Problem Set 2 Solutions

This problem set is not due and it will not be graded, but you are welcome to come to office hours to discuss your solutions. This is a set of practice problems for the midterm and final, and it covers the first quarter of the course.

The description of your proofs and algorithms should be as *clear* as possible (which does not mean *long* – in fact, typically, good clear explanations are also short.)

When a problem asks to give an algorithm, in your solution: (i) describe shortly and informally the main ideas in your solution; (ii) give a detailed description of the algorithm, using a style similar to the pseudo-code used in class or in a textbook; (iii) prove the correctness of the algorithm; (iv) prove a bound on the time complexity of the algorithm. You can omit the proof of correctness if it is clear from the description of the algorithm.

---

1. We are given a directed graph $G = (V, E)$ describing a network, and, for each edge $(u, v) \in E$ we are given the probability $p(u, v)$ that the link (u, v) may fail. These probabilities are independent, that is, the probability that no edge fails in the path $(u_1, u_2, \ldots, u_k)$ is $(1 - p(u_1, u_2))(1 - p(u_2, u_3)) \cdots (1 - p(u_{k-1}, u_k))$. Such a value is called the *reliability* of the path $(u_1, \ldots, u_k)$.

   Describe and analyse an algorithm that given G, the values $p(\cdot, \cdot)$ and two vertices $s, t \in V$, finds the path from s to t of maximum reliability.

   [Hint: if F is a monotone increasing function, then looking for the path $(s, u_1, \ldots, u_k, t)$ that maximizes $(1-p(s, u_1))(1-p(u_1, u_2)) \cdots (1-p(u_k, t))$ is the same as looking for the path that maximizes $F((1-p(s, u_1))(1-p(u_1, u_2)) \cdots (1-p(u_k, t)))$. Similarly, if F is monotone decreasing, the problem is the same as looking for the path $(s, u_1, \ldots, u_k, t)$ that minimizes $F((1 - p(s, u_1))(1 - p(u_1, u_2)) \cdots (1 - p(u_k, t)))$]

   Solution: create a new graph G' with the same vertex set and edge set as G, and such that each edge (u, v) has length $\ell_{u,v} = -\log(1 - p(u, v))$. Note that since the non-failure probabilities $1 - p(u, v)$ are ≤ 1, their logarithms are ≤ 0 and so all the lengths are ≥ 0. Then, observe that the length of a path $(s, u_1, \ldots, u_k, t)$ is $-\log(1 - p(s, u_1))(1 - p(u_1, u_2)) \cdots (1 - p(u_k, t))$, so a path from s to t of maximum reliability is a path from s to t in G' of minimum length (because $a \geq b$ if and only if $-\log a \leq -\log b$). We then just have to run Dijkstra's algorithm on the graph G' to find a shortest path in G' from s to t

2. We have seen that, using the Ford-Fulkerson algorithm, not only we can find the flow of maximum cost, but also the cut of minimum capacity. In certain applications, it is important to know if the minimum cut is unique, and, if not, how many minimum cuts there are.

   (a) Give an example of a network where the minimum cut is unique.

Solution: consider a network in which there are only the vertices s and t and an edge from s to t of capacity 1. Then the only cut, and hence the only minimum cut is $C = \{s\}$.

(b) Give an example of a network where there is more than one minimum cut.

Solution: Consider a network in which the set of nodes is $\{s, v_1, \ldots, v_n, t\}$ and we have the edges (s, v_1), v_i, v_{i+1} for $i = 1, \ldots, n-1$, and (v_n, t), all of capacity 1. Then the capacity of a minimum cut is 1, because all capacities are 1 and every cut is crossed by at least one edge, and the cut $C = \{s\}$ shows that capacity 1 is achievable. Every cut of the form $\{s, v_1, \ldots, v_i\}$ for $i = 1, \ldots$ is also optimal.

(c) Give a polynomial time algorithm that determines whether the minimum cut is unique, and, if not, finds at least two distinct minimum cuts.

Solution: The general idea is to first find a maximum flow f using the Ford-Fulkerson algorithm, and then a minimum cut C given by the set of vertices reachable from s in the residual network of f. Then we want to find a minimum cut $C' \neq C$ if C is not the unique one. This could be done in several possible ways.

One approach is to consider the set T of vertices *from which t is reachable* in the residual network of f. We argue that $C' = V - T$ of vertices from which t is not reachable is a minimum cut. First of all, $t \in T$ and $s \notin T$, so C' is a cut. We can proceed in the same way we proved that the set of vertices reachable from s in the residual network of a maximum flow is a minimum cut. We write

$$value(f) = \sum_{u \in C'} \sum_{v \in T} f_{u,v}$$

and then we observe that, for each edge (u, v) such that $u \in C'$ and $v \in T$ we must have $f_{u,v} = c_{u,v}$, or else the edge (u, v) would belong to the residual network, and $v \in T$ would imply that $u \in T$. Thus we have

$$value(f) = \sum_{u \in C'} \sum_{v \in T} f_{u,v} = capacity(C')$$

and so C' is an optimal cut.

