Algorithms Final review

Chapters 5–12 problem session

In class, April 30, 2025

Problems

1. A graph $G = (V, E)$ is *bipartite* if the vertices can be divided two sets A and B, such that $A \cup B = V$, $A \cap B = \emptyset$, and for every edge $uv \in E$, one end point is in A and the other in B. In other words, G is bipartite if the vertex set can be divided so that each half is an independent set.

   Give an algorithm to determine if a graph is bipartite.

2. Some friends of yours work on wireless networks, and they're currently studying the properties of a network among n mobile devices. As the devices move around, they define a graph at any point in time as follows: there is a node for each device, and there is an edge between device i and device j if the physical locations are less than 500 meters apart. (If so, we say that i and j are in range of each other.) They'd like it to be the case that the network of devices is connected at all times, and so they've constrained the motion to satisfy the following property: at all times, each device i must be in range of at least $n/2$ of the other devices. Our question is: does this property actually guarantee that the network will remain connected? In other words:

   Claim: Let G be a graph on n nodes, where n is an even number. If every node of G has degree at least $n/2$, then G is connected.

3. Let G be a directed acyclic graph with a unique source s and a unique sink t. A Hamiltonian path in G is a directed path in G that contains every vertex in G. Describe an algorithm to determine whether G has a Hamiltonian path

4. Let G be a connected undirected graph. Suppose we start with two coins on two arbitrarily chosen vertices of G. At every step, each coin must move to an adjacent vertex. Describe and analyze an algorithm to compute the minimum number of steps to reach a configuration where both coins are on the same vertex, or to report correctly that no such configuration is reachable. The input to your algorithm consists of a graph $G = (V, E)$ and two vertices $u, v \in V$ (which may or may not be distinct).

5. Suppose you are given a connected graph G, with edge costs that you may assume are all distinct. G has n vertices and m edges. A particular edge e of G is specified. Give an algorithm with running time $O(m + n)$ that decides whether e is contained in a minimum spanning tree of G.

6. Decide if you think the claim is true or false, and give a proof or counterexample.

(a) Suppose we are given an instance of the minimum spanning tree problem on a graph G, with all positive and distinct edge costs. Let T be a minimum spanning tree for this instance. Now suppose we replace each edge weight $w(e)$ by its square, $w(e)^2$, which has the same edge and vertex sets but a different weight function. True or false: T is still a minimum spanning tree for this new graph.

(b) Now suppose we have a weighted directed graph G with all positive distinct edge weights. Let P be a minimum cost path from vertex s to t in this graph. We now again replace the weight on each edge $w(e)$ with its square $w(e)^2$, creating a new graph with the same edge and vertex sets but new weights. True or false: P must still be a minimum cost s to t path in this new graph.

7. You are given a city map consisting of *intersections* and *roads*, where each road has a *travel time* in minutes and a *risk level* (an integer from 1 to 10, where 1 is very safe and 10 is very dangerous). You work for an emergency services team that must **minimize the total travel time** from a starting location to a destination, but **you may only use roads with a risk level at most** R, where R is a threshold specified by your team's safety guidelines.

   More precisely, the input is specified as follows:

   - A list of intersections (numbered from 1 to n).
   - A list of roads, where each road is a tuple (i, j, t_{ij}, r_{ij}), indicating a road between intersections i and j with travel time t_{ij} and risk level r_{ij}.
   - A starting intersection s, a target intersection t, and a risk threshold R.

   Give an algorithm, as efficient as possible, to find the shortest possible route which does not use any road with risk above R.

8. You are helping the medical consulting firm Doctors Without Weekends set up the work schedules of doctors in a large hospital. Daily schedules are finished, but they now need to make sure that at least one doctor is present on every holiday.

   Here's how we set it up. There are k vacation periods (e.g. the week of Christmas, the weekend of July 4, etc.) each spanning several contiguous days. Let D_j be the set of days included in the j^{th} vacation period; we will refer to the union of these days, $\cup_j D_j$, as the set of all vacation days.

   There are n doctors at the hospital, and doctor i has a set of vacation days S_i when she is available to work.

   (a) Give a polynomial time algorithm to determine if it is possible to select a single doctor to work on each vacation day, so that each doctor will work at most c vacation days total.

   (b) Now we add one restriction. Suppose we want to ensure that no doctor works for more than one day in any set D_j. (So each doctor works at most one vacation day in any holiday period.) Give a polynomial time algorithm to determine if such an assignment is possible.

9. Define the *escape problem* as follows: You are given a directed graph $G = (V, E)$ – picture this as based on a network of roads. A certain collection of nodes $X \subset V$ are designated as *populated* nodes, and a different subset $S \subset V$ are designated as *safe* nodes. (You may assume S and X are disjoint.) In case of an emergency, we want evacuation routes from the populated nodes to the safe nodes. A set of evacuation routes is defined as a set of paths in G so that (i) each node in X is the start of one path, (ii) the last node on each path lies in$S$, and (iii) the paths do not share any edges. Such a set of paths gives a way for the occupants of the populated nodes to "escape" to S, without overly congesting any edge in G.

   (a) Given G, X, and S, show how to decide in polynomial time whether such a set of evacuation routes exists.

   (b) Suppose we have exactly the same problem as in (a), but we want to enforce an even stronger version of the "no congestion" condition (iii). Thus we change (iii) to say "the paths do not share any nodes." With this new condition, show how to decide in polynomial lime whether such a set of evacuation routes exists

10. A graph is *tonian* if there is a cycle in the graph that visits at least half of the vertices. Show that decided if a graph G is tonian is NP-Complete.

11. Let $G = (V, E)$ be a graph. A dominating set in G is a subset S of the vertices such that every vertex in G is either in S or adjacent to a vertex in S. The *dominating set* problem asks, given a graph G and an integer k as input, whether G contains a dominating set of size k. Prove that this problem is NP-hard.

12. Consider the following problem: You are managing a communication network, modeled by a directed graph $G = (V, E)$. There are c users who are interested in making use of this network. User i (for $i \in \{1, \ldots c\}$) issues a request to reserve a specific path P_i in G on which to transmit data. You are interested in accepting as many of these path requests as possible, subject to the following restrictino: if you accept both P_i and P_j, then the two paths cannot share any nodes.

    This defines the *path selection problem*: Given a direct graph $G = (V, E)$, a set of requests $P_1, \ldots, P_c$, each of which is a path in G, and a number k, is it possible to select at least k of the paths so that no two of the paths share any nodes?

    Prove that path selection is NP-Complete.