(Exercises 10–12). Conditions such as those of recurrence and acceptability considered in §4 can, however, be used to guarantee the consistency of Comp(P) and to produce a completeness theorem for SLDNF–refutations with respect to the semantics given by Comp(P). (Again we refer the reader to Lloyd [1987, 5.4].)

Exercises

1. Show that no selection rule that always chooses the same literal from each goal clause can be fair.
(Hint: Consider the program P with three clauses:

$$(1)\ r :\!-\ p, q. \qquad (2)\ p :\!-\ p. \qquad (3)\ q :\!-\ q.\ ).$$

2. Describe a fair generalized selection rule and prove that it is fair.

(Hint: Always choose the first literal to appear in the proof so far that has not yet been chosen.)

3. Complete the proof of Lemma 6.8.

4. Verify that the relation $\equiv$ defined in the proof of Theorem 6.10 is an equivalence relation.

5. Verify that the equality axioms (1)–(6) are satisfied in the set M defined in the proof of Theorem 6.10 when "=" is interpreted as true equality of equivalence classes.

6. Prove that no set of sentences of predicate logic can imply axiom (7) of CWA. (Hint: Use the compactness theorem.)

7. Prove that the unique model for CWA(P) for a PROLOG program P is the minimal Herbrand model for P.

8. Prove that the minimal Herbrand model for a PROLOG program P is also a model of Comp(P).

9. Give a counterexample to the generalization of Theorem 6.10 to general goal clauses. (Hint: Write a short program and choose a general goal such that every attempted SLDNF–refutation flounders.)

10. Give an example of a general program P such that Comp(P) (and hence P) is satisfiable but CWA(P) is not.

11. Give an example of a general program P such that CWA(P) (and hence P) is satisfiable but Comp(P) is not.

12. Give an example of a satisfiable general program P such that neither Comp(P) nor CWA is satisfiable.

7 Negation and Nonmonotonic Logic

The general procedure of implementing negation as failure described in the last section is both useful and important. Nonetheless, it violates one of the most basic tenets of mathematical reasoning. In mathematical reasoning (and indeed in all the systems we consider elsewhere in this book) a conclusion drawn from a set of premises can be also be drawn from any larger set of premises. More information or axioms cannot invalidate deductions already made. This property of monotonicity of inferences is basic to standard mathematical reasoning, yet it is violated by many real life procedures as well as by the negation as failure rule.

In the absence of evidence to the contrary, we typically take consistency with the rest of our general belief system to provide grounds for a belief. The classic example concerns Tweety the bird. At some stage in the development of our knowledge we observe and learn about various birds. Based on this information we conclude that birds fly. One day we are told about Tweety the bird and naturally assume that he can fly. When we are later introduced to Tweety, we discover that he is a pet ostrich and can no more fly than we can. We reject our previous belief that all birds fly and revise our conclusions about Tweety. We now face the world with a new set of beliefs from which we continue to make deductions until new evidence once again proves our beliefs false. Such a process is typical of the growth of knowledge in almost all subjects except mathematics. Beliefs and conclusions are often based on a lack of evidence to the contrary.

A similar approach is embodied in the notion of negation as failure. If we have no evidence to the contrary (i.e., a deduction of L), we assume that L is false. This procedure clearly embodies a nonmonotonic system of reasoning. Minsky [1975, 5.5] was the first to propose such systems and beginning with McCarthy's study of circumscription [1980, 5.5] various researchers have proposed and studied a large number of nonmonotonic systems which have been suggested by various problems in computer science and AI. To list just a few: Hintikka's theory of multiple believers, Doyle's truth maintenance system, Reiter's default logic and Moore's autoepistemic logic as well as various versions of negation as failure in extensions of PROLOG by Apt, Clark and others.

We now briefly present a new approach to an abstract view of nonmonotonic systems as given in Marek, Nerode and Remmel [1990, 5.5]. It seems to capture the common content of many of the systems mentioned. The literature has dealt primarily with the propositional case and we restrict ourselves to it as well. For negation in PROLOG, this means that we are always looking at the set of ground instances of a given program in the appropriate Herbrand universe. After describing the general system we connect it to one interesting way of picking out a distinguished Herbrand model that captures many aspects of negation in PROLOG (although it is not precisely the same as the negation as failure rules of §6): the stable model semantics of Gelfond and Lifschitz [1988, 5.4].

We present the idea of nonmonotonic systems in the form of rules of inference such as resolution or the one given for classical monotonic logic in I.7. In such