The Role of Algorithms in Computing

What will we study?

- Look at some classical algorithms on different kinds of problems
- How to design an algorithm
- How to show that an algorithm works correctly
- How to analyze the performance of an algorithm

1.1 Algorithms

- Algorithm: Any well-defined computational procedure that takes some value, or set of values, as input and produces some value, or set of values, as output.
- *Or*; Algorithm: A method of solving a problem, using a sequence of well-defined steps
- Example: Sorting problem
- Input: A sequence of *n* numbers $< a_1, a_2, \ldots, a_n >$
- Output: A permutation $< a_1', a_2', \ldots, a_n' >$

of the input sequence such that $a_1' \leq a_2' \leq \ldots \leq a_n'$

Instance of a problem

- An ***instance of a problem*** consists of all inputs needed to compute a solution to the problem.
- An algorithm is said to be ***correct*** if for every input instance, it halts with the correct output.
- A correct algorithm ***solves*** the given computational problem. An incorrect algorithm might not halt at all on some input instance, or it might halt with other than the desired answer.

What kind of problem are solved by algorithms?

- The Human Genome Project
- The Internet applications
- Electronic commerce with public-key cryptography and digital signatures
- Manufacturing and other commercial settings

1.2 Algorithms as a technology

- Efficiency:
  - Different algorithms solve the same problem often differ noticeably in their efficiency.
  - These differences can be much more significant than difference due to hardware and software
- For example, in Chapter 2 we will see that *insertion sort* takes time roughly equal to $c_1 n^2$ (c_1 is constant) to sort n items. But, merge sort takes time roughly equal to $c_2 n \lg n$ (c_2 is constant).

1.2 Algorithms as a technology

- For example, assume a faster computer A (10^{10} instructions/sec) running insertion sort against a slower computer B (10^7 instructions/sec) running merge sort.
- Suppose that c_1=2，c_2=50 and $n = 10^7$.
  - the execution time of computer A is $2(10^7)^2/\ 10^{10}$ instructions/sec = 20,000 seconds
  - the execution time of computer B is $50 \cdot 10^7 \lg 10^7 / 10^7$ instructions/sec = 1,163 seconds

Homework

- Problem 1-1
- Due: 9/24