{a = b \quad S[a]}{S[b] \quad (eqsub)}$$

where $S[a]$ means a sentence S with one or more occurrences of a identified and $S[b]$ means those occurrences replaced by b. (There is no need to identify all occurrences of a in S.)

Any ground equation of the form $a = a$ can be introduced into a proof by the *reflex rule*

$$\frac{}{a = a \quad (reflex)}$$

The *reflex* rule is usually used in a backwards direction — a conclusion $a = a$ (say) can always be derived by using it.

Any equation $a = b$ means the same as the equation $b = a$. This is a consequence of the *Symmetry law of equality* which is derivable using the two new rules *eqsub* and *reflex*. See Figure 17.11. Line 3 is obtained by

$a, b \forall \mathcal{I}$	1	$a = b$	
	2	$a = a$	*reflex*
	3	$b = a$	*eqsub*
	4	$\forall x.\ \forall y.\ [x = y \rightarrow y = x]$	$\forall \rightarrow \mathcal{I}$

Figure 17.11 Proof of symmetry law

substituting b for the first a of line 2. The symmetry property means that $a = b$ and $b = a$ can be treated as the same equation, for although *eqsub* using $a = b$ is defined as substituting b for an occurrence of a, the use of symmetry allows $b = a$ to be derived and hence a can be substituted for an occurrence of b. The symmetry is not usually made explicit, equalities being used in whichever direction is most appropriate. Transitivity of $=$ $(\forall x.\forall y.\forall z.[x = y \wedge y = z \rightarrow x = z])$ can similarly be shown.

The symmetry of equations enables the *eqsub* rule to make sense whether it is used forwards (as described already) or backwards. In that case, we can use it to show $S[b]$ if we are given $b = a$, which is the same as being given $a = b$, and can show $S[a]$. The effect is to transform the current goal $S[b]$ (say) into a new goal $S[a]$ as at line 4 in the fragment shown in Figure 17.12.

1	$\vdots$	
2	$a = b$	
3	$\vdots$	
4	$S[a]$	
5	$S[b]$	*eqsub*

Figure 17.12

Show $P(a) \leftrightarrow \forall x.\ [x = a \rightarrow P(x)]$ (Figure 17.13)

This example illustrates the use of the *eqsub* and *reflex* rules. The final line of Figure 17.13 is derived by $\wedge\mathcal{I}$ followed by the use of the definition of $A \leftrightarrow B$ as $A \rightarrow B \wedge B \rightarrow A$. The first half of this proof is very useful as it shows how equality conditions of a particular kind can be eliminated. This is

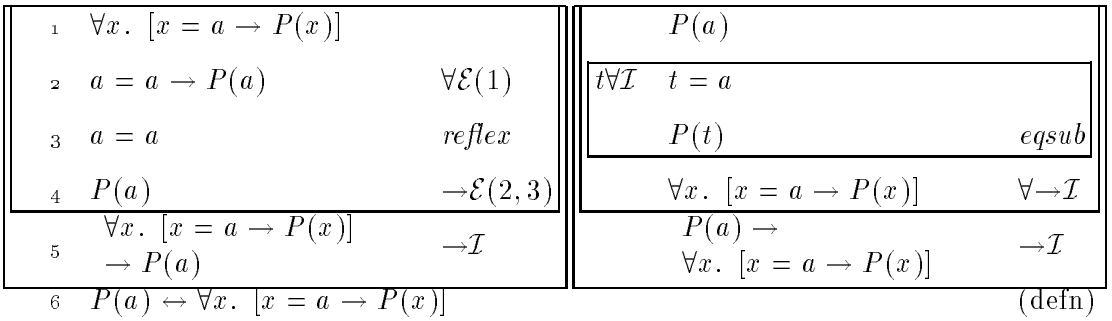

Figure 17.13

always the case for sentences of this sort which have conditions involving an equation with at least one variable argument. For example,

$$\forall x, y.\ [x = a \wedge y = b \wedge P(x, y) \rightarrow Q(x, y)]$$

will yield the simpler $P(a, b) \rightarrow Q(a, b)$.

In a similar way, $\exists x.\ [x = a \wedge P(x)] \leftrightarrow P(a)$ is also true.

Rewrite proofs

A method of showing that an equation is true, familiar from school mathematics, is to use rewriting. That is, to show $a_0 = b$, a_0 is rewritten into

1	$\forall xs, ys.\ [\texttt{rev}\ xs\texttt{++}ys = \texttt{rev}\ ys\texttt{++rev}\ xs]$	
2	$(z\texttt{:}zs) = \texttt{[}z\texttt{]++}zs$	
3	$\texttt{rev}\ (z\texttt{:}zs)$	
4	$= \texttt{rev}\ (\texttt{[}z\texttt{]++}zs)$	(2)
5	$= \texttt{rev}\ zs\texttt{++rev [}z\texttt{]}$	$\forall\mathcal{E}(1)$
6	$= \texttt{rev}\ zs\texttt{++[}z\texttt{]}$	prop of `rev`

Figure 17.14 A rewrite proof

a_1, and then a_1 is rewritten into a_2, and so on, until b is obtained. Each step implicitly uses the *eqsub* rule. A typical proof using this technique is used to derive $\texttt{rev}\ (z\texttt{:}zs) = \texttt{rev}\ zs\texttt{++[}z\texttt{]}$, shown in Figure 17.14.