If $C \neq C'$, we have found a second optimal cut.

It remains to argue that, if $C = C'$, then C is the unique minimum cut. Consider any minimum cut C''. By above reasoning, all edges (u, v) such that $u \in C''$ and $v \notin C''$ must be such that $f_{u,v} = c_{u,v}$. In particular, in the residual network, there is no path that can go from a vertex inside C'' to a vertex outside C''. This means that t is not reachable in the residual network from any vertex in C'' and $C'' \subseteq V - T = C' = C$. It also means that from s, in the residual network, we cannot reach any vertex outside C'' and so $C \subseteq C''$. Putting the two together we have $C = C''$.

3. A cohort of k spies resident in a certain country needs escape routes in case of emergency. They will be travelling using the railway system which we can think of as a directed graph $G = (V, E)$ with V being the cities. Each spy i has a starting point $s_i \in V$ and needs to reach the consulate of a friendly nation; these consulates are in a known set of cities $T \subseteq V$. In order to move undetected, the spies agree that at most c of them should ever pass through any one city. Our goal is to find a set of paths, one for each of the spies (or detect that the requirements cannot be met).

   Model this problem as a flow network. Specify the vertices, edges and capacities, and show that a maximum flow in your network can be transformed into an optimal solution for the original problem. You do not need to explain how to solve the max-flow instance itself.

   Solution: This was not said very clearly in the statement of the problem, but the starting points $s_1, \ldots, s_k$ are distinct nodes (that is, each spies starts at a different city).

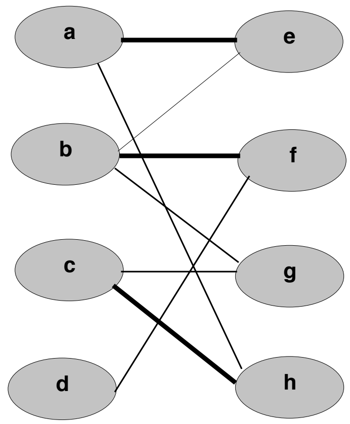
   We reduce the problem to a maximum flow problem with capacities on the vertices. We do so by creating a network that has all the vertices and edges of G, and additional new nodes s and t.

   The new node s has capacity-1 edges connecting it to each of the starting points s_i, and the new node t has edges going from each node in T to it, each of capacity c. Finally, all cities have vertex-capacity c and all other edges have capacity c. We then reduce the maximum flow problem with vertex capacities to a regular maximum flow problem as done in class (each node v is split into two nodes v_{in} and v_{out}, all the edges of the form (u, v) become of the form (u_{out}, v_{in}), for the same capacity as before, and for each vertex v of capacity c_v we have an edge (v_{in}, v_{out}) of capacity c_v.

   If there is a solution to the spies problem, then sending one unit of flow from s to each of the s_i, then one unit of flow along each of the paths from s_i to T, for $i = 1, \ldots, k$, and finally from nodes in T to t satisfying conservation constraints, we have a feasible flow of value k, which is optimal because the cut $\{s\}$ has capacity k. This means that if the maximum flow in the network is $\leq k - 1$ then the spies problem has no solution.

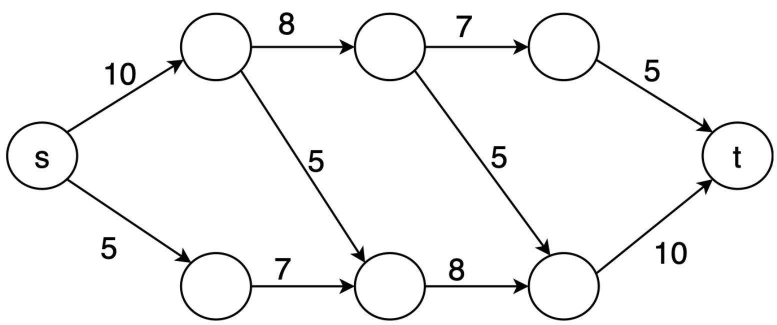
   Conversely, if the maximum flow in the network is k, will find the k paths as follows: we find a path from s to t made entirely of edges of the network that have positive flow; such a path contains a path from one of the s_i to a city in T. This will be the path for spy i. Then we reduce by one unit the flow on each edge of that path (so that we have a feasible flow of value $k - 1$) and remove s_i from the network. Then we do the same on this new flow. At each step, we have $k' < k$ spies left to rout, and a feasible flow of value k'.

4. Below we see a bipartite graph and a matching with 3 edges. Does the graph have a matching with 4 edges?



Solution: Yes, the matching $\{(a, e), (b, g), (c, h), (d, f)\}$.

5. Find a maximum flow and a minimum cut in the network below



Solution: A maximum flow is below, a minimum cut is $\{s\}$.