# COMS 4231: Analysis of Algorithms I, Fall 2021

## Problem Set 5, due Tuesday November 30, 11:59pm on Gradescope. No late days allowed on this homework.

Please follow the homework submission guidelines posted on Courseworks.

- *As usual, for each of the algorithms that you give, include an explanation of how the algorithm works and show its correctness.*
- *Make your algorithms as efficient as you can, state their running time, and justify why they have the claimed running time. All time bounds below refer to worst-case complexity, unless specified otherwise.*

**Problem 1**. [21 points] An undirected graph $G=(N,E)$ is called *bipartite* if its set $N$ of nodes can be partitioned into two subsets $N_1, N_2$ ($N_1 \cap N_2 = \emptyset$, $N_1 \cup N_2 = N$) so that every edge connects a node of $N_1$ with a node of $N_2$.

a. Prove that if a graph contains a cycle of odd length then it is not bipartite.

b. Give a O($n$+$e$)-time algorithm that determines whether a given graph is bipartite, where $n$ is the number of nodes and $e$ is the number of edges; the graph is given by its adjacency list representation.

If the graph is bipartite, then the algorithm should compute a bipartition of the nodes according to the above definition.

If the graph is not bipartite then the algorithm should output a cycle of odd length.

c. We are given a set of *non-equality* constraints of the form $x_i \neq x_j$ over a set of Boolean variables $x_1, x_2, ..., x_n$. We wish to determine if there is an assignment of Boolean values 0,1 to the variables that satisfies all the constraints, and compute such a satisfying assignment if there is one. Show that this problem can be solved in time O($n$+$m$), where $n$ is the number of variables and $m$ is the number of constraints.

**Problem 2**. [22 points] Do Problem 22-2, parts a-d in CLRS (page 622) on articulation points.

(Do for yourself the remaining parts also (e-h), but you do not have to turn them in.)

**Problem 3**. [22 points] We are given a set $V$ of $n$ variables $\{x_1, x_2, \ldots, x_n\}$ and a set $C$ of $m$ weak and strict inequalities between the variables, i.e., inequalities of the form $x_i \leq x_j$ or $x_i < x_j$. The set $C$ of inequalities is called *consistent* over the positive integers $Z^+$ iff there is an assignment of positive integer values to the variables that satisfies all the inequalities.

For example, the set $\{x_1 \leq x_3,\ x_2 < x_1\}$ is consistent, whereas $\{x_1 \leq x_3,\ x_2 < x_1,\ x_3 < x_2\}$ is not consistent.

a. Given a set $C$ of $m$ inequalities between $n$ variables as above, let $G$ be the directed graph with node set $V=\{x_1, x_2, \ldots, x_n\}$ and edge set $E = \{(x_i, x_j) \mid C \text{ contains } x_i \le x_j \text{ or } x_i < x_j\}$, i.e., $G$ has one node for every variable and one edge for every inequality of $C$. Show that $C$ is not consistent if and only if the graph $G$ contains a cycle where some edge $(x_i, x_j)$ of the cycle corresponds to a strict inequality $x_i < x_j$ of the set $C$.

b. Give an O($n$+$m$)-time algorithm to determine whether a given set $C$ of inequalities is consistent over the positive integers.

c. If the set of inequalities has a solution over the positive integers, then it has a unique minimum solution, i.e. a solution in which every variable has the minimum value among all possible solutions. For example the minimum solution for the set $\{ x_1 \le x_3,\ x_2 < x_1 \}$ is $x_1$=2, $x_2$=1, $x_3$=2. Give an O($n$+$m$)-time algorithm to compute the minimum solution of a given consistent set $C$.

(*Hint:* You may find the decomposition of a directed graph $G$ into strongly connected components useful in this problem.)

**Problem 4.** [20 points] We are given a weighted undirected graph G=(N,E) with weights w: E→R on its edges, and a minimum spanning tree $T$ of (G,w). Both G and $T$ are given by adjacency list representations.

a. Suppose that we increase the weight of an edge ($x$,$y$) of $T$ from w($x$,$y$) to w'($x$,$y$). Give an O($n$+$e$)-time algorithm that computes the minimum spanning tree of the modified weighted graph, where $n$ is the number of nodes and $e$ the number of edges of G.

b. Suppose that we decrease the weight of an edge ($u$,$v$) that is not in $T$ from w($u$,$v$) to w'($u$,$v$). Give an O($n$)-time algorithm that computes the minimum spanning tree of the modified weighted graph.

**Problem 5.** [15 points] A weighted graph may have many different shortest (minimum-weight) paths between two nodes. In this case we may prefer among them one that has the minimum number of edges. Modify Dijkstra's algorithm to solve the following problem:

*Input*: A weighted directed graph G=(N,E) with positive weights w: E→$R_+$ on its edges, and two nodes $s$, $t$.

*Output:* A minimum-weight path from $s$ to $t$ that contains as few edges as possible.

**Exercises for your practice (do not turn in):**

*Practice the graph algorithms that we learn on some graphs of your choice.*

For example, draw an arbitrary directed or undirected graph, pick a source node and apply Breadth-First Search. Compute the BFS tree, the distances, and the partition of nodes into layers. Does the tree depend on the order in which nodes appear in the adjacency lists? (Recall that the adjacency list of a node contains the adjacent nodes in arbitrary order.) How about the distances and the layers?

Similarly with Depth-First Search. Compute the DFS tree, the discovery and finish times of the nodes, classify the edges (tree, forward, back, cross). Compute the strongly connected components for a directed graph. For example, do exercises 22.3-2 and 22.5-2.

Draw a weighted undirected graph, and apply Kruskal's and Prim's algorithm to it.