CPSC 320 Midterm #1

February 4, 2015

Reminders (**but do not miss the problem also on this page!**):

- $f(n) \in O(g(n))$ (big-O, that is) exactly when there is a positive real constant c and positive integer n_0 such that for all integers $n \geq n_0$, $f(n) \leq c \cdot g(n)$.
- $f(n) \in o(g(n))$ (little-o, that is) exactly when for all positive real constants c, there is a positive integer n_0 such that for all integers $n \geq n_0$, $f(n) \leq c \cdot g(n)$.
- $f(n) \in \Omega(g(n))$ exactly when $g(n) \in O(f(n))$.
- $f(n) \in \omega(g(n))$ exactly when $g(n) \in o(f(n))$.
- $f(n) \in \Theta(g(n))$ exactly when $f(n) \in O(g(n))$ and $f(n) \in \Omega(g(n))$.

1 A Capital Idea [11 marks]

1. In each row below, circle the correct statement if we know that for all positive integers n, $f(n) < g(n)$. **[8 marks]**

   WARNING: one version of the exam had $f(n) > g(n)$ instead.

$f(n) \in O(g(n))$	$f(n) \notin O(g(n))$	$f(n)$ may or may not be in $O(g(n))$
$f(n) \in \Omega(g(n))$	$f(n) \notin \Omega(g(n))$	$f(n)$ may or may not be in $\Omega(g(n))$
$f(n) \in o(g(n))$	$f(n) \notin o(g(n))$	$f(n)$ may or may not be in $o(g(n))$
$f(n) \in \omega(g(n))$	$f(n) \notin \omega(g(n))$	$f(n)$ may or may not be in $\omega(g(n))$

2. Consider the following pseudocode:

```
Given an array A with length n:

Let TestCount = 0
For each index i from 1 to the length of A:
  For each index j from 1 to the length of A:
    If UnknownTest(A, A[i], A[j]) returns true:
      Increment TestCount
Halt and return TestCount
```

   If you're given no further information about `UnknownTest`, what can you say about the **asymptotic worst-case runtime** of this function? **Briefly** justify your answer. **[3 marks]**

2 Demi-Glace [15 marks]

The minimum spanning tree problem becomes somewhat strange in the presence of negative edge weights. Imagine, for example, that you are a telecommunications company creating a communications network by connecting particular cities with fiber-optic cable. You want to ensure that all cities are connected by some path (i.e., that you've created a spanning tree). There is a cost to laying the cable, but some pairs of cities are also willing to pay you to do the job; so, the **net** cost of a particular connection may be positive, zero, or even negative.

1. An **apparently unrelated** problem: We have a graph G with 25 nodes and 150 edges. We compute a spanning tree T of G with total cost 100 on its edges. We then add 2 to the cost of **every edge** in G. Explain why the new cost of T is 148. *Hint:* Draw a small example to gain insight! **[3 marks]**

2. Back to the problem above: Assuming that you insist on producing a fiber-optic network representing a minimum spanning **tree**, give a reduction from this problem to the problem of computing a minimum spanning tree over a graph with strictly **positive** edge weights. **[6 marks]**

   Remember: a reduction is two algorithms, not just one. *Hint:* the previous problem is **not** unrelated.

3. Finish this proof that your reduction—paired with an optimal solution to the positive edge weights MST problem—produces a **minimum** spanning tree. (Note: you may assume that your reduction produces a spanning tree, just not necessarily a **minimum** spanning tree.) **[6 marks]**

   Proof: Assume for contradiction that my reduction (paired with the optimal solution to the positive edge weight MST problem) produces a spanning tree T with cost c_T in the original graph but some other spanning tree U exists with cost $c_U < c_T$ in the original graph.

   Note: You're free to continue as you like from there, but if you want to also use the following, just circle it: "T's cost in the graph produced by my reduction is c'_T, and U's cost in the graph produced by my reduction is c'_U. We can compute c'_T and c'_U as follows..."

3 Greedy (Yet Plausible) Deniability [18 marks]

You're solving the interval scheduling problem except **minimizing** the number of jobs performed rather than maximizing it. In particular, we define a *conflict set* to be the set of all jobs that conflict with a particular job. Your solution should minimize the number of jobs performed while still performing exactly one job from each conflict set.

UNECESSARY FLAVOR TEXT: Your boss has just given you a list of jobs to perform. Each job has a start time and an end time. You can never do more than one job at a time. You're kind of tired; so, you'd like to do as few jobs as possible, but you can't just do **nothing** or you'll get fired. So, you want to find a list of the smallest number of jobs you can do so that every **other** job conflicts with (has times that overlap) at least one of the jobs you **are** doing.

1. Here is a greedy strategy that does **not** always choose the smallest number of jobs: (1) Sort the jobs in order by increasing **start** time. (2) Repeat until there are no jobs left: choose to do the first job left and then cross out it and all jobs with which it conflicts.

   Now, explain which jobs are in the conflict set of the job j with the first start time and where those jobs appear in the array that is sorted by start time. **[2 marks]**

2. Draw an example with **four jobs** that shows that this broken greedy strategy can succeed. Indicate what the strategy selects. **[1 mark]**

   WARNING: One version of the exam reversed the order of this question and the next.

