# Midterm 1 for CS 170

PRINT your name:

(last) (first)

SIGN your name:

WRITE your section number (e.g., 101):

WRITE your SID:

One page of notes is permitted. No electronic devices, e.g. cell phones and calculators, are permitted. Do all your work on the pages of this examination. If you need more space, you may use the reverse side of the page, but try to use the reverse of the same page where the problem is stated.

You have 50 minutes. The questions are of varying difficulty, so avoid spending too long on any one question.

In all algorithm design problems, you may use high-level pseudocode.

DO NOT TURN THE PAGE UNTIL YOU ARE TOLD TO DO SO.

## 1. True/False [2 points per problem]

For each statement below, say whether it is true or false by circling TRUE or FALSE. You do not need to provide any explanation for your answer.

(a) The topological sort of an arbitrary acylic directed graph can be computed in linear time.

(b) Consider an edge $(u, v)$ in a directed graph. In a depth-first search of this graph, if $u$ has (`pre`, `post`) times equal to (6,9) and $v$ has (`pre`, `post`) times equal to (5,10) then $(u, v)$ must be a back edge.

(c) In a depth-first search of a directed graph, it is possible to have an edge between two vertices $u$ and $v$ with (`pre`, `post`) times equal to (5,10) for $u$ and equal to (8,12) for $v$.

(d) In a depth-first search of an undirected graph, it is possible to have an edge between two vertices $u$ and $v$ with (`pre`, `post`) times equal to (5,10) for $u$ and equal to (15,20) for $v$.

(e) Let $G = (V, E)$ be a weighted directed graph. The Bellman-Ford algorithm will find the *longest* path (if it is finite, or report if no such finite path exists) from the start vertex $s$ to any other vertex $v \in V$ if we negate all the edge weights before running the algorithm.

(f) The paths computed by Bellman-Ford after the first iteration (i.e., after one relaxation of every edge) might be shortest paths.

## 2. True/False with Explanations [4 points per problem]

For each statement below, say whether it is true or false, *and provide a one-sentence and/or one-picture explanation.*

(a) We can find an undirected weighted graph $G = (V, E)$ and a vertex $v \in V$ such that the minimum spanning tree and the shortest path tree rooted at $v$ are disjoint (i.e., they do not have any edges in common).

(b) Suppose $T$ is a shortest paths tree for Dijkstra's algorithm. After adding $c > 0$ to every edge in the graph, $T$ is still a shortest paths tree for the modified graph.

(c) The heaviest edge in a graph cannot belong to a minimum spanning tree.

(d) Let $T$ be a minimum spanning tree of the weighted directed graph $G = (V, E)$. Let $G' = (V', E')$ be the graph obtained by augmenting $G$ with a new vertex $u$ and a set of weighted edges incident on $u$. (I.e., $V' = V \cup \{u\}$ and $E' = E \cup F$ where $F$ is the set of edges touching $u$). Then there is always a minimum spanning tree $T'$ of $G'$ such that $T \subset T'$.

(e) After DFS is run, the node with the smallest `post` number must be in a strongly connected component that is a sink.

### 3. A Bit of Math [3 points for (a), (b), (c) and (d); 5 points for (e)]

For each statement below, say whether it is true or false, *and provide a short justification.*

(a) $e^{cn^2}$ is $O(e^{n^2})$ for all $c > 0$.

(b) $\sum_{i=1}^{\sqrt{n}} i$ is $\Omega(n)$.

(c) The solution to the recurrence $T(n) = 6T(n/2) + n^2$ is $\Theta(n^2)$.

(d) The solution to the recurrence $T(n) = 2T(n/2) + g(n)$ is $\Theta(n)$ for any $g(n) = O(\sqrt{n})$.

(e) $\lceil \log n \rceil !$ is $O(n^2)$.

## 4. Divide and conquer [10 points]

(a) Design a divide-and-conquer algorithm for adding up the numbers in an array of length $n$. Your algorithm should break up the array into two half-arrays, and for simplicity you can assume that $n$ is a power of two.

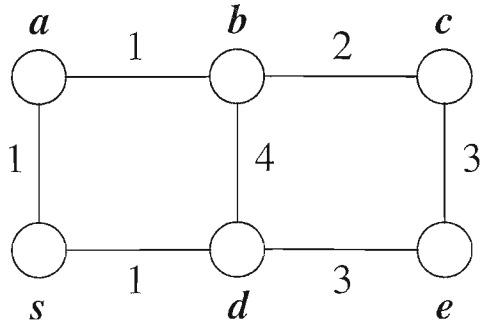
(b) Provide a running-time analysis for your algorithm.

(c) Is there any advantage to breaking up the array into $k$ pieces instead of 2 pieces at each step (in terms of asymptotic run time)?

## 5. DAGs [10 points]

Given an acyclic directed graph $G = (V, E)$, design an algorithm that determines whether it is possible to construct a path that visits each vertex exactly once.

## 6. Dijkstra and Prim [5 points per problem]




(a) In running Prim's algorithm in the graph above, what is the sequence of edges added to the minimum spanning tree? Assume that the algorithm starts with node $s$. (If you have a choice of nodes to pick at any point in the algorithm, pick the lexicographically smaller node).

(b) In running Dijkstra's algorithm from node $s$ in the graph above, which nodes will have their key values updated more than once? (If you have a choice of nodes to pick at any point in the algorithm, pick the lexicographically smaller node).

## 7. Bandwidth Worries [10 points]

Consider a communications network with $n$ nodes, which we model with an undirected graph $G = (V, E)$. For $(i, j) \in E$, let $b_{ij} = b_{ji}$ denote the bandwidth between nodes $i$ and $j$. For a given path between nodes $s$ and $t$, define the *path-bandwidth* as the minimum of the bandwidths along the path. Given fixed $s$ and $t$, design an efficient algorithm to find a path with maximum path-bandwidth between $s$ and $t$. Analyze the running time of your algorithm.

## 8. Bellman-Ford for $S$–$T$ paths [15 points]

Suppose that you have two sets of vertices $S$ and $T$ which are disjoint subsets of the vertex set $V$ of a directed graph. The graph may have negative edge weights. Suppose that you want to find the shortest path that starts at any vertex in $S$ and ends at any vertex in $T$.

Of course, you can solve this problem by running the Bellman-Ford algorithm $|S|$ times. This approach would require $O(mn|S|)$ time. But you can do better. Describe an algorithm that solves this problem in $O(mn)$ time, i.e., the time needed by a single run of the Bellman-Ford algorithm.