A rewrite proof can be seen as a contraction of a more cumbersome sequence of equations in which each follows from the next by the *eqsub* rule. The corresponding full proof of Figure 17.14 is given in Figure 17.15. The

1	$\forall xs, ys.\ [\texttt{rev}\ xs\texttt{++}ys = \texttt{rev}\ ys\texttt{++rev}\ xs]$	
2	$(z\texttt{:}zs) = \texttt{[}z\texttt{]++}zs$	defn of :
3	$\texttt{rev [}z\texttt{]} = \texttt{[}z\texttt{]}$	property of `reverse`
4	$\texttt{rev}\ zs\texttt{++[}z\texttt{]} = \texttt{rev}\ zs\texttt{++[}z\texttt{]}$	*reflex*
5	$\texttt{rev}\ zs\texttt{++rev [}z\texttt{]} = \texttt{rev}\ zs\texttt{++[}z\texttt{]}$	*eqsub*(3)
6	$\texttt{rev}\ (\texttt{[}z\texttt{]++}zs) = \texttt{rev}\ zs\texttt{++[}z\texttt{]}$	$\forall\mathcal{E}(1)$, *eqsub*
7	$\texttt{rev}\ (z\texttt{:}zs) = \texttt{rev}\ zs\texttt{++[}z\texttt{]}$	*eqsub*(2)

Figure 17.15

proof uses some properties of `rev`, one occurrence of the *reflex* rule and several applications of *eqsub*. It has the general pattern shown in Figure 17.16, where at each step *eqsub* is used to rewrite either the left or right side of an equation. (So either a_i is identical to a_{i-1} or b_i is identical to b_{i-1}.) The proof given in Figure 17.16 is naturally formed by working backwards from the conclusion, at each step applying *eqsub* to some term until the two sides are identical, when the *reflex* rule is used. It can quite naturally be contracted into the rewrite proof given in Figure 17.17.

various equations	
$\vdots$	
$a_n = a_n$	*reflex*
$a_{n-1} = b_{n-1}$	*eqsub*
$\vdots$	
$a_1 = b_1$	*eqsub*
$a_0 = b_0$	*eqsub*

Figure 17.16

Delete

We will illustrate the various features of natural deduction by proving that the `del` program meets its specification (that is, it deletes the first occurrence

various equations

$\vdots$

$a_0 = a_1$

$= a_2$

$\vdots = a_n$

$= b_{n-1}$

$\vdots = b_0$

Figure 17.17

of c from l). First of all the program and specification:

```
del ::  * -> [* ]-> [* ]
||pre:  c belongs to l
||post:  (E)m,n:[* ][z=m++n & l=m++[c ]++n &
|| not(c belongs to m)]
|| where z= del c l
del c (h:t) = t, c=h
= (h:  del c t), c ≠ t
```

Now the proof — the outline structure is given in Figure 17.18 and the two cases for the induction step are given in Figures 17.19 and 17.20. In the proof we use the following abbreviations:

$$P(l) \equiv \forall c : *.\ [c \in l \rightarrow Q(l)]$$

and

$$Q(l) \equiv \exists m, n : [*][\texttt{del}\ c\ l = m\texttt{++}n \wedge l = m\texttt{++}[c]\texttt{++}n \wedge \neg c \in m]$$

We also give the proof in English for comparison.

Proposition 17.1 `del` satisfies its specification. We have to show $\forall l : [*].\ P(l)$ and we use induction on l and show $P([])$ and $P(h{:}t)$.

The base case $P([])$ is vacuously true because $c \in []$ is always false. For the induction step we can assume as hypothesis $P(t)$:

$$\forall c.\ [c \in t \rightarrow \exists m, n : [*][\texttt{del}\ c\ t = m\texttt{++}n \wedge t = m\texttt{++}[c]\texttt{++}n \wedge \neg c \in m]]$$

So, fix c as a constant C and suppose $C \in h{:}t$. There are two cases: either $C = h$ or $C \neq h$. If $C = h$ then $l = [\ ]\texttt{++}[C]\texttt{++}t$ with $C \notin [\ ]$, and by definition $\texttt{del}\ C\ l = t = [\ ]\texttt{++}t$. Hence we can take $m = [\ ], n = t$. If $C \neq h$ then notice that because $C \in h{:}t$ we must have $C \in t$ and hence by the hypothesis there is some $m1$ and $n1$ such that

$$[\texttt{del}\ C\ t = m1\texttt{++}n1 \wedge t = m1\texttt{++}[C]\texttt{++}n1 \wedge \neg C \in m1]$$

		Base Case		Induction step	
$\forall\mathcal{I}$	2	$c1 : *$		$h : *, t : [*]$	
	3	$c1 \in [\,]$		$P(t)$	hypothesis
	4	$\bot$	prop. of lists	$\forall\mathcal{I}$ $C : *$	
	5	$Q([])$	$\bot\mathcal{E}$	$C \in (h{:}t)$	
	6	$\forall c : *.\ [c \in [] \rightarrow Q([])]$	$\forall{\rightarrow}\mathcal{I}$	$C = h \vee C \neq h$	
	7	$P([\,])$	defn	$C = h$ | $C \neq h$	
	8			$\vdots$ | $\vdots$	
	9			$Q(h{:}t)$ | $Q(h{:}t)$	
	10			$Q(h{:}t)$	$\vee\mathcal{E}(5)$
	11			$\forall c : *.\ [c \in (h{:}t) \rightarrow Q(h{:}t)]$	$\forall{\rightarrow}\mathcal{I}$
	12			$P(h{:}t)$	defn
	13	$\forall l : [*].\ P(l)$			induction

