Practice Problems 10

Problem 1. You are given an s-t network $G = (V, E)$, where each edge $(u, v) \in E$ stores a nonnegative capacity $c(u, v)$. Given any path in G, define its *capacity* to be the minimum capacity of any edge on the path. (The trivial path, consisting of no edges has a capacity of $+\infty$.)

Present an efficient algorithm which, given G, computes the path from s to t that has maximum capacity. (If there are multiple such paths of equal capacity, return any of them.) For example, the maximum capacity s-t path in the network shown in Fig. 1 has capacity 4.

What is your algorithm's running time? Present a formal proof that your algorithm is correct.

(**Hint:** This can be solved by a straightforward adaptation of Dijkstra's algorithm, particularly altering the meaning $d[v]$ for each vertex v, and how the relax operator works. Explain how to adapt the correctness proof of Dijkstra's algorithm to this context.)

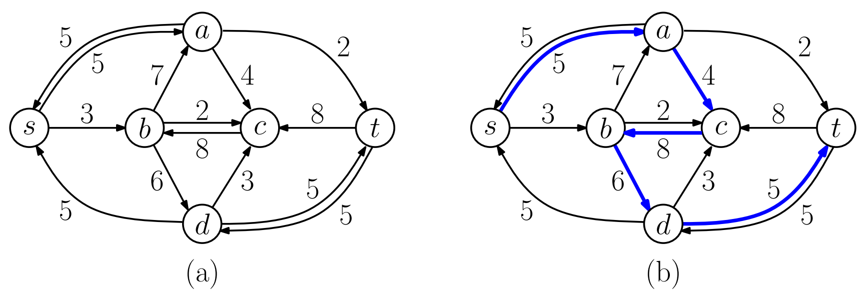

Figure 1: Maximum capacity s-t path.

Problem 2. Your friend has a new drone delivery startup, and they have asked you to help to design software to assist with scheduling this day's deliveries.

- There are m drone stations throughout the city. For $1 \leq i \leq m$, let d_i denote the (x, y) coordinates of the ith drone station (see Fig. 2(a)).
- There are n customers expecting to receive deliveries this day. For $1 \leq j \leq n$, let c_j denote the (x, y) coordinates of the jth customer.
- Customer j has ordered $o_j \geq 1$ deliveries. Each delivery requires a separate flight, which may come from any of the drone stations that is within 10 miles of the customer.
- FAA requirements state that each drone station can launch at total of at most 5 deliveries per day, and at most 2 flights can go from any one drone station to any one customer. (Note, however, that a customer can receive deliveries from different drone stations, but at most two per station.)
- Customers understand that drone deliveries are unreliable, and so a customer will be satisfied if at least $\max(1, o_j - 2)$ packages arrive.

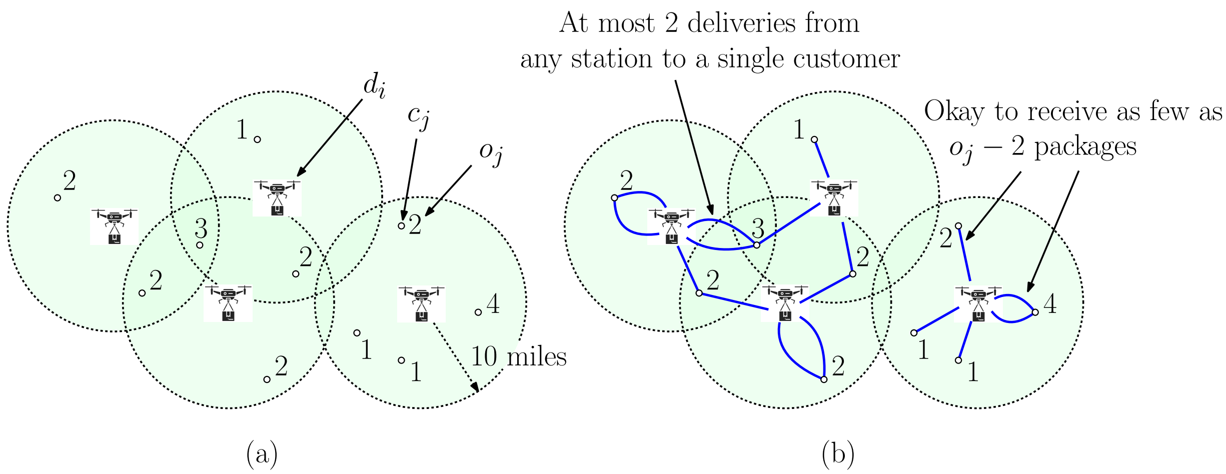

Figure 2: Drone delivery service. Black points are drone stations and hollow points are customers with package counts o_j: (a) Input and (b) Possible solution.

A *delivery schedule* is a multiset of station-customer pairs (d_i, c_j), called *deliveries*. Such a schedule is *valid* if the following constraints are observed:

(a) Drone station d_i can deliver to customer c_j only if $\text{dist}(d_i, c_j) \leq 10$.

(b) Each drone station makes at most 2 deliveries to any single customer and a total of at most 5 deliveries overall.

(c) Customer c_j receives a total of at least $\max(1, o_j - 2)$ deliveries and at most o_j deliveries.

Present an efficient algorithm which is given arrays of drone station coordinates $d[1..m]$, customer coordinates $c[1..n]$, and package order counts $o[1..n]$. The algorithm determines whether there is a valid delivery. (Hint: Reduce to circulations. Provide a *detailed proof* of your reduction's correctness. See the Survey-Design application in Lecture 14 for an example of what we are looking for.)

