American University of Beirut
MATHEMATICS 256
Advanced Algorithms and Data Structures
Fall 2001-2002

Final Exam

Date: Feb. 1st, 2002 8:00 – 10:00am.
Instructor: Jihad Boulos

Name: .....

ID #: .....

Section #: .....

Name of student to your left: .....

Name of student to your right: .....

This is an open-book, open-note exam. Your exam should have 15 pages (the last 3 are blank, so you can use them if you need additional space for any of the exercises), and there are 8 exercises totaling 100 points. You may use your notes, class handouts, and the main course textbook. You are **NOT** allowed to use any external notes. Your answers should be concise, and when possible should be a list of important points rather than prose. Solve as much problems as you can. I advise you to spend time on understanding the problem and budget your time for solving each problem, or else you will be wasting a lot of time on one problem and will run out of time for other problems. The problems are **NOT** provided with an incremental difficulty, so exercise 7 might be easier than exercise 4, for example.

Wordy and/or irrelevant answers will reduce your score for that problem. Your answers should be the summary of work done on scratch paper that you do not hand in. If I could not read your writing, I will just give a ZERO without bothering myself trying to understand what you are writing. The space allocated for answers should be sufficient for your answers. If not, use additional papers.

Exercise 1 (15 points): Asymptotics and Recurrences

Order these functions in order of increasing Θ-order. When two functions have the same Θ-order, please indicate this explicitly. Please write your calculations on scratch papers (e.g., on the back side of the proceeding page) and provide on this page your answers only.

a) Order the following 5 functions:

(1) $n/\log^4 n$

(2) $n2^{-n}$

(3) $\log\log n$

(4) $\log^2 n$

(5) $\sqrt{n}$

ANSWER:

b) Order the following 4 functions:

(1) $T(n) = n + 5T(n/6)$

(2) $T(n) = 1 + 3T(n/2)$

(3) $T(n) = n + T(n/7) + T(6n/7)$.

(4) $T(n) = n + T(3n/7) + T(4n/7)$.

ANSWER:

Exercise 2 (22 points):

True or False? Circle the correct answer. No explanation required. Each correct answer is worth 2 points, but 2 points will be subtracted for each wrong answer, so answer only if you are reasonably certain.

1. Let G be a directed graph with positive edge weights $W(e)$. Let $W_{max} = max_e\, W(e)$ be the maximum edge weight. Dijkstra's algorithm will find the longest path from the start vertex s to any other vertex v in a graph if we replace all the edge weights $W(e)$ by the new positive weights $1 + W_{max} - W(e)$ before running the algorithm.

ANSWER: TRUE FALSE

2. After we run DFS on a directed graph G, it is possible for a single vertex $v \in G$ to be simultaneously incident on at least one back edge, forward edge, cross edge and tree edge (i.e., all these types of edges together).

ANSWER: TRUE FALSE

3. Let $T(0) = 1$ and $T(n) = 2^{T(n-1)}$ for $n \geq 1$. Then the cost of computing $T(n)$ and printing it out as a binary integer is $\Theta(T(n-1))$.

ANSWER: TRUE FALSE

4. If $T(n) = 49T(n/7) + n^2 \cos\sqrt{n}$, then $T(n) = O(n^2 \log n)$.

ANSWER: TRUE FALSE

5. The path computed by the Bellman-Ford after the 2nd iteration might be the shortest.

ANSWER: TRUE FALSE

6. In a graph G, the sum of the degrees of all the nodes of G is an even number.

ANSWER: TRUE FALSE

7. Let $G(V, E)$ be an undirected graph. Then G has a cycle if and only if it is possible to put directions on all the edges in E so that every vertex in V has an edge pointing to it.

ANSWER: TRUE FALSE

8. Let X be a minimum spanning tree of the weighted undirected graph $G(V, E)$. Let $G'(V', E')$ be another graph defined be augmenting G with a new vertex u and some weighted edges incident on u. In other words, $V' = V \cup \{u\}$ and $E' = E \cup F$ where F is a

set of edges touching u. Then there is always a minimum spanning tree X' of G' such that $X \subset X'$.

ANSWER: TRUE FALSE

9. $e^{c\sqrt{n}}$ is $O(e^{\sqrt{n}})$ for all $c > 0$.

ANSWER: TRUE FALSE

10. Let G be an undirected graph, and let G' be another undirected graph. G' has one vertex for each biconnected component of G (and no other vertices). Two vertices in G' are connected by an edge if and only if the two corresponding biconnected components in G share a vertex. Then G' can have a cycle.

ANSWER: TRUE FALSE

11. Let G be a directed graph. The Bellman-Ford algorithm will find the longest path (if it is finite, or report if it is not) from the start vertex s to any other vertex v in a graph if we negate all the edge weights $W(e)$ before running the algorithm.

ANSWER: TRUE FALSE

Exercise 3 (3 points):

Your friend guesses an integer between 0 and N: You can ask questions like "is the number less than 100?" He will give YES NO answers.
How many questions can your friend force you to ask, if you are a smart person?

