Chapter 7

Gentzen's Sharpened Hauptsatz; Herbrand's Theorem

7.1 Introduction

We have mentioned in Chapter 6 that the cut elimination theorem shows the existence of normal forms for proofs, namely the fact that every LK-proof can be transformed to a cut-free proof (or a proof without essential cuts in LK_e).

In this chapter we shall use Gentzen's cut elimination theorem (also called Gentzen's Hauptsatz) to prove a version of Herbrand's theorem for LK and LK_e. A derivation of Herbrand's theorem from the cut elimination theorem has the advantage that it yields a constructive version of the result, in the spirit of Herbrand's original version (Herbrand, 1971). The proof given in this chapter using Gentzen's Hauptsatz is inspired from a method sketched in Kleene, 1967.

Herbrand's theorem is perhaps the most fundamental result of first-order logic because it shows how the provability of a formula of first-order logic reduces to the provability of a quantifier-free formula (obtained from the original formula by substitutions).

Before proceeding any further, we wish to emphasize that Herbrand's original theorem is concerned with *provability*, a proof-theoretic concept, and not *validity*, a semantic concept.

This is an important point because another theorem known as Skolem-Herbrand-Gödel theorem is often improperly referred to as Herbrand's theorem in the literature, thus causing a confusion. The Skolem-Herbrand-Gödel theorem is similar in form to Herbrand's theorem but deals with *unsatisfiability* (or validity), a semantic concept.

The reason for the confusion is probably that, by Gödel's completeness theorem, validity can be equated to provability. Hence, Herbrand's original theorem can also be stated for unsatisfiability (or validity).

However, Herbrand's original theorem is a deeper result, whose proof is significantly harder than the Skolem-Herbrand-Gödel theorem, and Herbrand's theorem also yields more information than the latter. More on this subject will be said at the end of Sections 7.5 and 7.6. For an illuminating discussion, the reader should consult Goldfarb's introduction to Herbrand, 1971.

In this chapter, we shall give in Section 7.5 a version of Herbrand's original theorem for prenex formulae, and in Section 7.6 a version of the Skolem-Herbrand-Gödel theorem for formulae in NNF (actually, half of this theorem is more like Herbrand's original theorem).

The Skolem-Herbrand-Gödel theorem can be viewed as the theoretical basis of most theorem proving methods, in particular the resolution method, one of the best known techniques in automatic theorem proving. In fact, the completeness of the resolution method will be shown in Chapter 8 by combining the Skolem-Herbrand-Gödel theorem and theorem 4.3.2.

A constructive version of Herbrand's theorem can be obtained relatively easily from Gentzen's sharpened Hauptsatz, which is obtained by analyzing carefully cut-free proofs of formulae of a special type.

Gentzen's sharpened Hauptsatz shows that provided the formulae in the bottom sequent have a certain form, a proof can be reorganized to yield another proof in *normal form* such that *all the quantifier inferences appear below all the propositional inferences*. The main obstacle to the permutation of inferences is the possibility of having a quantifier rule applied to the side formula arising from a propositional rule as illustrated below:

EXAMPLE 7.1.1

$$\dfrac{P(a) \rightarrow Q(f(g(a))) \qquad \dfrac{Q(f(g(a))) \rightarrow Q(f(g(a)))}{\forall x Q(f(x)) \rightarrow Q(f(g(a)))}}{P(a) \lor \forall x Q(f(x)) \rightarrow Q(f(g(a)))}$$

The obvious solution that consists in permuting the $\lor : left$ rule and the $\forall : left$ rule does not work since a quantifier rule is not allowed to apply to a subformula like $\forall x Q(f(x))$ in $P(a) \lor \forall x Q(f(x))$.

There are at least two ways of resolving this difficulty:

(1) Impose restrictions on formulae so that a quantifier inference cannot be applied to the side formula of a propositional inference.

(2) Allow more general quantifier rules applying to subformulae.

The standard approach has been to enforce (1) by requiring the formulae to be *prenex formulae*. A Prenex formula is a formula consisting of a (possibly empty) string of quantified variables followed by a quantifier-free formula, and it is easily seen that (1) holds.

The second approach is perhaps not as well known, but is possible. Smullyan has given quantifier rules (page 122 in Chapter 14 of Smullyan, 1968) that allow the permutation process to be performed for unrestricted formulae, and a general version of the extended Hauptsatz is also given (theorem 2, page 123). These rules are binary branching and do not seem very convenient in practice. We will not pursue this method here and refer the interested reader to Smullyan, 1968.

We have also discovered that Andrews's version of the Skolem-Herbrand-Gödel theorem (Andrews, 1981) suggests quantifier rules such that (2) holds for formulae in negation normal form. Such rules are quite simple, and since there is a refutation method based on Andrews's version of Skolem-Herbrand-Gödel's theorem (the method of matings), we shall give a proof of the extended Hauptsatz for such a system. Since every formula is equivalent to a prenex formula and to a formula in negation normal form, there is no loss of generality in restricting our attention to such normal forms. In this Chapter, it is assumed that no variable occurs both free and bound in any sequent (or formula).

7.2 Prenex Normal Form

First, we define the concept of a prenex formula.

Definition 7.2.1 (Prenex form) A formula is a *prenex formula* (or in *prenex form*) iff either it contains no quantifiers, or it is of the form $Q_1x_1...Q_nx_nB$, where B is a formula with no quantifiers (quantifier-free), $x_1, ..., x_n$ are (not necessarily distinct) variables, and $Q_i \in \{\forall, \exists\}$, for $i = 1, ..., n$.

In order to show that every formula is equivalent to a prenex formula, the following lemma will be used.

Lemma 7.2.1 The following formulae are valid:

(a)
$$\forall x(A \supset B) \equiv (\exists xA \supset B)$$
$$\exists x(A \supset B) \equiv (\forall xA \supset B)$$

where x does not occur free in B.

$$\forall x(A \supset B) \equiv (A \supset \forall xB)$$
$$\exists x(A \supset B) \equiv (A \supset \exists xB)$$

where x does not occur free in A.

(b)
$$\neg\forall xA \equiv \exists x\neg A$$
$$\neg\exists xA \equiv \forall x\neg A$$

(c)
$$(\forall xA \lor B) \equiv \forall x(A \lor B)$$
$$(\forall xA \land B) \equiv \forall x(A \land B)$$
$$(\exists xA \lor B) \equiv \exists x(A \lor B)$$
$$(\exists xA \land B) \equiv \exists x(A \land B)$$

where x does not occur free in B.

(d)
$$\forall x(A \land B) \equiv \forall xA \land \forall xB$$
$$\exists x(A \lor B) \equiv \exists xA \lor \exists xB$$

(e)
$$\forall xA \equiv \forall yA[y/x]$$
$$\exists xA \equiv \exists yA[y/x]$$

where y is free for x in A, and y does not occur free in A unless $y = x$ ($y \notin FV(A) - \{x\}$).

Proof: Some of these equivalences have already been shown. We prove two new cases, leaving the others as an exercise. A convenient method is to construct a proof tree (in G).

We give proofs for $(\forall xA \land B) \equiv \forall x(A \land B)$ and $(\forall xA \lor B) \equiv \forall x(A \lor B)$, where x is not free in B. As usual, to prove $X \equiv Y$, we prove $X \supset Y$ and $Y \supset X$.

(i) Proof of $(\forall xA \land B) \supset \forall x(A \land B)$:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{A[y/x], B \to A[y/x] \qquad\qquad A[y/x], B \to B}{A[y/x], B \to (A[y/x] \land B)}}{\forall xA, B \to (A \land B)[y/x]}}{\forall xA, B \to \forall x(A \land B)} \quad \text{for a new variable } y}{(\forall xA \land B) \to \forall x(A \land B)}}{\to (\forall xA \land B) \supset \forall x(A \land B)}$$

We have used the fact that since x is not free in B, $B[y/x] = B$.

(ii) Proof of $\forall x(A \land B) \supset (\forall xA \land B)$

$$\dfrac{\dfrac{\dfrac{\dfrac{A[y/x], B \rightarrow A[y/x]}{A[y/x] \land B \rightarrow A[y/x]}}{\forall x(A \land B) \rightarrow A[y/x]}}{\forall x(A \land B) \rightarrow \forall x A} \qquad \dfrac{\dfrac{A[y/x], B \rightarrow B}{A[y/x] \land B \rightarrow B}}{\forall x(A \land B) \rightarrow B}}{\rightarrow \forall x(A \land B) \supset (\forall x A \land B)}$$

Again, y is a new variable and we used the fact that $B[y/x] = B$.

(iii) Proof of $(\forall x A \lor B) \supset \forall x(A \lor B)$:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{A[y/x] \rightarrow A[y/x], B}{\forall x A \rightarrow A[y/x], B} \qquad B \rightarrow A[y/x], B}{(\forall x A \lor B) \rightarrow A[y/x], B}}{(\forall x A \lor B) \rightarrow A[y/x] \lor B}}{(\forall x A \lor B) \rightarrow \forall x(A \lor B)}}{\rightarrow (\forall x A \lor B) \supset \forall x(A \lor B)}$$

As in (ii) y is a new variable and we used the fact that $B[y/x] = B$.

(iv) Proof of $\forall x(A \lor B) \supset (\forall x A \lor B)$:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{A[y/x] \rightarrow A[y/x], B \qquad B \rightarrow A[y/x], B}{A[y/x] \lor B \rightarrow A[y/x], B}}{\forall x(A \lor B) \rightarrow A[y/x], B}}{\forall x(A \lor B) \rightarrow \forall x A, B}}{\forall x(A \lor B) \rightarrow (\forall x A \lor B)}}{\rightarrow \forall x(A \lor B) \supset (\forall x A \lor B)}$$

As in (iii) y is a new variable and we used the fact that $B[y/x] = B$. □

Theorem 7.2.1 For every formula A, a prenex formula B can be constructed such that $A \equiv B$ is valid (A and B are equivalent).

Proof: To simplify the proof, we will assume that $Q_1 x_1 ... Q_n x_n B$ denotes the quantifier-free formula B when $n = 0$. The proof proceeds using the induction principle applied to A. The base case is trivial since an atomic formula is quantifier free. The induction step requires six cases. We cover three of these cases, leaving the others as exercises.

(i) $A = (B \vee C)$. By the induction hypothesis, B is equivalent to a prenex formula $\mathrm{Q}_1x_1...Q_mx_mB'$ and C is equivalent to a prenex formula $R_1y_1...R_ny_nC'$, where B' and C' are quantifier free and $Q_i, R_j \in \{\forall, \exists\}$. By lemma 7.2.1(e), it can be assumed that the sets of variables $\{x_1, ..., x_m\}$ and $\{y_1, ..., y_n\}$ are disjoint and that both sets are also disjoint from the union of the sets $FV(B')$ and $FV(C')$ of free variables in B' and C' (by renaming with new variables). Using lemma 7.2.1(c) (several times), we get the prenex formula

$$Q_1x_1...Q_mx_mR_1y_1...R_ny_n(B' \vee C'),$$

which, by lemma 5.3.7 is equivalent to A.

(ii) $A = \neg B$. By the induction hypothesis, B is equivalent to a prenex formula $Q_1x_1...Q_mx_mB'$ where B' is quantifier free. For each Q_i, let

$$R_i = \begin{cases} \forall & \text{if } Q_i = \exists; \\ \exists & \text{if } Q_i = \forall. \end{cases}$$

Using lemma 7.2.1(b) (several times), by lemma 5.3.7 the prenex formula

$$R_1x_1...R_mx_m\neg B'$$

is equivalent to A.

(iii) $A = \forall xB$. By the induction hypothesis, B is equivalent to a prenex formula $Q_1x_1...Q_mx_mB'$. Hence, by lemma 5.3.7 A is equivalent to the prenex formula $\forall xQ_1x_1...Q_mx_mB'$. $\square$

EXAMPLE 7.2.1

Let

$$A = \forall x(P(x) \vee \neg\exists y(Q(y) \wedge R(x, y))) \vee \neg(P(y) \wedge \neg\forall xP(x)).$$

The prenex form of

$$\neg\exists y(Q(y) \wedge R(x, y))$$

is

$$\forall y\neg(Q(y) \wedge R(x, y)).$$

The prenex form of

$$\forall x(P(x) \vee \neg\exists y(Q(y) \wedge R(x, y)))$$

is

$$\forall x\forall y(P(x) \vee \neg(Q(y) \wedge R(x, y))).$$

The prenex form of

$$\neg\forall xP(x)$$

is

$$\exists x \neg P(x).$$

The prenex form of

$$(P(y) \land \neg \forall x P(x))$$

is

$$\exists x (P(y) \land \neg P(x)).$$

The prenex form of

$$\neg (P(y) \land \neg \forall x P(x))$$

is

$$\forall x \neg (P(y) \land \neg P(x)).$$

The prenex form of A is

$$\forall x \forall v \forall z ((P(x) \lor \neg (Q(v) \land R(x, v))) \lor \neg (P(y) \land \neg P(z))).$$

In a prenex formula $Q_1 x_1 ... Q_n x_n B$, B is sometimes called the *matrix*. Since B is quantifier free, it is equivalent to a formula in either conjunctive or disjunctive normal form. The disjunctive normal form of a formula can be obtained by using the *search* procedure as explained in theorem 3.4.2. The following lemma, which is the dual of lemma 3.4.5, provides another method for transforming a formula into disjunctive normal form.

Lemma 7.2.2 Every quantifier-free formula A (containing the connectives $\lor$, $\land$, $\neg$, $\supset$) can be transformed into an equivalent formula in disjunctive normal form, by application of the following identities:

$$\begin{gathered}
(A \supset B) \equiv (\neg A \lor B) \\
\neg\neg A \equiv A \\
\neg (A \land B) \equiv (\neg A \lor \neg B) \\
\neg (A \lor B) \equiv (\neg A \land \neg B) \\
A \land (B \lor C) \equiv (A \land B) \lor (A \land C) \\
(B \lor C) \land A \equiv (B \land A) \lor (C \land A) \\
(A \land B) \land C \equiv A \land (B \land C) \\
(A \lor B) \lor C \equiv A \lor (B \lor C)
\end{gathered}$$

Proof: The proof is dual to that of lemma 3.4.5 and is left as an exercise.
□

In the next sections, we consider versions of the sharpened Hauptsatz.

PROBLEMS

7.2.1. Convert the following formulae to prenex form:

$$(\neg\forall x P(x,y) \lor \forall x R(x,y))$$
$$\forall x(P(x) \supset \neg\exists y R(x,y))$$
$$(\neg\forall x\neg\forall y\neg\forall z P(x,y) \lor \neg\exists x\neg\exists y(\neg\exists z Q(x,y,z) \supset R(x,y)))$$

7.2.2. Convert the following formulae to prenex form:

$$(\exists x\forall y P(x,y) \land \forall y\exists x P(y,x))$$
$$(\neg(\forall x P(x) \lor \exists y\neg Q(y)) \lor (\forall z P(z) \lor \exists w\neg Q(w)))$$
$$(\neg\forall x(P(x) \lor \exists y\neg Q(y)) \lor (\forall z P(z) \lor \exists w\neg Q(w)))$$

7.2.3. Finish the proof of lemma 7.2.1.

7.2.4. Finish the proof of theorem 7.2.1.

7.2.5. Prove lemma 7.2.2.

7.2.6. Write a computer program for converting a formula to prenex form.

7.3 Gentzen's Sharpened Hauptsatz for Prenex Formulae

First, we will need a certain normal form for proofs.

7.3.1 Pure Variable Proofs

Gentzen's sharpened Hauptsatz for prenex formulae is obtained by observing that in a cut-free proof of a prenex sequent (in LK or LK_e), the quantifier rules can be interchanged with the propositional rules, in such a way that no propositional rule appears below (closer to the root of the proof tree than) a quantifier rule. There is actually a more general permutability lemma due to Kleene, but only part of this lemma is needed to obtain the sharpened Hauptsatz.

During the course of the proof of the sharpened Hauptsatz, it will be necessary to show that every provable sequent (in LK or LK_e) has a proof in which the variables introduced by $\forall : right$ and $\exists : left$ rules satisfy certain conditions. Proofs satisfying these conditions are called "pure-variable proofs." To illustrate the necessity of these conditions, consider the following example.

EXAMPLE 7.3.1

$$\dfrac{\dfrac{\Gamma \to P(y), Q}{\Gamma \to \forall x P(x), Q}\;\forall : \mathit{right} \qquad \Gamma \to \forall x P(x), R(y)}{\Gamma \to \forall x P(x), Q \wedge R(y)}\;\wedge : \mathit{right}$$

(To shorten proof trees, structural rules are often not shown.) The rule $\forall : \mathit{right}$ occurs above the rule $\wedge : \mathit{right}$. We wish to permute these two rules; that is, construct a proof with the same conclusion and the same premises, but with the $\forall : \mathit{right}$ rule occurring below the $\wedge : \mathit{right}$ rule. The following "proof" almost works, except that the variable y occurs free in the conclusion of the $\forall : \mathit{right}$ rule, violating the eigenvariable condition.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\Gamma \to P(y), Q}{\text{weakening and exchanges}}}{\Gamma \to P(y), \forall x P(x), Q} \qquad \dfrac{\dfrac{\Gamma \to \forall x P(x), R(y)}{\text{weakening and exchanges}}}{\Gamma \to P(y), \forall x P(x), R(y)}}{\Gamma \to P(y), \forall x P(x), Q \wedge R(y)}\;\wedge : \mathit{right}}{\Gamma \to \forall x P(x), \forall x P(x), Q \wedge R(y)}\;\forall : \mathit{right}}{\Gamma \to \forall x P(x), Q \wedge R(y)}\;\text{contraction}$$

The problem can be eliminated by making sure that for every eigenvariable y used in an application of the rule $\forall : \mathit{right}$ or $\exists : \mathit{left}$, that variable only occurs in the subtree rooted at the sequent constituting the premise of the rule (and y does not occur both free and bound, but this is already a condition required for constructing proof trees). If we rename the topmost occurrence of y in the first proof tree with the new variable z, the problem is indeed eliminated, as shown.

EXAMPLE 7.3.2

Pure-variable proof tree

$$\dfrac{\dfrac{\Gamma \to P(z), Q}{\Gamma \to \forall x P(x), Q}\;\forall : \mathit{right} \qquad \Gamma \to \forall x P(x), R(y)}{\Gamma \to \forall x P(x), Q \wedge R(y)}\;\wedge : \mathit{right}$$

New legal proof tree with $\wedge : \mathit{right}$ above $\forall : \mathit{right}$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\Gamma \to P(z), Q}{\text{weakening and exchanges}}}{\Gamma \to P(z), \forall x P(x), Q} \qquad \dfrac{\dfrac{\Gamma \to \forall x P(x), R(y)}{\text{weakening and exchanges}}}{\Gamma \to P(z), \forall x P(x), R(y)}}{\Gamma \to P(z), \forall x P(x), Q \land R(y)}\ \land : \mathit{right}}{\Gamma \to \forall x P(x), \forall x P(x), Q \land R(y)}\ \forall : \mathit{right}}{\Gamma \to \forall x P(x), Q \land R(y)}\ \text{contraction}$$

Definition 7.3.1 A proof tree (in LK, LK_e, G or $G_=$) is a *pure-variable proof tree* iff, for every variable y occurring as the eigenvariable in an application of the rule $\forall : right$ or $\exists : left$, y does not occur both free and bound in any formula in the proof tree, and y only occurs in the subtree rooted at the sequent constituting the premise of the rule.

Lemma 7.3.1 In LK or LK_e (or G or $G_=$), every proof of a sequent $\Gamma \to \Delta$ can be converted to a pure-variable proof of the same sequent, simply by replacing occurrences of variables with new variables.

Proof: First, recall that from lemma 6.5.1, given a sequent $\Gamma \to \Delta$ with proof tree T, where the variable x does not occur bound in T, for any variable y not occurring in T, the tree $T[y/x]$ obtained by substituting y for every free occurrence of x in T is a proof tree for the sequent $\Gamma[y/x] \to \Delta[y/x]$.

The rest of the proof proceeds by induction on the number k of inferences of the form $\forall : right$ or $\exists : left$ in the proof tree T for the sequent $\Gamma \to \Delta$. If $k = 0$, the lemma is obvious. Otherwise, T is of the form

$$\begin{array}{c} T_1 \quad \ldots \quad T_n \\ S \end{array}$$

where each tree T_i is of the form

$$\dfrac{\dfrac{Q_i}{\Gamma_i \to \Delta_i, A_i[y_i/x_i]}}{\Gamma_i \to \Delta_i, \forall x_i A_i}$$

or

$$\dfrac{\dfrac{Q_i}{A_i[y_i/x_i], \Gamma_i \to \Delta_i}}{\exists x_i A_i, \Gamma_i \to \Gamma_i}$$

where S does not contain any inferences of the form $\forall : right$ or $\exists : left$, the root of Q_i is labeled with $\Gamma_i \to \Delta_i, A_i[y_i/x_i]$ (or $A_i[y_i/x_i], \Gamma_i \to \Delta_i$), and the leaves of S are either axioms or are labeled with $\Gamma_i \to \Delta_i, \forall x_i A_i$ (or $\exists x_i A_i, \Gamma_i \to \Gamma_i$). Since each proof tree Q_i contains fewer applications of $\forall : right$ or $\exists : left$ than T, by the induction hypothesis, there is a pure-variable proof tree Q'_i for each $\Gamma_i \to \Delta_i, A_i[y_i/x_i]$ (or $A_i[y_i/x_i], \Gamma_i \to \Delta_i$). Note that no variable free in $\Gamma_i \to \Delta_i, A_i[y_i/x_i]$ (or $A_i[y_i/x_i], \Gamma_i \to \Delta_i$), including y_i, is used as an eigenvariable in Q'_i. For every $i = 1, ..., n$, let Q''_i be the proof tree obtained from Q'_i by renaming all eigenvariables in Q'_i, so that the set of eigenvariables in Q''_i is disjoint from the set of eigenvariables in Q''_j for $i \neq j$, and is also disjoint from the set of variables in S. Finally, for $i = 1, ..., n$, if y_i occurs in S (not below $\Gamma_i \to \Delta_i, \forall x_i A_i$ or $\exists x_i A_i, \Gamma_i \to \Delta_i$, since y_i is an eigenvariable), let z_i be a new variable (not in any of the Q''_i, and such that $z_i \neq z_j$ whenever $i \neq j$), and let T'_i be the tree

$$\cfrac{\cfrac{Q''_i[z_i/y_i]}{\Gamma_i \to \Delta_i, A_i[z_i/x_i]}}{\Gamma_i \to \Delta_i, \forall x_i A_i}$$

or

$$\cfrac{\cfrac{Q''_i[z_i/y_i]}{A_i[z_i/x_i], \Gamma_i \to \Gamma_i}}{\exists x_i A_i, \Gamma_i \to \Gamma_i}$$

obtained by substituting z_i for each occurrence of y_i in Q''_i. Then, using the above claim, one can verify that the following proof tree is a pure-variable proof tree:

$$\begin{array}{c} T'_1 \quad ... \quad T'_n \\ S \end{array}$$

We will also need the following lemma, which is analogous to lemma 6.3.1.