3. Draw an example with **four jobs** that shows that this broken greedy strategy can fail. Indicate (1) what the strategy selects and (2) what the optimal solution is. **[2 marks]**

4. Someone proposes a **new** greedy algorithm. They have **already** proven: "For any problem instance, an arbitrary optimal solution $\mathcal{O}$ **must** include at least one job c in common with the solution $\mathcal{G}$ created by their greedy algorithm."

   Here is a theorem that might be part of their proof of correctness: "Consider a new problem instance that is the same as the original one except that it excludes everything in c's conflict set. We now show that $\mathcal{O}$ with c removed (i.e., $\mathcal{O} - \{c\}$) is an optimal solution to this new problem instance."

   Finish these steps on the way to proving this theorem:

   (a) First, we show that $\mathcal{O} - \{c\}$ does not include two conflicting jobs. We know $\mathcal{O}$ itself does not include two conflicting jobs, and... **[2 marks]**

   (b) Next, we show that $\mathcal{O} - \{c\}$ includes at least one job from each conflict set. Assume for contradiction that $\mathcal{O} - \{c\}$ does not include a job from the conflict set of some job in the new instance. That job also appears in the original instance; so, ... **[5 marks]**

   (c) Finally, we show tht $\mathcal{O} - \{c\}$ is optimal for the new problem instance. Assume for contradiction that it is not. Then, some other solution $\mathcal{O}'$ exists that uses fewer jobs to solve the new instance, but... **[6 marks]**

4 Marriage Counselling [6 marks]

In this problem, we consider the Gale-Shapley algorithm with men proposing. For each statement, circle **one** answer to indicate whether the statement is **always** true, **never** true, or **sometimes** true (i.e., true for some instances but not for others).

Note: in some cases we restrict attention to just certain types of instances, in which case we're asking whether the statement is always, never, or sometimes true for instances **of that type**.

WARNING: one version of the exam asked slightly different questions below.

1. The last proposal made is to a woman who was previously engaged.

   always true **never** true **sometimes** true

2. Each man makes only one proposal.

   always true **never** true **sometimes** true

3. For any instance in which two women w_1 and w_2 both most prefer one man m, the one that m prefers **less** of these women does **not** marry m.

   always true **never** true **sometimes** true

5 Pairs of Apples and Oranges [10 marks]

For each of the following, indicate the most restrictive true answer of $f(n) \in O(g(n))$, $f(n) \in \Theta(g(n))$, and $f(n) \in \Omega(g(n))$. **NOTE:** there are only three options here (only the "big" bounds, not the "little" ones).

WARNING: one version of the exam reversed the left/right order of the answers below.

$\frac{\lg n}{\log n}$	$\in$ ____ (	2^{1000}	)
$n^3 - 2n^2 + 7$	$\in$ ____ (	$5n^2 + 53$	)
$\frac{n^2}{\lg n}$	$\in$ ____ (	$n \lg n$	)
$(n!)^2$	$\in$ ____ (	$(2n)!$	)
$(\lg 4)^n$	$\in$ ____ (	$(\lg 3)^n$	)

6 Bonus [Up to 7 Bonus Marks]

Bonus marks add to your exam total and also to your course bonus total. What course bonus points are worth still isn't clear, however! **WARNING**: These questions are too hard for their point values. We are free to mark these questions harshly. Finish the rest of the exam before attempting these questions. Do not **taunt** these questions.

1. In the practice exam problems, we found that the independent set of a graph G with n vertices where the maximum degree of any vertex is $d_{\max}$ is lower-bounded by $\lceil \frac{n}{d_{\max}+1} \rceil$. Give an algorithm that, given an n and a $d_{\max}$, constructs a graph with the given parameters whose largest independent set has exactly $\lceil \frac{n}{d_{\max}+1} \rceil$ nodes. (I.e., show that this lower-bound is tight.) You must clearly explain why the graph produced has the appropriate sized independent set. **[2 bonus marks]**

2. The **minimal** interval scheduling problem above suggests the following greedy approach: "Sort the jobs by start time and repeat until there are no jobs left: Of the jobs that conflict with the first job, choose the one with the last finish time and cross out all jobs that conflict with it."

   Prove or disprove that this strategy is optimal. **[2 bonus marks]**

3. In this problem you'll prove a tight, non-asymptotic upper bound on the number of iterations of the loop in G-S in terms of the number of men n in the input. (1) Give and prove the upper bound. (2) Give an algorithm to produce an instance of size n that achieves that bound and show that it does so, which establishes that the bound is tight. **[3 bonus marks]**

   Note: we will give 1 bonus mark for the first part for a **tight** bound, even if you do not prove that it's tight (i.e., finish the second part).

This page intentionally left (almost) blank.
If you write answers here, you must CLEARLY indicate on this page what question they belong with AND on the problem's page that you have answers here.

This page intentionally left (almost) blank.
If you write answers here, you must CLEARLY indicate on this page what question they belong with AND on the problem's page that you have answers here.