

Homework 5

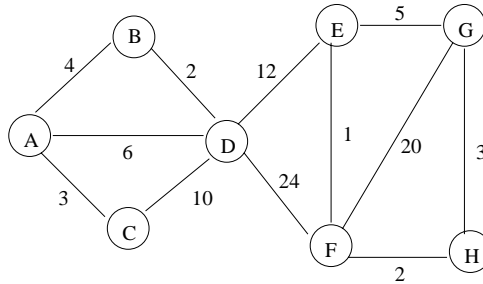
April 13, 1999

Due Date: April 22

Required Problems:

1. (10 pts) In a directed graph, the *in-degree* of a vertex is the number of edges entering it and the *out-degree* of a vertex is the number of edges leaving it. Let n be the number of vertices, and m the number of edges in the graph, let $d_I(v)$ be the in-degree of vertex v , and let $d_O(v)$ be the out-degree of vertex v .
 - (a) Given an **adjacency list representation** of a directed graph, and a specified vertex v , how would you best compute the out-degree of v ? Analyze the time complexity of your algorithm.
 - (b) Given an **adjacency list representation** of a directed graph, and a specified vertex v , how would you best compute the in-degree of v ? Analyze the time complexity of your algorithm.
 - (c) Given an **adjacency matrix** representation of a directed graph, and a specified vertex v , how would you best compute the out-degree of v ? Analyze the time complexity of your algorithm.
 - (d) Given an **adjacency matrix** representation of a directed graph, and a specified vertex v , how would you best compute the in-degree of v ? Analyze the time complexity of your algorithm.
2. (7 pts) Consider the unweighted directed graph given by the following adjacency list:
A: E
B: G C A
C: E
D: B F E
E:
F: C B E A
G: A
 - (a) Run DFS on this graph showing the discovery and finishing time for each vertex.
 - (b) Suppose that each vertex in the graph represents a task and an edge from i to j indicates the precedence constraint that task i must be done before task j . Give an order in which the steps could be sequentially performed (i.e. one step is performed at a time), so that all the precedence constraints are obeyed. You should briefly explain how you obtained your answer.
3. (10 pts) Describe how you could modify Dijkstra's shortest path algorithm so that if there is more than one shortest path from the source s to vertex v , it will select one that has the fewest edges. Argue that your resulting algorithm is correct.
4. (7 pts) Analyze the time complexity for Dijkstra's shortest path algorithm when the priority queue is implemented using a sorted doubly-linked list (assume that the linked list class provides a handle as did the binary heap implementation from Lab 4). Clearly explain your work.

5. (6 pts) Consider the following graph:



- (a) List the edges placed in the minimum spanning tree by Kruskal's MST algorithm in the order in which they are added.
- (b) List the edges placed in the minimum spanning tree by Prim's MST algorithm in the order in which they are added when the source vertex is A .
6. (10 pts) You are given a set of s sticks which are laying on top of each other in some configuration. You have been provided with a class `Stick` that has a method `on` such that for `Sticks a` and `b`, `a.on(b)` returns true exactly when stick `a` is resting directly on stick `b`. A stick may be picked up only if there is no stick resting on it. Design an algorithm to determine whether it is possible to pick up all the sticks and if so provides a legal sequence of stick pickups. Be sure to analyze the time complexity of your algorithm. You do NOT need to analyze any algorithms covered in class but rather can use their known time complexities.

NOTE: Assuming you formulate this as a graph problem from your solution the answers to the following questions should be clear. What do the vertices correspond to? What do the edges correspond to? Is it a directed or undirected graph (and if directed how are the edge directions defined)? Is it a weighted or unweighted graph (and if weighted how are the edge weights defined)?

Extra Credit Problems

- 7*. (5 pts) Suppose that all edge weights in a graph are integers in the range from 1 to W from some constant W . How fast (asymptotically) can you make Prim's algorithm run? (Justify your answer with an algorithm and its analysis).
- 8* (5 pts) *Arbitrage* is the practice by which one takes advantage of discrepancies in currency exchange rates to spend one unit of currency to buy more than one unit of the same currency. For example, suppose that 1 U.S. dollar buys .7 British pounds, 1 British pound buys 9.5 French francs and 1 French franc buys .16 U.S. dollars. Then by converting currencies with an initial investment of 1 U.S. dollar, one can buy $.7 \times 9.5 \times .16 = 1.064$ U.S. dollars.

Describe an algorithm that determines given a set of currencies and exchange rates whether or not there is an arbitrage scheme (and if so finds one). If you formulate this as a graph problem, very clearly describe the graph you form (including the answers to the questions given in problem 6)!

Hint: There is an algorithm, Bellman-Ford (page 532 of the text), which takes as input a weighted directed graph (negative weights are allowed) and either outputs a negative weight cycle or finds the shortest path between each pair of vertices.