CS 161 Summer 2012 Midterm Exam

July 25, 2012

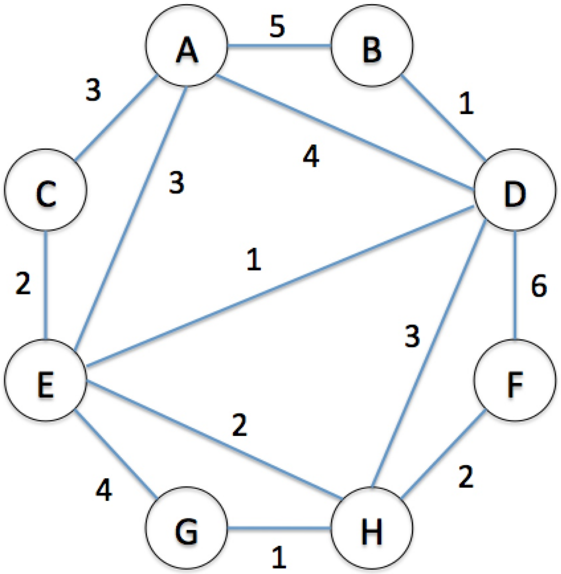
Full Name: ______________________

Stanford ID: ______________________

Instructions: This exam is open book, open notes, but closed computers/cell phones. Please answer all questions in the space provided. Any work outside of the space provided will not be graded. Write clearly and legibly. All statements made must be justified, and **any algorithm must be accompanied by an analysis of running time and a proof of correctness**. Any algorithm or theorem presented in class may be used without proof.

Problem	**Score**	**Out Of**
1		30
2		30
3		30
4		30
5		30
6		50
Total		200

Problem 1 (30 Points). Consider the following graph G:



(a) Suppose we run a DFS on G, where whenever we have a choice of nodes, we always pick the one alphabetically first. In what order are the nodes of G visited?

(b) Suppose we run Dijsktra's algorithm on G starting from A, breaking ties by picking the node that comes alphabetically first. In what order are nodes of G visited?

(c) Suppose we run Kruskal's algorithm on G, breaking ties by choosing the edge with the alphabetically first endpoint (if two nodes both share an alphabetically first endpoint, look at the other endpoint). In what order are edges added?

(d) Suppose we run Prim's algorithm on G, breaking ties the same way as in part (c). In what order are edges added?

Problem 2 (30 Points).

(a) Give a linear-time algorithm for the following task: Given a directed graph G, find a node v such that all other nodes in G are reachable from v, or report that no such v exists.

(b) Give a linear-time algorithm for the following task: Given a directed graph G, find two nodes u and v such that adding the edge (u, v) makes the G strongly-connected, or report that this is impossible.

Problem 3 (30 Points).

(a) Suppose we have a MST T for a graph G, and now we are told that the weight of some edge (u, v) (not necessarily part of T) has decreased. Give a linear-time algorithm for constructing a MST T' of the new graph.

(b) Suppose instead we are told that the weight of (u, v) has increased. Give a linear-time algorithm for constructing a MST T' of the new graph.

Problem 4 (30 Points). Let Γ be the alphabet consisting of characters A, B, C, D, and E. Let the string S be $EABADDCDAB$.

(a) Compute the frequencies f_A, f_B, f_C, f_D, and f_E of each character in the string S. Leaving frequencies as fractions is okay.

(b) Compute the optimal prefix-free encoding of the characters of Γ, and its associated binary tree. In the binary tree, from any node, the edge labeled 0 should point to the child with the alphabetically earliest descendant. For example, if one subtree contains the letters A and C, while the other contains B, the edge labeled 0 should point to the subtree containing A and C, and the edge labeled 1 should point to B.

(c) Compute the encoding of S under the encoding you found in part (b).

Problem 5 (30 Points). Solve the following recurrence relations, giving the tightest asymptotic bound possible. You do not need to show that your bound is tight.

(a) $T(n) = 3T(\lceil n/2 \rceil) + O(n^2)$

(b) $T(n) = 9T(n/3 + 2) + O(n^2)$

(c) $T(n) = T(\lceil \sqrt{n} \rceil) + O(n)$

(d) $T(n) = 5T(\lceil n/4 \rceil + 1) + O(n)$

(e) $T(n) = T(n - 1) + O(n^3)$

Problem 6 (50 Points). Determine whether each of the following statements is true for false, and explain why with a proof or counterexample.

(a) $(\log n)^{\log n} \in \Theta(n^{\log(\log n)})$. **True / False**

(b) If we perform a DFS on a dag, there will never be any cross edges. **True / False**

(c) Dijkstra's algorithm works on graphs with negative edges, provided any node with a negative incoming edge has no outgoing edges. **True / False**

(d) In the Huffman encoding algorithm where all frequencies are distinct, the character with the highest frequency is guaranteed to be one of the characters with the smallest codewords. **True / False**

(e) Every single operation in the disjoint-set data structure is guaranteed to take at most $O(\log^* n)$ time. **True / False**

(f) QuickSort runs in worst-case $O(n \log n)$ time. **True / False**

(g) If a graph G has a unique heaviest edge e, and e is in some MST, then e is in every MST. **True / False**

(h) If a weighted graph has edge lengths that are integers from 1 to 10, then we can compute shortest paths in linear (i.e. $O(|V| + |E|)$) time. **True / False**

(i) After running DFS on a directed graph, the node with the lowest `pre` number will be in a sink strongly connected component. **True / False**

(j) In a sparse graph ($|E| = O(|V|)$), we can compute shortest paths in linear time. **True / False**