Problem 3. The computer science department at a major university has a tutoring program. There are m tutors, $\{t_1, \ldots, t_m\}$ and n students who have requested the tutoring service $\{s_1, \ldots, s_n\}$. Each tutor t_i has a set T_i of topics that he/she knows, and each student s_j has a set of topics S_j that he/she wants help with. We say that tutor t_i is *suitable* to work with student s_j if $S_j \subseteq T_i$. (That is, the tutor t_i knows all the topics of interest to student s_j.) Finally, each tutor t_i has a range $[a_i, b_i]$, indicating that this tutor would like to work with at least a_i students and at most b_i students.

Given a list of students, a list of tutors, the ranges $[a_i, b_i]$ for the tutors, and a list of suitable tutors for each student, present an efficient algorithm that determines whether it is possible to generate a pairing of tutors to students such that:

- Each student is paired with *exactly* one tutor.
- Each tutor t_i is paired with *at least* a_i and *at most* b_i students.
- Each student is paired only with a *suitable* tutor.

Problem 4. Some thieves have stolen jewels from a famous French museum. You have been asked by the French police to help assist them set up roadblocks at various intersections in order to catch the thieves. The police believe that the thieves will drive from the museum to the airport along some unknown route. The police have provided you with a map of the city represented as an undirected graph $G = (V, E)$, where each vertex is an intersection and each edge is a road (see Fig. 3(a)). The museum is at vertex p, and the airport is at vertex q.

Your task is to determine the locations of a minimum number of roadblocks such that no matter what route the thieves take from p to q, they must pass through at least one of your roadblocks (see Fig. 3(b)). Note that you cannot place a roadblock either at p or q. (That would be too easy.)

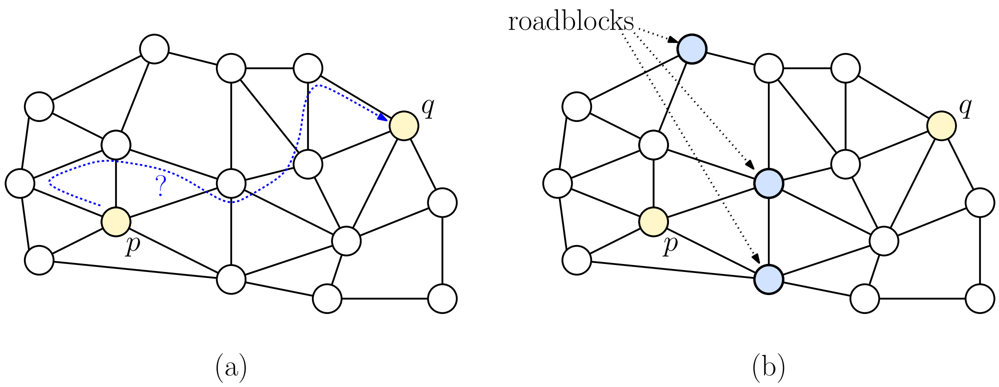

Figure 3: Roadblock problem. Three roadblocks suffice to detect any path from p to q.

(a) Present an efficient algorithm to compute these roadblocks. Justify your algorithm's correctness. Explain your reduction (including how your network is constructed, what the source and sink nodes are, what the capacities are, and how the output of the flow algorithm is interpreted.) Let $T(n, m)$ denote the time to compute the maximum flow on a network with n nodes and m edges. As a function of n, m, and $T(n, m)$, what is the running time of your algorithm?

(b) The French police have received new information that the theft may take place at any one of a number of vertices $P = \{p_1, \ldots, p_k\}$, and the thief's destination may be at any one of the vertices $Q = \{q_1, \ldots, q_\ell\}$. You may assume that P and Q are disjoint. (In this case you can place roadblocks at the elements of P and/or Q, but if k and ℓ are large, this may not be optimal.)

Explain how to modify your solution to part (a) to deal with this new complication. In particular, you wish to compute a minimum number of roadblocks so that any path from any $p_i \in P$ to any $q_j \in Q$ must pass through at least one of your roadblocks. As a function of k, ℓ, n, m, and $T(n, m)$, what is the running time of your modified algorithm?

Problem 5. You are given an s-t network G, which may contain cycles. Consider any flow f in G. Define $G[f]$ to be the network that results by removing from G all edges (u, v) such that $f(u, v) = 0$. (That is, we keep only the edges that are carrying flow.) Prove that if $G[f]$ contains any cycles, then there exists another flow f', such that $|f'| = |f|$, and $G[f']$ contains no cycles. (**Hint:** It may be helpful to recall that our definition of an s-t network assumes that no edge enters s and no edge leaves t, therefore such a cycle cannot pass through either s or t.)

Just for fun: You and your roommate are contestants in a game of wits. The devil gives each of you a card with a positive integer written on it. Neither person can see the other person's card, but the devil tells you that the difference in the two numbers is 1. For example, if your number is "53", then you know that your roommate may have one of the numbers "52"or "54". Otherwise, all you know about the possible numbers is that they must be 1 or larger.

The devil tells you that if either of you can determine the number on your roommate's card, you will receive a shiny fiddle made of gold, and otherwise you'll get a lump of coal. You and your roommate are pretty smart, so you take the devil's challenge. The devil starts his game:

- The devil asks you whether you know the number on your roommate's card. After thinking, you answer "no".
- The devil then asks your roommate whether they know your number. After thinking, your roommate answers "no".
- The devil is nice enough to give you another chance. After thinking, you again say "no".
- The devil gives your roommate another chance. After thinking, your roommate answers "no".

At this point, the devil has decided that you both have lost, and he holds out two lumps of coal. Suddenly, you exclaim, "I know the number on my roommate's card!" Your roommate then says the same. You reveal the numbers, and you both win your golden fiddles.

Explain how each of you determined the number on your cards. (**Hint:** This worked because you and your roommate had a particular combination of numbers.)