CSCI 570 Homework 4
Spring 2023

Due Date: Mar. 27, 2023 at 11:59 P.M.

1. You are given the following graph G. Each edge is labeled with the capacity of that edge.

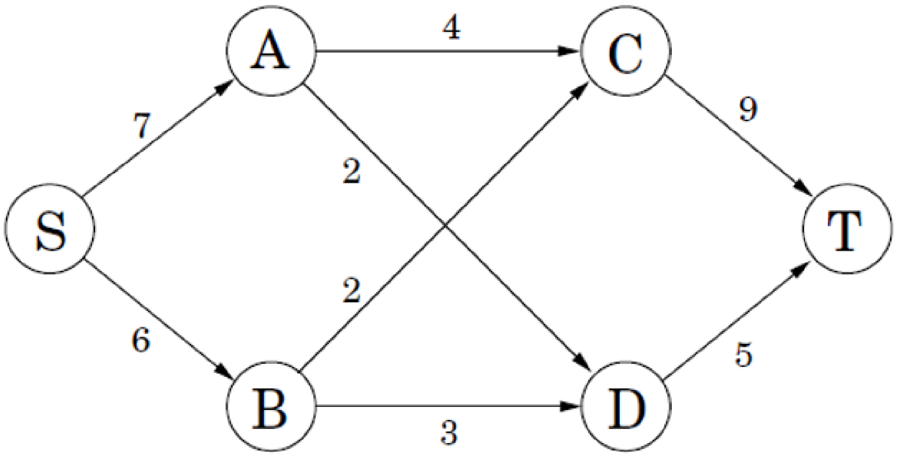


Figure 1: Graph G

   (a) Find a max-flow in G using the Ford-Fulkerson algorithm. Draw the residual graph G_f corresponding to the max flow. You do not need to show all intermediate steps.

   See fig 2

   (b) Find the max-flow value and a min-cut.
   f = 11, cut: (S, A, B, C, D, T)

   (c) Prove or disprove that increasing the capacity of an edge that belongs to a min cut will always result in increasing the maximum flow.
   Disprove by a counterexample. Increasing (B,D) by one won't increase the max-flow.

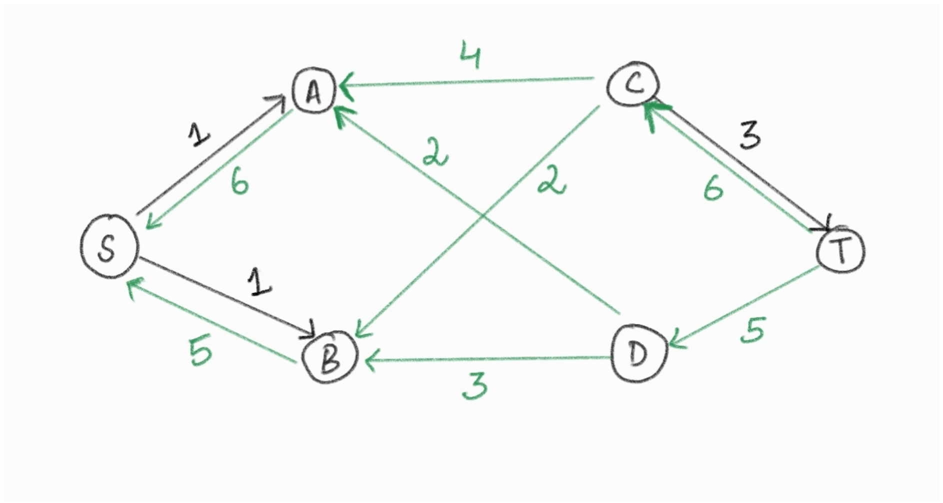


Figure 2: Graph G_f

2. Consider a set of mobile computing clients in a certain town who each need to be connected to one of several possible base stations. We'll suppose there are n clients, with the position of each client specified by its (x, y) coordinates in the plane. There are also k base stations; the position of each of these is specified by (x, y) coordinates as well. For each client, we wish to connect it to exactly one of the base stations. Our choice of connections is constrained in the following ways. There is a range parameter R which means that a client can only be connected to a base station that is within distance R. There is also a load parameter L which means that no more than L clients can be connected to any single base station. Given the positions of a set of clients and a set of base stations, as well as the range and load parameters, decide whether every client can be connected simultaneously to a base station.