Lemma 7.3.2 (i) Every sequent provable in LK is provable with a proof in which the axioms are sequents $A \to A$, where A is atomic.

(ii) Every sequent provable in LK_e is provable with a proof in which the axioms are either sequents $A \to A$ where A is atomic, or equality axioms.

Proof: The proof proceeds by induction on the weight of a proof as in lemma 6.3.1, and is very similar to the proof given in that lemma, except that LK (or LK_e) inferences are used. The details are left as an exercise. □

7.3.2 The Permutability Lemma

The following terminology will be used in proving the permutability lemma used for establishing the sharpened Hauptsatz.

Definition 7.3.2 If in a proof tree T the conclusion of a one-premise rule R_1 is a descendant of some premise of a rule R_2, and if any intermediate rules occur between R_1 and R_2, then they are structural rules, we say that R_1 *can be permuted with* R_2, iff a proof tree T' can be constructed from T as explained below.

If T is of the form

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{T_1}{S_3}}{S_1'}\;R_1}{ST}}{S_1}\qquad\qquad\dfrac{T_2}{S_2}}{\dfrac{S_0}{T_0}}\;R_2$$

or of the form

$$\dfrac{\dfrac{T_1}{S_3}\qquad\qquad\dfrac{\dfrac{\dfrac{\dfrac{T_2}{S_2}}{S_1'}\;R_1}{ST}}{S_1}}{\dfrac{S_0}{T_0}}\;R_2$$

where S_0, S_1, S_2, S_3, S_1' are sequents, T_0, T_1, T_2 are trees, and the only other rules in ST between R_1 and R_2 (if any) are structural rules, then T' is of the form

$$\dfrac{\dfrac{\dfrac{T_1}{S_3}\qquad\qquad\dfrac{T_2}{S_2}}{Q}}{\dfrac{S_0}{T_0}}$$

and in the tree Q, the conclusion of the rule R_1 is not a descendant of any premise of the rule R_2, and the other rules used in Q besides R_1 and R_2 are structural rules.

If T is of the form

$$\begin{array}{cl}
T_1 & \\ \cline{1-1}
S_2 & \\ \cline{1-1}
 & R_1 \\
S_1' & \\ \cline{1-1}
ST & \\ \cline{1-1}
S_1 & \\ \cline{1-1}
 & R_2 \\
S_0 & \\ \cline{1-1}
T_0 &
\end{array}$$

and the only rules in ST (if any) between R_1 and R_2 are structural rules, then T' is of the form

$$\begin{array}{c}
T_1 \\ \hline
S_2 \\ \hline
Q \\ \hline
S_0 \\ \hline
T_0
\end{array}$$

where in the tree Q, the conclusion of the rule R_1 is not a descendant of the premise of the rule R_2, and the other rules used in Q besides R_1 and R_2 are structural rules.

Lemma 7.3.3 In every pure-variable, cut-free proof T in LK (proof without essential cuts in LK_e) the following properties hold:

(i) Every quantifier rule R_1 can be permuted with a rule R_2 which is either a logical rule, or a structural rule, or the cut rule, provided that the principal formula of the quantifier rule is not a side formula of the lower rule R_2.

(ii) Every quantifier rule, contraction or exchange rule R_1 can be permuted with a weakening rule R_2 whose premise is the conclusion of R_1.

Proof: There are several cases to consider. We only consider some typical cases, leaving the others as exercises.

(i) $R_1 = \forall : right$, $R_2 = \wedge : right$.

We have the following tree:

$$\dfrac{\dfrac{\dfrac{\dfrac{\Gamma' \to \Delta', A[y/x]}{\Gamma' \to \Delta', \forall x A}\;\forall : \mathit{right}}{\text{structural rules } ST}}{\Gamma \to \Delta, \forall x A, \Lambda, B} \qquad \Gamma \to \Delta, \forall x A, \Lambda, C}{\Gamma \to \Delta, \forall x A, \Lambda, B \wedge C}\;\wedge : \mathit{right}$$

The permutation is achieved as follows:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\Gamma' \to \Delta', A[y/x]}{\text{weakening and exchanges}}}{\Gamma' \to A[y/x], \Delta', \forall x A}}{\text{structural rules ST}}}{\Gamma \to A[y/x], \Delta, \forall x A, \Lambda, B} \qquad \dfrac{\dfrac{\Gamma \to \Delta, \forall x A, \Lambda, C}{\text{weakening and exchanges}}}{\Gamma \to A[y/x], \Delta, \forall x A, \Lambda, C}}{\Gamma \to A[y/x], \Delta, \forall x A, \Lambda, B \wedge C}\;\wedge : \mathit{right}}{\text{exchanges}}}{\dfrac{\Gamma \to \Delta, \forall x A, \Lambda, B \wedge C, A[y/x]}{\Gamma \to \Delta, \forall x A, \Lambda, B \wedge C, \forall x A}\;\forall : \mathit{right}}}{\text{exchanges, contraction, and exchanges}}}{\Gamma \to \Delta, \forall x A, \Lambda, B \wedge C}$$

Since the first tree is part of a pure-variable proof, y only occurs above the conclusion of the $\forall : \mathit{right}$ rule in it, and so, y does not occur in the conclusion of the $\forall : \mathit{right}$ rule in the second tree and the eigenvariable condition is satisfied.

(ii) $R_1 = \forall : \mathit{left}$, $R_2 = \wedge : \mathit{right}$.

We have the following tree:

$$\dfrac{\dfrac{\dfrac{\dfrac{A[t/x], \Gamma' \to \Delta'}{\forall x A, \Gamma' \to \Delta'}\;\forall : \mathit{left}}{\text{structural rules } ST}}{\Gamma, \forall x A, \Delta \to \Lambda, B} \qquad \Gamma, \forall x A, \Delta \to \Lambda, C}{\Gamma, \forall x A, \Delta \to \Lambda, B \wedge C}\;\wedge : \mathit{right}$$

The permutation is achieved as follows:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{A[t/x],\Gamma'\to\Delta'}{\text{weakening and exchanges}}}{\forall xA,\Gamma',A[t/x]\to\Delta'}}{\text{structural rules ST}}}{\Gamma,\forall xA,\Delta,A[t/x]\to\Lambda,B} \qquad \dfrac{\dfrac{\Gamma,\forall xA,\Delta\to\Lambda,C}{\text{weakening and exchanges}}}{\Gamma,\forall xA,\Delta,A[t/x]\to\Lambda,C}}{\Gamma,\forall xA,\Delta,A[t/x]\to\Lambda,B\land C}\ \land:right}{\text{exchanges}}}{\dfrac{A[t/x],\Gamma,\forall xA,\Delta\to\Lambda,B\land C}{\forall xA,\Gamma,\forall xA,\Delta\to\Lambda,B\land C}\ \forall:left}}{\text{exchanges, contraction, and exchanges}}}{\Gamma,\forall xA,\Delta\to\Lambda,B\land C}$$

(iii) $R_1=\forall:left$, $R_2=\land:left$.

We have the following tree:

$$\dfrac{\dfrac{\dfrac{\dfrac{A[t/x],\Gamma'\to\Delta'}{\forall xA,\Gamma'\to\Delta'}\ \forall:left}{\text{structural rules ST}}}{B,\Gamma,\forall xA,\Delta\to\Lambda}}{B\land C,\Gamma,\forall xA,\Delta\to\Lambda}\ \land:left$$

The permutation is achieved as follows:

$$\begin{array}{cl}
A[t/x], \Gamma' \to \Delta' & \\ \cline{1-1}
\text{weakening and exchanges} & \\ \cline{1-1}
\forall x A, \Gamma', A[t/x] \to \Delta' & \\ \cline{1-1}
\text{structural rules ST} & \\ \cline{1-1}
B, \Gamma, \forall x A, \Delta, A[t/x] \to \Lambda & \\ \cline{1-1}
B \land C, \Gamma, \forall x A, \Delta, A[t/x] \to \Lambda & \land : \mathit{left} \\ \cline{1-1}
\text{exchanges} & \\ \cline{1-1}
A[t/x], B \land C, \Gamma, \forall x A, \Delta \to \Lambda & \\ \cline{1-1}
\forall x A, B \land C, \Gamma, \forall x A, \Delta \to \Lambda & \forall : \mathit{left} \\ \cline{1-1}
\text{exchanges, contraction, and exchanges} & \\ \cline{1-1}
B \land C, \Gamma, \forall x A, \Delta \to \Lambda &
\end{array}$$

(iv) $R_1 = \forall : \mathit{right}$, $R_2 = \mathit{weakening\ right}$.

We have the following tree:

$$\begin{array}{cl}
\Gamma \to \Delta, A[y/x] & \\ \cline{1-1}
\Gamma \to \Delta, \forall x A & \forall : \mathit{right} \\ \cline{1-1}
\Gamma \to \Delta, \forall x A, B & \text{weakening right}
\end{array}$$

The permutation is performed as follows:

$$\begin{array}{cl}
\Gamma \to \Delta, A[y/x] & \\ \cline{1-1}
\Gamma \to \Delta, A[y/x], B & \text{weakening right} \\ \cline{1-1}
\Gamma \to \Delta, B, A[y/x] & \text{exchange} \\ \cline{1-1}
\Gamma \to \Delta, B, \forall x A & \forall : \mathit{right} \\ \cline{1-1}
\Gamma \to \Delta, \forall x A, B & \text{exchange}
\end{array}$$

Since the first tree is part of a pure-variable proof, y only occurs above the conclusion of the $\forall : \mathit{right}$ rule in it, and so, y does not occur in the conclusion of the $\forall : \mathit{right}$ rule in the second tree and the eigenvariable condition is satisfied.

(v) $R_1 = \mathit{exchange\ right}$, $R_2 = \mathit{weakening\ right}$.

We have the following tree:

$$\begin{array}{cl}
\Gamma \to \Delta, A, B, \Lambda & \\ \hline
\Gamma \to \Delta, B, A, \Lambda & \text{exchange right} \\ \hline
\Gamma \to \Delta, B, A, \Lambda, C & \text{weakening right}
\end{array}$$

The permutation is performed as follows:

$$\begin{array}{cl}
\Gamma \to \Delta, A, B, \Lambda & \\ \hline
\Gamma \to \Delta, A, B, \Lambda, C & \text{weakening right} \\ \hline
\Gamma \to \Delta, B, A, \Lambda, C & \text{exchange right}
\end{array}$$

(vi) $R_1 =$ *contraction right*, $R_2 =$ *weakening right*.

We have the following tree:

$$\begin{array}{cl}
\Gamma \to \Delta, A, A & \\ \hline
\Gamma \to \Delta, A & \text{contraction right} \\ \hline
\Gamma \to \Delta, A, B & \text{weakening right}
\end{array}$$

The permutation is performed as follows:

$$\begin{array}{cl}
\Gamma \to \Delta, A, A & \\ \hline
\Gamma \to \Delta, A, A, B & \text{weakening right} \\ \hline
\text{exchanges} & \\ \hline
\Gamma \to \Delta, B, A, A & \\ \hline
\Gamma \to \Delta, B, A & \text{contraction right} \\ \hline
\Gamma \to \Delta, A, B & \text{exchange}
\end{array}$$

(vii) $R_1 = \forall : right$, $R_2 =$ *atomic cut*.

We have the following tree:

$$\dfrac{\dfrac{\dfrac{\Gamma' \to \Delta', A[y/x]}{\Gamma' \to \Delta', \forall x A}\ \forall : right}{\Gamma \to \Delta_1, \forall x A, \Delta_2, B}\ \text{structural rules } ST \qquad B, \Lambda \to \Theta}{\Gamma, \Lambda \to \Delta_1, \forall x A, \Delta_2, \Theta}\ \text{atomic cut } (B)$$

Since B is atomic, it is different from $\forall x A$.

The permutation is performed as follows:

$$
\begin{array}{cl}
\Gamma' \to \Delta', A[y/x] & \\ \cline{1-1}
\text{weakening and exchanges} & \\ \cline{1-1}
\Gamma' \to A[y/x], \Delta', \forall xA & \\ \cline{1-1}
\text{structural rules ST} & \\ \cline{1-1}
\Gamma \to A[y/x], \Delta_1, \forall xA, \Delta_2, B \qquad B, \Lambda \to \Theta & \text{atomic cut } (B) \\ \cline{1-1}
\Gamma, \Lambda \to A[y/x], \Delta_1, \forall xA, \Delta_2, \Theta & \\ \cline{1-1}
\text{exchanges} & \\ \cline{1-1}
\Gamma, \Lambda \to \Delta_1, \forall xA, \Delta_2, \Theta, A[y/x] & \forall : right \\ \cline{1-1}
\Gamma, \Lambda \to \Delta_1, \forall xA, \Delta_2, \Theta, \forall xA & \\ \cline{1-1}
\text{exchanges, contraction, and exchanges} & \\ \cline{1-1}
\Gamma, \Lambda \to \Delta_1, \forall xA, \Delta_2, \Theta &
\end{array}
$$

□

Remark: It is easily shown that the rules $\forall : right$ and $\exists : left$ permute with each other, permute with the rules $\forall : left$ and $\exists : right$, and the rules $\forall : left$ and $\exists : right$ permute with each other, provided that the main formula of the the upper inference is not equal to the side formula of the lower inference.

7.3.3 Gentzen's Sharpened Hauptsatz

We now prove Gentzen's sharpened Hauptsatz.

Theorem 7.3.1 (Gentzen's sharpened Hauptsatz) Given a sequent $\Gamma \to \Delta$ containing only prenex formulae, if $\Gamma \to \Delta$ is provable in LK (or LK_e), then there is a cut-free, pure-variable proof in LK (proof without essential cuts in LK_e) that contains a sequent $\Gamma' \to \Delta'$ (called the *midsequent*), which has the following properties:

(1) Every formula in $\Gamma' \to \Delta'$ is quantifier free.

(2) Every inference rule above $\Gamma' \to \Delta'$ is either structural, logical (not a quantifier rule), or an inessential cut.

(3) Every inference rule below $\Gamma' \to \Delta'$ is either a quantifier rule, or a contraction, or an exchange rule (but not a weakening).

Thus, the midsequent splits the proof tree into an upper part that contains the propositional inferences, and a lower part that contains the quantifier inferences (and no weakening rules).

Proof: From theorem 6.3.1, lemma 7.3.1, and lemma 7.3.2, we can assume that $\Gamma \rightarrow \Delta$ has a cut-free proof T in LK (without essential cuts in LK_e) that is a pure-variable proof, and such that the axioms are of the form either $A \rightarrow A$ with A atomic, or an equality axiom (it is actually sufficient to assume that A is quantifier free).

For every quantifier inference R in T, let $m(R)$ be the number of propositional inferences and $n(R)$ the number of weakening inferences on the path from R to the root (we shall say that such inferences are *below* R). For a proof T, $m(T)$ is the sum of all $m(R)$ and $n(T)$ the sum of all $n(R)$, for all quantifier inferences R in T. We shall show by induction on $(m(T), n(T))$ (using the lexicographic ordering defined in Subsection 2.1.10) that the theorem holds.

(i) $m(T) = 0$, $n(T) = 0$. Then, all propositional inferences and all weakening inferences are above all quantifier inferences. Let S_0 be the premise of the highest quantifier inference (that is, such that no descendant of this inference is a quantifier inference). If S_0 only contains quantifier-free formulae, S_0 is the midsequent and we are done. Otherwise, since the proof is cut free (without essential cuts in LK_e) and since the axioms only contain quantifier-free formulae (actually, atomic formulae), prenex quantified formulae in S_0 must have been introduced by weakening rules. If T_0 is the portion of the proof with conclusion S_0, by eliminating the weakening inferences from T_0, we obtain a (quantifier-free) proof T_1 of the sequent S_1 obtained by deleting all quantifed formulae from S_0. For every quantified prenex formula $A = Q_1x_1...Q_nx_nC$ occurring in S_0, let $B = C[z_1/x_1, ..., z_n/x_n]$ be the quantifier-free formula obtained from A by substituting new variables $z_1, ..., z_n$ for $x_1, ..., x_n$ in C (we are assuming that these new variables are distinct from all the variables occurring in T_1 and distinct from the variables used for other such occurrences of quantified formulae occurring in S_0). Let $\Gamma' \rightarrow \Delta'$ be the sequent obtained from S_0 by replacing every occurrence of a quantified formula A by the corresponding quantifier-free formula B, as above. It is clear that $\Gamma' \rightarrow \Delta'$ can be obtained from S_1 by weakening inferences, and that S_0 can be obtained from $\Gamma' \rightarrow \Delta'$ by quantifier inferences. But then, $\Gamma' \rightarrow \Delta'$ is the midsequent of the proof obtained from T_1 and the intermediate inferences from S_1 to $\Gamma' \rightarrow \Delta'$ and from $\Gamma' \rightarrow \Delta'$ to S_0. Since the weakening inferences only occur above $\Gamma' \rightarrow \Delta'$, all the conditions of the theorem are satisfied.

(ii) *Case* 1: $m(T) > 0$. In this case, there is an occurrence R_1 of a quantifier inference above an occurrence R_2 of a propositional inference, and intermediate inferences (if any) are structural. Since all formulae in the root sequent are prenex, the side formula (or formulae) of R_2 are quantifier free. Hence, lemma 7.3.3(i) applies, R_1 can be permuted with R_2, yielding a new proof T' such that $m(T') = m(T) - 1$. We conclude by applying the induction hypothesis to T'.

(ii) *Case* 2: $m(T) = 0$, $n(T) > 0$. In this case, all propositional inferences are above all quantifier inferences, but there is a quantifier inference R_1 above a weakening inference R_2, and intermediate inferences (if any) are exchange or contraction rules. Using lemma 7.3.3(ii), R_2 can be moved up until R_1 and R_2 are permuted, yielding a new proof T' such that $m(T') = 0$ and $n(T') = n(T) - 1$. We conclude by applying the induction hypothesis to T'. □

Remark: In the case where $m(T) > 0$, even though $m(T') = m(T) - 1$, $n(T')$ might be increased because the transformation may introduce extra weakenings. However, $n(T)$ is eventually reduced to 0, when $m(T)$ is reduced to 0. Also, note that the proof provides an algorithm for converting a pure-variable cut-free proof (proof without essential cuts in LK_e) into a proof having the properties stated in the theorem.

An example of the transformations of lemma 7.3.3 also showing the values of $m(T)$ and $n(T)$ is given below. In order to shorten proofs, some contractions and exchanges will not be shown explicitly.

EXAMPLE 7.3.3

Consider the following proof tree T in which $m(T) = 2$ and $n(T) = 0$:

$$\dfrac{\dfrac{\dfrac{P(f(a)) \to P(f(a))}{P(f(a)) \to P(f(a)), \exists y R(y)}}{P(f(a)) \to \exists x P(x), \exists y R(y)} \qquad \dfrac{\dfrac{R(g(a)) \to R(g(a))}{R(g(a)) \to R(g(a)), \exists x P(x)}}{R(g(a)) \to \exists x P(x), \exists y R(y)}}{P(f(a)) \lor R(g(a)) \to \exists x P(x), \exists y R(y)} \quad \lor : left$$

We first exchange the $\exists : right$ inference in the right subtree with $\lor : left$ obtaining the following tree

$$\dfrac{\dfrac{\dfrac{T_1 \qquad\qquad T_2}{P(f(a)) \lor R(g(a)) \to \exists x P(x), \exists y R(y), R(g(a))}}{P(f(a)) \lor R(g(a)) \to \exists x P(x), \exists y R(y), \exists y R(y)} \quad \exists : right}{P(f(a)) \lor R(g(a)) \to \exists x P(x), \exists y R(y)}$$

where T_1 is the tree

$$\dfrac{\dfrac{\dfrac{P(f(a)) \to P(f(a))}{P(f(a)) \to P(f(a)), \exists y R(y)}}{P(f(a)) \to \exists x P(x), \exists y R(y)} \quad \exists : right}{P(f(a)) \to \exists x P(x), \exists y R(y), R(g(a))} \quad \text{weakening}$$

and T_2 is the tree

$$\begin{array}{c}
R(g(a)) \to R(g(a)) \\ \hline
R(g(a)) \to R(g(a)), \exists x P(x) \\ \hline
\text{weakening, exchanges} \\ \hline
R(g(a)) \to \exists x P(x), \exists y R(y), R(g(a))
\end{array}$$

We now have $n(T) = 1$ and $m(T) = 1$, since a weakening was introduced below the $\exists : right$ inference in the tree T_1. We finally exchange the $\exists : right$ inference in T_1 with $\lor : left$. We obtain the following tree

$$\begin{array}{cl}
T_1 \qquad\qquad T_2 & \\ \hline
P(f(a)) \lor R(g(a)) \to \exists x P(x), \exists y R(y), R(g(a)), P(f(a)) & \lor : left \\ \hline
P(f(a)) \lor R(g(a)) \to \exists x P(x), \exists y R(y), R(g(a)), \exists x P(x) & \exists : right \\ \hline
P(f(a)) \lor R(g(a)) \to \exists x P(x), \exists y R(y), R(g(a)) & \\ \hline
P(f(a)) \lor R(g(a)) \to \exists x P(x), \exists y R(y), \exists y R(y) & \exists : right \\ \hline
P(f(a)) \lor R(g(a)) \to \exists x P(x), \exists y R(y) &
\end{array}$$

where T_1 is the tree

$$\begin{array}{c}
P(f(a)) \to P(f(a)) \\ \hline
P(f(a)) \to P(f(a)), \exists x P(x) \\ \hline
P(f(a)) \to P(f(a)), \exists x P(x), \exists y R(y) \\ \hline
P(f(a)) \to \exists x P(x), \exists y R(y), R(g(a)), P(f(a))
\end{array}$$

and T_2 is the tree

$$\begin{array}{c}
R(g(a)) \to R(g(a)) \\ \hline
R(g(a)) \to R(g(a))), \exists x P(x) \\ \hline
R(g(a)) \to R(g(a)), \exists x P(x), \exists y R(y) \\ \hline
R(g(a)) \to \exists x P(x), \exists y R(y), R(g(a)), P(f(a))
\end{array}$$

We now have $n(T) = 0$ and $m(T) = 0$ and the proof is in normal form.

