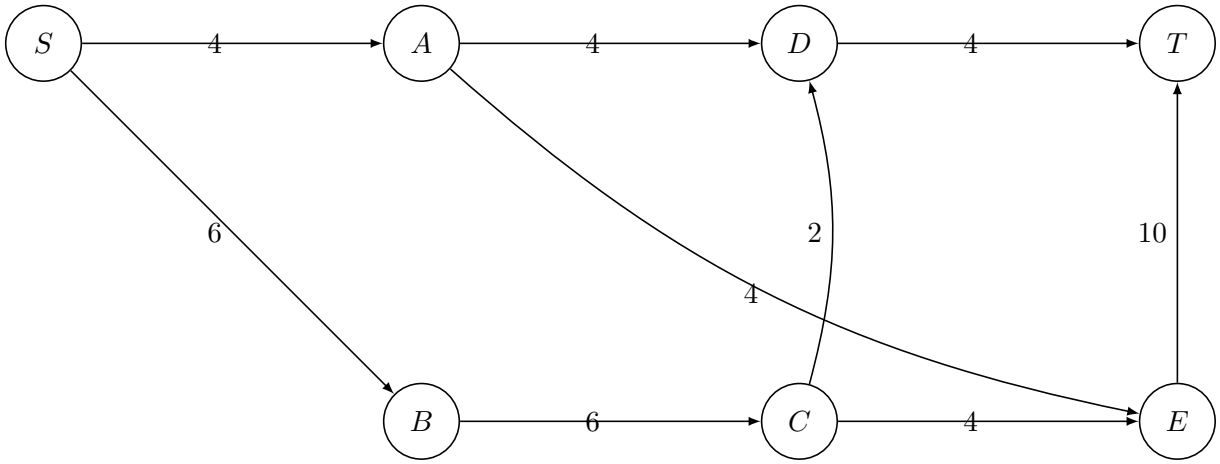
CS5800 Algorithms

Recitations from Oct 28, 2019 to Nov 1, 2019

Ravi Sundaram

Problem 1 (Network Flow)

1. Compute the maximum flow of the following two flow network using the Edmonds-Karp algorithm, where the source is S and the sink is T. Show the residual network after each flow augmentation. Write down the maximum flow value.



2. A school has S students, C classes, and T sports teams. Each student belongs to exactly one class, but is a member of one of more sports teams. Each team wants to select a member to be its representative in the student council. But a student cannot represent more than one team. The school also doesn't want to have more than 5 students from the same class in the student council. Describe an algorithm to determine if it is possible to form a student council subject to these constraints.

3. There have reports of an extremely dangerous swine flu outbreak in Boston (rumor has it the virus was created in a Boston University lab). The United States government is worried the diseases will spread to Washington DC (but they don't care what happens to the rest of the country). To stop the disease from spreading to Washington DC, they are going to close train stations. They want to hurt the economy as little as possible so they are trying to close the minimum number of stations. Boston is already too dangerous to close, and they don't want to close the station in Washington DC as that'd be too inconvenient.

   Describe and analyze an algorithm to the find the minimum number of stations that must be closed to block all rail travel from Boston to Washington DC. The rail network is represented by an undirected graph, with a vertex for each station and an edge for each rail connection between two stations.

Solution:

1. Edmonds-Karp chooses the shortest path (by shortest we mean the number of hops) when it looks for an augmenting path. So here, the paths through which we add flow are

$$S, A, D, T \text{ 4 units of flow}$$
$$S, B, C, E, T \text{ 4 units of flow}$$
$$S, B, C, D, A, E, T \text{ 2 units of flow}$$

2. We model this problem as a network flow problem. We create vertices s, t. Each set of the students, classes, and sports teams will be layers in the graph we create, and edges (and capacities) connecting the layers will be determined by the constraints we are given.
Here, we can solve the problem as follows. For each class $i \in C$, we create a vertex c_i and a directed edge from s to c_i with capacity 5. Our next layer will consist of the students. For each student j we create the vertex s_j and we create an edge from c_i to s_j if student j belongs to class i with capacity 1. Our next layer will be the sports teams. For each team k, we create the vertex t_k and an edge from s_j to t_k with capacity 1 if student j belongs to team k. Note that a student can be a part of multiple teams. And finally, we create a directed edge with capacity 1 from each t_k to t. Now we are done with graph.

   We can run max-flow on it, and if the max flow is equal to the number of teams, then it's possible to form a student council subject to the constraints. Otherwise its not. To see why this works, let's walk through the graph again. Each edge from s to c_i has capacity 5, which means each class gets at most 5 students in the student council. Each edge from c_i to s_j has capacity 1 if student j is in class i, which means a student is getting at most 1 flow (as a student can only be a representative once/represent 1 team). Each edge from s_j to t_k has capacity 1, which represents that a student can represent that team if they are on the team. Finally, each edge from t_k to t has capacity 1, which means a team gets at most one representative. Technically this wasn't a constraint, but the question is whether there's a way so that every team gets a representative, so doing it this way make sure the max flow won't be large even if not every team gets a representative.

3. So this problem looks like a min cut problem, but the issue is we want to find the minimum number of stations to close, which corresponds to vertices in the original graph. However, min cut is defined through edges. So we need to think of a way to convert the stations into edges somehow. We can do that in the following manner. For each station c, we create the vertices c_{in}, c_{out}, and a directed edge from c_{in} to c_{out} with capacity 1. Now suppose c is connected to stations a, b. Then we convert the original undirected edges in the graph by creating directed edges from c_{out} to a_{in} and b_{in}. Similarly, we create directed edges from a_{out}, b_{out} to c_{in}.
Now look at what happens if we delete the edge (c_{in}, c_{out}). If a train comes in to c_{in}, it will no longer be able to go anywhere, as all the outbound edges start at c_{out}. So deleting the edge (c_{in}, c_{out}) is equivalent to shutting down the station. So we can find the min cut between Boston and DC, which will corresponds to the minimum number of stations. But wait! If we are not careful about the capacities of the edges connecting different stations (the $(c_{out}, a_{in}), (b_{out}, c_{in})$ edges) then they might be included in the min cut, which is not what we want as we exclusively care about closing the stations, not any of the rail tracks between stations. So we set the capacities of all the edges connecting different cities to ∞. Then our min cut will only consist of edges of the form (c_{in}, c_{out}) which is exactly what we want.

...

■

Problem 2 (Extras...)

1. Let $G = (V, E)$ be a flow network with source s and sink t. We say that an edge e is a chokepoint if it crosses every minimum-capacity cut separating s from t. Give an efficient algorithm to determine if a given edge e is a chokepoint in G.