CS2040S: Data Structures and Algorithms

Discussion Group Problems for Week 11

For: March 27–March 31

Problem 1. DFS/BFS

Problem 1.a. Recall that when performing DFS or BFS, we may keep track of a `parent` pointer that indicates the very first time that a node was visited. Explain why these parent edges form a tree (i.e., why there are no cycles).

Problem 1.b. Do you remember learning about the BFS and DDS algorithm for trees? Turns out, trees are just a type of graph. How does BFS or DFS on a tree relate to BFS or DFS on a graph? Are they the same algorithm? Will the BFS/DFS algorithm for trees work on a graph? Likewise, will the BFS/DFS algorithm for graphs work on a tree? What happens if you run BFS/DFS on a tree but do not start at the root?

Problem 2. Graph components

(Relevant Kattis Problem: https://open.kattis.com/problems/countingstars)

Given an undirected graph $G = (V, E)$ as an adjacency list, give an algorithm to: (i) determine if the graph is connected; (ii) return the number of connected components (CC) in the graph.

Problem 3. Is it a tree?

(Relevant Kattis Problem: https://open.kattis.com/problems/flyingsafely)

Assume you are given a connected graph with n nodes and m edges as an adjacency list. (You are given n but not m; assume each adjacency list is given as a linked list, so you do not have access to its size.)

Give an algorithm to determine whether or not this graph is a *tree*. Recall that a tree is a connected graph with no cycles. Your algorithm should run in $O(n)$; particularly, it should be independent of m. Assume $O(n + m)$ is too slow.

Problem 4. Graph modelling

Here are a bunch of problems. How would you model them as a graph? (Do not worry about solving the actual problem. We have not studied these algorithms. Just think about how you would model it as a graph problem.) Invent some of your own problems that can be modeled as graph problems—the stranger, the better.

Problem 4.a. Imagine you have a population in which some few people are infected with this weird virus. You also have a list of locations that each of the sick people were in during the last 14 days. Determine if any of the sick people ever met.

Problem 4.b. Imagine you have a list of cities. Some pairs of cities are also connected to each other by roads, however, these roads are pay-per-use and one must pay a certain toll (different for each road) to travel on that road! Find the cheapest way to travel from city A to city B.

Problem 4.c. You are given a set of jobs to schedule. Each job j starts at some time s_j an ends at some time t_j. Many of these jobs overlap. You want to efficiently find large collections of non-overlapping jobs so that you can assign each collection to a single server.

Problem 4.d. An English professor complains that students in their class are cheating. The professor suspects that the cheating students are all copying their material from only a few different sources, but does not know where they are copying from. Students that are not cheating, on the other hand, all submit fairly different solutions. How should we catch the cheaters?

Problem 4.e. There are n children and n presents, and each child has a list of presents that they want. How can we assign presents to children so that each child receives a present that they want?

Problem 5. Word games
(Relevant Kattis problem: https://open.kattis.com/problems/sendmoremoney)

Consider the following two puzzles:

- Puzzle 1:

```
   S E N D
+  M O R E
----------
 M O N E Y
```

- Puzzle 2:

```
   F O R T Y
       T E N
+      T E N
------------
   S I X T Y
```

In each of these two puzzles, you can assign a digit (i.e., $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$) to each of the letters in the puzzle such that the given equation holds. (Each digit is only assigned to once letter.) The goal is to solve these puzzles. How should you model and solve these puzzles? What is the running time of your solution? Can you optimize your solution to find the answer more quickly, most of the time?

Problem 5.a. Explain how to model the problem as a graph search problem. What are the nodes? How many nodes are there? What are the edges? Where do you start? What are you looking for?

Problem 5.b. To solve this problem, should you use BFS or DFS? Why? How else can you make it run faster?

Problem 5.c. When does your search finish? Can you optimize the algorithm to minimize the amount of searching?