Another example of a proof in the normal form given by theorem 7.3.1 is shown below.

EXAMPLE 7.3.4

Consider the sequent

$$\rightarrow \exists x\forall y\neg P(x,y), \exists y_1\forall z\neg Q(y_1,z), \forall x_1\exists y_2\exists z_1(P(x_1,y_2)\land Q(y_2,z_1))$$

whose validity is equivalent to the validity of the formula

$$A = \exists x\forall y\neg P(x,y) \lor \exists y_1\forall z\neg Q(y_1,z) \lor \forall x_1\exists y_2\exists z_1(P(x_1,y_2)\land Q(y_2,z_1))$$

The following proof satisfies the conditions of theorem 7.3.1:

$$\begin{array}{c}
\dfrac{P(x_1,y) \rightarrow P(x_1,y)}{P(x_1,y),Q(y,z) \rightarrow P(x_1,y)} \qquad \dfrac{Q(y,z) \rightarrow Q(y,z)}{P(x_1,y),Q(y,z) \rightarrow Q(y,z)} \\ \hline
P(x_1,y),Q(y,z) \rightarrow P(x_1,y)\land Q(y,z) \\ \hline
P(x_1,y) \rightarrow \neg Q(y,z), P(x_1,y)\land Q(y,z) \\ \hline
\rightarrow \neg P(x_1,y), \neg Q(y,z), P(x_1,y)\land Q(y,z) \\ \hline
\rightarrow \neg P(x_1,y), \neg Q(y,z), \exists z_1(P(x_1,y)\land Q(y,z_1)) \\ \hline
\rightarrow \neg P(x_1,y), \neg Q(y,z), \exists y_2\exists z_1(P(x_1,y_2)\land Q(y_2,z_1)) \\ \hline
\rightarrow \neg P(x_1,y), \forall z\neg Q(y,z), \exists y_2\exists z_1(P(x_1,y_2)\land Q(y_2,z_1)) \\ \hline
\rightarrow \neg P(x_1,y), \exists y_1\forall z\neg Q(y_1,z), \exists y_2\exists z_1(P(x_1,y_2)\land Q(y_2,z_1)) \\ \hline
\rightarrow \forall y\neg P(x_1,y), \exists y_1\forall z\neg Q(y_1,z), \exists y_2\exists z_1(P(x_1,y_2)\land Q(y_2,z_1)) \\ \hline
\rightarrow \exists x\forall y\neg P(x,y), \exists y_1\forall z\neg Q(y_1,z), \exists y_2\exists z_1(P(x_1,y_2)\land Q(y_2,z_1)) \\ \hline
\rightarrow \exists x\forall y\neg P(x,y), \exists y_1\forall z\neg Q(y_1,z), \forall x_1\exists y_2\exists z_1(P(x_1,y_2)\land Q(y_2,z_1))
\end{array}$$

The midsequent is $\rightarrow \neg P(x_1,y), \neg Q(y,z), P(x_1,y)\land Q(y,z)$.

Notice that the proof was constructed so that the $\forall : right$ rule is always applied as soon as possible. The reason for this is that we have the freedom of choosing the terms involved in applications of the $\exists : right$ rule (as long as they are free for the substitutions), whereas the $\forall : right$ rule requires that the substituted term be a new variable. The above strategy has been chosen to allow ourselves as much freedom as possible in the substitutions.

PROBLEMS

7.3.1. Prove lemma 7.3.2.

7.3.2. Finish the proof of the cases in lemma 7.3.3.

7.3.3. Prove the following facts stated as a remark at the end of the proof of lemma 7.3.3: The rules $\forall : right$ and $\exists : left$ permute with each other, permute with the rules $\forall : left$ and $\exists : right$, and the rules $\forall : left$ and $\exists : right$ permute with each other, provided that the main formula of the the upper inference is not equal to the side formula of the lower inference.

7.3.4. Give proofs in normal form for the prenex form of the following sequents:

$$(\exists x \forall y P(x,y) \land \forall y \exists x P(y,x)) \rightarrow$$

$$(\neg(\forall x P(x) \lor \exists y \neg Q(y)) \lor (\forall z P(z) \lor \exists w \neg Q(w))) \rightarrow$$

$$(\neg \forall x (P(x) \lor \exists y \neg Q(y)) \lor (\forall z P(z) \lor \exists w \neg Q(w))) \rightarrow$$

7.3.5. Write a computer program converting a proof into a proof in normal form as described in theorem 7.3.1.

7.3.6. Design a *search* procedure for prenex sequents using the suggestions made at the end of example 7.3.4.

7.4 The Sharpened Hauptsatz for Sequents in NNF

In this section, it is shown that the quantifier rules of the system $G1^{nnf}$ presented in Section 6.4 can be extended to apply to certain subformulae, so that the permutation lemma holds. As a consequence, the sharpened Hauptsatz also holds for such a system. In this section, all formulae are rectified.

7.4.1 The System $G2^{nnf}$

First, we show why we chose to define such rules for formulae in NNF and not for arbitrary formulae.

EXAMPLE 7.4.1

Let

$$A = P(a) \lor \neg(Q(b) \land \forall x Q(x)) \rightarrow .$$

Since A is logically equivalent to $P(a) \lor (\neg Q(b) \lor \exists x \neg Q(x))$, the correct rule to apply to the subformula $\forall x Q(x)$ is actually the $\exists : left$ rule!

Hence, the choice of the rule applicable to a quantified subformula B depends on the parity of the number of negative subformulae that have the formula B has a subformula. This can be handled, but makes matters more complicated. However, this problem does not arise with formulae in NNF since only atoms can be negated.

Since there is no loss of generality in considering formulae in NNF and the quantifier rules for subformulae of formulae in NNF are simpler than for arbitrary formulae, the reason for considering formulae in NNF is clear.

The extended quantifier rules apply to maximal quantified subformulae of a formula. This concept is defined rigorously as follows.

Definition 7.4.1 Given a formula A in NNF, the set $QF(A)$ of *maximal quantified subformulae* of A is defined inductively as follows:

(i) If A is a literal, that is either an atomic formula B or the negation $\neg B$ of an atomic formula, then $QF(A) = \emptyset$;

(ii) If A is of the form $(B \lor C)$, then $QF(A) = QF(B) \cup QF(C)$;

(iii) If A is of the form $(B \land C)$, then $QF(A) = QF(B) \cup QF(C)$.

(iv) If A is of the form $\forall x B$, then $QF(A) = \{A\}$.

(v) If A is of the form $\exists x B$, then $QF(A) = \{A\}$.

EXAMPLE 7.4.2

Let

$$A = \forall u P(u) \lor (Q(b) \land \forall x(\neg P(x) \lor \forall y R(x, y))) \lor \exists z P(z).$$

Then

$$QF(A) = \{\forall u P(u), \forall x(\neg P(x) \lor \forall y R(x, y)), \exists z P(z)\}.$$

However, $\forall y R(x, y)$ is not a maximal quantified subformula since it is a subformula of $\forall x(\neg P(x) \lor \forall y R(x, y))$.

Since we are dealing with rectified formulae, all quantified subformulae differ by at least the outermost quantified variable, and therefore, they are all distinct. This allows us to adopt a simple notation for the result of substituting a formula for a quantified subformula.

Definition 7.4.2 Given a formula A in NNF whose set $QF(A)$ is nonempty, for any subformula B in $QF(A)$, for any formula C, the formula $A[C/B]$ is defined inductively as follows:

(i) If $A = B$, then $A[C/B] = C$;

(ii) If A is of the form $(A_1 \lor A_2)$, B belongs either to A_1 or to A_2 but not both (since subformulae in $QF(A)$ are distinct), and assume that B belongs to A_1, the other case being similar. Then, $A[C/B] = (A_1[C/B] \lor A_2)$;

(iii) If A is of the form $(A_1 \land A_2)$, as in (ii), assume that B belongs to A_1. Then, $A[C/B] = (A_1[C/B] \land A_2)$.

EXAMPLE 7.4.3

Let

$$A = \forall u P(u) \lor (Q(b) \land \forall x(\neg P(x) \lor \forall y R(x,y))),$$

$$B = \forall x(\neg P(x) \lor \forall y R(x,y)), \quad \text{and}$$

$$C = (\neg P(f(a)) \lor \forall y R(f(a),y)).$$

Then,

$$A[C/B] = \forall u P(u) \lor (Q(b) \land (\neg P(f(a)) \lor \forall y R(f(a),y))).$$

We now define the system $G2^{nnf}$.

Definition 7.4.3 (Gentzen system $G2^{nnf}$) The Gentzen system $G2^{nnf}$ is the system obtained from $G1^{nnf}$ by adding the weakening rules, by replacing the quantifier rules by the *quantifier rules for subformulae* listed below, and using axioms of the form $A \to A$, $\to A, \neg A$, $A, \neg A \to$, and $\neg A \to \neg A$, where A is atomic. Let A be any formula in NNF containing quantifiers, let C be any subformula in $QF(A)$ of the form $\forall xB$, and D any subformula in $QF(A)$ of the form $\exists xB$. Let t be any term free for x in B.

$$\frac{A[B[t/x]/C], \Gamma \to \Delta}{A, \Gamma \to \Delta} \quad (\forall : left)$$

The formula $A[B[t/x]/C]$ is the *side formula* of the inference. Note that for $A = C = \forall xB$, this rule is identical to the $\forall : left$ rule of $G1^{nnf}$.

$$\frac{\Gamma \to \Delta, A[B[y/x]/C]}{\Gamma \to \Delta, A} \quad (\forall : right)$$

where y is not free in the lower sequent.

The formula $A[B[y/x]/C]$ is the *side formula* of the inference. For $A = C = \forall xB$, the rule is identical to the $\forall : right$ rule of $G1^{nnf}$.

$$\frac{A[B[y/x]/D], \Gamma \to \Delta}{A, \Gamma \to \Delta} \quad (\exists : left)$$

where y is not free in the lower sequent.

The formula $A[B[y/x]/D]$ is the *side formula* of the inference. For $A = D = \exists xB$, the rule is identical to the $\exists : left$ rule of $G1^{nnf}$.

$$\frac{\Gamma \to \Delta, A[B[t/x]/D]}{\Gamma \to \Delta, A} \quad (\exists : right)$$

The formula $A[B[t/x]/D]$ is the *side formula* of the inference. For $A = D = \exists xB$, the rule is identical to the $\exists : right$ rule of $G1^{nnf}$.

7.4.2 Soundness of the System $G2^{nnf}$

The soundness of the quantifier rules of $G2^{nnf}$ of definition 7.4.3 is shown in the following lemma.

Lemma 7.4.1 The system $G2^{nnf}$ is sound.

Proof: We treat the case of the rules $\forall : left$ and $\forall : right$, the case of the rules $\exists : left$ and $\exists : right$ beeing similar. We need to prove that if the premise of the rule is valid, then the conclusion is valid. Equivalently, this can shown by proving that if the conclusion if falsifiable, then the premise is falsifiable.

Case 1: $\forall : left$.

To show that if the conclusion is falsifiable then the premise is falsifiable amounts to proving the following: For every structure **A**, for every assignment s:

(i) $A = C$: If $\forall xB$ is satisfied in **A** by s, then $B[t/x]$ is satisfied in **A** by s. This has been shown in lemma 5.4.2.

(ii) $A \neq C$: If A is satisfied in **A** by s, then $A[B[t/x]/C]$ is satisfied in **A** by s. We proceed by induction on A.

If A is of the form $(A_1 \vee A_2)$, we can assume without loss of generality that C is in $QF(A_1)$. But $(A_1 \vee A_2)$ is satisfied in **A** by s iff either A_1 is satisfied in **A** by s or A_2 is satisfied in **A** by s. Since

$$A[B[t/x]/C] = (A_1[B[t/x]/C] \vee A_2),$$

in the second case, $(A_1[B[t/x]/C] \vee A_2)$ is also satisfied in **A** by s. In the first case, by the induction hypothesis, since A_1 is satisfied in **A** by s, $A_1[B[t/x]/C]$ is also satisfied in **A** by s, and so $(A_1[B[t/x]/C] \vee A_2)$ is satisfied in **A** by s, as desired.

If A is of the form $(A_1 \wedge A_2)$, assume as in the previous case that C occurs in A_1. If A is satisfied in **A** by s, then both A_1 and A_2 are satisfied in **A** by the same assignment s. By the induction hypothesis, $A_1[B[t/x]/C]$ is satisfied in **A** by s, and so $A[B[t/x]/C]$ is satisfied in **A** by s.

If A is equal to C, then we have to show that if $\forall xB$ is satisfied in **A** by s, then $B[t/x]$ is satisfied in **A** by s, but this reduces to Case 1(i).

Case 2: $\forall : right$.

This time, showing that if the conclusion is falsifiable then the premise is falsifiable amounts to proving the following: For every structure **A**, if A is falsifiable in $\mathbf{A}$ by s, then $A[B[y/x]/D]$ is falsifiable in $\mathbf{A}$ by an assignment s' of the form $s[y := a]$, where a is some element in the domain of $\mathbf{A}$, and y is not free in A, Γ and Δ. There are two cases:

(i) $A = D$; If $\forall xB$ is falsifiable in $\mathbf{A}$ by s, there is some element a in the domain of $\mathbf{A}$ such that $B[\mathbf{a}/x]$ is falsified in $\mathbf{A}$, and since y does not occur free in the conclusion of the inference, by lemma 5.4.2, $B[y/x]$ is falsifiable in $\mathbf{A}$ by $s[y := a]$.

(ii) $A \neq D$; If A is falsifiable in $\mathbf{A}$ by s, then $A[B[y/x]/D]$ is falsifiable in $\mathbf{A}$ by an assignment s' of the form $s[y := a]$, where y does not occur free in A, Γ, Δ.

We prove by induction on formulae that for *every* subformula A' of A containing some formula in $QF(A)$ as a subformula, if A' is falsified in $\mathbf{A}$ by s, then $A'[B[y/x]/D]$ is falsified in $\mathbf{A}$ by an assignment s' of the form $s[y := a]$, where y is not free in A, Γ and Δ. Note that in the induction hypothesis, it is necessary to state that y is not free in A, and not just A'.

If A' is of the form $(A_1 \lor A_2)$, we can assume without loss of generality that D is in $QF(A_1)$. But $(A_1 \lor A_2)$ is falsified in $\mathbf{A}$ by s iff A_1 is falsified in $\mathbf{A}$ by s and A_2 is falsified in $\mathbf{A}$ by s. Since

$$A'[B[y/x]/D] = (A_1[B[y/x]/D] \lor A_2),$$

by the induction hypothesis $A_1[B[y/x]/D]$ is falsified in $\mathbf{A}$ by some assignment of the form $s[y := a]$ where y is not not free in A, Γ and Δ. But since A_2 is a subformula of A' and thus of A, y is not free in A_2 and by lemma 5.3.3, A_2 is also falsified in $\mathbf{A}$ by $s[y := a]$. Then $A'[B[y/x]/D]$ is falsified in $\mathbf{A}$ by the assignment $s[y := a]$.

If A' is of the form $(A_1 \land A_2)$, assume as in the previous case that D occurs in A_1. If A' is falsified in $\mathbf{A}$ by s, then either A_1 is falsified in $\mathbf{A}$ by s or A_2 is falsified in $\mathbf{A}$ by s. In the second case $(A_1[B[y/x]/D] \land A_2)$ is falsified in $\mathbf{A}$ by s. In the first case, since A_1 is falsified in $\mathbf{A}$ by s, by the induction hypothesis $A_1[B[y/x]/D]$ is falsified in $\mathbf{A}$ by some assignment s' of the form $s[y := a]$. Then $(A_1[B[y/x]/D] \land A_2)$ is falsified in $\mathbf{A}$ by $s[y := a]$, as desired.

If A' is equal to D, then we are back to case 1(i). □

It should be emphasized that the condition that y is not free in A, Γ and Δ, and not just D, Γ, Δ plays a crucial role in the proof. Note that the system $G1^{nnf}$ is a subsystem of $G2^{nnf}$.

We now prove a version of lemma 7.3.3 and a normal form theorem analogous to theorem 7.3.1 for the system $G2^{nnf}$.

7.4.3 A Gentzen-like Sharpened Hauptsatz for $G2^{nnf}$

The definition of an inference R_1 permuting with an inference R_2 given in definition 7.3.2 extends immediately to the new quantifier rules, except that the section ST of structural rules only contains weakenings. It is also easy to see that lemmas 7.3.1 and 7.3.2 extend to the system $G2^{nnf}$. This is left as an exercise to the reader.

Consider a quantifier rule of $G2^{nnf}$, say $\forall : left$:

$$\frac{A[B[t/x]/C], \Gamma \rightarrow \Delta}{A, \Gamma \rightarrow \Delta}$$

If A belongs to Γ, letting $\Lambda = \Gamma - \{A\}$, the rule can be written as:

$$\frac{A, A[B[t/x]/C], \Lambda \rightarrow \Delta}{A, \Lambda \rightarrow \Delta}$$

where A does *not* belong to Λ. In this version, we say that the rule is the $\forall$: *left rule with contraction*. The first version of the rule in which either $A \in \Gamma$ or A does not belong to the upper sequent is called the *rule without contraction*. The definition extends to the other quantifier rules.

Lemma 7.4.2 In every pure-variable, cut-free proof T in $G2^{nnf}$, the following holds: Every quantifier rule R_1 of $G2^{nnf}$ can be permuted with a propositional rule R_2 or a weakening.

Proof: The proof proceeds by cases. First, we prove that every quantifier rule of $G1^{nnf}$ without contraction can be permuted with a propositional rule R_2 or a weakening. It is not difficult to see that the cases treated in lemma 7.3.3 are still valid with the rules of $G1^{nnf}$. This is because sequents are pairs of sets, and contraction rules can be simulated. For example to perform the permutation involving $R_1 = \forall : right$ and $R_2 = \wedge : right$, the following deduction is used:

$$\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{\Gamma' \rightarrow \Delta', A[y/x]}{\Gamma' \rightarrow A[y/x], \Delta', \forall x A}}{\text{weakening rules}}}{\Gamma \rightarrow A[y/x], \Delta, \forall x A, \Lambda, B} \qquad \cfrac{\Gamma \rightarrow \Delta, \forall x A, \Lambda, C}{\Gamma \rightarrow A[y/x], \Delta, \forall x A, \Lambda, C}\ \text{weakening}}{\Gamma \rightarrow \Delta, \forall x A, \Lambda, B \wedge C, A[y/x]}\ \wedge : right}{\Gamma \rightarrow \Delta, \forall x A, \Lambda, B \wedge C}\ \forall : right$$

In the lowest inference, we took advantage of the fact that

$$\Gamma \rightarrow \Delta, \forall x A, \Lambda, B \wedge C, \forall x A$$

denotes the same sequent as

$$\Gamma \to \Delta, \forall x A, \Lambda, B \wedge C,$$

since a sequent is a pair of sets. The details are left as an exercise. We still have to consider the cases of quantifier rules of $G1^{nnf}$ with contractions, quantifier rules of $G2^{nnf}$ applied to the side formula of a propositional rule, and permutation with weakenings. We treat $\wedge : left$ and $\wedge : right$, the other cases being similar.

Case 1: $\wedge : left$, $\forall : left$. There are two subcases:

1.1: $A \in \Gamma$ or A does not belong to the top sequent:

$$\dfrac{\dfrac{A[B[t/x]/C], \Gamma \to \Delta}{A, \Gamma \to \Delta}\ \forall : left}{A \wedge E, \Gamma \to \Delta}\ \wedge : left$$

The permutation is achieved as follows:

$$\dfrac{\dfrac{A[B[t/x]/C], \Gamma \to \Delta}{A[B[t/x]/C] \wedge E, \Gamma \to \Delta}\ \wedge : left}{A \wedge E, \Gamma \to \Delta}\ \forall : left$$

1.2: $A \notin \Gamma$ and A occurs in the top sequent.

$$\dfrac{\dfrac{A, A[B[t/x]/C], \Gamma \to \Delta}{A, \Gamma \to \Delta}\ \forall : left}{A \wedge E, \Gamma \to \Delta}\ \wedge : left$$

The permutation is achieved as follows:

$$\dfrac{\dfrac{\dfrac{A, A[B[t/x]/C], \Gamma \to \Delta}{A \wedge E, A[B[t/x]/C], \Gamma \to \Delta}\ \wedge : left}{A \wedge E, A[B[t/x]/C] \wedge E, \Gamma \to \Delta}\ \wedge : left}{A \wedge E, \Gamma \to \Delta}\ \forall : left$$

Case 2: $\wedge : left$, $\exists : left$.

2.1: $A \in \Gamma$ or A does not belong to the top sequent:

$$\dfrac{\dfrac{A[B[y/x]/D],\Gamma\rightarrow\Delta}{A,\Gamma\rightarrow\Delta}\;\exists:left}{A\land E,\Gamma\rightarrow\Delta}\;\land:left$$

The variable y is a new variable occurring only above $A[B[y/x]/D],\Gamma\rightarrow\Delta$. The permutation is achieved by performing the substitution of $B[y/x]$ for D in $A\land E$, yielding $A[B[y/x]/D]\land E$. The inference is legitimate since, y being a variable that only occurs above $A[B[y/x]/D],\Gamma\rightarrow\Delta$ in the previous proof tree, y does not occur free in A, E, Γ and Δ. The permutation is achieved as follows:

$$\dfrac{\dfrac{A[B[y/x]/D],\Gamma\rightarrow\Delta}{A[B[y/x]/D]\land E,\Gamma\rightarrow\Delta}\;\land:left}{A\land E,\Gamma\rightarrow\Delta}\;\exists:left$$

2.2: $A\notin\Gamma$ and A occurs in the top sequent.

$$\dfrac{\dfrac{A,A[B[y/x]/D],\Gamma\rightarrow\Delta}{A,\Gamma\rightarrow\Delta}\;\exists:left}{A\land E,\Gamma\rightarrow\Delta}\;\land:left$$

The permutation is performed as follows:

$$\dfrac{\dfrac{\dfrac{A,A[B[y/x]/D],\Gamma\rightarrow\Delta}{A\land E,A[B[y/x]/D],\Gamma\rightarrow\Delta}\;\land:left}{A\land E,A[B[y/x]/D]\land E,\Gamma\rightarrow\Delta}\;\land:left}{A\land E,\Gamma\rightarrow\Delta}\;\exists:left$$

Case 3: $\land:right$, $\forall:right$.

