

CSE 830: Homework #4

Due: Tuesday, November 10th 2009, 10:20am

1. Is the path between a pair of vertices in a minimum spanning tree necessarily the shortest path between the two vertices in the full graph? Give a proof or a counter-example.
2. In breadth-first and depth-first search, an *undiscovered* node is marked *discovered* when it is first encountered, and marked *completely-explored* when it has been searched. At any given moment, various numbers of nodes can be in any of these states. In all cases below, describe a graph on n vertices with a particular starting vertex v that has the properties described.
 - a. At some point during a breadth-first search, $\Theta(n)$ are simultaneously in the discovered state.
 - b. At some point during a depth-first-search $\Theta(n)$ nodes are simultaneously in the discovered state.
 - c. At some point during a breadth-first-search $\Theta(n)$ nodes are in the undiscovered state, $\Theta(n)$ nodes are in the completely explored state, and only $\Theta(1)$ nodes are discovered.
3. Suppose G is a connected undirected graph. An edge e whose removal disconnects the graph is called a *bridge*. Must every bridge e be an edge in a depth-first search tree of G , or can e be a back edge? Give a proof or a counter-example.
4. An articulation vertex of a graph G is a vertex whose deletion disconnects G . Let G be a graph with n vertices and m edges.
 - a. Give a simple $O(n+m)$ algorithm for finding a vertex of G that is *not* an articulation vertex, i.e. whose deletion does not disconnect G .
 - b. Now expand this algorithm to one that finds a deletion order for the n vertices, also in $O(n+m)$ time, such that no deletion disconnects the graph. (Hint: Think BFS or DFS).
5. The *Maximum Clique* of a graph G is the largest sub-graph where all pairs of vertices in that sub-graph have an edge connecting them. Finding the maximum clique is known to be a hard problem, but sometimes an application requires instances of it to be solved. A brute force approach might test all possible combinations of vertices. Describe optimizations to solve maximum clique as fast as possible. Specifically, be sure to consider backtracking, bounding, the ordering in which the vertices are tested, and polynomial-time simplifications. You do not have to write out the full algorithm, just describe the optimizations.
6. Textbook problems 22.1-5 and 22-3
7. (*Extra credit*) A tournament is a directed graph formed by taking the complete undirected graph and assigning arbitrary directions on the edges, i.e. a graph $G = (V; E)$ such that for all u and v in V , exactly one of $(u; v)$ or $(v; u)$ is in E . Show that every tournament has a Hamiltonian path, that is, a path that visits every vertex exactly once. Give an algorithm to find this path.