CMPT 307 (Summer 2015)
Final (August 14, 2015)
Total marks = 90
Time: 12 to 3pm

This exam has two parts. You must answer all Part I questions. Answer questions worth 40 **points from Part II section.**

Part I section

1. (15 points) Answer True or False to the following questions and briefly JUSTIFY each answer. A correct answer with no or totally incorrect justification will get you 1 out of the total 3 points.

   (a) Depth First Search (DFS) is a linear time algorithm.
   True, the running time is $O(|V|+|E|)$

   (b) n is $O((logn)^{logn})$.
   True: we know $n = 2^{\log n}$**.**

   (c) Consider the following divide-and-conquer algorithm for finding the minimum spanning tree of a given weighted connected undirected graph G. Partition the graph into two parts by using a cut. Then construct the minimum spanning tree for each part. Then connect the two parts by adding to them the minimum weight cross edge of the cut. The resulting spanning tree is a minimum spanning tree of the whole graph.
   False: the smallest edge connecting S and V-S is an MST edge. There could be more than one between S and V-S. Consider $S = \{(0,0),(0,4)\}$ **and** $V - S = \{(1,0),(1,4)\}$**.**

   (d) Consider a weighted undirected graph $G = (V,E)$ with distinct edge weights. The two smallest cost edges of G are always contained in the minimum spanning tree of G.
   True: If the second smallest edge e is not an mst edge, adding e to the mst will realize a cycle with at least three edges. The cost of one of them must be greater than that of e.

   (e) All Depth First Search (DFS) forests of a directed graph $G = (V,E)$ (for traversal starting at different vertices) will have the same number of trees.
   False: Consider a directed tree with root s. The DFS from s will realize one tree. If you start dfs from any other node, you get more than one trees.

2. (4 points) Let P be a problem whose lower bound is known to be $\Omega(n \log n)$. Let A be an algorithm that solves P. Which subset of he following statements are consistent with this information about the complexity of P?

   (a) A has worst-case time complexity $O(n^2)$.

(b) A has worst-case time complexity $O(n^{3/2})$.

(c) A has worst-case time complexity $O(n)$.

(d) A has worst-case time complexity $\Theta(n^2)$.

(a), (b) and (d) are consistent.

3. (7 points) Consider the problem of evaluating a^b for given integers a and b: we want the whole answer, not modulo a third integer. We know two algorithms for doing this: the iterative algorithm which performs $b-1$ multiplications by a; and the recursive algorithm based on the binary expansion of b. What are the running times of the two algorithms? You can assume that b is a perfect power of 2. You use the fact that the time to multiply an n-bit number by an m-bit number is $O(mn)$ and the size of the product is an $(n+m)$-bit number.

   First algorithm: Total cost: $n^2+(2n)\times n+(3n)\times n+\ldots+((b-1)n)\times n$**.**
   Second algorithm: It is given by the recurrence relation: $T(n)=2T(n/2)+((b/2-1)n)^2$**.**

   Both the approaches give the same running time.

4. (5 points) Give the state of the disjoint-set data structure after the following sequence of operations starting from singleton sets $\{1\},\{2\},\ldots,\{8\}$. Use path compression. In case of ties, always make the lower numbered root point to the higher numbered one.

```
union(1,2), union(3,4), union(5,6), union(7,8), union(1,4),
            union(6,7), union(4,5), find(1)
```

   It is a pointer tree; described in the text.

5. (7 points) Design a 2-3 tree data structure that supports the following operations.

   (a) The elements at the leaves are kept unordered.

   (b) The minimum and maximum values can be found in $O(1)$ time.

   (c) An element can be inserted in $O(\log n)$ time.

   (d) The minimum or maximum element can be deleted in $O(\log n)$ time.

   Give details of the insert and delete operations.

6. (5 points) For each of the methods of divide-and-conquer, greedy algorithm and dynamic programming, determine the properties from the following list that are appropriate for the method.

   (a) Make a choice at each step
   Greedy

(b) Bottom up solution, from smaller to larger subproblems
DC and DP

(c) Solve the subproblem arising after the choice is made
Greedy

(d) Top down solution, problems decrease in size
Greedy, DC

(e) Subproblems are disjoint.
DC

7. (7 points) Consider a set valuable items with weights $w_1, w_2, \ldots, w_n$ and values $v_1, v_2, \ldots, v_n$. You are supposed to fill in a knapsack that holds a total weight W. You are allowed to break an item, if necessary, a broken item retains its fractional value (so, for example, a third of item i has weight $w_i/3$ and value $v_i/3$).