3.1: $A\in\Gamma$ or A does not occur in the top sequent:

$$\dfrac{\dfrac{\Gamma\rightarrow\Delta,A[B[y/x]/C]}{\Gamma\rightarrow\Delta,A}\;\forall:right\qquad\qquad\Gamma\rightarrow\Delta,E}{\Gamma\rightarrow\Delta,A\land E}\;\land:right$$

The variable y is new and occurs only above $\Gamma\rightarrow\Delta,A[B[y/x]/C]$. The permutation is achieved as follows:

$$\dfrac{\dfrac{\Gamma \to \Delta, A[B[y/x]/C] \qquad \Gamma \to \Delta, E}{\Gamma \to \Delta, A[B[y/x]/C] \wedge E}\ \wedge : \mathit{right}}{\Gamma \to \Delta, A \wedge E}\ \forall : \mathit{right}$$

The application of the $\forall$: *right* is legal because y does not occur free in $\Gamma \to \Delta, A \wedge E$, since in the previous proof it occurs only above $\Gamma \to \Delta, A[B[y/x]/C]$.

3.2: $A \notin \Gamma$ and A occurs in the top sequent.

$$\dfrac{\dfrac{\Gamma \to \Delta, A, A[B[y/x]/C]}{\Gamma \to \Delta, A}\ \forall : \mathit{right} \qquad \Gamma \to \Delta, E}{\Gamma \to \Delta, A \wedge E}\ \wedge : \mathit{right}$$

The permutation is performed as follows:

$$\dfrac{\dfrac{\dfrac{\Gamma \to \Delta, A, A[B[y/x]/C] \quad \dfrac{\Gamma \to \Delta, E}{\Gamma \to \Delta, A[B[y/x]/C], E}}{\Gamma \to \Delta, A[B[y/x]/C], A \wedge E} \qquad \dfrac{\Gamma \to \Delta, E}{\Gamma \to \Delta, E, A \wedge E}}{\Gamma \to \Delta, A[B[y/x]/C] \wedge E, A \wedge E}}{\Gamma \to \Delta, A \wedge E}\ \forall : \mathit{right}$$

Case 4: $\wedge : \mathit{right}$, $\exists \mathit{right}$:

4.1: $A \in \Gamma$ or A does not occur in the top sequent:

$$\dfrac{\dfrac{\Gamma \to \Delta, A[B[t/x]/D]}{\Gamma \to \Delta, A}\ \exists : \mathit{right} \qquad \Gamma \to \Delta, E}{\Gamma \to \Delta, A \wedge E}\ \wedge : \mathit{right}$$

The permutation is achieved as follows:

$$\dfrac{\dfrac{\Gamma \to \Delta, A[B[t/x]/D] \qquad \Gamma \to \Delta, E}{\Gamma \to \Delta, A[B[t/x]/D] \wedge E}\ \wedge : \mathit{right}}{\Gamma \to \Delta, A \wedge E}\ \exists : \mathit{right}$$

4.2: $A \notin \Gamma$ and A occurs in the top sequent.

$$\dfrac{\dfrac{\Gamma \to \Delta, A, A[B[t/x]/D]}{\Gamma \to \Delta, A}\ \exists : \mathit{right} \qquad \Gamma \to \Delta, E}{\Gamma \to \Delta, A \wedge E}\ \wedge : \mathit{right}$$

The permutation is achieved as follows:

$$\dfrac{\dfrac{\dfrac{\Gamma \to \Delta, A, A[B[t/x]/D] \quad \dfrac{\Gamma \to \Delta, E}{\Gamma \to \Delta, A[B[t/x]/D], E}}{\Gamma \to \Delta, A[B[t/x]/D], A \wedge E} \quad \dfrac{\Gamma \to \Delta, E}{\Gamma \to \Delta, E, A \wedge E}}{\Gamma \to \Delta, A[B[t/x]/D] \wedge E, A \wedge E}}{\Gamma \to \Delta, A \wedge E}\ \exists : \mathit{right}$$

The other cases are similar and left as an exercise. □

We can now prove the following type of normal form theorem for the systems $G1^{nnf}$ and $G2^{nnf}$.

Theorem 7.4.1 (A sharpened Hauptsatz for $G1^{nnf}$ and $G2^{nnf}$) Given a sequent $\Gamma \to \Delta$ containing only formulae in NNF, if $\Gamma \to \Delta$ is provable in $G1^{nnf}$, then there is a cut-free, pure-variable proof in $G2^{nnf}$ that contains a sequent $\Gamma' \to \Delta'$ (called the *midsequent*), which has the following properties:

(1) Every formula in $\Gamma' \to \Delta'$ is quantifier free.

(2) Every inference rule above $\Gamma' \to \Delta'$ is either a weakening or propositional (not a quantifier rule).

(3) Every inference rule below $\Gamma' \to \Delta'$ is a quantifier rule.

Thus, in such a proof in normal form, the midsequent splits the proof tree into an upper part that contains the propositional inferences (and weakenings), and a lower part that contains the quantifier inferences.

Proof: First we apply theorem 6.4.1 to obtain a cut-free $G1^{nnf}$-proof. Next, we use induction as in theorem 7.3.1. However, there is a new difficulty: Exchanges involving contractions introduce two propositional rules and sometimes a weakening above a quantifier rule, and a simple induction on $m(T)$ does not work. This difficulty can be resolved as follows. Let us assign the letter a to a quantifier rule, and the letter b to a propositional rule or a weakening. Consider the set of all strings obtained by tracing the branches of the proof tree from the root, and concatenating the letters corresponding to the quantifier and other inferences in the proof tree. The fact that a propositional inference (or a weakening) occurs below a quantifier inference is indicated by an occurrence of the substring ba. Note that every string obtained is of the form

$$a^{n_0} b^{m_1} a^{n_1} b^{m_2} a^{n_2} \ldots b^{m_l} a^{n_l} b^{m_{l+1}}.$$

By inspection of the exchanges given by lemma 7.4.2, note that a string ba is replaced either by ab or abb or $abbb$. From this, it is easily seen that $b^m a$ is replaced by some string ab^k, where $m \leq k \leq 3m$. Hence, $b^m a^n$ is replaced by some string $a^n b^{mk}$, where $1 \leq k \leq 3^n$.

To permute quantifier rules and propositional rules, proceed as follows: Consider all lowest occurrences of inferences of the form $b^{m_1} a^{n_1}$, and perform the exchanges using lemma 7.4.2. After this step is completed, each path

$$a^{n_0} b^{m_1} a^{n_1} b^{m_2} a^{n_2} ... b^{m_l} a^{n_l} b^{m_{l+1}}$$

is now of the form

$$a^{n_0} a^{n_1} b^{p_1} b^{m_2} a^{n_2} ... b^{m_l} a^{n_l} b^{m_{l+1}}.$$

Since the number of blocks of the form $b^{m_i} a^{n_i}$ has decreased, if we repeat the above process, we will eventually obtain a proof in which all quantifier inferences are below all propositional inferences and weakenings. When such a proof is obtained, we pick the premise of the highest quantifier inference. If this sequent M only contains quantifier-free formulae, it is the midsequent. Otherwise, since the axioms only contain literals and no quantifier rule is applied in the part of the proof above the sequent M, all quantified subformulae in M are introduced by weakening. As in the proof of theorem 7.3.1, we can consider the part of the proof above M, and form another proof of the sequent M' obtained from M by deleting all quantified formulae. It is now possible (as in theorem 7.3.1) to form a quantifier-free sequent $\Gamma' \to \Delta'$ from M' which is also provable, from which M is provable by applying the quantifier rules of $G2^{nnf}$. □

Remarks: (i) The theorem deals with the two systems $G1^{nnf}$ and $G2^{nnf}$, whereas theorem 7.3.1 deals with the single system LK. Theorem 7.4.1 shows how to convert a $G1^{nnf}$-proof with cut into a cut-free normal $G2^{nnf}$-proof. This is easier to prove than the extended Hauptsatz for $G2^{nnf}$. Also, this is sufficient to derive part of Herbrand's theorem for $G1^{nnf}$.

(ii) The resulting proof may have a depth exponential in the size of the original proof.

As an illustation of theorem 7.4.1, consider the following example.

EXAMPLE 7.4.4

The following is a proof of the sequent

$$(P(a) \lor \forall x(Q(x) \lor \forall y R(y))) \to P(a), Q(a), R(f(a)) \land R(f(b))$$

which is not in normal form:

$$\dfrac{\dfrac{\dfrac{P(a) \to P(a)}{P(a) \to P(a), Q(a)}}{P(a) \to P(a), Q(a), R(f(a)) \land R(f(b))} \qquad T_1}{(P(a) \lor \forall x(Q(x) \lor \forall y R(y))) \to P(a), Q(a), R(f(a)) \land R(f(b))}$$

where T_1 is the tree

$$\dfrac{\dfrac{\dfrac{\dfrac{Q(a) \;\to\; Q(a)}{Q(a) \;\to\; Q(a), R(f(a)) \land R(f(b))} \qquad T_2}{Q(a) \lor \forall y R(y) \;\to\; Q(a), R(f(a)) \land R(f(b))}}{\forall x(Q(x) \lor \forall y R(y)) \;\to\; Q(a), R(f(a)) \land R(f(b))}}{\forall x(Q(x) \lor \forall y R(y)) \;\to\; P(a), Q(a), R(f(a)) \land R(f(b))}$$

and T_2 is the tree

$$\dfrac{\dfrac{\dfrac{R(f(a)) \;\to\; R(f(a))}{\forall y R(y) \;\to\; R(f(a))} \quad \dfrac{R(f(b)) \;\to\; R(f(b))}{\forall y R(y) \;\to\; R(f(b))}}{\forall y R(y) \;\to\; R(f(a)) \land R(f(b))}}{\forall y R(y) \;\to\; Q(a), R(f(a)) \land R(f(b))}$$

The proof below is in normal form:

$$\dfrac{\dfrac{\dfrac{\dfrac{\text{propositional part}}{P(a) \lor (Q(a) \lor R(f(a))), P(a) \lor (Q(a) \lor R(f(b))) \;\to\; \Delta}}{P(a) \lor (Q(a) \lor \forall y R(y)), P(a) \lor (Q(a) \lor R(f(a))) \;\to\; \Delta}}{P(a) \lor (Q(a) \lor \forall y R(y)) \;\to\; P(a), Q(a), R(f(a)) \land R(f(b))}}{(P(a) \lor \forall x(Q(x) \lor \forall y R(y))) \;\to\; P(a), Q(a), R(f(a)) \land R(f(b))}$$

where $\Delta = P(a), Q(a), R(f(a)) \land R(f(b))$. The midsequent is

$$\begin{aligned} &P(a) \lor (Q(a) \lor R(f(a))), P(a) \lor (Q(a) \lor R(f(b))) \to \\ &\qquad\qquad\qquad\qquad\qquad\qquad P(a), Q(a), R(f(a)) \land R(f(b)), \end{aligned}$$

which is equivalent to

$$\begin{aligned} &(P(a) \lor (Q(a)) \lor (R(f(a)) \land R(f(b))) \to \\ &\qquad\qquad\qquad\qquad\qquad\qquad P(a), Q(a), R(f(a)) \land R(f(b)). \end{aligned}$$

We now consider the case of languages with equality.

7.4.4 The Gentzen System $G2^{nnf}_{=}$

The system $G2^{nnf}_{=}$ has the following axioms and inference rules.

Definition 7.4.4 The system $G2^{nnf}_{=}$ is obtained from $G1^{nnf}_{=}$ by adding the quantifier rules of definition 7.4.3 and the weakening rules. It is easy to see that $G2^{nnf}_{=}$ is sound.

The following version of lemma 7.4.2 holds for $G2^{nnf}_{=}$.

Lemma 7.4.3 For every pure-variable proof T in $G2^{nnf}_{=}$ without essential cuts, the following holds: Every quantifier rule of $G2^{nnf}_{=}$ can be permuted with a propositional rule, an inessential cut, or a weakening.

Proof: The case not handled in lemma 7.4.2 is an exchange with an inessential cut. This is handled as in lemma 7.3.3. □

7.4.5 A Gentzen-like Sharpened Hauptsatz for $G2^{nnf}_{=}$

We also have the following sharpened Hauptsatz for $G2^{nnf}_{=}$.

Theorem 7.4.2 (A sharpened Hauptsatz for $G1^{nnf}_{=}$ and $G2^{nnf}_{=}$) Given a sequent $\Gamma \rightarrow \Delta$ containing only formulae in NNF, if $\Gamma \rightarrow \Delta$ is provable in $G1^{nnf}_{=}$, then there is a pure-variable proof in $G2^{nnf}_{=}$ without essential cuts that contains a sequent $\Gamma' \rightarrow \Delta'$ (called the *midsequent*), which has the following properties:

(1) Every formula in $\Gamma' \rightarrow \Delta'$ is quantifier free.

(2) Every inference rule above $\Gamma' \rightarrow \Delta'$ is either a weakening, a propositional rule, or an inessential cut (but not a quantifier rule).

(3) Every inference rule below $\Gamma' \rightarrow \Delta'$ is a quantifier rule.

Proof: The proof is essentially identical to that of theorem 7.4.1, but using lemma 7.4.3 instead of lemma 7.4.2. The details are left as an exercise. □

We shall see in Section 7.6 how theorems 7.4.1 and 7.4.2 can be used to yield Andrews's version of the Skolem-Herbrand-Gödel theorem (See Andrews, 1981), and also half of a version of Herbrand's original theorem.

PROBLEMS

7.4.1. Finish the proof of lemma 7.4.1.

7.4.2. Finish the proof of the cases left out in the proof of lemma 7.4.2

7.4.3. Prove that lemma 7.3.1 and 7.3.2 extend to the system $G2^{nnf}$.

7.4.4. Fill in the details in the proof of theorem 7.4.1.

7.4.5. Fill in the details in the proof of lemma 7.4.3.

7.4.6. Fill in the details in the proof of theorem 7.4.2.

7.4.7. Give a proof in normal form for the sequent

$$\forall x(P(x) \lor \forall y Q(y, f(x))) \land (\neg P(a) \land (\neg Q(a, f(a)) \lor \neg Q(b, f(a))) \to .$$

7.4.8. It is tempting to formulate the $\forall : left$ rule so that a formula of the form $\forall x A$ can be instantiated to the formula $A[t_1/x] \land ... \land A[t_k/x]$, for any k terms $t_1, ..., t_k$ free for x in A, and never apply contractions. However, this does not work. Indeed, the resulting system is not complete. Consider the following sequent provided by Dale Miller:

$$\forall x \exists y(\neg P(x) \land P(y)) \to .$$

The following is a proof of the above sequent using contractions:

$$\begin{array}{c}
P(u) \to \neg P(x), P(u)), P(v) \\
\hline
\neg P(x), P(u), \neg P(u), P(v) \to \\
\hline
\neg P(x), P(u), (\neg P(u) \land P(v)) \to \\
\hline
(\neg P(x) \land P(u)), (\neg P(u) \land P(v)) \to \\
\hline
(\neg P(x) \land P(u)), \exists y(\neg P(u) \land P(y)) \to \\
\hline
(\neg P(x) \land P(u)), \forall x \exists y(\neg P(x) \land P(y)) \to \\
\hline
\exists y(\neg P(x) \land P(y)), \forall x \exists y(\neg P(x) \land P(y)) \to \\
\hline
\forall x \exists y(\neg P(x) \land P(y)), \forall x \exists y(\neg P(x) \land P(y)) \to \\
\hline
\forall x \exists y(\neg P(x) \land P(y)) \to
\end{array}$$

Show that a derivation involving no contractions cannot lead to a proof tree, due to the eigenvariable restriction.

7.5 Herbrand's Theorem for Prenex Formulae

In this section, we shall derive a constructive version of Herbrand's theorem from Gentzen's Hauptsatz, using a method inspired by Kleene (Kleene, 1967).

In presenting Herbrand's theorem, it is convenient to assume that we are dealing with sentences.

7.5.1 Preliminaries

The following lemma shows that there is no loss of generality in doing so.

Lemma 7.5.1 Let A be a rectified formula, and let $FV(A) = \{y_1, ..., y_n\}$ be its set of free variables. The sequent $\rightarrow A$ is (LK or LK_e) provable if and only if the sequent $\rightarrow \forall y_1 ... \forall y_n A$ is provable.

Proof: We proceed by induction on the number of free variables. The induction step consists in showing that if A is a rectified formula and has a single free variable x, then $\rightarrow A$ is provable iff $\rightarrow \forall x A$ is provable.

We can appeal to the completeness theorem and show that A is valid iff $\forall x A$ is valid, which is straightforward. We can also give a the following proof using the pure-variable lemma 7.3.1 and lemma 6.5.1.

Assume that $\rightarrow A$ is provable. Since A is rectified, the variable x is not free in $\forall x A$, and the following inference is valid.

$$\frac{\rightarrow A[x/x]}{\rightarrow \forall x A}$$

By putting the proof of $A = A[x/x]$ on top of this inference, we have a proof for $\forall x A$.

Conversely, assume that $\forall x A$ is provable. There is a minor problem, which is that the eigenvariable z used in the lowest inference in the proof of $\forall x A$ is not necessarily x, even though x is not free in $\forall x A$. However, by lemma 7.3.1, there is a pure-variable proof of A in which all eigenvariables are new and distinct. Hence, the variable x does not occur in the part of the proof above $\rightarrow A[z/x]$. Using lemma 6.5.1, we can substitute x for all occurrences of z in this part of the proof, and we get a proof of A. □

We will also use the following fact which is easily shown: Given a sequent $A_1, ..., A_m \rightarrow B_1, ..., B_n$,

$$A_1, ..., A_m \rightarrow B_1, ..., B_n$$

is provable if and only if

$$\rightarrow \neg A_1 \lor ... \lor \neg A_m \lor B_1 \lor ... \lor B_n$$

is provable. Using this fact and lemma 7.5.1, a sequent consisting of formulae is provable if and only if a sentence is provable.

Hence, we will assume without loss of generality that the sequent Γ to be proved does not have sentences on the left of $\rightarrow$, that the sentences occurring on the righthand side of $\rightarrow$ are all distinct, and that they are rectified.

The main idea of Herbrand's theorem is to encode (with some inessential loss of information) the steps of the proof in which the $\forall :$ *rule* is applied. For this, some new function symbols called Herbrand functions or Skolem functions are introduced.

7.5.2 Skolem Function and Constant Symbols

Skolem symbols are defined as follows.

Definition 7.5.1 For every prenex formula $A = Q_n x_n ... Q_1 x_1 B$ occurring in a sequent $\to \Gamma$, for every occurrence Q_i of a universal quantifier in A then:

(i) If $n > 0$ and $m_i > 0$, where m_i is the number of existential quantifiers in the string $Q_n ... Q_{i+1}$, a new function symbol f_i^A having a number of arguments equal to m_i is created.

(ii) If $m_i = 0$ or $i = n$, the constant symbol f_i^A is created.

Such symbols are called *Skolem function symbols* and *Skolem constant symbols* (for short, *Skolem functions*).

The essence of Herbrand's theorem can be illustrated using example 7.3.4.

EXAMPLE 7.5.1

For the sequent of example 7.3.4, the unary function symbol f is associated with $\forall$ in $\exists x \forall y \neg P(x, y)$, the unary function symbol g is associated with $\forall$ in $\exists y_1 \forall z \neg Q(y_1, z)$, and the constant a with $\forall$ in $\forall x_1 \exists y_2 \exists z_1 (P(x_1, y_2) \land Q(y_2, z_1))$.

Now, we shall perform alterations to the proof in example 7.3.4. Moving up from the bottom sequent, instead of performing the $\forall : right$ rules, we are going to perform certain substitutions. For the instance of $\forall : right$ applied to

$$\forall x_1 \exists y_2 \exists z_1 (P(x_1, y_2) \land Q(y_2, z_1)),$$

substitute the Skolem constant a for all occurrences of x_1 in the proof; for the instance of $\forall : right$ applied to

$$\forall y \neg P(a, y),$$

substitute $f(a)$ for all occurrences of y in the proof; for the instance of $\forall : right$ applied to

$$\forall z \neg Q(f(a), z),$$

substitute $g(f(a))$ for all occurrences of z in the proof. Note that the resulting tree is no longer a legal proof tree because the applications of $\forall : right$ rules have been spoiled. However, the part of the proof from the new midsequent up is still a valid proof involving only propositional (and structural) rules (extending our original language with the symbols a, f, g).

The following definition will be useful for stating Herbrand's theorem.

Definition 7.5.2 Given a prenex sequent $\to \Gamma$, the *functional form* of $\to \Gamma$ is obtained as follows: For each prenex formula $A = Q_n x_n ... Q_1 x_1 B$ in

$\rightarrow \Gamma$, for each occurrence of a variable x_i bound by an occurrence Q_i of a universal quantifier, the term $f_i^A(y_1, ..., y_m)$ is substituted for x_i in B and $Q_i x_i$ is deleted from A, where f_i^A is the Skolem function symbol associated with Q_i, and $y_1, ..., y_m$ is the list of variables (in that order) bound by the existential quantifiers occurring in the string $Q_n ... Q_{i+1}$ (if $m = 0$ or $i = n$, f_i^A is a constant symbol).

EXAMPLE 7.5.2

Again, referring to example 7.3.4, the sequent

$$\rightarrow \exists x \neg P(x, f(x)), \exists y_1 \neg Q(y_1, g(y_1)), \exists y_2 \exists z_1 (P(a, y_2) \land Q(y_2, z_1))$$

is the functional form of our original sequent. Note that the provable sequent

$$\rightarrow \neg P(a, f(a)), \neg Q(f(a), g(f(a))), P(a, f(a)) \land Q(f(a), g(f(a)))$$

is obtained from the functional form of the original sequent by deleting quantifiers and substituting the terms a, $f(a)$, $f(a)$ and $g(f(a))$ for x, y_1, y_2, and z_1 respectively. Example 7.5.2 illustrates part of Herbrand's theorem.

Informal statement of Herbrand's theorem: If a prenex sequent $\rightarrow \Gamma$ is provable, then a disjunction of quantifier-free formulae constructible from Γ is provable. Furthermore, this disjunction of quantifier-free formulae consists of formulae obtained by substituting ground terms (built up from the original language extended with Skolem functions) for the bound variables of the sentences occurring in the functional form of the original sequent. The converse is also true, as we shall see shortly.

