Analysis of Algorithms

Homework 5 – Networks

Abraham Murciano

January 24, 2021

Question 1

Part A

We are given the network in figure 1. We are to find the maximal flow using the Edmonds-Karp algorithm.

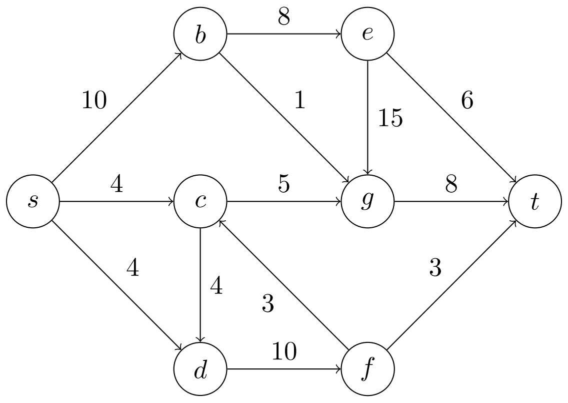

Figure 1: A network

The first augmenting path which the algorithm finds can be (s, b, e, t) (it may vary depending on the order that neighbours are selected in the BFS), through which a flow of 6 can be sent. The states of the graph at each point are shown in figure 2. (Saturated edges are omitted, and only remaining capacity is shown on each edge.)

The next augmenting path could be (s, b, g, t) with a flow of 1. After this one, another possible augmenting path is (a, c, g, t), which has a flow of 4. Next is (s, d, f, t) with a flow of 3. The next augmenting path is s, b, e, g, t. A flow of 2 can be sent through this one. The final augmenting path has a flow of 1, and is (s, d, f, c, g, t).

Once there are no more augmenting paths to be found, we can sum up the flows through all the augmenting paths to find the maximal flow, which for this network is $6 + 1 + 4 + 3 + 2 + 1 = 17$.

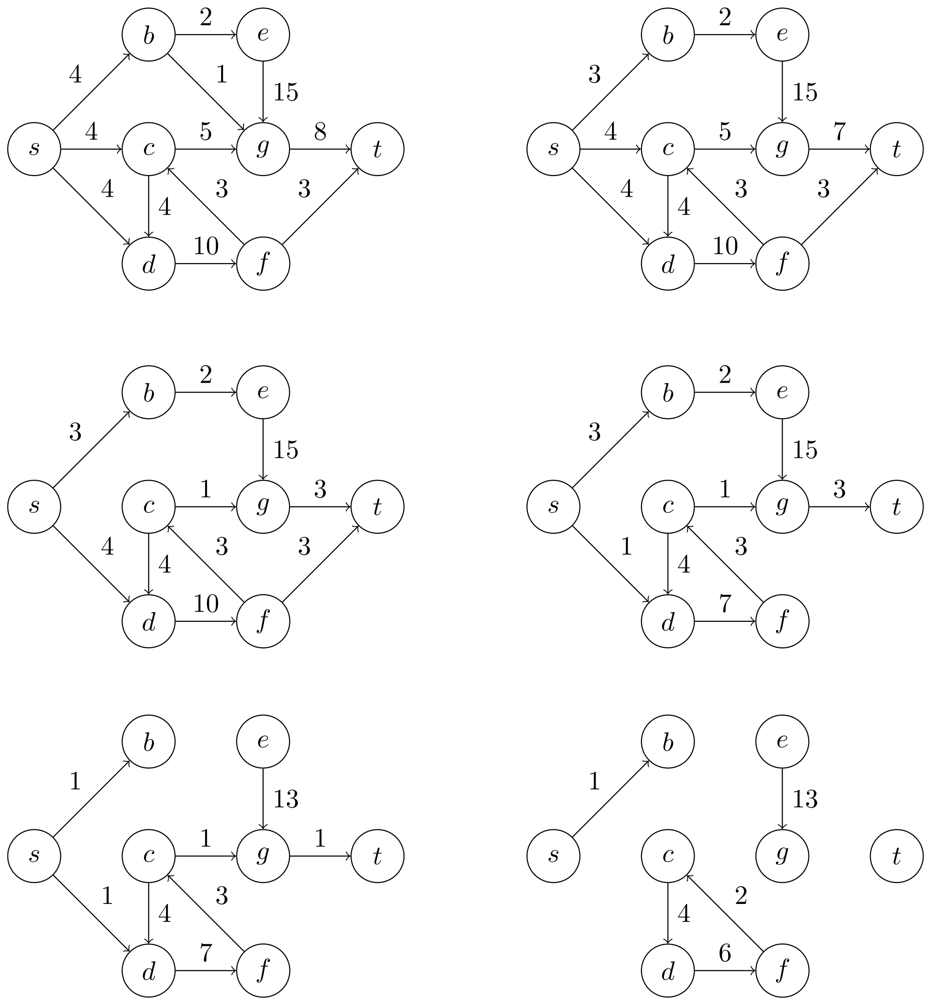

Figure 2: The stages of the graph after sending sending flow through each shortest path