Figure 17.18 Outline proof of delete

1	First part of $\vee\mathcal{E}$	
2	$C = h$	
3	$\texttt{del}\ C\ (C{:}t) = [\,]\texttt{++}t \wedge$ $(C{:}t) = [\,]\texttt{++}[C]\texttt{++}t \wedge \neg C \in [\,]$	$\wedge\mathcal{I}\ (\texttt{del}\ C\ (C{:}t) = t)$
4	$\texttt{del}\ C\ (h{:}t) = [\,]\texttt{++}t \wedge$ $(h{:}t) = [\,]\texttt{++}[C]\texttt{++}t \wedge$ $\neg C \in [\,]$	eqsub(2)
5	$\exists m, n : [*] \left[\begin{array}{l} \texttt{del}\ C\ (h{:}t) = m\texttt{++}n \wedge \\ (h{:}t) = m\texttt{++}[C]\texttt{++}n \wedge \\ \neg C \in m \end{array} \right]$	$\exists\mathcal{I}(m = [], n = t)$

Figure 17.19

Since $\texttt{del}\ C\ (h{:}t) = (h{:}\texttt{del}\ C\ t) = (h{:}m1)\texttt{++}n1$ and $h{:}t = (h{:}m1)\texttt{++}[C]\texttt{++}n1$ with $C \notin h{:}m1$, we can take $m = h : m1, n = n1$ to satisfy the conclusion. □

In Exercise 10 you are asked to identify the corresponding steps in the formal and informal proofs.

	1	second part of $\vee\mathcal{E}$	
	2	$C \neq h$	
	3	$C \in t$	$(C \neq h$ and $C \in h{:}t)$
	4	$\exists \mathtt{m}, \mathtt{n} : [*] \left[\begin{array}{l} \mathtt{del}\ C\ \ t = m \mathtt{++} n \wedge \\ t = m \mathtt{++} [C] \mathtt{++} n \wedge \\ \neg C \in m \end{array}\right]$	$\rightarrow\mathcal{E}$(hypothesis)
$m1, n1\exists\mathcal{E}$	5	$\begin{array}{l} \mathtt{del}\ C\ \ t = m1 \mathtt{++} n1 \wedge \\ t = m1 \mathtt{++} [C] \mathtt{++} n1 \wedge \\ \neg C \in m1 \end{array}$	
	6	$\mathtt{del}\ C\ \ t = m1 \mathtt{++} n1$	$\wedge\mathcal{E}$
	7	$(h{:}\mathtt{del}\ C\ \ t) = (h{:}m1) \mathtt{++} n1$	properties of lists
	8	$\mathtt{del}\ C\ \ (h{:}t) = (h{:}m1) \mathtt{++} n1$	program
	9	$t = m1 \mathtt{++} [C] \mathtt{++} n1$	$\wedge\mathcal{E}$
	10	$(h{:}t) = (h{:}m1) \mathtt{++} [C] \mathtt{++} n1$	properties of lists
	11	$\neg C \in m1$	$\wedge\mathcal{E}$
	12	$\neg C \in (h{:}m1)$	$(C \neq h)$
	13	$\begin{array}{l} \mathtt{del}\ C\ \ (h{:}t) = (h{:}m1) \mathtt{++} n1 \wedge \\ (h{:}t) = (h{:}m1) \mathtt{++} [C] \mathtt{++} n1 \wedge \\ \neg C \in (h{:}m1) \end{array}$	$\wedge\mathcal{I}$
	14	$Q(h{:}t)$	$\exists\mathcal{I}\ \ (m = (h{:}m1), n = n1)$
	15	$Q(h{:}t)$	$\exists\mathcal{E}(4)$

Figure 17.20

17.5 Summary

- The natural deduction rules for quantifiers are collected in Appendix C.
- The rules $\forall\mathcal{I}$ and $\exists\mathcal{E}$ are automatic, whereas $\forall\mathcal{E}$ and $\exists\mathcal{I}$ are not and require some ingenuity in their use. A useful tactic for dealing with quantifiers is

  > Apply the automatic $\forall\mathcal{I}$ and $\exists\mathcal{E}$ rules as soon as possible for they will yield constants that can be used in $\exists\mathcal{I}$ and $\forall\mathcal{E}$ steps later.

- It can be helpful to apply equivalences to quantified sentences so that the quantifiers qualify the smallest subsentences possible. For example, $\forall x.\ [(\exists y.\ Q(x,y)) \rightarrow P(x)]$ might be easier to deal with than $\forall x.\ \forall y.\ [Q(x,y) \rightarrow P(x)]$.
- The *eqsub* and *reflex* natural deduction rules are also listed in Appendix C.

- Equality is used to express uniqueness and functionhood.
- The equality rules can be used to show the symmetry and transitivity of $=$.
- The equality rules can be used to give a rewrite proof.

