CSE 331 - Summer 2015

Homework 1

Due Thursday, June 18, 2015 @ 2:10pm

IMPORTANT: The stated deadline on assignments will be strictly enforced. I will go over solutions at the deadline while the problems are still fresh in your minds and will not accept submissions after the solutions have been presented.

Homework can be submitted at any time before the deadline as either a hard copy or electronically. If submitting electronically, use a recognizable filename (ex. "homework1.pdf") and submit with submit_cse331 or as an email attachment.

---

1. (40 points)

   (a) (20pts) Exercise 1 in Chapter 1: Decide whether the following statement is true or false:

   > In every instance of the Stable Matching Problem, there is a stable matching containing a pair (m, w) such that m is ranked first on the preference list of w and w is ranked first on the preference list of m.

   If you state true then you will have to formally argue why the statement is correct. If you state false, then you have give a counter-example.

   (b) (20pts) Decide whether the following statement is true or false:

   > For any instance of the Stable Matching Problem, if there exists a perfect matching that is not stable due to m and w wanting to be together, then (m, w) will be in *every* stable matching of that instance. In other words, if (m, w') and (m', w) are both in a perfect matching and m prefers w over w' and w prefers m over m', then (m, w) will be in every stable matching.

   If you state true then you will have to formally argue why the statement is correct. If you state false, then you have give a counter-example.

2. (45 points)

   Order the following running times in ascending order in terms of rates of growth so that if one (let's call it $f(n)$) comes before another (let's call it $g(n)$), then $f(n)$ is $O(g(n))$:

   $$2^{100n},\ n^3,\ \log_{10}(n^4),\ 2^{\log_4 n},\ e^{\pi^{4096}},\ n! + 12^{1000}.$$

   Argue why your order is correct (formal proof is not needed).

   (*Note*: It may help to review the properties of logarithms.)

   Grading: This question will be graded based on how many pairs of functions are in the correct order. This means each of the 15 pairs of functions will be worth 3 points each. For each pair, the points are distributed as follows:

   Correct order: 1 pt

   Correct justification: 2 pts

(*Note*: You don't have to justify each of the 15 pairs seperatly if some of them are implied by transitivity, though having one function in the wrong place could cost a lot of points since it potentially makes 5 pairs incorrect, so be careful.).

(*Hint:* You can utilize a $g'(n)$ such that $g'(n)$ is $\Theta(g(n))$.)

3. (15 points)

   You are on a battlefield where the enemy ranks have established n bases. A recent wave of spies infiltrated the bases and were able to radio back some information about the enemy forces. They informed you that the enemy has either 1, 2, or 3 platoons of soldiers guarding each base. Moreover, one of the spies lets you know which base is housing the enemy flag! This flag is inexplicably important and capturing it means victory for your side.

   For this mission, you are given charge over a large brigade of soldiers to invade the enemy bases and capture their flag. You can choose how your brigade travels from base to base, though there are a limited number of roads connecting the bases and your soldiers can only travel on these roads. These roads are *one-way* in the sence that going the *wrong* way on any of these roads would give too much of a strategic advantage to the enemy to be a viable option (i.e. you can think of this as a directed graph).

   Design an algorithm that finds a path through the bases for your brigade that minimizes the number of enemy platoons encountered. The starting point for your brigade is fixed, and the path should end at the enemy flag. You only have to capture this flag, not visit every base. Once your soldiers arrive at the flag the battle will be over, so don't worry about an extraction plan.

   There are n bases and m roads connecting the bases.

   You must design an algorithm to find the a path in $O(m+n)$ time that minimizes the number of enemy platoons encountered. An $O(m \log n)$ time algorithm would be ok, but an $O(m+n)$ time algorithm is ideal. For full credit, an $O(m+n)$ runtime algorithm must be given. No need for a formal proof of correctness, but you must give an argument for the runtime.