   (a) Describe a $O(n \log n)$ time greedy algorithm to solve this problem.

   (b) Show that your algorithm works.

   (c) (Bonus) (4 points) Improve the running time of your algorithm to linear.

Part II section

8. (8 points) Consider a divide and conquer algorithm A whose running time is $T(n) = 4T(\frac{n}{2}) + \Theta(n \log n)$. In the following you can assume that n is a perfect power of 2.

   (a) Draw the recursion tree of A.

   (b) Determine its height, the number of leaf nodes, and the amount of work done at level i.

   (c) Assume that the structure of the program and its asymptotic performance cannot be changed. As a programmer, you are constrained to improve only the constant factor in program's running time. Where would you put your focus to make your program run faster? Justify. (Your choices for improving the performance are divide (partition) step, merge (combine) step, basis of the recursion.)

9. (8 points) Let $G = (V, E)$ be an undirected unweighted graph. Design breadth-first-search-based algorithms (along with their analyses) to perform the following tasks.

   (a) Is G 2-colorable?

   (b) Find the shortest cycle that contains a specified vertex s.

   (c) Find the shortest cycle in G.

10. (8 points) You are given as input n real numbers $x_1, x_2, \ldots, x_n$. Design an efficient algorithm that uses the minimum number m of unit intervals $[a_i, a_i + 1)(1 \leq i \leq m)$ that cover all the input numbers. A number x_j is covered by an interval $[a_i, a_i + 1)$ if $a_i \leq x_j < a_i + 1$. Formally argue that your algorithm is correct. What is the running time of the algorithm?

11. (8 points) Let $G = (V, A)$ be a directed acyclic graph that has an edge between every pair of vertices and whose vertices are labeled $1, 2, \ldots, n$, where $n = |V|$. To determine the direction of an edge between two vertices in V, you are only allowed to ask a *query*. A query consists of two specified vertices u and v and is answered with:

    "from u to v" if (u, v) is in A, or
    "from v to u" if (v, u) is in A.

    (a) Give an example of a complete directed acyclic graph with 5 vertices.

    (b) What is the topological sort of the graph in the above example?

    (c) Give an upper bound for the number of queries required to find a topological sort of a complete directed acyclic graph G with n vertices. (I am looking for a sub-quadratic upper bound answer.)

12. (8 points)

    (a) Suppose that the weighted graph $G = (V, E)$ is represented as an adjacency matrix. Give a simple implementation of Dijkstra's algorithm to solve the point-to-point shortest path problem for this case that runs in $O(|V|^2)$ time.

    (b) Describe the landmark algorithm discussed in the class.

13. (8 points) Consider a directed graph $G = (V, E)$ where $|V| = n$ and $|E| = m$. Let $r_{ij} > 0$ be the cost of the edge (i, j), $i \neq j$. The cost of a cycle C in G is $\prod_{(i,j) \in C} r_{ij}$. A cycle C is an opportunity cycle if and only if

    $$\prod_{(i,j) \in C} r_{ij} > 1$$

    Formulate this problem as a problem in G' such that G has an opportunity cycle if and only if G' has a negative cycle. Design an algorithm to determine if G has an opportunity cycle. What is the running time?

14. (8 points) Give a dynamic programming algorithm that on input S, where $S = \{s_0 = 0 \leq s_1 \leq \ldots \leq s_n = m\}$ is a finite set of positive integers, determines whether it is possible to place gas stations along an m-mile highway such that:

    (a) A gas station can only be placed at a distance $s_i \in S$ from the start of the highway.

    (b) There must be a gas station at the beginning of the highway ($s_0 = 0$) and at the end of the highway ($s_n = m$).

    (c) The distance between every two consecutive gas stations on the highway is between 15 and 25 miles.

    For example, suppose the input is $\{0, 15, 40, 50, 60\}$. Then your algorithm should output "yes", because we can place gas stations at distances $\{0, 15, 40, 60\}$ from the beginning of the highway.

    However, if the input is $\{0, 25, 30, 55, 70\}$, then your algorithm should output "no", because there is no subset of the distances that satisfies the conditions listed above.

    Remember to analyze the running time of your algorithm.

    Note that this question should be rephrased: should ask for the minimum number of gas stations.