CS41 Lab 4

September 21, 2021

In typical labs this semester, you'll be working on a number of problems in groups of 3-4 students. You will not be handing in solutions; the primary purpose of these labs is to have a low-pressure space to discuss algorithm design. However, it will be common to have some overlap between lab exercises and homework sets.

The goal of this lab session is to gain more experience with algorithm design, particularly using graphs. You should focus on the first three problems, and only consider the fourth (much more open-ended) problem if your group is convinced you have solved the first three.

1. **Paths in Graphs.** Recall that a path P in a graph $G = (V, E)$ is a sequence of vertices $P = [v_1, \ldots, v_k]$ such that for all $1 \leq i < k$ there is an edge $(v_i, v_{i+1}) \in E$. P is *simple* if all v_i's are distinct.

   In this problem, you will examine different graphs and consider how many different paths can exist in the graph.

   (a) Describe a graph G_1 on n vertices where between any two distinct vertices there are zero simple paths.

   (b) Describe a graph G_2 on n vertices where between any two distinct vertices there is exactly one simple path.

   (c) Describe a graph G_3 on n vertices where between any two distinct vertices there are exactly two simple paths.

   (d) Describe a graph G_4 on n vertices and two distinct vertices $s, t \in V$ such that there are $2^{\Omega(n)}$ simple $s \rightsquigarrow t$ paths.

2. **Connectivity.** A graph $G = (V, E)$ is *connected* if there is a path between any two vertices. Design an algorithm to detect whether an **undirected** graph is *connected*. Your algorithm should return YES if the graph is connected; otherwise, return NO. Provide low-level pseudocode which specifies the data structures you are using. Your algorithm should run in $O(m + n)$ time on a graph with n vertices and m edges.

3. **Strongly Connected Components.** Let $G = (V, E)$ be a **directed** graph. Vertices u and v are *strongly connected* if there are $u \rightsquigarrow v$ and $v \rightsquigarrow u$ paths in G. A *strongly connected component* is a set of vertices $C \subseteq V$ such that u, v are strongly connected for all $u, v \in C$ (and no other vertices are strongly connected to a vertex $u \in C$.)

   Design and analyze an algorithm to identify all strongly connected components in G. What is the runtime of your algorithm?

4. **Claire's Coffee Corner.** Claire has a coffee shop, famous for its elaborate special-order coffees. Each beverage costs \$2, no matter what unusual ingredients are used or how long it takes to prepare the drink. Needless to say, this deal attracts a lot of regular customers, some of whom order very unusual concoctions that take a long time to prepare.

   Claire's regular customers are impatient and demanding. They want coffees made *exactly* according to their special order, and if their drinks aren't started as soon as they enter the shop, they get angry and storm out.

Claire used to have many employees to help her make the coffees, but unfortunately, all of Claire's coffee clerks went on strike, leaving Claire alone to make the coffees. Claire wants to make as many coffees as possible, but now can only prepare one drink at a time. Coffee preparation requires a lot of attention, so once Claire begins to make a drink, she must focus on that drink until it is ready. Only after it is prepared can Claire begin the next drink.

Design and analyze a strategy to help Claire quickly decide which drinks to make. Your strategy should take as input the information about regular customers (when they arrive and how long their drink takes to make) and output which drinks Claire should make. Does your strategy guarantee that Claire prepares the maximum number of coffees? If so, say why. Otherwise, give an example when it does not maximize the number of coffees prepared.