The key observation used in showing the converse of the above statement is to notice that the terms a, $f(a)$, $g(f(a))$ record the order in which the $\forall : right$ rules were applied in the original proof. The more nested the symbol, the earlier the rule was applied.

Warning: Many authors introduce Skolem function symbols to eliminate *existential quantifiers*, and not *universal quantifiers* as we do. This may seem confusing to readers who have seen this other definition of Skolem function symbols. However, there is nothing wrong with our approach. The reason the dual definition is also used (eliminating existential quantifiers using Skolem function symbols), as in the resolution method, is that the dual approach consists in showing that a formula A is valid by showing that $B = \neg A$ is *unsatisfiable*. This is equivalent to showing that the sequent $B \rightarrow$ is valid, or equivalently that $\rightarrow \neg B$ (that is $\rightarrow A$) is valid. Since in the dual approach, the *existential quantifiers* in B are Skolemized, in $\neg B = A$, the *universal quantifiers* are Skolemized. Since our approach consists in showing directly that A is valid, and not that $\neg A$ is unsatisfiable, we have defined Skolem function symbols for that purpose, and this is why they are used to eliminate *universal quantifiers*. What we have defined in definition 7.5.2 is often called in the literature the *validity functional form*, the dual form for eliminating existential quantifiers being called the *satisfiability functional form* (Herbrand, 1971). To avoid confusion, we shall call the first (the *validity functional form*) simply the *functional form*, and the second (the *satisfiability functional form*) the *Skolem normal form* (see definition 7.6.2).

7.5.3 Substitutions

Before stating and proving Herbrand's theorem, we need to define substitution functions.

Definition 7.5.3 A *substitution function* (for short, a *substitution*) is any function

$$\sigma : \mathbf{V} \rightarrow TERM_{\mathbf{L}}$$

assigning terms to the variables in $\mathbf{V}$. By theorem 2.4.1, there is a unique homomorphism

$$\widehat{\sigma} : TERM_{\mathbf{L}} \rightarrow TERM_{\mathbf{L}}$$

extending σ and defined recursively as follows:

For every variable $x \in \mathbf{V}$, $\widehat{\sigma}(x) = \sigma(x)$.

For every constant c, $\widehat{\sigma}(c) = \sigma(c)$.

For every term $ft_1...t_n \in TERM_{\mathbf{L}}$,

$$\widehat{\sigma}(ft_1...t_n) = f\widehat{\sigma}(t_1)...\widehat{\sigma}(t_n).$$

By abuse of language and notation, the function $\widehat{\sigma}$ will also be called a substitution, and will often be denoted by σ.

The subset X of $\mathbf{V}$ consisting of the variables such that $\sigma(x) \neq x$ is called the *support* of the substitution. In what follows, we will be dealing with substitutions of finite support. If a substitution σ has finite support $\{y_1, ..., y_n\}$ and $\sigma(y_i) = s_i$, for $i = 1, .., n$, for any term t, $\widehat{\sigma}(t)$ is also denoted by $t[s_1/y_1, ..., s_n/y_n]$.

Substitutions can be extended to formulae as follows. Let A be a formula, and let σ be a substitution with finite support $\{x_1, ..., x_n\}$. Assume that the variables in $\{x_1, ..., x_n\}$ are free in A and do not occur bound in A, and that each s_i ($s_i = \sigma(x_i)$) is free for x_i in A. The *substitution instance* $A[s_1/x_1, ..., s_n/x_n]$ is defined recursively as follows:

If A is an atomic formula of the form $Pt_1...t_m$, then

$$A[s_1/x_1, ..., s_n/x_n] = Pt_1[s_1/x_1, ..., s_n/x_n]...t_m[s_1/x_1, ..., s_n/x_n].$$

If A is an atomic formula of the form $(t_1 \doteq t_2)$, then

$$A[s_1/x_1, ..., s_n/x_n] = (t_1[s_1/x_1, ..., s_n/x_n] \doteq t_2[s_1/x_1, ..., s_n/x_n]).$$

If $A = \perp$, then

$$A[s_1/x_1, ..., s_n/x_n] = \perp .$$

If $A = \neg B$, then

$$A[s_1/x_1, ..., s_n/x_n] = \neg B[s_1/x_1, ..., s_n/x_n].$$

If $A = (B * C)$, where $* \in \{\vee, \wedge, \supset, \equiv\}$, then

$$A[s_1/x_1, ..., s_n/x_n] = (B[s_1/x_1, ..., s_n/x_n] * C[s_1/x_1, ..., s_n/x_n]).$$

If $A = \forall yB$ (where $y \notin \{x_1, ..., x_n\}$), then

$$A[s_1/x_1, ..., s_n/x_n] = \forall yB[s_1/x_1, ..., s_n/x_n].$$

If $A = \exists yB$ (where $y \notin \{x_1, ..., x_n\}$), then

$$A[s_1/x_1, ..., s_n/x_n] = \exists yB[s_1/x_1, ..., s_n/x_n].$$

Remark: $A[s_1/x_1][s_2/x_2]...[s_n/x_n]$, the result of substituting s_1 for x_1, ... ,s_n for x_n (as defined in definition 5.2.6) in that order, is usually different from $A[s_1/x_s, ..., s_n/x_n]$. This is because the terms s_1,...,s_n may contain some of the variables in $\{x_1, ..., x_n\}$. However, if none of the variables in the support of the substitution σ occurs in the terms s_1,...,s_n, it is easy to see that the order in which the substitutions are performed is irrelevant, and in this case,

$$A[s_1/x_1][s_2/x_2]...[s_n/x_n] = A[s_1/x_1, ..., s_n/x_n].$$

In particular, this is the case when σ is a *ground substitution*, that is, when the terms s_1,...,s_n do not contain variables.

Given a formula A and a substitution σ as above, the pair (A, σ) is called a *substitution pair*. The substitution instance defined by A and σ as above is also denoted by $\sigma(A)$. Notice that distinct substitution pairs can yield the same substitution instance.

A minor technicality has to be taken care of before proving Herbrand's theorem. If the first-order language does not have any constants, the theorem fails. For example, the sequent $\rightarrow \exists x \exists y (P(x) \vee \neg P(y))$ has a proof whose midsequent is $\rightarrow P(x) \vee \neg P(x)$, but if the language has no constants, we cannot find a ground substitution instance of $P(x) \vee \neg P(x)$ that is valid. To avoid this problem, we will assume that if any first-order language **L** does not have constants, the special constant # is added to it.

7.5.4 Herbrand's Theorem for Prenex Formulae

Before stating and proving Herbrand's theorem, recall that we can assume without loss of generality that the sequent $\rightarrow \Gamma$ to be proved consists of sentences, does not have sentences on the left of $\rightarrow$, that the sentences occurring on the righthand side of $\rightarrow$ are all distinct, and that distinct quantifiers bind occurrences of distinct variables.

Theorem 7.5.1 (Herbrand's theorem for prenex formulae) Given a sequent $\rightarrow \Gamma$ such that all sentences in Γ are prenex, $\rightarrow \Gamma$ is provable (in LK or LK_e) if and only if there is some finite sequence $< (B_1, \sigma_1), ..., (B_N, \sigma_N) >$ of substitution pairs such that the sequent $\rightarrow H$ consisting of the substitution instances $\sigma_1(B_1), ..., \sigma_N(B_N)$ is provable, where each $\sigma_i(B_i)$ is a quantifier-free substitution instance constructible from the sentences in Γ (H is called a *Herbrand disjunction*). Furthermore, each quantifier-free formula $\sigma_i(B_i)$ in the Herbrand disjunction H is a substitution instance $B_i[t_1/x_1, ..., t_k/x_k]$ of a quantifier-free formula B_i, matrix of some sentence $\exists x_1 ... \exists x_k B_i$ occurring in the functional form of $\rightarrow \Gamma$. The terms $t_1, ..., t_k$ are ground terms over the language consisting of the function and constant symbols in **L** occurring in $\rightarrow \Gamma$, and the Skolem function (and constant) symbols occurring in the functional form of $\rightarrow \Gamma$.

Proof: Using Gentzen's sharpened Hauptsatz (theorem 7.3.1), we can assume that we have a pure-variable cut-free proof in LK (proof without essential cuts in LK_e) with midsequent $\rightarrow \Gamma'$. We alter this proof in the following way. Starting with the bottom sequent and moving up, for every instance of the rule $\forall : \mathit{right}$ applied to a formula

$$\forall x_i Q_{i-1} x_{i-1} ... Q_1 x_1 B[s_1/y_1, ..., s_m/y_m]$$

which has been obtained from a prenex sentence

$$A = Q_n x_n ... Q_{i+1} x_{i+1} \forall x_i Q_{i-1} x_{i-1} ... Q_1 x_1 C$$

in Γ, substitute the term

$$f_i^A(s_1, ..., s_m)$$

for all occurrences of x_i in the proof (or Skolem constant f_i^A if $m = 0$), where f_i^A is the Skolem function symbol associated with x_i in A, $y_1, ..., y_m$ is the list of all the variables (all distinct by our hypothesis on proofs) bound by existential quantifiers in the string $Q_n x_n ... Q_{i+1} x_{i+1}$, and s_1,..., s_m is the list of terms that have been substituted for $y_1, ..., y_m$ in previous steps. Since the only inference rules used below the midsequent are quantifier, contraction or exchange rules, the resulting midsequent $\rightarrow H$ is indeed composed of substitutions instances of matrices of sentences occurring in the functional form of

$\rightarrow \Gamma$, and since the modified part of the proof above the midsequent is still a proof, the new midsequent $\rightarrow H$ is provable. If the midsequent contains variables, substitute any constant for all of these variables (by a previous assumption, the language has at least one constant, perhaps the special constant #). □

We now prove the converse of the theorem. We can assume without loss of generality that the disjuncts are distinct, since if they were not, we could suppress the duplications and still have a provable disjunct. Let $<(B_1, \sigma_1), ..., (B_N, \sigma_N)>$ be a sequence of distinct substitution pairs where each B_i is the matrix of the functional form of some sentence in $\rightarrow \Gamma$, let H be the corresponding sequence of substitution instances and assume that $\rightarrow H$ is provable. Notice that the conditions assumed before the statement of theorem 7.5.1 guarantee that every substitution pair (B, σ) corresponds to the unique pair $(Q_n x_n ... Q_1 x_1 C, \sigma)$, where $\exists y_1 ... \exists y_m B$ is the functional form of some sentence $Q_n x_n ... Q_1 x_1 C$ in $\rightarrow \Gamma$, and σ is a substitution with support $\{y_1, ..., y_m\}$.

Given the prenex sentence $A = Q_n x_n ... Q_1 x_1 C$ in $\rightarrow \Gamma$, its functional form is a sentence of the form

$$\exists y_1 ... \exists y_m C[r_1/z_1, ..., r_p/z_p],$$

where the union of $\{y_1, ..., y_m\}$ and $\{z_1, ..., z_p\}$ is $\{x_1, ..., x_n\}$, and $\{z_1, ..., z_p\}$ is the set of variables which are universally quantified in A. Each term r_i is rooted with the Skolem function symbol f_i^A. Consider the set HT composed of all terms of the form

$$r_i[s_1/y_1, ..., s_m/y_m]$$

occurring in the Herbrand disjunction $\rightarrow H$, where r_i is a Skolem term associated with an occurrence of a universal quantifier in some prenex sentence A in Γ, and s_1,...,s_m are the terms defining the substitution σ involved in the substitution pair $(C[r_1/z_1, ..., r_p/z_p], \sigma)$.

We define a partial order on the set HT as follows: For every term in HT of the form $f_i^A(t_1, ..., t_m)$, every subterm of $t_j \in HT$ (possibly t_j itself, $1 \leq i \leq m$) precedes $f_i^A(t_1, ..., t_m)$;

Every term in HT rooted with f_i^A precedes any term in HT rooted with f_j^A if $i > j$.

In order to reconstruct a proof, we will have to eliminate the Skolem symbols and perform $\forall : right$ rules with new variables. For this, we set up a bijection v between the set HT and a set of new variables not occurring in Γ. For every term t occurring in the Herbrand disjunction H (not only terms in HT), let $\bar{t}$ be the result of substituting the variable $v(s)$ for every maximal subterm $s \in HT$ of t (a maximal subterm in HT of t is a subterm of t in HT, which is not a proper subterm of any other subterm in HT of t). Let $<(B'_1, \sigma'_1), ..., (B'_N, \sigma'_N)>$ be the list of substitution pairs obtained from

$< (B_1, \sigma_1), ..., (B_N, \sigma_N) >$ by replacing each term $f_j^{A_i}(y_1, ..., y_m)$ occurring in B_i (where A_i is the sentence in Γ whose functional form is $\exists y_1 ... \exists y_m B_i$) by $v(\sigma_i(f_j^{A_i}(y_1, ..., y_m)))$, and each term s involved in defining the substitution σ_i by $\overline{s}$ (B'_i is actually a substitution instance of the matrix of a prenex formula in Γ, where the substitution is a bijection). Let $\rightarrow H'$ be the resulting sequent of substitution instances.

We are going to show that a deduction of $\rightarrow \Gamma$ can be constructed from $\rightarrow H'$ (really $< (B'_1, \sigma'_1), ..., (B'_N, \sigma'_N) >$) and the partially ordered set HT. First, notice that since $\rightarrow H$ is provable in the propositional part of LK (or LK_e), $\rightarrow H'$ is also provable since the above substitutions do not affect inferences used in the proof of $\rightarrow H$. The following definition will be needed.

Definition 7.5.4 Given a substitution pair (A, σ), where A is a prenex formula of the form $Q_n x_n ... Q_1 x_1 C$ $(n \geq 1)$ occurring in the sequent $\rightarrow \Gamma$ and σ is a substitution with support the subset of variables in $\{x_1, ..., x_n\}$ bound by existential quantifiers, the *σ-matrix of the functional form of A up to i* $(0 \leq i \leq n)$ is defined inductively as follows:

For $i = 0$, the σ-matrix of the functional form of A up to 0 is the substitution instance of C obtained by substituting the new variable $v(\sigma(f_j^A(y_1, ..., y_m)$ $))$ for x_j in C, for each occurrence of a variable x_j bound by a universal quantifier in A.

For $0 \leq i \leq n - 1$:

(i) If $Q_{i+1} = \exists$ and if the σ-matrix of the functional form of A up to i is B, the σ-matrix of the functional form of A up to $i + 1$ is $\exists x_{i+1} B$;

(ii) If $Q_{i+1} = \forall$, if the σ-matrix of the functional form of A up to i is $B[v(\sigma(f_{i+1}^A(y_1, ..., y_m)))/x_{i+1}]$, where the set of variables bound by existential quantifiers in $Q_n x_n ... Q_{i+2} x_{i+2}$ is $\{y_1, ..., y_m\}$, the σ-matrix of the functional form of A up to $i + 1$ is $\forall x_{i+1} B$.

Note that the σ-matrix of the functional form of A up to n is A itself.

Next, we define inductively a sequence Π of lists of substitution pairs $< (B_1, \sigma_1), ..., (B_p, \sigma_p) >$, such that the tree of sequents $\rightarrow \sigma_1(B_1), ..., \sigma_p(B_p)$ is a deduction of $\rightarrow \Gamma$ from $\rightarrow H'$. During the construction of Π, terms will be deleted from HT, eventually emptying it. Each B_j in a pair (B_j, σ_j) is the σ-matrix of the functional form up to some i of some sentence $A = Q_n x_n ... Q_i x_i ... Q_1 x_1 C$ in $\rightarrow \Gamma$, where σ is one of the original substitutions in the list $< (B'_1, \sigma'_1), ..., (B'_N, \sigma'_N) >$. Each σ_j is a substitution whose support is the set of variables in $\{x_n, ..., x_{i+1}\}$ which are bound by existential quantifiers in A.

The first element of Π is

$$< (B'_1, \sigma'_1), ..., (B'_N, \sigma'_N) > .$$

If the last element of the list Π constructed so far is $< (B_1, \sigma_1), ..., (B_p, \sigma_p) >$ and the corresponding sequent of substitution instances is $\rightarrow \Delta$, the next list of substitution pairs is determined as follows:

If Δ differs from Γ and no formula B_j for some pair (B_j, σ_j) in the list $< (B_1, \sigma_1), ..., (B_p, \sigma_p) >$ is the σ-matrix of the functional form up to i of some sentence

$$A = Q_n x_n ... \exists x_{i+1} Q_i x_i ... Q_1 x_1 C$$

occurring in $\rightarrow \Gamma$, select the leftmost formula

$$B_j = Q_i x_i ... Q_1 x_1 B[v(f^A_{i+1}(s_1, ..., s_m))/x_{i+1}]$$

which is the σ-matrix of the functional form up to i of some sentence

$$A = Q_n x_n ... \forall x_{i+1} Q_i x_i ... Q_1 x_1 C$$

in $\rightarrow \Gamma$, and for which the variable $v(f^A_{i+1}(s_1, ..., s_m))$ corresponds to a maximal term in (the current) HT. Then, apply the $\forall : right$ rule to $v(f^A_{i+1}(s_1, ..., s_m))$, obtaining the sequent $\rightarrow \Delta'$ in which

$$\sigma_j(\forall x_{i+1} Q_i x_i ... Q_1 x_1 B)$$

replaces

$$\sigma_j(Q_i x_i ... Q_1 x_1 B[v(f^A_{i+1}(s_1, ..., s_m))/x_{i+1}]).$$

At the end of this step, delete $f^A_{i+1}(s_1, ..., s_m)$ from HT, and perform contractions (and exchanges) if possible. The new list of substitution pairs is obtained by first replacing

$$(B_j, \sigma_j)$$

by

$$(\forall x_{i+1} Q_i x_i ... Q_1 x_1 B, \sigma_j),$$

and performing the contractions and exchanges specified above.

Otherwise, there is a formula

$$B_j = Q_i x_i ... Q_1 x_1 B$$

which is the σ-matrix of the functional form up to i of some sentence

$$A = Q_n x_n ... \exists x_{i+1} Q_i x_i ... Q_1 x_1 C$$

in $\rightarrow \Gamma$. Then, apply the $\exists : right$ rule to the leftmost substitution instance

$$\sigma_j(B_j)$$

in $\rightarrow \Delta$ of such a formula. The conclusion of this inference is the sequent $\rightarrow \Delta'$ in which

$$\sigma'_j(\exists x_{i+1} Q_i x_i ... Q_1 x_1 B)$$

replaces

$$\sigma_j(Q_i x_i ... Q_1 x_1 B),$$

where σ'_j is the restriction of the substitution σ_j obtained by eliminating x_{i+1} from the support of σ_j (Hence, $\sigma'_j(x_{i+1}) = x_{i+1}$). The pair

$$(\exists x_{i+1} Q_i x_i ... Q_1 x_1, \sigma'_j)$$

replaces the pair

$$(B_j, \sigma_j)$$

in the list of substitution pairs. After this step, perform contractions (and exchanges) if possible. Note that in this step, no term is deleted from HT.

Repeat this process until a list of substitution pairs $< (B_1, \sigma_1), ..., (B_p, \sigma_p) >$ is obtained, such that every substitution σ_j has empty support and $\rightarrow B_1, ..., B_p$ is the sequent $\rightarrow \Gamma$.

We claim that Π defines a deduction of $\rightarrow \Gamma$ from $\rightarrow H'$. First, it is easy to see that the sequence Π ends with the sequent $\rightarrow \Gamma$, since we started with substitution instances of matrices of functional forms of sentences in $\rightarrow \Gamma$, and since every step brings some formula in Δ "closer" to the corresponding formula in Γ. We leave the details as an exercise.

To show that the eigenvariable condition is satisfied for every application of the $\forall : right$ rule, we show the following claim by induction on the number of $\forall : right$ steps in Π.

Claim: Just before any application of a $\forall : right$ rule, the set of terms of the form $f_i^A(s_1, ..., s_m)$ such that $v(f_i^A(s_1, ..., s_m))$ occurs (free) in Δ is the current set HT, and for every maximal term $f_i^A(s_1, ..., s_m) \in HT$, the variable $v(f_i^A(s_1, ..., s_m))$ occurs free in at most one formula in Δ of the form

$$Q_{i-1} x_{i-1} ... Q_1 x_1 B[v(f_i^A(s_1, ..., s_m))/x_i].$$

Proof of claim: Just before the first $\forall : right$ step, since all the formulae in $\rightarrow \Delta$ are of the form $\exists x_{i-1} ... \exists x_1 B$, and since the formulae in $\rightarrow H$ are substitution instances of matrices of sentences occurring in the functional form of $\rightarrow \Gamma$, it is clear that the set of terms of the form $f_i^A(s_1, ..., s_m)$ such that $v(f_i^A(s_1, ..., s_m))$ occurs in $\rightarrow \Delta$ is the initial set HT. Since a term $f_i^A(s_1, ..., s_m)$ is maximal in HT if and only if it corresponds to the rightmost occurrence of a universal quantifier in A, $v(f_i^A(s_1, ..., s_m))$ occurs free at most in a single formula

$$\exists x_{i-1} ... \exists x_1 B[v(f_i^A(s_1, ..., s_m))/x_i]$$

(substitution instance of the σ-matrix of the functional form up to $i-1$ of the sentence $A = Q_n x_n ... \forall x_i \exists x_{i-1} ... \exists x_1 C$ in $\rightarrow \Gamma$). Next, assuming the induction hypothesis, let $\rightarrow \Delta_1$ be the sequent and HT_1 the set of terms just before an application of a $\forall : right$ step, and $\rightarrow \Delta_2$ be the sequent and HT_2 the set of terms just before the next $\forall : right$ step. Since the maximal term $f_i^A(s_1, ..., s_m)$ is deleted from HT_1 during the $\forall : right$ step applied to $\rightarrow \Delta_1$, and since the following steps until the next $\forall : right$ step are $\exists : right$ rules which do not affect $HT_1 - \{f_i^A(s_1, ..., s_m)\}$,

$$HT_2 = HT_1 - \{f_i^A(s_1, ..., s_m)\}.$$

Since a term $f_i^A(s_1, ..., s_m)$ is maximal in HT_2 if and only if it corresponds to the rightmost occurrence of a universal quantifier in the prefix $Q_n x_n ... Q_{i+1} x_{i+1} \forall x_i$ of the formula

$$A = Q_n x_n ... Q_{i+1} x_{i+1} \forall x_i Q_{i-1} x_{i-1} ... Q_1 x_1 C$$

in $\rightarrow \Gamma$, it must correspond to $\forall x_i$. If the variable $v(f_i^A(s_1, ..., s_m))$ occurs free in some other formula $R_{j-1} x_{j-1} ... R_1 x_1 B'$ in $\rightarrow \Delta_2$, since $R_j = \forall$, $v(f_i^A(s_1, ..., s_m))$ occurs within a term of the form $f_j^{A'}(s'_1, ..., s'_q)$, contradicting the maximality of $f_i^A(s_1, ..., s_m)$, since $f_j^{A'}(s'_1, ..., s'_q)$ is also in HT_2 (as a result of the induction hypothesis). Therefore, $v(f_i^A(s_1, ..., s_m))$ may only occur free in the formula

$$Q_{i-1} x_{i-1} ... Q_1 x_1 B[v(f_i^A(s_1, ..., s_m))/x_i]$$

in $\rightarrow \Delta_2$, substitution instance of the σ-matrix of the functional form up to $i - 1$ of the formula

$$A = Q_n x_n ... \forall x_i Q_{i-1} x_{i-1} ... Q_1 x_1 C$$

in $\rightarrow \Gamma$. Hence, the eigenvariable condition is satisfied. In a $\exists : step$, since the variables occurring in the term s are distinct from the variables occurring bound in the formulae in Γ, the term s is free for x_i in the substitution, and the inference is valid. Hence, Π yields a deduction of $\rightarrow \Gamma$ from $\rightarrow H'$, which can be extended to a proof of $\rightarrow \Gamma$ from axioms, since $\rightarrow H'$ is provable. This concludes the proof of Herbrand's theorem. □

The method for reconstructing a proof from a list of substitution pairs is illustrated in the following example.