Part B

The minimal cut in the network must be a cut such that the sum of the capacities of the forward facing edges is minimal. We know however, that the capacity of the minimal cut must always be the same as the maximal flow. Therefore a minimal cut of the network in figure 1 must have a capacity of 17. One such cut is the following one.

$$(\{s, b\}, \{c, d, e, f, g, t\})$$

Question 2

Part A

Supposing we have a network N, we are told that N' is a modified version of N such that all the capacities of N are multiplied by some constant $c > 1$. If $f_{\max}$ and $f'_{\max}$ are the maximal flows of N and N' respectively, then $f'_{\max} = cf_{\max}$

To demonstrate this, suppose P is the set of all augmenting paths found by a run of the Edmonds-Karp algorithm on either N or N' (P would be the same regardless of which network is used).

Now for any $p \in P$, we will denote by f_p and f'_p the amount of flow that can be sent through p on the networks N and N' respectively. This is equal to the smallest capacity of an edge along the path p. But since all the capacities in N' are c multiplied by the corresponding capacity in N, it follows that $f'_p = cf_p$.

Finally, we know the following.

$$f_{\max} = \sum_{p \in P} f_p$$
$$f'_{\max} = \sum_{p \in P} f'_p = \sum_{p \in P} cf_p = c \sum_{p \in P} f_p = cf_{\max}$$

Part B

If we tweak the problem in part A such that $0 < c < 1$, the answer would be the same, with the same explanation, because the explanation given for part A at no point applies the assumption that $c > 1$. It holds for all $c > 0$.

Question 3

We are tasked with finding a maximal set of disjoint paths for the graph in figure 3 from vertices 1 to 6. The maximal set for this graph is the set $\{p_1, p_2, p_3\}$, such that:

$$p_1 = ((1,2),(2,6))$$
$$p_2 = ((1,5),(5,6))$$
$$p_3 = ((1,8),(8,7),(7,6))$$

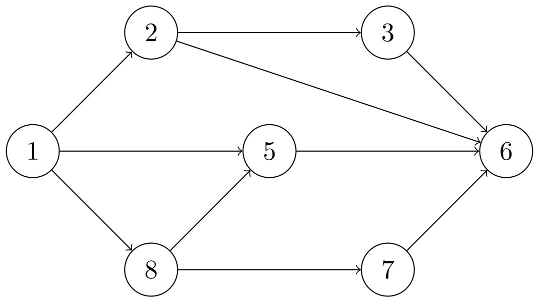

Figure 3: A sample graph for question 3

Part B

We are to suggest an algorithm based on maximal flow that will find a maximal set of disjoint paths from s to t given a directed graph.

If we construct a network out of the given graph, such that the capacities of each edge are all some constant c, then we attempt to find the maximal flow using the Ford–Fulkerson method, we can return a set of all augmenting paths as the maximal set of disjoint paths.

This set must be disjoint, because once an edge has been used with a flow of c, it is saturated and will not be used as part of another augmenting path. This set must also me maximal because we can send c units of flow through each augmenting path, so the maximal flow must be sent through the maximal number of augmenting paths.

Question 4

Suppose that for a network, in addition to capacities being assigned to edges, capacity is also assigned to each vertex, so that maximal flow that can pass through vertex v is c_v.

We are to suggest an algorithm to find maximal flow through this network, specifying its run-time complexity.

We can calculate the maximal flow of such a network by replacing each vertex v by two vertices v_1 and v_2. Then each edge of the form (x, v) or (v, y)

is replaced with (x, v_1) and (v_2, y) respectively. We then add a single edge (v_1, v_2) with the capacity c_v. Thus we translate all the vertex capacities into edge capacities, transforming our problem into one we can already solve.

The preprocessing stage can be done in $O(|V| + |E|)$, and then if we use the Edmonds-Karp algorithm for the next stage, the total complexity would be that of the Edmonds-Karp algorithm; i.e. $O(|V||E|^2)$.