CMPSC 465 Data Structures & Algorithms
Spring 2024 Mehrdad Mahdavi and David Koslicki

HW 6

Due February 29, 10:00 pm

Instructions: You are encouraged to solve the problem sets on your own, or in groups of up to five people, but you must write your solutions strictly by yourself. You must explicitly acknowledge in your write-up all your collaborators, as well as any books, papers, web pages, etc. you got ideas from.

Formatting: Each problem should begin on a new page. Each page should be clearly labeled with the problem number. The pages of your homework submissions must be in order. You risk receiving no credit for it if you do not adhere to these guidelines.

Late homework will not be accepted. Please, do not ask for extensions since we will provide solutions shortly after the due date. Remember that we will drop your lowest two scores.

This homework is due Thursday, February 8, 10:00 pm electronically. You need to submit it via Gradescope (Class code XX7RVV). Please ask on Campuswire about any details concerning Gradescope and formatting.

For each algorithm question, explain your algorithm and analyze its correctness and running time. Pseudocode is not required, but you may include it if you feel it makes your written explanation more clear.

1. (20 pts.) **Forest Trail Network.** A park ranger provides you with a map of a forest trail network, represented as an undirected tree $T = (V, E)$ (in adjacency list format), where each node corresponds to a specific location (e.g., a cafe) along the trail and edges denote connections between locations. Furthermore, the ranger designates a specific starting point $r \in V$ where all hikers commence their journey. It's crucial for the ranger to efficiently determine whether one location can be reached from an "early checkpoint", meaning that the path from the starting point r to the location v in T passes through the early checkpoint u. Additionally, hikers should not visit a location more than once so they do not get bored. How can you preprocess the trail map to allow the ranger to answer checkpoint queries of the form "can v be reached by going through checkpoint u starting from r without visiting locations more than once" in constant $O(1)$ time, with the preprocessing itself taking $O(|V|)$?

2. (20 pts.) **One-way Streets.** The police department in the city of Comtopia, a city with two main districts, the West district and the East district, has made all streets one-way. The mayor contends that there is still a way to drive legally from any intersection in the West part of the city to any other intersection in the West part of the city. Similarly, the mayor contends that there is still a way to drive legally from any intersection in the East part of the city to any other intersection in the East part of the city, but the opposition is not convinced. A computer program is needed to determine whether the mayor is right. However, the city elections are coming up soon, and there is just enough time to run a *linear-time* algorithm.

(a) Formulate this problem graph-theoretically, and explain why it can indeed be solved in linear time (in terms of the number of roads and intersections).

(b) Suppose it now turns out that the mayor's original claim is false. She next claims something weaker: if you start driving from the park in the West district, navigating one-way streets, then no matter where you reach, there is always a way to drive legally back to the West park. Similarly, if you start driving from the park in the East district, navigating one-way streets, then no matter where you reach, there is

always a way to drive legally back to the East park. Formulate this weaker property in a graph-theoretic setting, and carefully show how it too can be checked in linear time.

3. (20 pts.) **Connected Computers.** Provide a linear time algorithm that, given a network of interconnected computers represented as a directed graph $G = (V, E)$, identifies if there exists a central server $s \in V$ from which all other computers can be accessed. Make sure to prove that your algorithm is correct.

4. (20 pts.) **BFS on Directed Graph.** Consider a directed graph $G = (V, E)$. Depth first search allows us to label edges as tree, forward, back and cross edges. We can make similar definitions for breadth first search on a directed graph:

1. A BFS tree edge is an edge that is present in the BFS tree.
2. A BFS forward edge leads from a node to a non-child descendant in the BFS tree.
3. A BFS back edge leads to an ancestor in the BFS tree.
4. All other edges are BFS cross edges.

(a) Explain why it is impossible to have BFS forward edges.

(b) Give an efficient algorithm that classifies all edges in G as BFS tree edges, back edges, or cross edges.

5. (20 pts.) **2SAT.** In the 2SAT problem, you are given a set of clauses, where each clause is the disjunction (OR) of two literals (a literal is a Boolean variable or the negation of a Boolean variable). You are looking for a way to assign a value **true** or **false** to each of the variables so that all clauses are satisfied – that is, there is at least one true literal in each clause. For example, here's an instance of 2SAT with five clauses and four variables:

$$(x_1 \vee \hat{x}_2) \wedge (\hat{x}_1 \vee \hat{x}_3) \wedge (x_1 \vee x_2) \wedge (\hat{x}_3 \vee x_4) \wedge (\hat{x}_1 \vee x_4).$$

This instance has a satisfying assignment: set x_1, x_2, x_3, and x_4 to **true**, **false**, **false**, and **true**, respectively.

Given an instance ϕ of 2SAT with n variables and m clauses, construct a directed graph $G_\phi = (V, E)$ as follows:

- G_ϕ has $2n$ nodes, one for each variable and one for its negation.
- G_ϕ has $2m$ edges: for each clause $(\alpha \vee \beta)$ of ϕ (where α and β are literals), G_ϕ has an edge from the negation of α to β, and one from the negation of β to α.

(a) Show that if G_ϕ has a strongly connected component containing both x and its negation $\hat{x}$ for some variable x, then ϕ has no satisfying assignment.

(b) Now show the converse of (a): namely, that if none of G_ϕ's strongly connected components contain both a literal and its negation, then the instance ϕ must be satisfiable. To prove this, show that the following algorithm results in a satisfying assignment: repeatedly pick a sink strongly connected component of G_ϕ. Assign the value true to all literals in the sink and assign false to their negations, delete all of the nodes in the sink, and repeat. (*Hint:* Consider case analysis on when each literal of a clause is assigned a value.)

(c) Starting from a given instance of 2SAT, use everything discussed above to show that it can be solved in linear time.

6. (0 pts.) **Acknowledgments.** The assignment will receive a 0 if this question is not answered.
(a) If you worked in a group, list the members of the group. Otherwise, write "I did not work in a group."
(b) If you received significant ideas about the HW solutions from anyone not in your group, list their names

here. Otherwise, write “I did not consult with anyone other than my group members.”
(c) List any resources besides the course material that you consulted in order to solve the material. If you did not consult anything, write “I did not consult any non-class materials.”