EXAMPLE 7.5.3

Consider the sequent $\rightarrow \Gamma$ given by:

$$\rightarrow \exists x \forall y \neg P(x, y), \exists y_1 \forall z \neg Q(y_1, z), \forall x_1 \exists y_2 \exists z_1 (P(x_1, y_2) \wedge Q(y_2, z_1)),$$

whose functional form is:

$$\rightarrow \exists x \neg P(x, f(x)), \exists y_1 \neg Q(y_1, g(y_1)), \exists y_2 \exists z_1 (P(a, y_2) \wedge Q(y_2, z_1)).$$

The provable sequent $\rightarrow H$ given by

$$\rightarrow \neg P(a, f(a)), \neg Q(f(a), g(f(a))), P(a, f(a)) \wedge Q(f(a), g(f(a)))$$

is obtained from the functional form of the original sequent by deleting quantifiers and substituting the terms a, $f(a)$, $f(a)$ and $g(f(a))$ for x, y_1, y_2, and z_1 respectively. We have $HT = \{a, f(a), g(f(a))\}$, with the ordering $a < f(a)$, $a < g(f(a))$, $f(a) < g(f(a))$.

Define the bijection v' such that $v'(g(f(a))) = u$, $v'(f(a)) = v$ and $v'(a) = w$. The result of replacing in $\rightarrow H$ the maximal terms in HT by the variables given by v' is the sequent $\rightarrow H'$ given by

$$\rightarrow \neg P(w, v), \neg Q(v, u), P(w, v) \wedge Q(v, u).$$

The formula $P(w, v) \wedge Q(v, u)$ is the σ-matrix of the functional form up to 0 of the formula $\forall w \exists v \exists u (P(w, v) \wedge Q(v, u))$. Hence we have a $\exists$: *right* step.

$$\frac{\rightarrow \neg P(w, v), \neg Q(v, u), P(w, v) \wedge Q(v, u)}{\rightarrow \neg P(w, v), \neg Q(v, u), \exists z_1 (P(w, v) \wedge Q(v, z_1))}$$

Similarly, $\exists z_1 (P(w, v) \wedge Q(v, z_1))$ is the σ-matrix of the functional form up to 1 of $\forall w \exists v \exists z_1 (P(w, v) \wedge Q(v, z_1))$. Hence, we have another $\exists$: *right* step.

$$\begin{array}{c}
\rightarrow \neg P(w, v), \neg Q(v, u), P(w, v) \wedge Q(v, u) \\ \hline
\rightarrow \neg P(w, v), \neg Q(v, u), \exists z_1 (P(w, y_2) \wedge Q(y_2, z_1)) \\ \hline
\rightarrow \neg P(w, v), \neg Q(v, u), \exists y_2 \exists z_1 (P(w, y_2) \wedge Q(y_2, z_1))
\end{array}$$

Now, only a $\forall$: *right* step can be applied. According to the algorithm, we apply it to the leftmost formula for which the variable $v'(t)$ corresponds to a maximal term $t \in HT$. This must be $\neg Q(v, u)$, since $v'(g(f(a)) = u$ and $g(f(a))$ is the largest element of HT. We also delete $g(f(a))$ from HT.

Note that it would be wrong to apply the $\forall$: *right* rule to any of the other formulae, since both w and v would occur free in the conclusion of that inference.

$$\begin{array}{c}
\rightarrow \neg P(w, v), \neg Q(v, u), P(w, v) \wedge Q(v, u) \\ \hline
\rightarrow \neg P(w, v), \neg Q(v, u), \exists z_1 (P(w, y_2) \wedge Q(y_2, z_1)) \\ \hline
\rightarrow \neg P(w, v), \neg Q(v, u), \exists y_2 \exists z_1 (P(w, y_2) \wedge Q(y_2, z_1)) \\ \hline
\rightarrow \neg P(w, v), \forall z \neg Q(v, z), \exists y_2 \exists z_1 (P(w, y_2) \wedge Q(y_2, z_1))
\end{array}$$

Now, we can apply a $\exists : right$ step to $\forall z \neg Q(v, z)$.

$$\begin{array}{c}
\to \neg P(w, v), \neg Q(v, u), P(w, v) \land Q(v, u) \\ \hline
\to \neg P(w, v), \neg Q(v, u), \exists z_1 (P(w, y_2) \land Q(y_2, z_1)) \\ \hline
\to \neg P(w, v), \neg Q(v, u), \exists y_2 \exists z_1 (P(w, y_2) \land Q(y_2, z_1)) \\ \hline
\to \neg P(w, v), \forall z \neg Q(v, z), \exists y_2 \exists z_1 (P(w, y_2) \land Q(y_2, z_1)) \\ \hline
\to \neg P(w, v), \exists y_1 \forall z \neg Q(y_1, z), \exists y_2 \exists z_1 (P(w, y_2) \land Q(y_2, z_1))
\end{array}$$

At this point, a $\forall : right$ step is the only possibility. Since the next largest term in $HT = \{a, f(a)\}$ is $f(a)$, we apply it to $\neg P(w, v)$.

$$\begin{array}{c}
\to \neg P(w, v), \neg Q(v, u), P(w, v) \land Q(v, u) \\ \hline
\to \neg P(w, v), \neg Q(v, u), \exists z_1 (P(w, y_2) \land Q(y_2, z_1)) \\ \hline
\to \neg P(w, v), \neg Q(v, u), \exists y_2 \exists z_1 (P(w, y_2) \land Q(y_2, z_1)) \\ \hline
\to \neg P(w, v), \forall z \neg Q(v, z), \exists y_2 \exists z_1 (P(w, y_2) \land Q(y_2, z_1)) \\ \hline
\to \neg P(w, v), \exists y_1 \forall z \neg Q(y_1, z), \exists y_2 \exists z_1 (P(w, y_2) \land Q(y_2, z_1)) \\ \hline
\to \forall y \neg P(w, y), \exists y_1 \forall z \neg Q(y_1, z), \exists y_2 \exists z_1 (P(w, y_2) \land Q(y_2, z_1))
\end{array}$$

We can now apply a $\exists : right$ step to $\forall y \neg P(w, y)$.

$$\begin{array}{c}
\to \neg P(w, v), \neg Q(v, u), P(w, v) \land Q(v, u) \\ \hline
\to \neg P(w, v), \neg Q(v, u), \exists z_1 (P(w, y_2) \land Q(y_2, z_1)) \\ \hline
\to \neg P(w, v), \neg Q(v, u), \exists y_2 \exists z_1 (P(w, y_2) \land Q(y_2, z_1)) \\ \hline
\to \neg P(w, v), \forall z \neg Q(v, z), \exists y_2 \exists z_1 (P(w, y_2) \land Q(y_2, z_1)) \\ \hline
\to \neg P(w, v), \exists y_1 \forall z \neg Q(y_1, z), \exists y_2 \exists z_1 (P(w, y_2) \land Q(y_2, z_1)) \\ \hline
\to \forall y \neg P(w, y), \exists y_1 \forall z \neg Q(y_1, z), \exists y_2 \exists z_1 (P(w, y_2) \land Q(y_2, z_1)) \\ \hline
\to \exists x \forall y \neg P(x, y), \exists y_1 \forall z \neg Q(y_1, z), \exists y_2 \exists z_1 (P(w, y_2) \land Q(y_2, z_1))
\end{array}$$

Finally, since $HT = \{a\}$ and only a $\forall : right$ step is possible, a $\forall : right$ step is applied to $\exists y_2 \exists z_1 (P(w, y_2) \land Q(y_2, z_1))$.

$$\begin{array}{c}
\rightarrow \neg P(w,v), \neg Q(v,u), P(w,v) \land Q(v,u) \\
\hline
\rightarrow \neg P(w,v), \neg Q(v,u), \exists z_1(P(w,y_2) \land Q(y_2,z_1)) \\
\hline
\rightarrow \neg P(w,v), \neg Q(v,u), \exists y_2 \exists z_1(P(w,y_2) \land Q(y_2,z_1)) \\
\hline
\rightarrow \neg P(w,v), \forall z \neg Q(v,z), \exists y_2 \exists z_1(P(w,y_2) \land Q(y_2,z_1)) \\
\hline
\rightarrow \neg P(w,v), \exists y_1 \forall z \neg Q(y_1,z), \exists y_2 \exists z_1(P(w,y_2) \land Q(y_2,z_1)) \\
\hline
\rightarrow \forall y \neg P(w,y), \exists y_1 \forall z \neg Q(y_1,z), \exists y_2 \exists z_1(P(w,y_2) \land Q(y_2,z_1)) \\
\hline
\rightarrow \exists x \forall y \neg P(x,y), \exists y_1 \forall z \neg Q(y_1,z), \exists y_2 \exists z_1(P(w,y_2) \land Q(y_2,z_1)) \\
\hline
\rightarrow \exists x \forall y \neg P(x,y), \exists y_1 \forall z \neg Q(y_1,z), \forall x_1 \exists y_2 \exists z_1(P(x_1,y_2) \land Q(y_2,z_1))
\end{array}$$

This last derivation is a deduction of $\rightarrow \Gamma$ from $\rightarrow H'$. Observe that this proof is identical to the proof of example 7.3.4.

Remarks: (1) The difference between first-order logic without equality and first-order logic with equality noted in the paragraph following the proof of theorem 5.6.1 shows up again in Herbrand's theorem. For a language without equality, in view of the second corollary to theorem 5.5.1, the hard part in finding a proof is to find appropriate substitutions yielding a valid Herbrand disjunction. Indeed, as soon as such a quantifier-free formula is obtained, there is an algorithm for deciding whether it is provable (or valid). However, in view of the remark following the corollary, Church's theorem implies that there is no algorithm for finding these appropriate substitutions.

For languages with equality, the situation is worse! Indeed, even if we can find appropriate substitutions yielding a quantifier-free formula, we are still facing the problem of finding an algorithm for deciding the provability (or validity) of quantifier-free formulae if equality is present. As we mentioned in Chapter 5, there is such an algorithm presented in Chapter 10, but it is nontrivial. Hence, it appears that automatic theorem proving in the presence of equality is harder than automatic theorem proving without equality. This phenomenon will show up again in the resolution method.

(2) Note that the last part of the proof of theorem 7.5.1 provides an algorithm for constructing a proof of $\rightarrow \Gamma$ from the Herbrand disjunction H (really, the list of substitution pairs) and its proof. Similarly, the first part of the proof provides an algorithm for constructing an Herbrand disjunction and its proof, from a proof satisfying the conditions of Gentzen's sharpened Hauptsatz. Actually, since the proof of the sharpened Hauptsatz from Gentzen's cut elimination theorem is entirely constructive, a Herbrand disjunction and its proof can be constructed from a pure-variable, cut-free proof. The only step that has not been justified constructively in our presentation is the fact that a provable sequent has a cut-free proof. This is because even though the *search* procedure yields a cut-free proof of a provable sequent, the correctness and termination of the *search* procedure for provable sequents is established by

semantic means involving a nonconstructive step: the existence of the possibly infinite counter-example tree (considering the case where the sequent is falsifiable). However, Gentzen gave a completely constructive (syntactic) proof of the cut elimination theorem, and so, the version of Herbrand's theorem given in this section is actually entirely constructive, as is Herbrand's original version (Herbrand, 1971). See also lemma 7.6.2.

As mentioned at the beginning of this chapter, there is a theorem similar in form to Herbrand's theorem and known as the Skolem-Herbrand-Gödel theorem. Since a version of that theorem will be proved in Section 7.6, we postpone a discussion of the relationship between the two theorems to the end of Section 7.6.

PROBLEMS

7.5.1. Prove the following fact: Given a sequent $A_1, ..., A_m \rightarrow B_1, ..., B_n$, $A_1, ..., A_m \rightarrow B_1, ..., B_n$ is provable (in LK) if and only if $\rightarrow \neg A_1 \lor ... \lor \neg A_m \lor B_1 \lor ... \lor B_n$ is provable (in LK).

7.5.2. The method given in Section 7.2 for converting a formula to prenex form used in conjunction with the Skolemization method of Section 7.5 tends to create Skolem functions with more arguments than necessary.

(a) Prove that the following method for Skolemizing is correct:

Step 1: Eliminate redundant quantifiers; that is, quantifiers $\forall x$ or $\exists x$ such that the input formula contains a subformula of the form $\forall x B$ or $\exists x B$ in which x does not occur in B.

Step 2: Rectify the formula.

Step 3: Eliminate the connectives $\supset$ and $\equiv$.

Step 4: Convert to NNF.

Step 5: Push quantifiers to the right. By this, we mean: Replace

$$\exists x(A \lor B) \text{ by } \begin{cases} A \lor \exists x B & \text{if } x \text{ is not free in } A, \\ \exists x A \lor B & \text{if } x \text{ is not free in } B. \end{cases}$$

$$\forall x(A \lor B) \text{ by } \begin{cases} A \lor \forall x B & \text{if } x \text{ is not free in } A\ , \\ \forall x A \lor B & \text{if } x \text{ is not free in } B. \end{cases}$$

$$\exists x(A \land B) \text{ by } \begin{cases} A \land \exists x B & \text{if } x \text{ is not free in } A, \\ \exists x A \land B & \text{if } x \text{ is not free in } B. \end{cases}$$

$$\forall x(A \land B) \text{ by } \begin{cases} A \land \forall x B & \text{if } x \text{ is not free in } A, \\ \forall x A \land B & \text{if } x \text{ is not free in } B. \end{cases}$$

Step 6: Eliminate universal quantifiers using Skolem function and constant symbols.

Step 7: Move existential quantifiers to the left, using the inverse of the transformation of step 5.

(b) Compare the first method and the method of this problem for the formula

$$\forall x_2 \exists y_1 \forall x_1 \exists y_2 (P(x_1, y_1) \land Q(x_2, y_2)).$$

Note: Step 5 is the step that reduces the number of arguments of Skolem functions.

7.5.3. Prove that the following formulae are valid using Herbrand's theorem:

$$\neg(\exists x \forall y P(x, y) \land \forall y \exists x P(y, x))$$
$$\neg(\neg(\forall x P(x) \lor \exists y \neg Q(y)) \lor (\forall z P(z) \lor \exists w \neg Q(w)))$$
$$\neg(\neg\forall x(P(x) \lor \exists y \neg Q(y)) \lor (\forall z P(z) \lor \exists w \neg Q(w)))$$

7.5.4. Give an example in which $A[s_1/x_1][s_2/x_2]...[s_n/x_n]$, the result of substituting s_1 for x_1, ... ,s_n for x_n (as defined in definition 5.2.6) in that order, is different from $A[s_1/x_s, ..., s_n/x_n]$.

Show that if none of the variables in the support of the substitution σ occurs in the terms s_1,...,s_n, the order in which the substitutions are performed is irrelevant, and in this case,

$$A[s_1/x_1][s_2/x_2]...[s_n/x_n] = A[s_1/x_1, ..., s_n/x_n].$$

7.5.5. Fill in the missing details in the proof of theorem 7.5.1.

7.5.6. Consider the following formula given by

$$\neg\exists y \forall z (P(z, y) \equiv \neg\exists x (P(z, x) \land P(x, z))).$$

(a) Prove that the above formula is equivalent to the following prenex formula A:

$$\forall y \exists z \forall u \exists x [(P(z, y) \land P(z, x) \land P(x, z)) \lor (\neg P(z, y) \land (\neg P(z, u) \lor \neg P(u, z)))].$$

(b) Show that A can be Skolemized to the formula

$$B = \exists z \exists x [(P(z, a) \land P(z, x) \land P(x, z)) \lor (\neg P(z, a) \land (\neg P(z, f(z)) \lor \neg P(f(z), z)))],$$

and that the formula C given by

$$[(P(a, a) \land P(a, f(a)) \land P(f(a), a)) \lor (\neg P(a, a) \land (\neg P(a, f(a)) \lor \neg P(f(a), a)))]$$

is valid.

(c) Using the method of theorem 7.5.1, reconstruct a proof of A from the valid Herbrand disjunction C.

7.5.7. Consider a first-order language without equality. Show that Herbrand's theorem provides an algorithm for deciding the validity of prenex sentences of the form

$$\forall x_1 ... \forall x_m \exists y_1 ... \exists y_n B.$$

7.5.8. Write a computer program implementing the method given in the proof of theorem 7.5.1 for reconstructing a proof from a Herbrand's disjunction.

7.6 Skolem-Herbrand-Gödel's Theorem for Formulae in NNF

In this section, we shall state a version of the Skolem-Herbrand-Gödel theorem for unsatisfiability as opposed to validity.

7.6.1 Skolem-Herbrand-Gödel's Theorem in Unsatisfiability Form

Using the results of Section 7.4, we shall derive a version of the Herbrand-Skolem-Gödel theorem for formulae in NNF due to Andrews (Andrews, 1981). Actually, we shall prove more. We shall also give half of a version of Herbrand's theorem for sentences in NNF, the part which states that if a sequent $A \rightarrow$ is provable, then a quantifier-free formula C whose negation $\neg C$ is provable can be effectively constructed.

We believe that it is possible to give a constructive Herbrand-like version of this theorem similar to theorem 7.5.1, but the technical details of the proof of the converse of the theorem appear to be very involved. Hence we shall use a mixed strategy: Part of the proof will be obtained constructively from theorem 7.4.1, the other part by a semantic argument showing that a sentence is satisfiable iff its Skolem form is satisfiable. This last result is also interesting in its own right, and can be used to prove other results, such as the compactness theorem, and the Löwenheim-Skolem theorem (see the problems).

Since a formula A is valid iff $\neg A$ is unsatisfiable, any unsatisfiability version of the Skolem-Herbrand-Gödel theorem yields a validity version of the theorem, and vice versa. Since one of the most important applications of the Skolem-Herbrand-Gödel theorem is the completeness of refutation-oriented procedures such as the resolution method (to be presented in Chapter 8) and the method of matings (Andrews, 1981), it will be useful for the reader to see a treatment of this theorem for unsatisfiablity.

As discussed in Section 7.5, since our goal is now to prove that a formula A is valid by showing that $\neg A$ is unsatisfiable, we are going to use the dual of the method used in Section 7.5, that is, eliminate *existential quantifiers* using Skolem functions. However, it is not quite as simple to define the conversion of a formula in NNF to Skolem normal form (satisfiability functional form) as it is to convert a formula in prenex form into (validity) functional form. We present an example first.

EXAMPLE 7.6.1

Consider the formula

$$A = \exists x((P(x) \lor \exists y R(x,y)) \supset (\exists z R(x,z) \lor P(a))).$$

The NNF of its negation is

$$B = \forall x((P(x) \lor \exists y R(x,y)) \land (\forall z \neg R(x,z) \land \neg P(a))).$$

The following is a $(G2^{nnf})$ proof in normal form of the sequent $B \rightarrow$:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{P(a), \neg P(a) \rightarrow}{P(a), \neg R(a,y), \neg P(a) \rightarrow} \qquad \dfrac{R(a,y), \neg R(a,y) \rightarrow}{R(a,y), \neg R(a,y), \neg P(a) \rightarrow}}{(P(a) \lor R(a,y)), \neg R(a,y), \neg P(a) \rightarrow}}{(P(a) \lor R(a,y)), (\neg R(a,y) \land \neg P(a)) \rightarrow}}{(P(a) \lor R(a,y)) \land (\neg R(a,y) \land \neg P(a)) \rightarrow}}{(P(a) \lor R(a,y)) \land (\forall z \neg R(a,z) \land \neg P(a)) \rightarrow}}{(P(a) \lor \exists y R(a,y)) \land (\forall z \neg R(a,z) \land \neg P(a)) \rightarrow}}{\forall x((P(x) \lor \exists y R(x,y)) \land (\forall z \neg R(x,z) \land \neg P(a))) \rightarrow}$$

(In order to shorten the proof, the $\land : left$ rule of G was used rather than the $\land : left$ rule of $G2^{nnf}$. We leave it as an exercise to make the necessary alterations to obtain a pure $G2^{nnf}$-proof.) The midsequent is

$$(P(a) \lor R(a,y)) \land (\neg R(a,y) \land \neg P(a)) \ \rightarrow .$$

The existential quantifier can be eliminated by introducing the unary Skolem function symbol f, and we have the following sequent:

$$(*) \qquad \forall x((P(x) \lor R(x, f(x))) \land (\forall z \neg R(x,z) \land \neg P(a))) \rightarrow$$

If in the above proof we replace all occurrences of the eigenvariable y by $f(a)$, we obtain a proof of the sequent $(*)$ whose midsequent is:

$$(P(a) \lor R(a, f(a))) \land (\neg R(a, f(a)) \land \neg P(a)) \to .$$

This illustates the Skolem-Herbrand-Gödel's theorem stated in unsatisfiability form: A formula B is unsatisfiable iff some special kind of quantifier-free substitution instance of B is unsatisfiable.

Such instances are called *compound instances* by Andrews (Andrews, 1981). We shall now define precisely all the concepts mentioned in the above example.

