CPSC 413-Fall 2020

Problem Set 3 — Graphs

Total marks: 40

1. **(20 marks)** Review of undirected graphs, breadth-first search, and depth-first search.

   (a) Let G be a graph whose vertices are the integers 1 through 8, and let the adjacency list of each vertex be given by the table below:

vertex	adjacent vertices
1	$(2, 3, 4)$
2	$(1, 3, 4)$
3	$(1, 2, 4)$
4	$(1, 2, 3, 6)$
5	$(6, 7, 8)$
6	$(4, 5, 7)$
7	$(5, 6, 8)$
8	$(5, 7)$

   Assume that, in a traversal of G, the adjacent vertices of a given vertex are returned in the same order as they are listed in the above table.

   i. **(2 marks)** Give the sequence of vertices of G visited using a DFS traversal starting at vertex 1.

   ii. **(2 marks)** Give the sequence of vertices visited using a BFS traversal starting at vertex 1.

   iii. **(2 marks)** Is G connected? If yes, give a shortest list of edges to remove such that the graph is no longer connected. If no, give a shortest list of edges to add such that the graph is connected.

   iv. **(2 marks)** Does G contain a cycle? Explain your answer.

   v. **(2 marks)** Is G a tree? Explain your answer.

   (b) An undirected graph is *complete* if it contains an edge between every pair of distinct vertices.

   i. **(2 marks)** What does a depth-first search tree of a complete graph look like?

   ii. **(2 marks)** What does a breadth-first search tree of a complete graph look like?

   (c) **(3 marks)** Let G be an undirected graph with n vertices named $\{0, 1, \ldots, n-1\}$ where edges are represented as an adjacency list. Describe a data structure to store the adjacency list such that we can check in $O(\lg n)$ time if two vertices are adjacent. Justify your answer.

   (d) **(3 marks)** Exercise 7 on page 108-109.

2. **(20 marks)** Exercise 2 on page 107. Your solution must include:

   (a) **(3 marks)** a precise specification of the problem

   (b) **(6 marks)** specification of the algorithm (using pseudocode)

   (c) **(6 marks)** a proof that your algorithm is correct (if your algorithm is a modification of a known algorithm, your proof of correctness may assume that the known algorithm is correct and focus on arguing that the adaptations solve the new problem),

   (d) **(5 marks)** a proof that your algorithm runs in time $O(m + n)$.