17.6 Exercises

1. Show:

   (a) $dragon(Puff),\ \forall x.\ [dragon(x) \rightarrow fly(x)] \vdash\ \exists x.\ fly(x)$

   (b) $\forall x.\ \neg(man(x) \wedge woman(x)),$
   $man(tom),\ woman(jill),\ woman(sophia) \vdash \exists x.\ \neg man(x)$

   (c) $\forall x, y.\ [arc(x, y)] \rightarrow path(x, y),$
   $\forall x, y.\ [\exists z.\ [arc(x, z) \wedge path(z, y)] \rightarrow path(x, y)],$
   $arc(A, B),\ \ arc(B, D),\ \ arc(B, C),\ \ arc(D, C) \vdash \exists u.\ path(u, C)$

   How many different proofs are there?

   (d) $\forall x, y, z.\ [R(x, y) \wedge R(y, z) \rightarrow R(z, x)],\ \forall w.\ R(w, w)$
   $\vdash \forall x, y.\ [R(x, y) \rightarrow R(y, x)]$

   (e) $On(A, B), On(B, C),$
   $\forall x.\ \neg(Blue(x) \wedge Green(x)),\quad Green(A),\quad Blue(C),$
   $\forall x, y.\ [On(x, y) \wedge Green(x) \wedge \neg Green(y) \rightarrow Ans(x, y)]$
   $\vdash \exists x.\ \exists y.\ Ans(x, y)$

   (f) $\forall x, y.\ [\forall z[z \in x \rightarrow z \in y] \rightarrow x \subseteq y],$
   $\forall x.\ \neg(x \in \varnothing),\ \forall y.\ y \in U\ \vdash \forall r.\ \varnothing \subseteq r \wedge \forall s.\ s \subseteq U \wedge \forall t.\ t \subseteq t$

2. Show

   $\forall x, y, z.\ [less(x, z) \wedge less(z, y) \rightarrow between(x, y, z)],$
   $\forall x.\ less(x, s(x)),\ \forall x, y.\ [less(x, y) \rightarrow less(x, s(y))]$
   $\vdash$ (a) $\wedge$ (b) $\wedge$ (c)

   where

   (a) $= between(s(0), s(s(s(0))), s(s(0)))$

   (b) $= \exists x.\ [between(0, x, s(0)) \wedge between(s(s(0)), s(s(s(s(0)))), x)]$

   (c) $= \exists x.\ \exists y.\ [between(0, x, y) \wedge between(s(0), s(s(s(0))), x)]$

3. Use Natural Deduction to show:

   (a) $\forall x.\ \neg P(x) \vdash \neg\exists x.\ P(x)$

   (b) $\neg\exists x.\ P(x) \vdash \forall x.\ \neg P(x)$

   (c) $\forall x.\ [F(x) \wedge G(x)] \vdash \forall x.\ F(x) \wedge \forall x.\ G(x)$

   (d) $\forall x.\ F(x) \vee \forall x.\ G(x) \vdash \forall x.\ [F(x) \vee G(x)]$

(e) $\exists x.\ [F(x) \wedge G(x)] \vdash \exists x.\ F(x) \wedge \exists x.\ G(x)$

(f) $\exists x.\ F(x) \vee \exists x.\ G(x) \vdash \exists x.\ [F(x) \vee G(x)]$

(g) $\forall x, y.\ F(x, y) \vdash \forall u, v.\ F(v, u)$

(h) $\exists x.\ \exists y.\ F(x, y) \vdash \exists u.\ \exists v.\ F(v, u)$

(i) $\exists x.\ \forall y.\ G(x, y) \vdash \forall u.\ \exists v.\ G(v, u)$

(j) $\forall x, y.\ [S(y) \rightarrow F(x)] \vdash \exists y.\ S(y) \rightarrow \forall x.\ F(x)$

(k) $\forall x.\ \neg P(x) \vdash \neg \exists x.\ P(x)$

(l) $\neg \forall x.\ P(x) \vdash \exists x.\ \neg P(x)$
(HINT: assume that $\neg \exists x.\ \neg P(x)$ and derive a contradiction; this time the only way to use the negated premiss.)

(m) $P \rightarrow \forall x.\ Q(x) \vdash \forall x.\ [P \rightarrow Q(x)]$

(n) $\exists x.\ [P \rightarrow Q(x)] \vdash P \rightarrow \exists x.\ Q(x)$

(o) $P \rightarrow \exists x.\ Q(x)\ ,\ \exists z.\ \top \vdash \exists x.\ [P \rightarrow Q(x)]$

(p) $(\exists x.\ P(x)) \rightarrow Q \vdash \forall x.\ [P(x) \rightarrow Q]$

(q) $\exists x.\ [P(x) \rightarrow Q] \vdash (\forall x.\ P(x)) \rightarrow Q$

(r) $\forall x.\ P(x) \rightarrow Q,\ \exists z.\ \top \vdash \exists x.\ [P(x) \rightarrow Q]$

(s) $\forall x.\ [F(x) \vee G(x)] \vdash \forall x.\ F(x) \vee \exists y.\ G(y)$.
(HINT: use $\forall x.F(x) \vee \neg \forall x.\ F(x)$.)