7.6.2 Skolem Normal Form

We begin with the notion of universal scope of a subformula.

Definition 7.6.1 Given a (rectified) formula A in NNF, the set $US(A)$ of pairs $< B, L >$ where B is a subformula of A and L is a sequence of variables is defined inductively as follows:

$$\begin{aligned}
US_0 &= \{< A, <>>\};\\
US_{k+1} &= US_k \cup \{< C, L >, < D, L > \mid\ < B, L > \in US_k,\\
&\qquad B \text{ is of the form } (C \land D) \text{ or } (C \lor D)\}\\
&\qquad \cup \{< C, L > \mid\ < \exists x C, L > \in US_k\}\\
&\qquad \cup \{< C, < y_1, ..., y_m, x >> \mid\ < \forall x C, < y_1, ..., y_m >> \in US_k\}.
\end{aligned}$$

For every subformula B of A, the sequence L of variables such that $< B, L >$ belongs to $US(A) = \bigcup US_k$ is the *universal scope* of B.

In the process of introducing Skolem symbols to eliminate existential quantifiers, we shall consider the subset of US consisting of all the pairs $< \exists x B, L >$, where $\exists x B$ is a subformula of A.

EXAMPLE 7.6.2

Let

$$A = \forall x(P(a) \lor \exists y(Q(y) \land \forall z(P(y, z) \lor \exists u Q(x, u)))) \lor \exists w Q(a, w).$$

Then,

$$\begin{gathered}
< \exists y(Q(y) \land \forall z(P(y, z) \lor \exists u Q(x, u))), < x >>,\\
< \exists u Q(x, u), < x, z >> \quad \text{and}\\
< \exists w Q(a, w), <>>
\end{gathered}$$

define the universal scope of the subformulae of A of the form $\exists xB$. We now define the process of Skolemization.

Definition 7.6.2 Given a rectified sentence A, the *Skolem form* $SK(A)$ of A (or *Skolem normal form*) is defined recursively as follows using the set $US(A)$. Let A' be any subformula of A:

(i) If A' is either an atomic formula B or the negation $\neg B$ of an atomic formula B, then

$$SK(A') = A'.$$

(ii) If A' is of the form $(B * C)$, where $* \in \{\lor, \land\}$, then

$$SK(A') = (SK(B) * SK(C)).$$

(iii) If A' is of the form $\forall xB$, then

$$SK(A') = \forall x SK(B).$$

(iv) If A' is of the form $\exists xB$, then if $< y_1, ..., y_m >$ is the universal scope of $\exists xB$ (that is, the sequence of variables such that $< \exists xB, < y_1, ..., y_m >> \in US(A)$) then

(a) If $m > 0$, create a new *Skolem function symbol* $f_{A'}$ of rank m and let

$$SK(A') = SK(B[f_{A'}(y_1, ..., y_m)/x]).$$

(b) If $m = 0$, create a new *Skolem constant symbol* $f_{A'}$ and let

$$SK(A') = SK(B[f_{A'}/x]).$$

Observe that since the sentence A is rectified, all subformulae A' are distinct, and since the Skolem symbols are indexed by the subformulae A', they are also distinct.

EXAMPLE 7.6.3

Let

$$A = \forall x(P(a) \lor \exists y(Q(y) \land \forall z(P(y, z) \lor \exists uQ(x, u)))) \lor \exists wQ(a, w),$$

as in example 7.6.2.

$$SK(\exists wQ(a, w)) = Q(a, c),$$
$$SK(\exists uQ(x, u)) = Q(x, f(x, z)),$$
$$SK(\exists y(Q(y) \land \forall z(P(y, z) \lor \exists uQ(x, u)))) =$$
$$(Q(g(x)) \land \forall z(P(g(x), z) \lor Q(x, f(x, z)))), \quad \text{and}$$
$$SK(A) =$$
$$\forall x(P(a) \lor (Q(g(x)) \land \forall z(P(g(x), z) \lor Q(x, f(x, z))))) \lor Q(a, c).$$

The symbol c is a Skolem constant symbol, g is a unary Skolem function symbol, and f is a binary Skolem function symbol.

7.6.3 Compound Instances

At the end of example 7.6.1, we mentioned that the midsequent of a proof in normal form of a sequent $B \rightarrow$ where B is in Skolem normal form consists of certain formulae called compound instances. The formal definition is given below.

Definition 7.6.3 Let A be a rectified sentence and let B its Skolem form. The set of *compound instances* (for short, *c-instances*) of B is defined inductively as follows:

(i) If B is either an atomic formula C or the negation $\neg C$ of an atomic formula, then B is its only c-instance;

(ii) If B is of the form $(C * D)$, where $* \in \{\vee, \wedge\}$, for any c-instance H of C and c-instance K of D, $(H * K)$ is a c-instance of B;

(iii) If B is of the form $\forall x C$, for any k closed terms t_1,...,t_k, if H_i is a c-instance of $C[t_i/x]$ for $i = 1, ..., k$, then $H_1 \wedge ... \wedge H_k$ is a c-instance of B.

EXAMPLE 7.6.4

Let

$$B = \forall x(P(x) \vee \forall y Q(y, f(x))) \wedge (\neg P(a) \wedge (\neg Q(a, f(a)) \vee \neg Q(b, f(a)))).$$

Then,

$$(P(a) \vee (Q(a, f(a)) \wedge Q(b, f(a)))) \wedge (\neg P(a) \wedge (\neg Q(a, f(a)) \vee \neg Q(b, f(a))))$$

is a c-instance of B.

Note that c-instances are quantifier free. The following lemma shows that in a certain sense, c-instances are closed under conjunctions.

Lemma 7.6.1 Let A be a sentence in NNF and in Skolem normal form. For any two c-instances K and L of A, a c-instance D such that $D \supset (K \wedge L)$ is provable can be constructed.

Proof: We proceed by induction on A.

(i) If A is either of the form B or $\neg B$ for an atomic formula B, then $K = L = A$ and we let $D = A$.

(ii) If A is of the form $(B * C)$, where $* \in \{\vee, \wedge\}$, then K is of the form $(B_1 * C_1)$ and L is of the form $(B_2 * C_2)$, where B_1 and B_2 are c-instances of B, and C_1 and C_2 are c-instances of C. By the induction hypothesis, there are c-instances D_1 of B and D_2 of C such that $D_1 \supset (B_1 * B_2)$ and

$D_2 \supset (C_1 * C_2)$ are provable. But then, $D = (D_1 * D_2)$ is a c-instance of A such that $D \supset (K * L)$ is provable.

(iii) If A is of the form $\forall xC$, then K has the form $H_1 \wedge ... \wedge H_m$, where H_i is a c-instance of $C[s_i/x]$ for $i = 1, ..., m$, and L has the form $K_1 \wedge ... \wedge K_n$, where K_j is a c-instance of $C[t_j/x]$ for $j = 1, ..., n$. It is clear that $D = K \wedge L$ satisfies the lemma. □

We are now ready for the constructive part of Herbrand's theorem

7.6.4 Half of a Herbrand-like Theorem for Sentences in NNF

In the rest of this section, it is assumed that first-order languages have at least one constant symbol. This can always be achieved by adjoining the special symbol # as a constant. First, we prove the constructive half of Herbrand's theorem announced in the introduction to this section. This part asserts a kind of completeness result: If a sequent $A \rightarrow$ is provable, then this can be demonstrated constructively by providing a (propositional) proof of the negation $\neg C$ of a quantifier-free formula C obtained from A.

Lemma 7.6.2 Let **L** be a first-order language with or without equality. Let A be an **L**-sentence in NNF, and let B be its Skolem normal form. If $A \rightarrow$ is provable (in $G1^{nnf}$ or $G1^{nnf}_{=}$), then a (quantifier-free) c-instance C of B can be constructed such that $\neg C$ is provable.

Proof: From theorems 7.4.1 and 7.4.2, if $A \rightarrow$ is provable, it has a proof in normal form, in which all quantifier rules are below all propositional rules. Let us now perform the following alteration to the proof:

In a bottom-up fashion, starting from $B \rightarrow$, whenever the $\exists : left$ rule is applied to a subformula A' of the form $\exists xB$ with eigenvariable z, if the universal scope of $\exists xB$ is $< y_1, ..., y_m >$ and the terms t_1,...,t_m have been substituted for y_1,...,y_m in previous $\forall : right$ steps, substitute $f_{A'}(t_1, ..., t_m)$ for all occurrence of z in the proof. If the midsequent of the resulting tree still has variables, substitute any constant for all occurrences of these variables.

It is not difficult to prove by induction on proof trees that the resulting deduction is a proof of the sequent $B \rightarrow$, where B is the Skolem form of A. This is left as an (easy) exercise.

We can also prove that the midsequent consists of c-instances of B. For this, we prove the following claim by induction on proof trees:

Claim: For every proof in normal form of a sequent $\Gamma \rightarrow$ consisting of Skolem forms of sentences, the formulae in the midsequent are c-instances of the sentences in Γ.

Proof of claim: We have three cases depending on the sentence B to which the quantifier rule is applied.

The result is trivial if B is either an atomic formula or the negation of an atomic formula.

If B is of the form $(C * D)$, where $* \in \{\wedge, \vee\}$, assume without loss of generality that the $\forall : left$ rule is applied to C. Hence, in the premise of the rule, $(C * D)$ is replaced by $(C[F[t/x])/E] * D)$, where E is a maximal subformula of C of the form $\forall xF$. By the induction hypothesis, the midsequent consists of c-instances of $(C[F[t/x])/E] * D)$ and of the other formulae in Γ. However, one can easily show by induction on the formula C that a c-instance of $(C[F[t/x])/E] * D)$ is also a c-instance of $(C * D)$. This is left as an exercise to the reader. Hence, the result holds.

If B is of the form $\forall xC$, then in the premise of the rule, B is replaced by $C[t/x]$ for some closed term t. By the induction hypothesis, the midsequent consists of c-instances of $C[t/x]$ and of the other formulae in Γ. By definition, a c-instance of $C[t/x]$ is a c-instance of B. Hence, the midsequent consists of c-instances of the sentences in Γ. □

If C_1,...,C_m are the c-instances of B occurring in the midsequent, since $C_1, ..., C_m \rightarrow$ is provable, $\neg(C_1 \wedge ... \wedge C_m)$ is provable. Applying lemma 7.6.1 $m - 1$ times, a c-instance C of B can be constructed such that $C \supset (C_1 \wedge ... \wedge C_m)$ is provable. But then, $\neg C$ is provable since $\neg(C_1 \wedge ... \wedge C_m)$ is provable. □

Remark: In Andrews, 1981, a semantic proof of lemma 7.6.2 is given for languages *without equality*. Our result applies to languages with equality as well, and is constructive, because we are using the full strength of the normal form theorems (theorems 7.4.1 and 7.4.2).

7.6.5 Skolem-Herbrand-Gödel's Theorem (Sentences in NNF)

In order to prove the converse of the above lemma, we could as in the proof of theorem 7.5.1 try to reconstruct a proof from an unsatisfiable c-instance C of B. This is a rather delicate process whose justification is very tedious, and we will follow a less constructive but simpler approach involving semantic arguments. Hence, instead of proving the converse of the half of Herbrand's theorem shown in lemma 7.6.2, we shall prove a version of the Skolem-Herbrand-Gödel theorem.

The following fact will be used:

$(*)$ If the Skolem form B of a sentence A is unsatisfiable then A is unsatisfiable.

Since it is easy to prove that if C is a c-instance of B then $(B \supset C)$ is valid, if C is unsatisfiable, then B must be unsatisfiable, and by $(*)$ A is also unsatisfiable.

We shall actually prove not only $(*)$ but also its converse. This is more than we need for the part of the proof of the Skolem-Herbrand-Gödel theorem, but since this result can be used to give a semantic proof of the Skolem-Herbrand-Gödel theorem (see the problems), it is interesting to prove it in full.

Lemma 7.6.3 Let **L** be a first-order language with or without equality. Let A be a rectified **L**-sentence in NNF, and let B be its Skolem normal form. The sentence A is satisfiable iff its Skolem form B is satisfiable.

Proof: Let C be any subformula of A. We will show that the following properties hold:

(a) For every structure $\mathbf{A}$ such that all function, predicate, and constant symbols in the Skolem form $SK(C)$ of C receive an interpretation, for every assignment s, if $\mathbf{A} \models SK(C)[s]$ then $\mathbf{A} \models C[s]$.

(b) For every structure $\mathbf{A}$ such that exactly all function, predicate, and constant symbols in C receive an interpretation, for every assignment s (with range A), if $\mathbf{A} \models C[s]$ then there is an expansion $\mathbf{B}$ of $\mathbf{A}$ such that $\mathbf{B} \models SK(C)[s]$.

Recall from definition 5.4.6 that if $\mathbf{B}$ is an expansion of $\mathbf{A}$, then $\mathbf{A}$ and $\mathbf{B}$ have the same domain, and the interpretation function of $\mathbf{A}$ is a restriction of the interpretation function of $\mathbf{B}$.

The proof proceeds by induction on subformulae of A.

(i) If C is either of the form D or $\neg D$ where D is atomic, $SK(C) = C$ and both (a) and (b) are trivial.

(ii) If C is of the form $(D \land E)$, then $SK(C) = (SK(D) \land SK(E))$.

(a) If $\mathbf{A} \models SK(C)[s]$ then $\mathbf{A} \models SK(D)[s]$ and $\mathbf{A} \models SK(E)[s]$. By the induction hypothesis, $\mathbf{A} \models D[s]$ and $\mathbf{A} \models E[s]$. But then, $\mathbf{A} \models C[s]$.

(b) Let $\mathbf{A}$ be a structure such that exactly all function, predicate, and constant symbols in C receive an interpretation. Since $\mathbf{A} \models C[s]$, we have $\mathbf{A} \models D[s]$ and $\mathbf{A} \models E[s]$. By the induction hypothesis, there are expansions $\mathbf{B}_1$ and $\mathbf{B}_2$ of $\mathbf{A}$ such that $\mathbf{B}_1 \models SK(D)[s]$ and $\mathbf{B}_2 \models SK(E)[s]$. Since $\mathbf{B}_1$ and $\mathbf{B}_2$ are both expansions of $\mathbf{A}$, their interpretation functions agree on all the predicate, function, and constant symbols occurring in both D and E, and since the sets of Skolem symbols in D and E are disjoint (because A is rectified), we can take the union of the two interpretation functions to obtain an expansion $\mathbf{B}$ of $\mathbf{A}$ such that $\mathbf{B} \models (SK(D) \land SK(E))[s]$, that is, $\mathbf{B} \models SK(C)[s]$.

(iii) C is of the form $(D \lor E)$. This case is similar to case (ii) and is left as an exercise.

(iv) C is of the form $\forall x D$. Then $SK(C) = \forall x SK(D)$.

(a) If $\mathbf{A} \models SK(C)[s]$, then $\mathbf{A} \models SK(D)[s[x := a]]$ for all $a \in A$. By the induction hypothesis, $\mathbf{A} \models D[s[x := a]]$ for all $a \in A$, that is, $\mathbf{A} \models C[s]$.

(b) If $\mathbf{A} \models C[s]$, then $\mathbf{A} \models D[s[x := a]]$ for all $a \in A$. By the induction hypothesis, there is an expansion $\mathbf{B}$ of $\mathbf{A}$ such that for any assignment s', if $\mathbf{A} \models D[s']$ then $\mathbf{B} \models SK(D)[s']$. But then, for all $a \in A$, we have $\mathbf{B} \models SK(D)[s[x := a]]$, that is, $\mathbf{B} \models SK(C)[s]$.

(v) C is of the form $\exists xD$. Then $SK(C) = SK(D)[f_C(y_1, ..., y_m)/x]$, where $< y_1, ..., y_m >$ is the universal scope of C in A, and f_C is the Skolem symbol associated with C.

(a) If $\mathbf{A} \models SK(C)$, then by lemma 5.4.1, letting

$$a = (f_C(y_1, ..., y_m))_{\mathbf{A}}[s],$$

we have $\mathbf{A} \models (SK(D)[\mathbf{a}/x])[s]$, which by lemma 5.3.1 is equivalent to $\mathbf{A} \models (SK(D))[s[y := a]]$. By the induction hypothesis, $\mathbf{A} \models D[s[y := a]]$, that is, $\mathbf{A} \models (\exists xD)[s]$.

(b) Assume that $\mathbf{A} \models C[s]$ for every s. Then, for some $a \in A$, $\mathbf{A} \models D[s[x := a]]$. By the induction hypothesis, there is an expansion $\mathbf{B}$ of $\mathbf{A}$ such that $\mathbf{B} \models (SK(D))[s[x := a]]$. Observe that by the recursive definition of the Skolem form of a formula, $FV(SK(D)) = \{x, y_1, ..., y_m\}$, where $< y_1, ..., y_m >$ is the universal scope of C in A. In order to expand $\mathbf{B}$ to a structure for $SK(C)$, we need to interpret f_C in $\mathbf{B}$. For any $(a_1, ..., a_m) \in A^m$, let s be any assignment such that $s(y_i) = a_i$, for $i = 1, .., m$. By the induction hypothesis, there is some $a \in A$ such that

$$\mathbf{B} \models (SK(D))[s[x := a]].$$

Define the value of $f_C(a_1, ..., a_m)$ in $\mathbf{B}$ as any chosen $a \in A$ such that

$$(*) \qquad \mathbf{B} \models (SK(D))[s[x := a]].$$

Since the only free variables in $SK(D)$ are $\{x, y_1, ..., y_m\}$, by lemma 5.3.3, this definition only depends on the values of $y_1, ..., y_m$. Given the interpretation for f_C given in $(*)$, for any assignment s, for $a = (f_C(y_1, ..., y_m))_{\mathbf{B}}[s]$, we have $\mathbf{B} \models (SK(D))[s[x := a]]$. By lemma 5.3.1 and lemma 5.4.1, we have

$$\begin{aligned} &\mathbf{B} \models (SK(D))[s[x := a]] \quad \text{iff} \\ &\mathbf{B} \models (SK(D)[\mathbf{a}/x])[s] \quad \text{iff} \\ &\mathbf{B} \models (SK(D)[f_C(y_1, ..., y_m)/x])[s]. \end{aligned}$$

Hence, $\mathbf{B} \models SK(C)[s]$, as desired. □

We are now ready to prove the following version of the Skolem-Herbrand-Gödel theorem extending Andrews's theorem (Andrews, 1981) to first-order languages with equality.

Theorem 7.6.1 (Skolem-Herbrand-Gödel theorem, after Andrews) Let **L** be a first-order language with or without equality. Given any rectified sentence A, if B is the Skolem form of A, then the following holds:

(a) A is unsatisfiable if and only if some compound instance C of B is unsatisfiable.

(b) Given a ($G1^{nnf}$ or $G1^{nnf}_{=}$) proof of the sequent $A \rightarrow$, a compound instance C of B such that $\neg C$ is provable can be effectively constructed from the proof of $A \rightarrow$.

Proof: Part (b) is lemma 7.6.2. Part (a) is proved as follows.

If A is unsatisfiable, then $\neg A$ is valid and by the completeness theorem, $\neg A$ is provable. Hence, $A \rightarrow$ is provable, and by lemma 7.6.2, a compound instance C of B such that $\neg C$ is provable can be effectively constructed. By soundness, $\neg C$ is valid, and so C is unsatisfiable.

Conversely, assume that some compound instance C of B is unsatisfiable. We prove that for any compound instance C of B, $(B \supset C)$ is valid. This is shown by induction on B.

If B is either an atomic formula or the negation of an atomic formula, $C = B$ and $(B \supset C)$ is valid.

If B is of the form $(D * E)$, where $* \in \{\vee, \wedge\}$, then C is of the form $(K * L)$ where K is a compound instance of D and L is a compound instance of D. By the induction hypothesis, both $(D \supset K)$ and $(E \supset L)$ are valid. But then, $(D * E) \supset (K * L)$ is valid.

If B is of the form $\forall x D$, then C is of the form $H_1 \wedge ... \wedge H_k$, where H_i is a compound instance of $D[t_i/x]$, for some closed terms t_i, $i = 1, ..., k$. By the induction hypothesis, $(D[t_i/x] \supset H_i)$ is valid for $i = 1, ..., k$. But $(\forall x D \supset D[t/x])$ is valid for every closed term t (in fact for every term t free for x in D), as shown in the proof of lemma 5.4.2. Therefore, $(B \supset H_i)$ is valid for $i = 1, ..., k$, which implies that $(B \supset (H_1 \wedge ... \wedge H_k))$ is valid.

This concludes the proof that $(B \supset C)$ is valid. □

Now, since C is unsatisfiable and $(B \supset C)$ is valid, B is also unsatisfiable. By lemma 7.6.3 (part (a)), if A is satisfiable then B is satisfiable. Since B is unsatisfiable, A must be unsatisfiable. □

If the sentence A in NNF is also prenex, observe that a compound instance C of the Skolem form B of A is in fact a conjunction of ground substitution instances of the matrix of B. Hence, we obtain the following useful corollary.

Corollary If A is a prenex sentence in NNF, A is unsatisfiable if and only if a finite conjunction of ground substitution instances of the matrix of the Skolem form B of A is unsatisfiable. □

Remarks: (1) The first remark given at the end of the proof of theorem 7.5.1 regarding the difference between logic without equality and logic with equality also applies here. There is an algorithm for deciding whether a c-instance is unsatisfiable if equality is absent, but in case equality is present, such an algorithm is much less trivial. For details, see Chapter 10.

(2) The version of theorem 7.6.1(a) for languages without equality is due to Andrews (Andrews, 1981). Andrews's proof uses semantic arguments and assumes the result of lemma 7.6.3. Instead of using lemma 7.6.2, assuming that every compound instance of B is satisfiable, Andrews constructs a model of B using the compactness theorem. His proof is more concise than ours, but there is more to the theorem, as revealed by part (b). Indeed, there is a constructive aspect to this theorem reflected in the part of our proof using lemma 7.6.2, which is not brought to light by Andrews's semantic method.

Actually, we have not pushed the constructive approach as far as we could, since we did not show how a proof of $A \rightarrow$ can be reconstructed from a proof of a compound instance $C \rightarrow$. This last construction appears to be feasible, but we have not worked out the technical details, which seem very tedious. Instead, we have proved a version of the Skolem-Herbrand-Gödel theorem using the easy semantic argument that consists of showing that $(B \supset C)$ is valid for every compound instance of B, and part (a) of lemma 7.6.3.