   (a) Describe how to construct a flow network

   Introduce a vertex b_i for every base station and a vertex c_i for every client. Connect each client to a base station if that station is within the range R. Assign capacity 1 to those edges. Let s be a source vertex and t be a sink vertex. For every client vertex, add an edge (s, c_i) of capacity 1. For every base station vertex, add an edge (bi, t) of capacity L.

   (b) Make a claim of how the original problem is related to the max-flow problem.

The problem has a solution if and only if the network has a max-flow of value n.

(c) Prove the above claim in both directions

Assume that there is a solution. It means that every client is connected to a correspondent base station. So we can push a flow of 1 from the source to each client. On the edges between clients and stations, we assign a flow of 1 or 0, depending whether a client is connected by a particular station or not. On the edges between stations and the sink, we assign a flow value equal to the number of clients covered by that station. This is possible, since we have a valid solution. It follows the max-flow is n.

Conversely, assume there is a max-flow of value n. This means that each client will get a flow of one unit. We also observe that no base station is overloaded due to the capacity condition L.

3. There are N students at the USC Viterbi school who want to celebrate the Indian festival of colour called Holi. There are M unique colors (call the set of colors C) available at the USC Bookstore and i^{th} color has c_i packets left at the store (e.g there are c_1 packets left of color 1, c_2 packets left of color 2, and so on till c_m packets of color M). The i^{th} student has a set $F_i \subseteq C$ representing their favourite colors and wishes to buy a total of b_i packets from their favourite color set (it doesn't matter which colors out of F_i as long as the total number of packets is b_i). However, to ensure fair availability of all colors, the USC Bookstore restricts each student to buy a maximum of 2 packets of the same color. Design an algorithm that determines if all the students can have their wish granted.

   (a) Describe how to construct a flow network

   There are N student nodes with have demand b_i and there are M color nodes with supply c_i. Then we can solve the problem using the demand-supply flow feasibility problem with the max capacities between the edges connecting colors to students being 2.

   Alternatively, we can ask if the following (see figure 3) max-flow problem has a max flow of $\sum_i^N b_i$

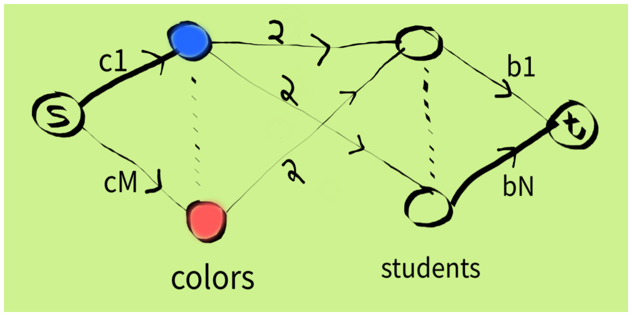


Figure 3: color-student network

   (b) Make a claim of how the original problem is related to the max-flow problem.

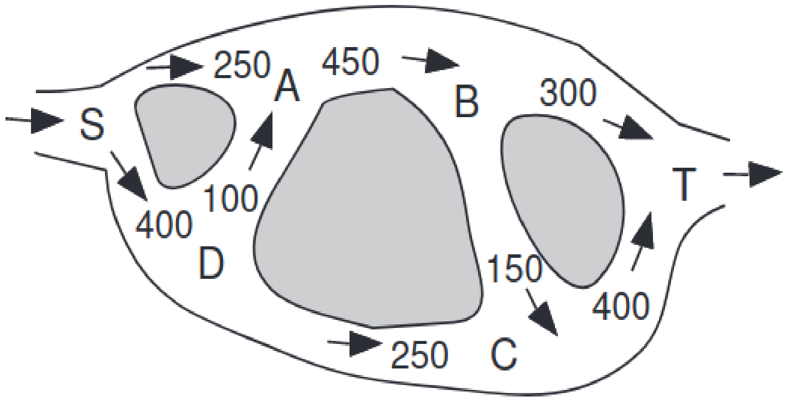
   The wishes can be granted if and only if the above constructed network has a max-flow (min-cut) of value $\sum_i^N b_i$.

