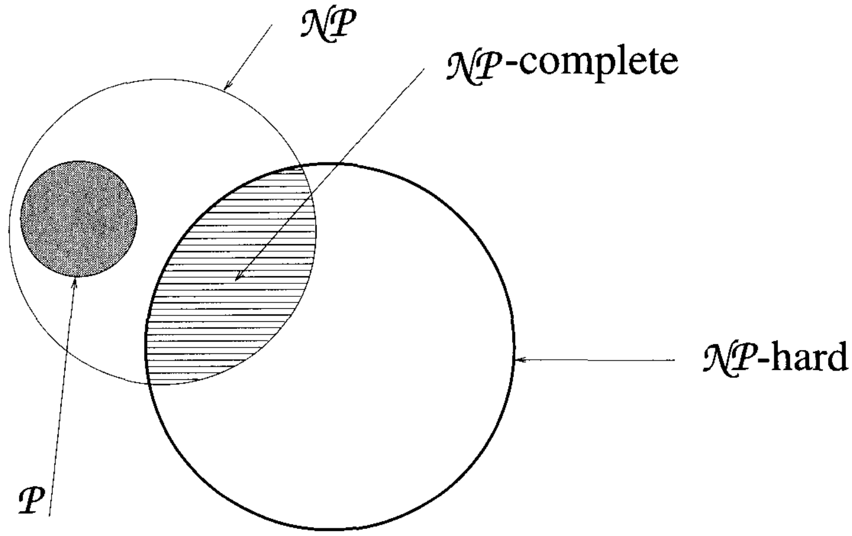

Figure 11.2 Commonly believed relationship among $\mathcal{P}$, $\mathcal{NP}$, $\mathcal{NP}$-complete, and $\mathcal{NP}$-hard problems

It is easy to see that there are $\mathcal{NP}$-hard problems that are not $\mathcal{NP}$-complete. Only a decision problem can be $\mathcal{NP}$-complete. However, an optimization problem may be $\mathcal{NP}$-hard. Furthermore if L_1 is a decision problem and L_2 an optimization problem, it is quite possible that $L_1 \propto L_2$. One can trivially show that the knapsack decision problem reduces to the knapsack optimization problem. For the clique problem one can easily show that the clique decision problem reduces to the clique optimization problem. In fact, one can also show that these optimization problems reduce to their corresponding decision problems (see the exercises). Yet, optimization problems cannot be $\mathcal{NP}$-complete whereas decision problems can. There also exist $\mathcal{NP}$-hard decision problems that are not $\mathcal{NP}$-complete. Figure 11.2 shows the relationship among these classes.

Example 11.10 As an extreme example of an $\mathcal{NP}$-hard decision problem that is not $\mathcal{NP}$-complete consider the halting problem for deterministic algorithms. The *halting problem* is to determine for an arbitrary deterministic algorithm A and an input I whether algorithm A with input I ever terminates (or enters an infinite loop). It is well known that this problem is undecidable. Hence, there exists no algorithm (of any complexity) to solve this problem. So, it clearly cannot be in $\mathcal{NP}$. To show satisfiability $\propto$ the halting problem, simply construct an algorithm A whose input is a propositional formula X. If X has n variables, then A tries out all 2^n possible truth assignments and verifies whether X is satisfiable. If it is, then A stops. If it is not, then A enters an infinite loop. Hence, A halts on input X if and only