CSE 331: Introduction to Algorithm Analysis and Design Fall 2010

Homework 3
Due Friday, October 1, 2010 by 1:15pm in class

IMPORTANT: Please submit each problem separately, i.e. each problem should begin on a new page and only the pages for one problem should be stapled together. Failure to do so might result in some problem(s) not being graded.
For general homework policies and our suggestions, please see the policy document.
For this homework, you can assume that addition, subtraction, multiplication and division of two numbers can be done in $O(1)$ time. (Side question: Is this assumption justified?)

---

1. (40 points) You have been assigned the task of designing an algorithm for the following task. Someone has built an array of the person numbers of all the n students enrolled in 331 this fall. In particular, you have no information about the order in which the person numbers are stored. The only operation that you are allowed is to be able to query the person number at the ith location of the array (for any $1 \le i \le n$).

   Present a $\Theta(n)$ time algorithm that given any input person number p, will report whether or not p is enrolled in the class. Prove both the $O(n)$ and $\Omega(n)$ bounds on the running time of your algorithm. (You can assume that every access to the database takes 1 step.)

2. ($30 + 15 = 45$ points) Polynomials are formal objects that even though pretty abstract have a lot of applications in practice. For example, the "error correcting code" that allows for CDs and DVDs to play even with scratches at its mathematical core is based on properties of polynomials. In this problem we will consider a fundamental operation on polynomials called *polynomial evaluation*.

   Recall that a polynomial of degree $n-1$ is defined by n coefficients $a_0, \ldots, a_{n-1}$ and is defined as the formal sum in some variable X as follows: $\sum_{i=0}^{n-1} a_i \cdot X^i$. For example, the polynomial corresponding to $a_0 = 1, a_1 = 0.2, a_2 = 100, a_3 = .5$ is $.5X^3 + 100X^2 + .2X + 1$.

   For concreteness, given n real numbers $a_0, \ldots, a_{n-1}$, let us denote the polynomial $\sum_{i=0}^{n-1} a_i \cdot X^i$ by $P_{a_0,\ldots,a_{n-1}}(X)$. Now the evaluation of the polynomial $P_{a_0,\ldots,a_{n-1}}$ at a real number y is obtained from the formal sum by replacing X with y, i.e. $P_{a_0,\ldots,a_{n-1}}(y) = \sum_{i=0}^{n-1} a_i \cdot y^i$. For example, for the polynomial considered above when evaluated at 2 gives

   $$P_{1,.2,100,.5}(2) = .5 \times 2^3 + 100 \times 2^2 + .2 \times 2 + 1 = 405.4$$

   Consider the following algorithm for polynomial evaluation:

   ```
   /* Input: Real numbers $a_0, \ldots, a_{n-1}$ and $y$ */
   (a) eval $\leftarrow 0$.
   (b) For $i = 0 \cdots n-1$
       i. temp $\leftarrow 1$.
       ii. For $j = 1 \cdots i$
   ```

```
        • temp ← temp · $y$
    iii. eval ← eval + $a_i$ · temp
(c) Return eval
```

Convince yourself that the algorithm above does indeed output $P_{a_0,\ldots,a_{n-1}}(y)$. Now consider the following questions:

- Prove that the algorithm above has a running time of $\Theta(n^2)$.
- The polynomial fairy tells you that there is a faster algorithm for the polynomial evaluation problem than the one above.[1] Design an algorithm with a running time of $O(f(n))$ such that $\lim_{n\to\infty} \frac{f(n)}{n^2} = 0$. You will receive more credit the (asymptotically) smaller your function $f(n)$ is.

3. (15 points) In Q3 on HW 1 we saw that for every n there exists certain inputs to the stable matching problem such that there exists a "dumb" way of choosing free women at each iteration of the while loop in the Gale-Shapley algorithm that forces the while loop to iterate $\Omega(n^2)$ time. It is natural to wonder if for every input there is always a smart way to choose the free women such that for any input there are asymptotically fewer iterations of the while loop.

   Prove or dis-prove the following statement:

   > For every $n \geq 1$ and every possible input to the stable matching problem on n men and n women, there is always a way to choose the free women at the beginning of the while loop in the Gale-Shapley algorithm such that the number of iterations of the while loop is bounded by $g(n)$ such that $\lim_{n\to\infty} \frac{g(n)}{n^2} = 0$.

   (*Note:* If you prove the statement above then you have to give an *algorithm* that picks one among the multiple free women who can propose at the beginning of the while loop in the Gale-Shapley algorithm. Other than this change, you are *not* allowed to change the Gale-Shapley algorithm in any way. You have to prove the bound on number of iterations of the while loop with your change above. If you dis-prove the statement then you have to show that for every $n \geq 1$, there exists an input to the stable matching problem such that *no matter* how a free woman is chosen at the beginning of the while loop in the Gale-Shapley algorithm, there will be $\Omega(n^2)$ iterations of the while loop.)

[1] The polynomial fairy never lies.