CS109B Midterm

Friday, May 1, 2:15 – 3:05 pm

Directions

The exam is *open book*; any written materials may be used. Answer all 8 questions on the exam paper itself. The total number of points is 100. You may make additional comments at the bottoms of pages or at the end if you feel the need to clarify or justify your answers. Do not forget to **sign the pledge** below.

I acknowledge and accept the honor code.

____

Print your name here: ____

1	/10
2	/10
3	/15
4	/15
5	/10
6	/15
7	/10
8	/15
Total	/100

Problem 1: (10 points) An undirected graph G is said to be *tripartite* if we can divide the nodes of G into three disjoint sets X, Y, Z, such that if $\{u, v\}$ is any edge of G, then u and v are in different sets. Equivalently, G is tripartite if and only if it can be colored with three colors; the sets X, Y, and Z are each the set of nodes colored in one of the three colors.

a) K_3, the complete graph of three nodes (i.e., a triangle) is tripartite. Briefly justify this observation.

---

b) K_4, the complete graph of four nodes is not tripartite. Prove this fact.

---

c) Are there any other undirected graphs of four nodes that are not tripartite? ______
Below, either describe such a graph or justify why there is none.

Problem 2: (10 points) A *complete* tripartite graph is an undirected graph G whose nodes can be divided into three disjoint sets, X, Y, and Z. No edge $\{u, v\}$ has both u and v in the same set, and every edge consisting of two nodes from different sets is actually present in G.

a) Under what conditions does G have an Euler circuit? *Hint*: The condition can be expressed in terms of x, y, and z, the numbers of nodes in sets X, Y, and Z, respectively.

---

b) Justify your answer to (a).

Problem 3: (15 points) Let G be a directed graph that is acyclic and has weights (labels) for its arcs. We wish to find for each node u of G the quantity $maxDist(u)$, which is the largest distance from u to any node of G. That is, the "distance" along any path is the sum of the weights of the arcs along that path, and we want to take the maximum over all paths (to anywhere) that start at u. One efficient algorithm to compute $maxDist(u)$ for all nodes u is to begin by finding a topological order $v_1, v_2, \ldots, v_n$ for the nodes of G.

a) Complete the description of this algorithm, below. To receive full credit, your algorithm must be as efficient as possible.

b) Assuming that G is represented by adjacency lists, that n is the number of nodes, and m is the larger of the number of nodes and arcs, what is the running time of your algorithm as a function of n and m? ____________

c) Briefly justify your answer to (b).

Problem 4: (15 points) In the space below draw an automaton (may be nondeterministic) that accepts all and only the strings of 0's and 1's such that each 1 is immediately followed by at least two 0's. Examples of accepted strings are ϵ, 00, and 01001000, while 01010 and 00110000 are not accepted.

Briefly explain what the intuitive meaning of each of your states is; that is, what strings get you to each of the states?

Problem 5: (10 points) While in general, concatenation of regular expressions is not idempotent, give three examples of regular expressions R such that $RR \equiv R$. Your three examples must denote different languages.

Problem 6: (15 points) In the following problem you *must* use the regular-expression-to-automaton conversion algorithm described in Section 10.8 of the text. We shall convert the regular expression $\mathbf{ab^*a}$.

a) Show the expression tree for this regular expression:

b) Convert the expression to an automaton with ϵ-transitions:

c) Eliminate ϵ-transitions from your answer to (b):

Problem 7: (10 points) Write a regular expression for the set of strings of 0's and 1's that have at least one 0 and at least one 1.

Problem 8: (15 points) Let G be a directed graph. Let T be a depth-first search tree for G constructed by starting at some node a (from which we happen to be able to reach all the nodes).† Suppose that $u \to v$ is NOT an arc of G, and let G^+ be G with the addition of the arc $u \to v$. For some arcs $u \to v$, when starting at node a to construct a depth-first search tree for G^+ we can construct the same tree T if we choose the proper order in which to consider the of successors of each node. For other arcs $u \to v$ it is not possible.

a) Under what condition will it be possible to construct the tree T? (*Hint*: The condition can be expressed in terms of the structure of T, rather than G.)

---

b) Briefly explain why it is possible to construct T in this case.

---

c) Under what conditions is it impossible to construct T?

---

d) Briefly explain why it is impossible to construct T in this case.

---

† To anticipate some possible questions: T is a tree consisting of the nodes of G and the tree arcs of G, thought to run from parent to child. The solid arcs in Fig. 9.29 (p. 470) are an example of what T might be. The order of children in T is significant.