School of Computing and Information Systems
COMP90038 Algorithms and Complexity Tutorial Week 4

19–23 August 2019

Plan

We provide more exercises than we can cover in the tutorial, so that you have more to practice on in your own time.

The exercises

15. For each of the following pairs f, g, determine whether $f(n) \in O(g(n))$, or $g(n) \in O(f(n))$, or both:

    (a) $f(n) = (n^2 + 1 - n^2)/2$ and $g(n) = 2n$
    (b) $f(n) = n^2 + n\sqrt{n}$ and $g(n) = n^2 + n$
    (c) $f(n) = n \log n$ and $g(n) = \frac{n}{4}\sqrt{n}$
    (d) $f(n) = n + \log n$ and $g(n) = \sqrt{n}$
    (e) $f(n) = 4n \log n + n$ and $g(n) = (n^2 - n)/2$
    (f) $f(n) = (\log n)^2$ and $g(n) = 2 + \log n$

16. Show the steps of selection sort, when given the keys S, O, R, T, X, A, M, P, L, E.

17. One possible way of representing a polynomial

$$p(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0$$

    is as an array A of length $n + 1$, with $A[i]$ holding the coefficient a_i.

    (a) Design a brute-force algorithm for computing the value of $p(x)$ at a given point x. Express this as a function PEVAL(A, n, x) where A is the array of coefficients, n is the degree of the polynomial, and x is the point for which we want the value of p.
    (b) If your algorithm is $\Theta(n^2)$, try to find a linear algorithm.
    (c) Is it possible to find an algorithm that solves the problem in sub-linear time?

18. Trace the brute-force string search algorithm on the following input: The path p is '`needle`', and the text t is '`there_need_not_be_any`'. How many comparisons (successful and unsuccessful) are made?

19. Assume we have a text consisting of one million zeros. For each of these patterns, determine how many character comparisons the brute-force string matching algorithm will make:

(a) `010001` (b) `000101` (c) `011101`

20. Give an example of a text of length n and a pattern of length m, which together constitute a worst-case scenario for the brute-force string matching algorithm. How many character comparisons, as a function of n and m, will be made for the worst-case example. What is the value of m (the length of the pattern) that maximises this function? i.e. What is the worst case pattern length?

21. The *assignment problem* asks how to best assign n jobs to n contractors who have put in bids for each job. An instance of this problem is an $n \times n$ *cost matrix* C, with $C[i, j]$ specifying what it will cost to have contractor i do job j. The aim is to minimise the total cost. More formally, we want to find a permutation $\langle j_1, j_2, \ldots j_n \rangle$ of $\langle 1, 2, \ldots, n \rangle$ such that $\sum_{i=1}^{n} C[i, j_i]$ is minimized. Use brute force to solve the following instance:

	Job 1	**Job 2**	**Job 3**	**Job 4**
Contractor 1	9	2	7	8
Contractor 2	6	4	3	7
Contractor 3	5	8	1	8
Contractor 4	7	6	9	4

22. Give an instance of the assignment problem for which the smallest item $C[i, j]$ of its cost matrix is not included in its solution.