(t) $\forall x.\ \exists y.\ [F(x) \vee G(y)],\ \exists z.\ \top \vdash \exists y.\ \forall x.\ [F(x) \vee G(y)]$
(HINT: use the theorem $X \vee \neg X$ where X is the conclusion $\exists y.\ \forall x.\ [F(x) \vee G(y)]$.)

4. Show by natural deduction

(a) $\forall x.\ P(a, x, x), \forall x, y, z.\ [P(x, y, z) \rightarrow P(f(x), y, f(z))] \vdash P(f(a), a, f(a))$

(b) $\forall x.\ P(a, x, x), \forall x, y, z.\ [P(x, y, z) \rightarrow P(f(x), y, f(z))]$
$\vdash \exists z.\ [P(f(a), z, f(f(a)))]$

(c) $\forall y.\ L(b, y), \forall x, z.\ [L(x, y) \rightarrow L(s(x), s(y))] \vdash \exists z.\ [L(b, z) \wedge L(z, s(s(b)))]$

5. One of the convenient ideas incorporated in Natural Deduction is that it is possible to use 'derivation patterns' (or derivation schemes); for example, the pattern $\neg A, A \vee B \vdash B$ can be derived. Such schemes enable larger steps to be taken in a proof than are possible using only the basic rules. If the scheme is very common it is sometimes called a derived rule and given a name. (The benefit lies in the fact that any sentence can be substituted throughout the scheme for A or B (for example) and the scheme remains true. For example, in (a) below we could have $\forall x.\ [P(b, x) \rightarrow Q(x, x)], P(b, a) \vdash Q(a, a)$.)

Some useful schemes are given below; in each case give a Natural Deduction proof of the scheme. The notation $P[x]$ means that x occurs in the arguments of P if P is a predicate, or, more generally, in P if it is a sentence:

(a) $\forall x.\ [P[x] \to Q[x]], \exists x.\ P[x] \vdash \exists x.\ Q[x]$ or $\forall x.\ [P[x] \to Q[x]], P[a] \vdash Q[a]$, where a is a constant

(b) $\forall x.\ [P[x] \wedge R[x] \to Q[x]], P[a], R[a] \vdash Q[a]$, where a is a constant.
Why doesn't $\forall x.\ [P[x] \wedge R[x] \to Q[x]], \exists x.\ P[x], \exists x.\ R[x] \vdash \forall x.\ Q[x]$ work?

Collecting lots of these schemes together enables more concise derivations to be obtained that are still sure to be correct. There are lots of schemes for arguing about arrays, too. For example,

(c) If $n > 0$ then $\forall i[0 < i < n + 1 \to P[i]] \vdash \forall i[0 < i < n \to P[i]] \wedge P[n]$ holds in both this direction and the opposite one and is useful for dealing with situations when $P[i]$ is a sentence about array values.

6. Use natural deduction to show the following:

(a) $\forall x.\ [x = a \vee x = b]\ ,\ \neg P(b),\ Q(a) \vdash \forall x.\ [P(x) \to Q(x)]$
(HINT: Use the $\vee\mathcal{E}$ and $\perp\mathcal{E}$ rules.)

(b) (1) $\forall x.\ \neg B(x,x) \vdash \forall x.\ \forall y.\ [B(x,y) \to x \neq y]$
(2) $\forall x.\ \forall y.\ [B(x,y) \to x \neq y] \vdash \forall x.\ \neg B(x,x)$

(c) KB is either at home or at college, KB is not at home
$\vdash$ home $\neq$ college.

(d) Everyone likes John, John likes no-one but Jack $\vdash$ John = Jack.

(e) S is green, S is the only thing in the box
$\vdash$ Everything in the box is green.

(f) $\forall x.\ \forall y.\ \forall z.\ [R(x,y) \wedge R(x,z) \to z = y],\ R(a,b),\ b \neq c \vdash\ \neg R(a,c)$.

(g) $a = b \vee a = c,\ a = b \vee c = b,\ P(a) \vee P(b) \vdash P(a) \wedge P(b)$

(h) $\vdash \forall x.\ \exists y.\ y = f(x)$

(i) $\vdash \forall y.\ [y = f(a) \to \forall z.\ [z = f(a) \to y = z]]$

(j) $\forall x.\ [x = a \vee x = b],\ g(a) = b,$
$\forall x.\ \forall y.\ [g(x) = g(y) \to x = y] \vdash g(g(a)) = a$
(HINT: You will need to use $\forall\mathcal{E}$ in the first sentence with $g(b)$ substituted for x.)

7. Express in logic:

(a) For each x there is at most one y such that $y = f(x)$.

(b) For each x there is exactly one y such that $y = f(x)$.

