CS3230 : Design and Analysis of Algorithms (Spring 2015)

Homework Set #3

[Lectures 5 to 7]

OUT: 12-Feb-2015 **DUE:** Thursday, 12-Mar-2015, before 12:00noon

IMPORTANT: Write your NAME, Matric No, Tut. Gp (G1-G7) in your Answer Sheet.

Solve all the S-Problems in every homework set.

- Start each of the problems on a *separate* page.
- Make sure your name and matric number is on each sheet.
- Write legibly. If we cannot read what you write, we cannot give points. In case you CANNOT write legibly, please type out your answers and print out hard copy.
- To hand the homework in, **staple them together** and drop them into the CS3230 dropbox outside my office (COM1 03-17) before the due date and time.

IMPORTANT:

You are advised to start on the homework *early*. For some problems, you need to let it incubate in your head before the solution will come to you. Also, start early so that there is time to consult the teaching staff during office hours. Some students might need some pointers regarding writing the proofs, others may need pointers on writing out the algorithm (idea, example, pseudo-code), while others will need to understand *more deeply* the material covered in class before they apply them to solving the homework problems. Use the office hours for these purposes!

It is always a good idea to email the teaching staff before going for office hours or if you need to schedule other timings to meet. Do it in advance.

HOW TO "Give an Algorithm":

When asked to "give an algorithm" to solve a problem, your write-up should *NOT* be just the code. (*This will receive very low marks.*) Instead, it should be a short essay. The first paragraph should summarize the problem and what your results are. The body of the essay should consist of the following:

- A description of the algorithm *in English* and, if helpful, pseudo-code.
- One *worked example or diagram* to show more precisely how your algorithm works.
- A *proof* (or indication) of the correctness of the algorithm
- An *analysis* of the running time of the algorithm.

Remember, your goal is to communicate. Full credit will be given only to correct solutions which are described clearly. Convoluted and obtuse descriptions will receive low marks.

In CS3230, you learn to develop high-level abstractions when describing algorithms. Try not to speak in ML/AL (machine/assembly language) or "for (j=0; j<n; j++) do". Instead give names to your sets (of objects or things or data structures), talk about Depth-First Search, Binary Search, traverse the graph, sort the set, use a priority queue, etc. You are no longer in CS1010, CS1020, CS2010 or CS2020. Speak with greater sophistication, and at a higher level of abstraction.

(**Note:** Each problem is worth 10 marks. So the problems are NOT equally weighted. So, adjust your strategy accordingly.)

Printed 2/12/2015 1:12 PM

Routine Practice Problems *-- do not turn these in -- but make sure you know how to do them.*

R1. Modification of Exercise 17.1-3 of [CLRS]

A sequence of n operations is performed on a data structure. The *i*th operation costs *i* if *i* is an exact power of 2, and 1 otherwise.

(a) What is the worst case cost per the operation?

(b) Use aggregate analysis to determine the amortized cost per operation?

R2. The clique problem is in NP

A k-clique is a complete graph with k vertices. Given a graph G=(V, E) and integer k, the clique problem asks if G contains a k-clique as its subgraph. Can you show that the clique problem is in NP?

Standard-Problems -- Solve these problems and turn them in by the due-date.

S1. [Bigger Optimal Decision Trees]

In the lecture notes and the book, we presented a decision tree for sorting *any* permutation of 3 numbers $A[1..3]$. The decision tree has 6 leaf nodes. It is an *optimal* decision tree for sorting 3 numbers, meaning that it has *minimum* height. In this problem, we want to design an *optimal* decision tree for sorting any permutation of 4 numbers $A[1..4]$.

(a) What is the *minimum number of leaves* in the decision tree for sorting 4 numbers? Hence, what is the *minimum height* of the decision tree for $A[1..4]$.

(b) Give an optimal decision tree for sorting 4 numbers $A[1..4]$ **where the comparison at the root node of the tree MUST be 1:4** (namely, compare A[1] with A[4] at the root node).

[*Hint on getting optimal decision trees:* When making a decision on which two items to compare at a given node in the decision tree, you should try to balance (as much as possible) the "possible outcomes" in the left and right sub-trees. Both puns intended.]

[*Hmm...* Why 1:4? Is it because Prof. Leong likes the numbers 1 and 4? While that may be true, you should also *be warned* that this could be his *built-in copy-detector* mechanism.]

S2. [Amortized analysis for the set data structure]

We want a data structure to support the following three operations on a set S of integers, in constant $\Theta(1)$ *amortized* time. Namely, we want that *any arbitrary sequence* of n operations to run in $\Theta(n)$ time. (This sequence is any arbitrary sequences of the three operations.)

INSERT(S, x)	: *inserts* x into set S
FIND-MIN(S)	: returns value of *smallest* element in the set S
DELETE-LARGER-HALF(S)	: *deletes the larger* $\lceil \frac{\lvert S \rvert}{2} \rceil$ elements from S.

You are given an *unsorted array* $A[1..N]$ to store the elements in S. (You can assume that size of S will not exceed N.) Using a variable MV to store the current *minimum element* in the set S, it is trivial to implement INSERT(S, x) and MIN(S) in constant $\Theta(1)$ time. The non-trivial part is DELETE-LARGER-HALF(S).

(a) Design algorithms for INSERT(S, x), MIN(S), DELETE-LARGER-HALF(S) that will achieve *constant amortized time* for all three operations. Describe your algorithms for the three operations. (You ***do not need*** to give detailed code; you can freely quote any algorithm covered in the course.)
[*Hint:* Median-find algorithm runs in time ≤ Dn for some fixed constant D.]

(b) Using the *aggregate* method or the *accounting* method, show that the amortized cost of *each* of the three operation is *constant*, namely, $\Theta(1)$.

S3. [P or NP or co-NP]
For the following six decision problems, please state whether they are in P, NP or co-NP? Please give a brief explanation of your answer. (Note that a problem can belong to more than one complexity class.)

Shortest path problem 1: Consider an undirected graph G=(V, E), an integer k, and two nodes s and t. Is the shortest simple path between s and t shorter than k?
Shortest path problem 2: Consider an undirected graph G=(V, E), an integer k, and two nodes s and t. Is the shortest simple path between s and t longer than k?
Euler cycle problem: Consider an undirected graph G=(V, E). Is there a cycle in G that covers every edge in E exactly once?
Hamiltonian cycle problem: Consider an undirected graph G=(V, E). Is there a simple cycle in G that covers every vertex in V exactly once?
Longest path problem 1: Consider an undirected graph G=(V, E), an integer k, and two nodes s and t. Is the longest simple path between s and t shorter than k?
Longest path problem 2: Consider an undirected graph G=(V, E), an integer k, and two nodes s and t. Is the longest simple path between s and t longer than k?