Problem 6. Good students, bad students
(Relevant Kattis problem: https://open.kattis.com/problems/amanda)

There are good students and bad students[1]. And at the end of every year, we have to divide the students into two piles: G, the good students who will get an A, and B, the bad students who will get an F. (We only give two grades in this class.)

To help with this process, your friendly tutors have each created a set of notecards. Each card contains the names of two students. One of the two names is a good student, and the other is a bad student. Unfortunately, they do not indicate which is which.

Since the notecards come from thirty eight different tutors, it is not immediately certain that the cards are consistent. Maybe one tutor thinks that Humperdink is a good student, while another tutor thinks that Humperdink is a bad student. (And Humperdink may appear on several different cards.) In addition, the tutors do not provide cards for every student.

Assume you can read the names on a card in $O(1)$ time and that there are more good students than bad students.
Devise an algorithm to determine the answers for the following questions:

- Are the notecards consistent, i.e., is there *any* way we can assign students to G and B that is consistent with the cards?
- Are the notecards sufficient? i.e., is there only one or are there more than one ways to assign students to the sets G and B?
- Assuming that the notecards are consistent and sufficient, determine which set each student belongs in.

[1] No, not really. This sort of binary distinction is silly.

Problem 7. Gone viral (*more challenging*)

There are n students in the National University of Singapore. Among them, there are $n-1$ friendships. Note that friendship is a symmetric relation, but it is not necessarily transitive.

Any two people in the National University of Singapore are either directly or indirectly friends. Formally, between any two different people x and y, either x is friends with y or there exists a sequence $q_1, q_2, \ldots, q_k$ such that x is friends with q_1, q_i is friends with q_{i+1} for all $i < k$ and q_k is friends with y.

It was discovered today that **two** people were found to have the flu in the National University of Singapore.

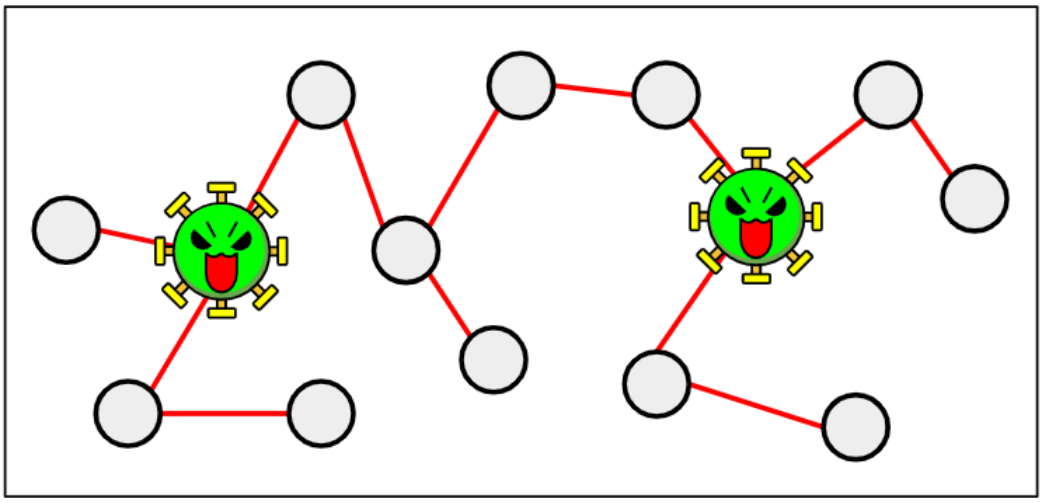

Figure 1: *Gone Viral.* (Matthew Ng Zhen Rui)

Every day, every person can meet with **at most one friend.** When these two people meet, if exactly one of them has the flu, it will be transmitted to the other.

Give an $O(n \log^2 n)$ algorithm to determine the minimum possible number of days before it is possible that *everyone* has the flu.

Hint: First, solve the case where there is only a single person was infected at the start in $O(n \log n)$