8. Show (a) (1) $\vdash$ (2), (b) (2) $\vdash$ (3) and (c) (3) $\vdash$ (1) by natural deduction:

   (1) $\exists x.\ [g(x) \wedge \forall z.\ [g(z) \rightarrow z = x]]$

   (2) $\exists x.\ \forall z.\ [g(z) \leftrightarrow z = x]$

   (3) $\exists x.\ [g(x)] \wedge \forall z.\ \forall y.\ [g(z) \wedge g(y) \rightarrow z = y]$

9. Show by natural deduction that

$$\begin{array}{l} \forall y.\ \forall x.\ \forall z.\ [\mathit{mother\text{-}of}(x,y) \wedge \mathit{mother\text{-}of}(z,y) \rightarrow z = y] \\ \forall y.\ \exists x.\ \mathit{mother\text{-}of}(x,y) \\ \vdash \quad \forall u.\ [\mathit{mother\text{-}of}(u, \mathit{Ann}) \leftrightarrow \mathit{mother\text{-}of}(u, \mathit{Jeremy})] \ \leftrightarrow \\ \quad\ \exists u.\ [\mathit{mother\text{-}of}(u, \mathit{Ann}) \wedge \mathit{mother\text{-}of}(u, \mathit{Jeremy})] \end{array}$$

10. Identify the corresponding steps between the English and box proofs in Section 17.4, in which it was shown that `del` meets its specification.
11. Give Miranda programs for the functions given below and then use box proofs to prove, using induction if appropriate, that the functions meet their specifications. That is, show that the specification follows from any assumed pre-conditions and the execution and termination of the program. (Show that the program terminates as well.)

    (a) `last :: [char] -> char`; `last` x is the last character of x

$$\forall x : \texttt{[*]}.\ [x \neq \texttt{[]} \rightarrow \exists y : \texttt{[*]}.\ x = y\texttt{++[last } x\texttt{]}\ ]$$

    (b) `odd:: num -> num`; `odd` x is the least odd number larger than x

$$\forall x : \mathit{num} \left[\begin{array}{l} \mathit{odd}(\texttt{odd } x) \wedge x < \texttt{odd } x \wedge \\ \neg \exists y : \mathit{num}.\ [\mathit{odd}(y) \wedge y > x \wedge y < \texttt{odd } x] \end{array} \right]$$

    (c) `prime:: num` $\rightarrow$ `Bool`; `prime` x is true iff x is prime

$$\forall x : \mathit{num}.\ [\texttt{prime } x \leftrightarrow \neg \exists z : \mathit{num}.\ [\mathit{divisor}(z,x) \wedge z \geq 1 \wedge z < x]]$$

    (d) `uni:: [char] -> Bool: uni` x is true iff x has no duplicates

$$\forall x : [\mathit{char}] \left[\begin{array}{l} \texttt{uni } x \leftrightarrow \neg \exists y : \mathit{char}. \\ \exists m : \texttt{[}\mathit{char}\texttt{]}.\ \exists n : \texttt{[}\mathit{char}\texttt{]}.\ \exists p : \texttt{[}\mathit{char}\texttt{]}. \\ [x = m\texttt{++[}y\texttt{]++}n\texttt{++[}y\texttt{]++}p] \end{array} \right]$$

Chapter 18

Models

18.1 Validity of arguments

So far, we have used natural deduction to justify that a conclusion C follows from some premisses P and when we successfully derive C from P we write $P \vdash C$.

We justified the natural deduction rules from an informal idea of meaning: $P \vdash C$ is intended to capture the fact that in any situation where P holds, C must hold, too. But the relation $P \vdash C$ that we ended up defining — 'C can be proved from P by natural deduction' makes no mention of 'situations' or of sentences 'holding' and is purely formal: to apply the rules correctly (though to do it successfully and reach the desired conclusion is another matter) you just need to manipulate the syntactic structure of the sentences, the symbols used to write them down. So how do we know that $P \vdash C$ means what we intended? To give any kind of answer we need a more mathematical account of the meanings of the symbols, and this will enable us to give a precise definition of an independent relation $P \models C$ that more plainly says 'in any situation where P holds then C holds, too'. Our question, then, is whether $\vdash$ and $\models$ are equivalent:

- If we prove $P \vdash C$ by natural deduction, do we really know $P \models C$? (that is, is natural deduction *sound*?)
- If $P \models C$ is it possible to prove $P \vdash C$ by natural deduction? (that is, is natural deduction *complete*?)

We call the relationship $\models$ *logical implication* or *logical entailment*. When $P \models C$ is true, we say that it is a *valid* statement or argument.

Informal predicate structures

When you write a set of sentences in logic, you usually have in mind some interpretations which can be attached to the symbols used. For example, in writing *lives(John, Fort William)* $\rightarrow$ *likes(John,climbing)* you might have in mind that *John* referred to a particular person called John, *Fort William* referred to the place in Scotland, *climbing* referred to a sport, and *lives* and *likes* were predicates with their usual interpretations. But this need not be so. Perhaps the sentence is secret code for something else, and *John* refers to a place, *Fort William* and *climbing* to a time, *lives* to the predicate 'good weather at' and *likes* to the predicate 'will smuggle at'. Then the sentence could be saying that if the weather at some place and time is predicted to be good, that place and another time will be used for smuggling! The reader of such a sentence can only understand it if a precise interpretation of the symbols is given.

More usually, we indicate the particular interpretation we have in mind by using standard notation. For instance, a constant called 0 would suggest the number zero, a binary function called $+$ and written infix $(x + y)$ would suggest numeric addition and a binary predicate called $\leq$ and written infix would suggest numeric comparison. Moreover, these implicitly introduce a domain of objects (the numbers) that the sentences are about.

If you are writing your sentences about numbers, you would certainly expect ordinary facts about numbers such as $\forall x.\ x \leq x$ to be available for use without being explicitly written down. But for the moment we are going to look at what pure logic can do on its own, without knowing any implicit premisses. The idea behind logical implication is to be able to forget about intended meanings and to focus on the logical structure instead.

