

2

Algorithms

In this chapter we shall study:

- fundamental principles of algorithms: problems that can or cannot be solved by algorithms, properties of algorithms themselves, and notation for algorithms (Section 2.1)
- analysis of algorithms to determine their time and space efficiency (Section 2.2)
- the notion of complexity of algorithms (Section 2.3)
- recursive algorithms and their complexity (Section 2.4).

2.1 Principles of algorithms

In Section 1.1 we encountered a variety of algorithms. In this section we briefly discuss some fundamental issues concerned with problems, algorithms, and notation.

Problems

Concerning problems, we can state the following principles:

- An algorithm must be designed to solve a stated *problem*, which is a well-defined task that has to be performed.
- The problem must be *solvable* by an algorithm.

We have already (in Section 1.1) encountered a number of problems that can be solved by algorithms. We can also pose some problems that are *not* solvable by algorithms. To say that a problem is unsolvable does not just mean that an algorithm has *not yet* been found to solve it. It means that such an algorithm can *never* be found. A human might eventually solve the problem, but only by applying insight and creativity, not by following a step-by-step procedure; moreover, there can be no guarantee that a solution will be found. Here is an example of a problem that is unsolvable by an algorithm.