   (c) Prove the above claim in both directions

If the edges from colors to students that have non-zero flow give the assignment of the color packets to the students based on the amount of flow in that color to student edge. Also since the capacities on the color to student nodes are each 2, no student is assigned more than two packets of the same color. Since the max flow is $\sum_i^N b_i$, all the edges in the students to target are saturated and hence every students wish is granted.

If there is a valid assignment of the color packets to the students then assign the links connecting color to student as the number of color packet assigned to them. Since the number of color packets assigned cannot be greater than 2 for a valid assignment, we note that the edge capacity constraints on the color to student nodes are not exceeded. Finally, since there is a valid assignment, the total number of assignments are $\sum_i^N b_i$ which means the total flow from color nodes to student nodes is $\sum_i^N b_i$. Since the min-cut consists of the edges from student to target nodes value $\sum_i^N b_i$, we have a max-flow situation.

4. In a museum S and T denote the entrance and exits of a hall. There are exhibits placed at the positions A, B, C, D. The hall has three gardens as shown in gray in the figure. There are hallways around the gardens that can support the movement of a specified number of people per hour which are marked in the figure. For example, people wanting to visit exhibit D can enter at S and go to D via the hallway which supports 400 visitors to pass through every hour. Similarly the hallway from exhibit B to C support 150 visitors per hour. The museum wants to attract 6000 visitors every day and closes when the target is reached for that day. If the museum opens at 8 am on a specific day, when is the earliest it can close on that day to support 6000 visitors?



The figure 4 shows the network representation of the museum hallways and also shows the maximum cut. Figure **??** shows the maximum flow. Since the max-flow is 600, the amount of hours required to support 6000 visitors is $\frac{6000}{600} = 10$ hours. Hence the earliest the museum can close is 6pm.

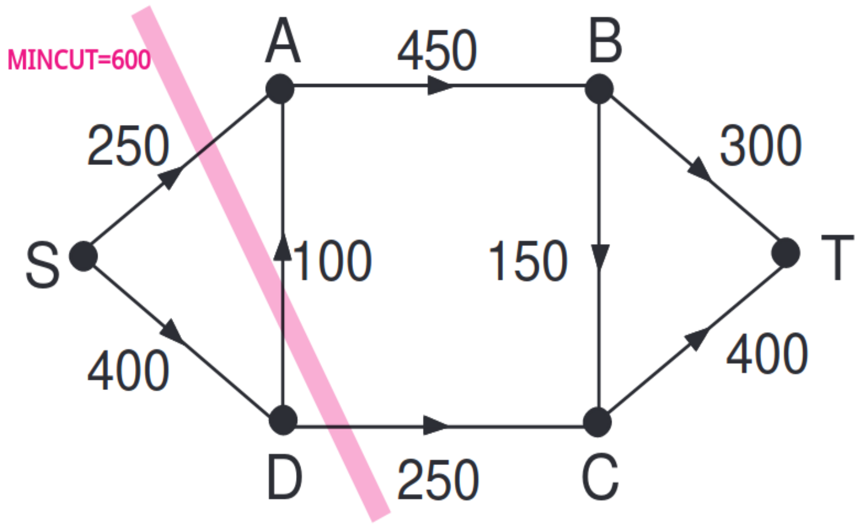


Figure 4: Flow rates in different paths

5. In addition to the edge capacity constraints, we introduce vertex capacities b_i constraints for each vertex v_i in a network-flow problem. The total amount of flow passing via vertex v_i cannot exceed b_i. Describe an algorithm to solve the vertex capacity constrained max-flow problem. Hint: Use reduction to edge capacity network flow problem.

   Create a new network where each vertex v is replaced by two vertices v_{in} and v_{out}. Connect a directed edge from v_{in} to v_{out} with capacity b_i. Connect all incoming edges on v to v_{in} and all outgoing edges from v to v_{out}. Now this problem is just an edge capacity constrained max-flow problem and can be solved by any standard max-flow algorithm.

6. The edge connectivity of an undirected graph is the minimum number of edges whose removal disconnects the graph. Describe an algorithm to compute the edge connectivity of an undirected graph with n vertices and m edges in $O(m^2n)$ time.

   For a cut $(S, \bar{S})$, let $c(S, \bar{S})$ denote the number of edges crossing the cut. By definition, the edge connectivity

   $$k = \min_{S \subset V} c(S, \bar{S})$$

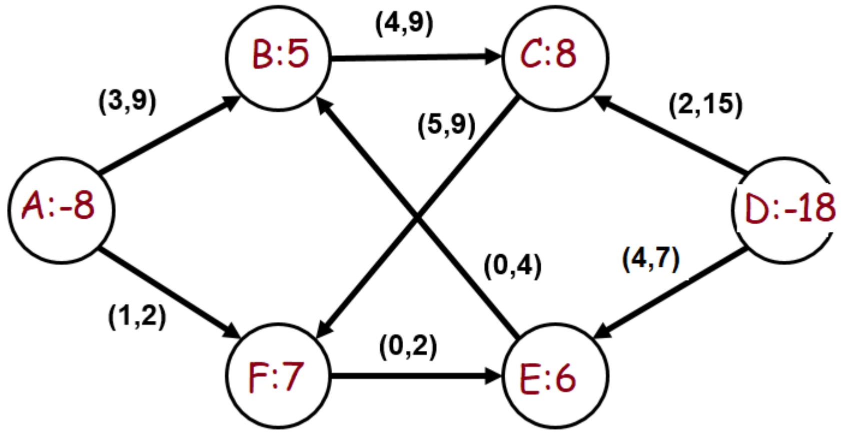
   Fix a vertex $u \in V$. For every cut $(S, \bar{S})$, there is a vertex $v \in V$ such that u and v are on either side of the cut. Let $C_{u,v}$ denote the value of the min u-v cut. Thus

   $$k = \min_{v \in V, v \neq u} C_{u,v}$$

   For each $v \neq u$, $C_{u,v}$ can be determined by computing the max flow from u to v. Since G is undirected, we need to implement an undirected variant of the max flow algorithm. Set all edge capacities to 1. During each step of the flow computation, search for an undirected augmenting path using breadth first search or depth first search and send a flow of 1 through this path. Keep track of the minimum size cut found during the max-flow computation for each vertex pair.

   There are $n - 1$ flow computations and each flow computation takes at most $O(m^2)$ time. So the overall time complexity is $O(m^2n)$.

7. In the network below, the demand values are shown on vertices. Lower bounds on flow and edge capacities are shown as (lower bound, capacity) for each edge.



Answer the following questions:

(a) Remove the lower bounds on each edge. Write down the new demands on each vertex A, B, C, D, E, and F in this order.

D(A)=-4, D(B)=6, D(C)=7, D(D)=-12, D(E)=2, D(F)=1;

(b) Solve the circulation problem without lower bounds. Write down the max-flow value (without lower bound).

16

(c) Is there is a feasible circulation in the original graph? Explain your answer.

It is feasible because the maxflow value equals the sum of demands.

8. Consider a student dormitory with n students and m rooms. A student can only be assigned to one room. Each room has the capacity to hold either one or two students. Each student has a subset of rooms as their possible choice. We also need to make sure that there is at least one student assigned to each room. Give a polynomial time algorithm that determines whether a feasible assignment of students to rooms is possible that meets all of the above constraints. If there is a feasible assignment, describe how your solution can identify which student is assigned to which room.

   Construct a flow network as follows,

   - For each student i create a vertex ai, for each room j create a vertex bj , if room j is one of student i's possible choices, add an edge from ai to bj with upper bound 1.
   - Create a super source s, connect s to all ai with an edge of lower bound 0 and upper bound 1.
   - Create a super sink t, connect all bj to t with an edge of lower bound 1 and upper bound 2.
   - Connect t to s with an edge of lower bound and upper bound n.
   - Find an integral circulation of the graph, because all edges capacity and lower bound are integer, this can be done in polynomial time. If there is one, for each ai and bj , check whether the flow from ai to bj is 1, if yes, assign student i to room j.

   We can identify the student assignment by checking the edges from ai to bj having a non-zero flow value.

9. Suppose that you have just bought a new computer and you want to install software on that. Specifically, two companies, which you can think of like Microsoft and Apple, are trying to sell their own copy of n different products, like Operation System. Spread Sheet, Web Browser. For each product i, $i \in \{1, 2, \dots, n\}$, we have

   - the price $p_i \geq 0$ that Microsoft charges and the price $p'_i \geq 0$ that Apple charges.

- the quality $q_i \geq 0$ of Microsoft version and the quality $q'_i \geq 0$ of Apple version.

For example, Apple may provide a better Web Browser Safari, but Microsoft a better Word Processor. You want to assemble your favorite computer by installing exactly one copy of each of the n products, e.g. you want to buy one operating system, one Web Browser, one Word Processor, etc. However, you don't want to spend too much money on that. Therefore, your goal is to maximize the quality minus total price.

However, as you may know, the products of different companies may not be compatible. More concretely, for each product pair (i, j), we will suffer a penalty $\tau_{ij} \geq 0$ if we install product i of Microsoft and product j of Apple. Note that τ_{ij} may not be equal to τ_{ji} just because Apple's Safari does not work well on Microsoft Windows doesn't mean that Microsoft's Edge does not work well in Mac-OS. We assume that products are always compatible internally, which means that there is no penalty for installing two products from the same company. All pairwise penalties will be subtracted from the total quality of the system.

Your task is then to give a polynomial-time algorithm for computing which product i to purchase from which of the two companies (Apple and Microsoft) for all $i \in \{1, 2, \dots, n\}$, to maximize the total system quality (including the penalties) minus the total price. Prove the correctness of your algorithm.

(a) Describe how to model this problem as a min-cut problem.

In this problem, we actually need to separate the objects into two parts A (Microsoft) and B (Apple) so that we can maximize the following:

$$\max \sum_{i \in A} q_i + \sum_{j \in B} q'_j - \sum_{i \in A, j \in B} \tau_{ij} - \sum_{i \in A} p_i - \sum_{j \in B} p'_j.$$

With a little calculation, we can transform the above objective function as follows:

$$\sum_{i\in A} q_i + \sum_{j\in B} q'_j - \sum_{i\in A, j\in B} \tau_{ij} - \sum_{i\in A} p_i - \sum_{j\in B} p'_j$$
$$= \sum_{i\in[n]} (q_i + q'_i) - \left(\sum_{j\in B} q_j + \sum_{i\in A} q'_i + \sum_{i\in A, j\in B} \tau_{ij} + \sum_{i\in A} p_i + \sum_{j\in B} p'_j \right).$$

Note that $\sum_{i\in[n]}(q_i + q'_i)$ is a constant in this problem. Therefore, to maximize the original objective, we are to minimize the following:

$$\min \left(\sum_{j\in B} q_j + \sum_{i\in A} q'_i + \sum_{i\in A, j\in B} \tau_{ij} + \sum_{i\in A} p_i + \sum_{j\in B} p'_j \right).$$

Then we can model this problem as a min-cut problem by constructing the graph as follows. First, we add a source node s and a sink node t. For each object j, we add one node A_j. We add an edge from S to A_j with capacity $p'_i + q_i$ and add an edge from each A_j to T with capacity $p_j + q'_j$. For the correlation between objects, we add an edge from A_i to A_j with capacity τ_{ij}.

(b) Make a claim of how the original problem is related to the min-cut problem.

The above minimization problem has an optimal value v if and only if the above constructed network has a max-flow (min-cut) of value v.

(c) Prove the above claim in both directions

If there is a cut of value v in the network, then actually this cut contains two parts of nodes. One part contains S and the other part contains T. Then, we just assign the objects corresponding to the nodes belonging to S to A and assign the objects corresponding to the nodes belonging to T to B. For the cut value, if node i is in A and node j is in B, we will cut the edges from S to A_j, A_i to T and A_i to A_j. These edges are of total capacity $p'_j + q_j + p_i + q'_i + \tau_{ij}$, which is exactly the corresponding term in the above objective function. Therefore, we can achieve v by assigning n objects to either A or B.

On the other hand, if the optimal value of the minimization problem is v, then there must be an assignment of the n objects. Then we just cut the network into two parts. One part includes S and objects in A and the other part includes T and objects in B. It is similar to the former direction to show that the cut value here equals v. Based on the above observations, we prove the claim.

Therefore, our algorithm is to first construct the above network, use Edmund-Karp (as here we require polynomial algorithm) to obtain a max-flow of that graph, traverse along the max-flow to obtain a corresponding min-cut and assign the objects belonging to S to Microsoft and the others to Apple. The construction of graph, Edmund-Karp algorithm, traversing and assigning can all be done in polynomial time. The correctness is proved by the above analysis.

10. **Online Questions.** Please go to DEN (https://courses.uscden.net/) and take the online portion of your assignment.