Formal predicate structures

Logic itself provides us with connectives and quantifiers, but the predicates, functions and constants used in sentences are 'extralogical' — outside logic. Hence to know exactly what sentences we are allowing, we need to know what extralogical symbols we are using and how they are used — whether they are predicates, functions or constants, and (for predicates and functions) what their *arities* are. A specification of this extralogical information is called a *signature*. For instance, the sentence $\forall x.\ [P(x) \rightarrow \exists y.\ Q(x, f(y))]$ uses a signature that comprises (at least) a unary (unary means one argument — of arity 1) function $f(\ )$ and two predicates, $P(\ )$ and $Q(\ ,\ )$.

To find the meaning of a sentence we need to know both the range of possible values over which variables can vary, and the meanings, or *interpretations*, of the extralogical symbols. We provide these through the idea

of a *structure* for a signature: the structure comprises

- a set D, known as the *domain*;
- for each constant in the signature, a corresponding element of the domain;
- for each function symbol in the signature, an actual function from D^n to D (where n is the arity of the function); and
- for each predicate P, an n-ary relation on D, that is, a subset of D^n (where n is the arity of P).

D^n here is the set of n-tuples of elements from D: so in Miranda notation, D^2, the set of pairs, is (D, D), D^3 is (D, D, D), and so on. Also, D^1 is D and D^0 has only one element, the unique '0-tuple' ( ).

The idea for the predicates is that $P(u, v, \ldots)$ should be true if and only if the tuple $(u, v, \ldots)$ is in the corresponding subset of D^n. Note that if $n = 0$ (the predicate has no arguments — it is a proposition) then P is interpreted either as true (the subset is $\{( )\}$) or false (the subset is $\{\ \}$).

Example 18.1 of Signatures

1. Suppose we have a signature with predicates $P(\ )$ and $Q(\ ,\ )$, no functions, and a constant A. Two possible structures are

   (a) The Domain is the set of authors of this book
   $P(v)$ means 'v is female'
   $Q(u, v)$ means 'u lives further away from College than v'
   A is the first in alphabetical order (that is, *hessam*)

   (b) Domain is the set of positive integers
   $P(v)$ means 'v is even'
   $Q(u, v)$ means '$u < v$'
   A is the number 1

2. Suppose the signature has predicate $P(\ ,\ ,\ )$, function $s(\ )$ and constant a then two different structures are

   (a) Domain is the set of positive or zero integers
   $P(x, y, z)$ means $x + y = z$
   $s(n)$ means $n + 1$
   a is the number 0

   (b) Domain is the set of integers ≥ 1
   $P(x, y, z)$ means $x \times y = z$
   $s(n)$ means $2 \times n$
   a is the number 1

Once we have a structure for a sentence S, that is to say a structure for a signature that includes all the extralogical symbols used in S, then we can determine the truth or falsity of S by using the rules given earlier and repeated in Figure 18.1.

- $\forall x.\ S$ is true iff for each d in D, $S(d/x)$ is true, where $S(d/x)$ means d replaces every occurrence of x in S that is bound by $\forall x$.
- $\exists x.\ S$ is true iff for some d in D, $S(d/x)$ is true
- $A \wedge B$ is true iff both A and B are true.
- $A \vee B$ is true iff at least one of A or B is true.
- $A \rightarrow B$ is true iff A is false or both A and B are true.
- $\neg A$ is true iff A is false.
- $A \leftrightarrow B$ is true if A and B are both true or both false.
- $t = u$ is true iff they are identified with the same element in the domain.

Figure 18.1 Determining the truth value of a sentence

Example 18.2

1. Find the truth or falsity of $P(A) \wedge \forall x.\ \exists y.\ [P(x) \rightarrow Q(y, x)]$ using the first pair of structures of Example 18.1.

   (a) $P(A)$ means 'hessam is female', which is false, hence the whole sentence is false. But let us find the truth value of the other constituent $\forall x.\ \exists y.\ [P(x) \rightarrow Q(y, x)]$ anyway. It means $\forall x.\ \exists y.\ [female(x) \rightarrow lives\text{-}further\text{-}from\text{-}college(y, x)]$ and its truth value will depend on the value for each x in the domain, that is, for $x = hessam$, $x{=}krysia$, $x{=}steve$ and $x{=}susan$.

   $x = hessam$: $\exists y.\ [female(hessam) \rightarrow lives\text{-}further\text{-}from\text{-}college(y, hessam)]$ is true for any y as $female(hessam)$ is false.
   Similarly for $x = steve$.

   $x = krysia$: $\exists y.\ [female(krysia) \rightarrow lives\text{-}further\text{-}from\text{-}college(y, krysia)]$ is true as $female(krysia) \rightarrow lives\text{-}further\text{-}from\text{-}college(steve, krysia)$ is true, as $lives\text{-}further\text{-}from\text{-}college(steve, krysia)$ is true.
   Similarly for $x = susan$.

   Thus $\forall x.\ \exists y.\ [P(x) \rightarrow Q(y, x)]$ is true in this structure.