ANSWER:

Exercise 4 (8 points):

A B-tree with parameter $t = 64$ has 256 million elements in it. Give the minimum and maximum possible height of the tree. The parameter t is the minimum branching factor for a node (i.e., the minimum degree of a B-tree as it is defined in the book). You may use the approximation 1 million $\approx 2^{20}$. If you do rough calculations you may be off by one either way. This is fine, but show me your logic in computing these maximum and minimum, so that you can get some grades even if the answers are not correct.

ANSWER:

Exercise 5 (10 points):

In a *full* binary tree, each node is either a leaf or has exactly two children (i.e., there are no nodes with just one child). Consider a full binary tree with n leaves and suppose we give each leaf ℓ a *weight* of $1/2^{depth(\ell)}$ (the root is defined to have depth 0).

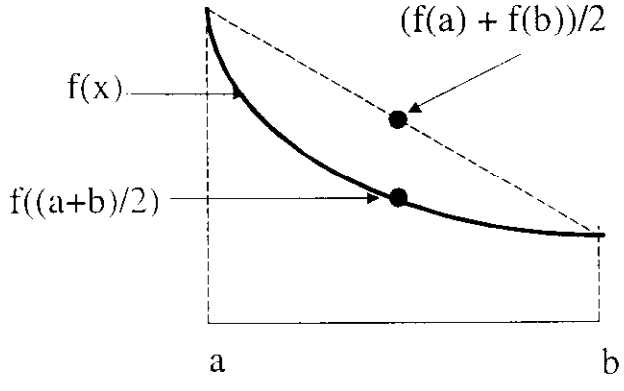
a) What is the sum of the weights of all the leaves?

ANSWER:

b) What is the average weight of all the leaves?

ANSWER:

c) The function $f(x) = 1/2^x$ is "convex". This means that for any set of points S, $average_{x \in S}(f(x)) \geq f(average(S))$. For example,



Use this fact and your answer to question **(b)** to complete the following statement: In a binary tree with n leaves, the average depth of a leaf is at least (circle the largest correct answer):

$\frac{1}{2}\log_2 n$ $\log_2 n$ $2\log_2 n$

ANSWER:

d) The answer to **(c)** implies that for any comparison-based sorting algorithm, the expected number of comparisons needed to sort a random permutation of n inputs is at least (choose the largest correct answer):

$\frac{1}{2}\log_2 n$ $\log_2 n$ $2\log_2 n$ $\frac{1}{2}\log_2(n!)$ $\log_2(n!)$ $2\log_2(n!)$

ANSWER:

Exercise 6 (12 points): Inverting upper triangular matrices

A matrix U is upper triangular if $u_{ij} = 0$ for $i > j$. Consider the following recursive algorithm to compute the inverse of an upper triangular matrix U. We will assume that n is a power of 2. [You need not know the definition of the inverse of a matrix to solve this question.]

Procedure Inv(U)

if $n = 1$ **then** return u_{11}^{-1}

Decompose U into four $n/2 \times n/2$ matrices such that: $U = \begin{pmatrix} A & B \\ 0 & C \end{pmatrix}$

return $\begin{pmatrix} Inv(A) & -Inv(A) \bullet B \bullet Inv(C) \\ 0 & Inv(C) \end{pmatrix}$

The $\bullet$ is matrix multiplication, which we will assume is performed using the "standard" algorithm that takes $O(n^3)$ arithmetic operations to multiply two $n \times n$ matrices.

a) Write a recurrence for the number of arithmetic operations used by procedure Inv() on an $n \times n$ matrix. (The base case is $T(1) = 1$.)

ANSWER:

b) Solve the recurrence to determine the number of arithmetic operations used by procedure Inv() on an $n \times n$ matrix. Write your answer using big-O notation.

ANSWER:

Exercise 7 (15 points): Running to AUB

To get in shape, you have decided to start running to the university. You want a route that goes entirely uphill and then entirely downhill so that you can work up a sweat going uphill and then get a nice wind at the end of your run as you run faster downhill. Your run will start at home and end at the university (for those who live on or near the campus, assume that you will be running around the university) and you have a map detailing the roads with m road segments (any existing road between two intersections) and n intersections. Each road segment has a positive length, and each intersection has a distinct elevation.

a) Assuming that every road segment is either uphill or downhill, give an efficient algorithm to find the shortest route that meets your specifications.

ANSWER:

b) Give an efficient algorithm to solve the problem if some roads may be level (i.e., both intersections at the end of the road segments are at the same elevation) and therefore can be taken at any point.

ANSWER:

Exercise 8 (15 points): Checking the Shortest Path

The single source shortest path algorithm we studied requires time $O((|E| + |V|) \lg |V|)$ to find the minimum distance $SP(v_0, v_i)$ from the source v_0 to each node v_i. *Checking* the answer seems easier than finding it. Give an $O(|V| + |E|)$ time algorithm that, given a directed, connected, and weighted graph (with non-negative weights), and a sequence of distances d_i for each $0 \leq i \leq n - 1$, will verify whether $d_i = SP(v_0, v_i)$, i.e., whether d_i really is the minimum distance from the source to vertex v_i. Argue the correctness of your algorithm (i.e., prove that (a) if the d_i 's are the shortest path values, your algorithm returns TRUE, and (b) if the d_i 's are not correct shortest path values, your algorithm returns FALSE.) Explain why your algorithm is $O(|V| + |E|)$.

ANSWER: