Report on "Introduction to Algorithms and Data Structures"

January 28–February 1, 2019
Ryuhei Uehara (`uehara@jaist.ac.jp`)

Do not forget to write your name, student ID, problems, and answers on your report. Choose any problems that make 60 in total and answer them in English. (If you choose more, I'll take from better scores.) Submit the final report to Ms. Yu May Paing (`yumaypaing@uit.edu.mm`) up to February 8 (Friday). Checked reports will be returned to you.

Problem 1 (10 pts): Suppose that the array $a[0], a[1], \ldots, a[n-1]$ consists of n real numbers. We like to compute the function $f(x) = a[0] + a[1]x + a[2]x^2 + \cdots + a[n-1]x^{n-1}$. Consider the following two algorithms:

1. Following the definition, compute $a[0]+a[1]\times x+a[2]\times x\times x+a[3]\times x\times x\times x+\cdots+a[n-1]\times x\times\cdots\times x$ step by step.
2. Compute after the following modification: $a[0] + x \times (a[1] + x \times (a[2] + x \times (a[3] + x \times (a[4] + \cdots + x \times (a[n-2] + x \times a[n-1])))))$

Evaluate the number of summation and multiplication operations respectively as functions of n, and discuss which is a better way.

Problem 2 (10 pts): Prove the following according to the definition:

$$\begin{aligned} 26n^2 + n + 2019 &\in O(n^3) \\ 26n^2 + n + 2019 &\notin O(n) \end{aligned}$$

Problem 3 (10 pts): In an example of the implementation of quick sort (qsort), we use $x = a[(i+j)/2]$ as a pivot. Show an example of data that gives the worst case for this way of choosing pivot.

Problem 4 (10 pts): In the merge sort, you have to merge two arrays, say, $a[i,j]$ and $a[k,\ell]$. Describe the merge process and evaluate the running time of your algorithm.

Problem 5 (10 pts): Fibonatti number $F(n)$ is defined by $F(0) = F(1) = 1$, and $F(i) = F(i-1) + F(i-2)$ for $i > 1$. Then show an algorithm that computes $F(n)$ for a given n using dynamic programming technique. Show the running time of your algorithm.

Problem 6 (20 pts): You want to shuffle the data which is in $a[0], a[1], \ldots, a[n-1]$ by randomization. You can use a random generator function `random`(k) that returns an integer i with $0 \le i < k$ uniformly at random. Then, show a shuffle algorithm for $a[]$. That is, the algorithm outputs each possible permutations of $a[]$ with the same probability. Evaluate the running time of your algorithm. (Hint: there are several ways, but there exists a simple $O(n)$ time algorithm, which is bit similar to bubble sort.)

Problem 7 (20 pts): Let $a[0], a[1], \ldots, a[n-1]$ be the array of data. Then the partition problem is defined as follows: For the given input array $a[]$, determine if you can partition data into two subsets that makes the same value. That is, when $S = \sum_{i=0}^{n-1} a[i]$, determine if there is a subset I of $\{0, 1, \ldots, n-1\}$ such that $\sum_{i \in I} a[i] = S/2$. It is intractable in general. Now, suppose that you know that $0 \leq a[i] \leq D$ for all indices i and some positive integer D. Then there is a polynomial time algorithm that solves the partition problem. Show the algorithm, and prove that it runs in polynomial time.