   (b) After interpreting the symbols P, A, Q we have

   $even(1) \wedge \forall x.\ \exists y.\ [even(x) \rightarrow y < x]$ is again false since 1 is not an even integer.
   However, $\forall x.\ \exists y.\ [even(x) \rightarrow y < x]$ is true:

   even integers $x \geq 2$: $\exists y.\ [even(x) \rightarrow y < x]$ is true, for y can always be $x - 1$.

   odd integers $x \geq 1$: $\exists y.\ [even(x) \rightarrow y < x]$ is true for any choice of y, for $even(x)$ is false.

2. Find a structure with Domain = {*james*,*edward*} that makes both (i) and (ii) true.

   (i) $Dr\ Jekyll = Mr\ Hyde$

   (ii) $\exists x.\ killed(Mr\ Hyde,\ x)$

   Either both Dr Jekyll and Mr Hyde must be *edward* or both must be *james* in order to satisfy (i). Say they are both interpreted as *edward*. To make (ii) true, at least one of *killed(edward,edward)* or *killed(edward,james)* must be true.

At last we come to the important notion of *model*: a model for a sentence S is a structure in which S is true. We can now say that

> $A \models B$ is true if each structure of $\{A, B\}$ that is a model of A is also a model of B.

and

> $A \models B$ is false if some structure of $\{A, B\}$ that is a model of A is not a model of B.

In general, it is rather difficult to test directly whether $A \models B$ is true for there are very many structures to check. Natural deduction allows us to circumvent this difficulty. The two relations $\models$ and $\vdash$ between a set of sentences S and a conclusion T are the same. That is, if you want to show $S \models T$ you can show $S \vdash T$ instead, that is, if $S \vdash T$ then $S \models T$. It is also the case that if $S \models T$ then $S \vdash T$ so that natural deduction is an adequate alternative to checking models.

These properties are, respectively, called *soundness* and *completeness* of natural deduction and their proofs are discussed in Sections 18.6 and 18.7.

18.2 Disproving arguments

By now you will have tried to prove all sorts of arguments by natural deduction and may well be finding that sometimes it is just not possible to find a proof. In other words, for some problem to show $P \vdash C$, there seems no way to derive C from premisses P by natural deduction. In this case, what can you conclude? Can you conclude that $P \nvdash C$? Well, no, you cannot. For in any proof that appears to be stuck you can, for example, go on introducing theorems of the form $X \vee \neg X$ for all kinds of exotic formulas X and one of them just might lead to a proof of C — you never can tell. Instead, you might try to show that, after all, $P \nvdash C$ does not hold. You can do that by finding a counter-example interpretation of $\{P, C\}$ which makes P true but C false. We might call this the 'failed natural deduction by counter-example' technique.

Certainly, if $P \not\vdash C$ then it will not be possible to show $P \vdash C$, for if it were, $P \models C$ *would* hold (by soundness, which we shall prove in Section 18.6). The next few examples show some typical situations in derivations that cannot be completed successfully. Very often, the apparent impasse provides some help as to what the counter-example interpretation might be.

Try to show $\forall x.\ P(x,x) \vdash \forall u.\ \forall y.\ P(u,y)$

	1	$\forall x.\ P(x,x)$	
$a\forall\mathcal{I}$	2		
$b\forall\mathcal{I}$	3	$\vdots$	
	4	{cannot show $P(a,b)$}	
	5	$P(a,b)$	
	6	$\forall y.\ P(a,y)$	$\forall\mathcal{I}$
	7	$\forall u.\ \forall y.\ P(u,y)$	$\forall\mathcal{I}$

Figure 18.2 Failure to prove $\forall x.\ P(x,x) \vdash \forall u.\ \forall y.\ P(u,y)$

The failure in Figure 18.2 occurs because no instances of $\forall x.\ P(x,x)$ will yield $P(a,b)$. When b is introduced, it is in a context that now includes a and so b cannot be the same as a. In this case, from the failed derivation a counter-example situation can be found:

Let the domain be the set of constants $\{a,b\}$ and suppose $P(a,a)$ and $P(b,b)$ are true and other atoms are false; then this is a situation in which $\forall x.\ P(x,x)$ is true but $\forall u.\ \forall y.\ P(u,y)$ is false.

Try to show $\exists x.\ P(x) \vdash \forall x.\ P(x)$

Here, a is introduced in a context which includes b and so a must be different from b and no successful derivation can be found. If instead of using a $\forall\mathcal{I}$ step first a $\exists\mathcal{E}$ step using $\exists x.\ P(x)$ is made, a similar difficulty arises. A counter-example situation can be found here as well — suppose that the domain is again $\{a,b\}$ and take $P(a)$ to be true (as assumed in the proof attempt) and $P(b)$ to be false. Then $\exists x.\ P(x)$ is true but $\forall x.\ P(x)$ is not.

Try to show $\{\exists z.\ \top, \forall x.\ \exists y.\ P(x,y)\} \vdash \exists u\ \forall v.\ P(u,v)$

(see Figures 18.4 and 18.5). In Figure 18.4, after c has been introduced at line 3 it is natural to use it in a $\forall\mathcal{E}$ step and then in a corresponding $\exists\mathcal{I}$ step, in order to try and make $P(c,d)$ and $P(u,v)$ match. But the term used in place of v in the $\forall\mathcal{I}$ step has to be new and so cannot be the same as d. It is easy to see that a counter-example situation must have a domain of