

Problem 1: Adjacency List Representation (20 Points)

(a) Let a_1, \dots, a_n be an array of integers in the range $1, \dots, k$. Give an algorithm that sorts the input array and runs in time $O(n+k)$. Remark: You may find helpful to read paragraphs 8.2 and 8.3 of CLRS.

(b) Let $G(V, E)$ and $G'(V, E')$ be two directed graphs given by their adjacency list representations. These graphs are defined on the same set of vertices. We make the assumption that the names of vertices are integers in the range $1, \dots, |V|$. We do not make any assumption about the order in which the vertices appear in the adjacency list representations. Give an $O(|V| + |E| + |E'|)$ time algorithm that determines if G and G' are equivalent. By equivalent we mean that, for each $v \in V$, the sets of vertices incident to v according to G and the set of vertices incident to v in E' are the same. Remark: You may find helpful to read the key to Quiz 2.

Problem 2: Dijkstra's Single Source Shortest Paths (20 Points)

(a) Give a simple example of a directed graph with negative-weight edges for which Dijkstra's algorithm for single source shortest paths produces incorrect answers.

(b) Suppose that we are given a weighted directed graph $G(V, E)$ in which edges that leave the source vertex s may have negative weights, all other edge weights are non-negative, and there are no negative-weight cycles. Argue that Dijkstra's algorithm correctly finds shortest paths in this graph.

(c) From part (a) we saw that, in general, when edges have negative weights Dijkstra's algorithm for single source shortest paths produces incorrect answers. Suppose that we consider the following transformation to solve the single source shortest path problem for the case of arbitrary edge weights. Let $G(V, E)$ be a weighted directed graph and s a specified source. Let w_{\max}^- be the absolute value of the weight of the most negative edge. Let us add w_{\max}^- to the weights of all the edges of G , thus producing a graph G' with non-negative weights of its edges. Now run Dijkstra's algorithm on G' . Does this transformation produce correct single source shortest paths for the original graph G ? If yes, then give an argument. If no, then give a counter-example.

Problem 3: Reliability Paths (20 Points)

Let $G(V, E)$ be a directed graph representing a network and s a specified source vertex. Suppose that each edge has an associated weight p_e which denotes the reliability of the edge: this is the probability that the edge is working properly. For a path e_1, e_2, \dots, e_k the reliability of the path is then defined as the product $\prod_{i=1}^k p_{e_i}$. Give an efficient algorithm to find the most reliable path from s to every other vertex $v \in V$. Argue that your algorithm is correct and analyze its running time.

Problem 4: Negative Cycles and Arbitrage (20 Points)

(a) Let $G(V, E)$ be a directed graph with arbitrary weights on its edges (i.e., positive and negative). Give an efficient algorithm to determine if G has a negative cycle. Argue that your algorithm is correct and analyze its running time.

Arbitrage is the use of discrepancy in currency exchange rates to transform one unit of a currency into more than one unit of the same currency. For example, suppose that 1 U.S. dollar buys .7 British pound, 1 British pound buys 9.5 French francs, and 1 French franc buys .16 U.S. dollar. Then, by converting currencies, a trader can start with 1 U.S. dollar and buy $.7 \times 9.5 \times .16 = 1.064$ U.S. dollars, thus turning a profit of 6.4%.

Suppose that we are given n currencies c_1, c_2, \dots, c_n and $n \times n$ table R of exchange rates, such that one unit of currency c_i buys $R(i, j)$ units of currency c_j .

Give an efficient algorithm to determine whether or not there exists a sequence of currencies $c_{i_1}, c_{i_2}, \dots, c_{i_k}$ such that $R(i_1, i_2) \times R(i_2, i_3) \times \dots \times R(i_{k-1}, i_k) \times R(i_k, i_1) > 1$. Analyze the running time of your algorithm.

Problem 5: Shortest Paths (20 Points)

(a) To get in shape you have decided to start running to work. You want a route that goes entirely uphill and then entirely downhill. Your run will start at home and end at work and you have a map detailing the roads with m road segments and n intersections. Each road segment has a positive length and each intersection has a unique elevation. Assuming that every road segment is either uphill or downhill, give an efficient algorithm to find the shortest route that meets your specifications. You may also assume that your home and your work lie at intersections. Analyze the running time of your algorithm.

(b) Similar to question above. To get in shape you have decided to start running to work. You want a route that goes entirely uphill then remains on the same level and then goes entirely downhill. Your run will start at home and end at work and you have a map detailing the roads with m road segments and n intersections. Each road segment has a positive length and each intersection has a unique elevation. Assuming that every road segment is either uphill or straight or downhill, give an efficient algorithm to find the shortest route that meets your specifications. You may also assume that your home and your work lie at intersections. Analyze the running time of your algorithm.