7.6.6 Comparison of Herbrand and Skolem-Herbrand-Gödel Theorems

We now wish to discuss briefly the differences between Herbrand's original theorem and the Skolem-Herbrand-Gödel theorem.

First, Herbrand's theorem deals with provability whereas the Skolem-Herbrand-Gödel deals with unsatisfiability (or validity). Herbrand's theorem is also a deeper result, whose proof is harder, but it yields more information. Roughly speaking, Herbrand's theorem asserts the following:

Herbrand's original theorem:

(1) If a formula A is provable in a formal system Q_H defined by Herbrand, then a Herbrand disjunction H and its proof can be obtained constructively via primitive recursive functions from the proof of A.

(2) From a Herbrand disjunction H and its proof, a proof of A in Q_H can be obtained constructively via a primitive recursive function.

The concept of a primitive recursive function is covered in some detail in Section 7.7, and in the following discussion, we shall content ourselves with an informal definition. Roughly speaking, a primitive recursive function is a function over the natural numbers whose rate of growth is reasonably well behaved. The rate of growth of each primitive recursive function is uniformly bounded, in the sense that no primitive recursive function can grow faster than a certain given function (which itself is not primitive recursive). The class of primitive recursive function contains some simple functions, called the base functions, and is closed under two operations: composition and primitive recursion (see Section 7.7). Primitive recursion is a certain constrained type of recursion, and this is why the rate of growth of the primitive recursive functions is not arbitrary.

The fact that in Herbrand's theorem proofs can be obtained constructively and with reasonable complexity via primitive recursive functions, is a very essential part of the theorem. This last point is well illustrated by the history of the theorem, discussed extensively by Goldfarb, in Herbrand, 1971.

Herbrand's original version of the theorem was sometimes difficult to follow and its proof contained errors. As a matter of fact, Herbrand's original statement of the theorem did not refer to the concept of a primitive recursive function. Denton and Dreben were able to repair the defective proofs, and they realized the fact that the constructions are primitive recursive (see Note G and and Note H, in Herbrand, 1971). In his thesis, Herbrand mistakenly claimed simpler functions. It is also interesting to know that Herbrand did not accept the concept of validity because it is an infinitistic concept, and that this is the reason he gave an argument that is entirely proof-theoretic. However, Herbrand had an intuitive sense of the semantic contents of his theorem. As mentioned by Goldfarb in his introduction to Herbrand, 1971:

"Herbrand intends the notion of expansion to furnish more, namely a finitistic surrogate for the model-theoretic notion of infinite satisfiability."

We close this discussion with a final remark showing the central role occupied by Herbrand's theorem. First, observe that it is possible to prove the Skolem-Herbrand-Gödel theorem without appealing to the completeness theorem (see problem 7.6.11). Then, the following hold:

(1) Herbrand's theorem together with the Skolem-Herbrand-Gödel theorem implies the completeness theorem.

(2) Herbrand's theorem together with the completeness theorem implies the Skolem-Herbrand-Gödel theorem (see problem 7.6.15).

Of course, such proofs are a bit of an overkill, but we are merely illustrating the depth of Herbrand's theorem.

The version of Herbrand's theorem that we have presented in theorem 7.5.1 (and in the part in lemma 7.6.2) has the constructive nature of Herbrand's original theorem. What has not been shown is the primitive recursive nature of the functions yielding on the one hand the Herbrand disjunction H and its proof from the prenex sequent $\rightarrow \Gamma$, and on the other hand the proof of $\rightarrow \Gamma$ from a proof of the Herbrand disjunction H. However, we have shown the recursive nature of these functions. In view of the above discussion regarding Herbrand's original version of the theorem, it would be surprising if these functions were not primitive recursive.

For details on Herbrand's original theorem, the interested reader is referred to Herbrand, 1971; Van Heijenoort, 1967; and Joyner's Ph.D thesis (*Automatic theorem Proving and The Decision Problem*, Ph.D thesis, W. H. Joyner, Harvard University, 1974), which contains a Herbrand-like theorem for the resolution method.

PROBLEMS

7.6.1. Show that the Skolem form of the negation of the formula

$$A = \exists x \forall y [(P(x) \equiv P(y)) \supset (\exists x P(x) \equiv \forall y P(y))]$$

is the formula

$$C = \forall y [(\neg P(c) \lor P(y)) \land (\neg P(y) \lor P(c))] \land \\ [(P(d) \land \neg P(e)) \lor (\forall z P(z) \land \forall x \neg P(x))].$$

Using Skolem-Herbrand-Gödel's theorem, prove that A is valid.

7.6.2. Convert the negation of the following formula to Skolem form:

$$\neg \exists y \forall z (P(z, y) \equiv \neg \exists x (P(z, x) \land P(x, z))).$$

7.6.3. Convert the negation of the following formula to Skolem form:

$$\exists x \exists y \forall z ([P(x, y) \supset (P(y, z) \land P(z, z))] \land \\ [(P(x, y) \land Q(x, y)) \supset (Q(x, z) \land Q(z, z))]).$$

7.6.4. Write a computer program for converting a formulae in NNF to SKolem normal form, incorporating the optimization suggested in problem 7.5.2.

7.6.5. Fill in the missing details in the proof of lemma 7.6.2.

7.6.6. Prove the following fact: A c-instance of $(C[F[t/x]]/E] * D)$ is a c-instance of $(C * D)$.

7.6.7. Use the Skolem-Herbrand-Gödel theorem to show that the formula of problem 7.6.2 is valid.

7.6.8. Use the Skolem-Herbrand-Gödel theorem to show that the formula of problem 7.6.3 is valid.

7.6.9. Use lemma 7.6.3 to prove that a set of sentences is satisfiable iff the set of their Skolem forms is satisfiable. (Assume that for any two distinct sentences, the sets of Skolem symbols are disjoint.)

7.6.10. Let $\mathbf{L}$ be a first-order language without equality. A *free structure* $\mathbf{H}$ is an $\mathbf{L}$-structure with domain the set $H_{\mathbf{L}}$ of all closed $\mathbf{L}$-terms, and whose interpretation function satisfies the following property:

(i) For every function symbol f of rank n, for all t_1,...,$t_n \in H$,

$$f_{\mathbf{H}}(t_1, ..., t_n) = ft_1...t_n, \text{ and}$$

(ii) For every constant symbol c,

$$c_{\mathbf{H}} = c.$$

(a) Prove that the Skolem form B of a sentence A in NNF is satisfiable iff B is satisfiable in a free structure.

(b) Prove that a set of sentences is satisfiable iff it is satisfiable in a free structure. (Use problem 7.6.9.)

(c) Prove that (b) and (c) are false for languages with equality, but that they are true if we replace free structure by quotient of a free structure.

$*$ **7.6.11.** Let $\mathbf{L}$ be a first-order language without equality. Given a set S of $\mathbf{L}$-sentences in NNF, let H be the free structure built up from the set of function and constant symbols occurring in the Skolem forms of the sentences in S. The *Herbrand expansion* $E(C, H)$ of a quantifier-free formula C in NNF over the free universe H, is the set of all formulae of the form $C[t_1/x_1, ..., t_m/x_m]$, where $\{x_1, ..., x_m\}$ is the set of free variables in C, and t_1,...,$t_m \in H$. For each sentence $A \in S$, let $E(B^*, H)$ be the Herbrand expansion of the quantifier-free formula B^* obtained by deleting the universal quantifiers in the Skolem form B of A. The Herbrand expansion $E(A, H)$ of the sentence A is equal to $E(B^*, H)$.

(a) Prove that S is satisfiable iff the union of all the expansions $E(B^*, H)$ defined above is satisfiable. (Use problem 7.6.10.)

(b) Using the compactness theorem for propositional logic, prove the following version of the Skolem-Herbrand-Gödel theorem:

A sentence A is unsatisfiable iff some finite conjunction of quantifier-free formulae in the Herbrand Expansion $E(A, H)$ is unsatisfiable.

(c) Use (a) to prove the Löwenheim-Skolem theorem.

$*$ **7.6.12.** Let $\mathbf{L}$ be a first-order language without equality. Let $\mathbf{L}'$ be an expansion of $\mathbf{L}$ obtained by adding function and constant symbols. Let H be the set of all closed $\mathbf{L}$-terms, and H' the set of all closed $\mathbf{L}'$-terms.

Prove that for any sentence A, if the Herbrand expansion $E(A, H)$ is satisfiable, then $E(B^*, H')$ is also satisfiable, where B^* is the quantifier-free formula obtained by deleting the universal quantifiers in the Skolem form B of A.

Hint: Define a function $h : H' \to H$ as follows: Let t_0 be any fixed term in H.

(i) $h(t) = t_0$ if t is either a constant in H' not occurring in B^* or a term of the form $f(t_1, ..., t_k)$ such that f does not occur in B^*.

(ii) $h(t) = t$ if t is a constant occurring in B^*, or $f(h(t_1), .., h(t_n))$ if t is of the form $f(t_1, ..., t_n)$ and f occurs in B^*.

Assume that $E(A, H)$ is satisfiable in a free structure **A**. Expand **A** to an **L**′-structure **B** using the following definition: For every predicate symbol of rank n, for all t_1,...,$t_n \in H'$,

$$\mathbf{B} \models Pt_1...t_n \quad \text{iff} \quad \mathbf{A} \models Ph(t_1)...h(t_n).$$

Prove that $E(B^*, H')$ is satisfied in **B**.

7.6.13. Let **L** be a first-order language without equality. Prove the compactness theorem using problems 7.6.11, 7.6.12, and the compactness theorem for propositional logic.

7.6.14. State and prove a validity version of theorem 7.6.1.

7.6.15. Consider first-order languages without equality. In this problem, assume that Gentzen's original proof of the cut elimination theorem is used to avoid the completeness theorem in proving theorem 7.5.1.

(a) Prove that Herbrand's theorem (theorem 7.5.1) and the Skolem-Herbrand-Gödel theorem proved in problem 7.6.11 (without the completeness theorem) yield the completeness theorem (for prenex formulae).

(b) Prove that Herbrand's theorem (theorem 7.5.1) and the completeness theorem yield the Skolem-Herbrand-Gödel theorem.

∗ 7.7 The Primitive Recursive Functions

First, we discuss informally the notion of computability.

7.7.1 The Concept of Computability

In the discussion at the end of Section 7.6, the concept of a primitive recursive function was mentioned. In this section, we discuss this concept very briefly. For more details on recursive function theory and complexity theory, the reader is referred to Lewis and Papadimitriou, 1981; Davis and Weyuker, 1983; Machtey and Young, 1978; or Rogers, 1967. For excellent surveys, we recommend Enderton and Smorynski's articles in Barwise, 1977.

At the end of the nineteenth century, classical mathematics was shaken by paradoxes and inconsistencies. The famous mathematician Hilbert proposed the following program in order to put mathematics on solid foundations: Formalize mathematics completely, and exploit the finitist nature of proofs to prove the consistency of the formalized theory (that is, the absence of a contradiction) in the theory itself. In 1930, the famous logician Kurt Gödel made a major announcement; Hilbert's consistency program could not be carried out. Indeed, Gödel had proved two incompleteness theorems that showed the impossibility of Hilbert's program. The second theorem roughly states that in any consistent formal theory T containing arithmetic, the sentence asserting the consistency of T is not provable in T.

To prove his theorems, Gödel invented a technique now known as *Gödel-numbering*, in which syntactic objects such as formulae and proofs are encoded as natural numbers. The functions used to perform such encodings are definable in arithmetic, and are in some intuitive sense computable. These functions are the primitive recursive functions. To carry out Hilbert's program, it was also important to understand what is a computable function, since one of the objectives of the program was to check proofs mechanically.

The concern for providing logical foundations for mathematics prompted important and extensive research (initiated in the early thirties) on the topic of computability and undecidability, by Herbrand, Gödel, Church, Rosser, Kleene, Turing, and Post, to name only the pioneers in the field. Summarizing more than 60 years of research in a few lines, the following important and surprising facts (at least at the time of their finding) were discovered:

Several *models of computations* were proposed by different researchers, and were shown to be equivalent, in the sense that they all define the same class of functions called the *partial recursive functions*.

Among these models are the Turing machine already discussed in Subsection 3.3.5, and the class of partial recursive functions (due to Herbrand, Kleene, Gödel).

The above led to what is usually known as the *Church-Turing thesis*, which states that any "reasonable" definition of the concept of an effectively (or algorithmically) computable function is equivalent to the concept of a partial recursive function.

The Church-Turing thesis cannot be proved because the notion of a reasonable definition of computability is not clearly defined, but most researchers in the field believe it.

The other important theme relevant to our considerations is that of the complexity of computing a function. Indeed, there are computable functions (such as Ackermann's function, see Davis and Weyuker, 1983) that require so much time and space to be computed that they are computable only for very small arguments (may be $n = 0, 1, 2$). For some of these functions, the rate of growth is so "wild" that it is actually beyond imagination. For an entertaining article on this topic, consult Smorynski, 1983.

The class of *primitive recursive functions* is a subclass of the computable functions that, for all practical purposes, contains all the computable functions that one would ever want to compute. It is generally agreed that if an algorithm corresponds to a computable function that is not primitive recursive, it is not a simple algorithm. Herbrand's theorem says that the algorithms that yields a Herbrand's disjunction from a proof of the input formula and its converse are primitive recursive functions. Hence, from a computational point of view, the transformation given by Herbrand's theorem are reasonably simple, or at least not too bad.

The primitive recursive functions also play an important role in Gödel's incompleteness results (see Enderton, 1972, or Monk, 1976).

7.7.2 Definition of the Primitive Recursive Functions

In the rest of this section, we are considering functions of the natural numbers. In order to define the class of primitive recursive functions we need to define two operations on functions:

(1) Composition;

(2) Primitive Recursion.

Definition 7.7.1 Given a function f of $m > 0$ arguments and m functions g_1,...,g_m each of $n > 0$ arguments, the *composition*

$$f \circ (g_1, .., g_m)$$

of f and g_1,...,g_m is the function h of n arguments such that, for all $x_1, .., x_n \in$ **N**,

$$h(x_1, ..., x_n) = f(g_1(x_1, ..., x_n), ..., g_m(x_1, ..., x_n)).$$

Primitive recusion is defined as follows.

Definition 7.7.2 Given a function f of n arguments ($n > 0$), and a function g of $n+1$ arguments, the function h of $n+1$ arguments is defined by *primitive recursion* from f and g iff the following holds: For all $x_1, ..., x_n, y \in$ **N**,

$$\begin{aligned} h(x_1, ..., x_n, 0) &= f(x_1, ..., x_n); \\ h(x_1, ..., x_n, y+1) &= g(y, h(x_1, ..., x_n, y), x_1, ..., x_n). \end{aligned}$$

In the special case $n = 0$, let m be any given integer. Then

$$\begin{aligned} h(0) &= m; \\ h(y+1) &= g(y, h(y)). \end{aligned}$$

We also define the base functions.

Definition 7.7.3 The *base functions* are the following functions:

(i) The *successor function* S, such that for all $x \in \mathbf{N}$,

$$S(x) = x + 1;$$

(ii) The *zero function* Z, such that for all $x \in \mathbf{N}$,

$$Z(x) = 0;$$

(iii) The *projections functions*. For every $n > 0$, for every i, $1 \leq i \leq n$, for all $x_1, ..., x_n \in \mathbf{N}$,

$$P_i^n(x_1, ..., x_n) = x_i.$$

The class of primitive recursive functions is defined inductively as follows.

Definition 7.7.4 The class of *primitive recursive functions* is the least class of total functions over $\mathbf{N}$ containing the base functions and closed under composition and primitive recursion.

It can be shown that composition and primitive recursion preserve totality, so the definition makes sense.

7.7.3 The Partial Recursive Functions

In order to define the partial recursive functions, we need one more operation, the operation of minimization.

Definition 7.7.5 Given a function g of $n + 1$ arguments, the function f of $n > 0$ arguments is defined by *minimization* from g iff the following holds: For all $x_1, ..., x_n \in \mathbf{N}$:

(i) $f(x_1, ..., x_n)$ is defined iff there is some $y \in \mathbf{N}$ such that $g(x_1, ..., x_n, z)$ is defined for all $z \leq y$ and $g(x_1, ..., x_n, y) = 0$;

(ii) If $f(x_1, ..., x_n)$ is defined, then $f(x_1, ..., x_n)$ is equal to the least y satisfying (i). In other words,

$$\begin{array}{c} f(x_1, ..., x_n) = y \quad \text{iff} \quad g(x_1, ..., x_n, y) = 0 \quad \text{and} \\ \text{for all } z < y,\ g(x_1, ..., x_n, z) \text{ is defined and nonzero.} \end{array}$$

The condition that $g(x_1, ..., x_n, z)$ is defined for all $z \leq y$ is essential to the definition, since otherwise one could define noncomputable functions. The function f is also denoted by

$$min_y(g(x_1, ..., x_n, y) = 0)$$

(with a small abuse of notation, since the variables $x_1, ..., x_n$ should not be present).

Definition 7.7.6 The class of *partial recursive functions* is the least class of partial functions over $\mathbf{N}$ containing the base functions and closed under composition, primitive recursion, and minimization.

A function is *recursive* iff it is a total partial recursive function.

Obviously, the class of primitive recursive functions is a subclass of the class of recursive functions. Contrary to the other closure operations, if f is obtained by minimization from a total function g, f is not necessarily a total function. For example, if g is the function such that $g(x, y) = x + y + 1$, $min_y(g(x, y) = 0)$ is the partial function undefined everywhere.

It can be shown that there are (total) recursive functions that are not primitive recursive. The following function kown as *Ackermann's function* is such as example: (See example 2.1.1, in Chapter 2.)

EXAMPLE 7.7.1

$$\begin{aligned}A(x, y) = if\ x = 0\ then\ y + 1 \\ else\ if\ y = 0\ then\ A(x - 1, 1) \\ else\ A(x - 1, A(x, y - 1))\end{aligned}$$

A problem A (encoded as a set of natural numbers) is said to be *decidable* iff there is a (total) recursive function h_A such that for all $n \in \mathbf{N}$,

$$n \in A \quad \text{iff} \quad h_A(n) = 1, \text{ otherwise } h_A(n) = 0.$$

A problem A is *partially decidable* iff there is a partial recursive function h_A such that for all $n \in \mathbf{N}$,

$$n \in A \quad \text{iff} \quad h_A(n) = 1, \text{ otherwise either } h_A \text{ is undefined or } h_A(n) = 0.$$

Church's theorem states that the problem of deciding whether a first-order formula is valid is partially decidable, but is not decidable (see Enderton, 1972; Monk, 1976; Lewis and Papadimitriou, 1981).

We conclude with a short list of examples of primitive recursive functions. One of the unpleasant properties of definition 7.7.4 is the rigid format of primitive recursion, which forces one to use projections and composition to permute or drop arguments. Since the purpose of this section is only to give a superficial idea of what the primitive recursive functions are, we will ignore these details in the definitions given below. We leave as a (tedious) exercise to the reader the task to rewrite the definitions below so that they fit definition 7.7.4.

7.7.4 Some Primitive Recursive Functions

EXAMPLE 7.7.2

(a) Addition:

$$\begin{aligned} x + 0 &= P_1^1(x) \\ x + (y + 1) &= S(x + y) \end{aligned}$$

(b) Multiplication:

$$\begin{aligned} x * 0 &= Z(x) \\ x * (y + 1) &= (x * y) + x \end{aligned}$$

(c) Exponentiation:

$$\begin{aligned} exp(x, 0) &= 1 \\ exp(x, y + 1) &= exp(x, y) * x \end{aligned}$$

(d) Factorial:

$$\begin{aligned} fact(0) &= 1 \\ fact(y + 1) &= fact(y) * S(y) \end{aligned}$$

(e) Iterated exponentiation:

$$\begin{aligned} ex(0) &= 0 \\ ex(y + 1) &= exp(2, ex(y)) \end{aligned}$$

(f) N-th prime number:

$$pr(x) = \text{ the } x\text{-th prime number.}$$

PROBLEMS

7.7.1. Prove that composition and primitive recursion applied to total functions yield total functions.

7.7.2. Prove that the functions given in example 7.7.2 are primitive recursive.

Notes and Suggestions for Further Reading

Gentzen's cut elimination theorem, Gentzen's sharpened Hauptsatz, and Herbrand's theorem are perhaps the most fundamental proof-theoretic results of first-order logic.

Gentzen's theorems show that there are normal forms for proofs, and reduce the provability of a first-order sentence to the provability of a quantifier-free formula. Similarly, Herbrand's theorem provides a deep characterization of the notion of provability, and a reduction to the quantifier-free case.

Interestingly, Herbrand's proof (Herbrand, 1971) and Gentzen's proofs (Szabo, 1969) are significantly different. It is often felt that Herbrand's arguments are difficult to follow, whereas Gentzen's arguments are crystal clear.

Since Gentzen's Hauptsatz requires formulae to be prenex, but Herbrand's original theorem holds for arbitrary formulae, it is often said that Herbrand's theorem is more general than Gentzen's sharpened Hauptsatz. However, using Kleene's method (presented in Section 7.5) and the method partially developed in Section 7.4, it appears that this is not the case.

For more on Gentzen systems, the reader is referred to Szabo, 1969; Takeuti, 1975; Kleene, 1952; Smullyan, 1968; Prawitz, 1965; and Schwichtenberg's article in Barwise, 1977. Another interesting application of Herbrand's theorem is its use to find decidable classes of formulae. The reader is referred to Dreben and Goldfarb, 1979. A companion book by Lewis (Lewis, 1979) deals with unsolvable classes of formulae.

We have not explored complexity issues related to Herbrand's theorem, or Gentzen's theorems, but these are interesting. It is known that a proof of a sentence can be transformed into a proof of a disjunction of quantifier-free (ground) instances of that sentence, and that the transformation is primitive recursive, but how complex is the resulting proof?

A result of Statman (Statman, 1979) shows that a significant increase in the length of the proof can occur. It is shown in Statman, 1979, that for a certain set X of universally quantified equations, for each n, there is a closed equation E_n such that E_n is provable from X in a proof of size linear in n, but that for any set Y of ground instances of equations in X such that $Y \rightarrow E_n$ is provable, Y has cardinality at least $ex(n)/2$, where $ex(n)$ is the iterated exponential function defined at the end of Section 7.7. For related considerations, the reader is referred to Statman's article in Barwise, 1977.

Another interesting topic that we have not discussed is the possibility of extending Herbrand's theorem to higher-order logic. Such a generalization is investigated in Miller, 1984.