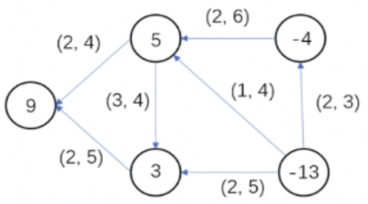
# Homework 9

1. In the network $G$ below, the demand values are shown on vertices (supply value if negative). Lower bounds on flow and edge capacities are shown as (lower bound, capacity) for each edge. Determine if there is a feasible circulation in this graph. You need to show all your steps. (20 pt)

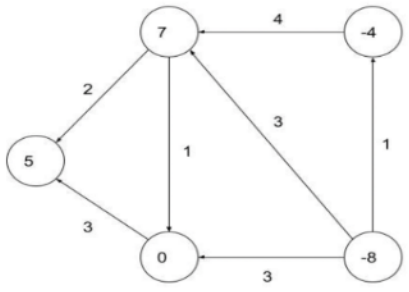


   (a) Reduce the Feasible Circulation with Lower Bounds problem to a Feasible Circulation problem without lower bounds. (8 pt)

   Solution: Steps to follow :-

   i. Assign flows to edges to satisfy lower bounds.
   ii. At each node $v$, $L_v = (f_{in} - f_{out})$, followed by $D' = D - L_v$
   iii. At each edge, $cap = cap - LB$

   Resultant network:

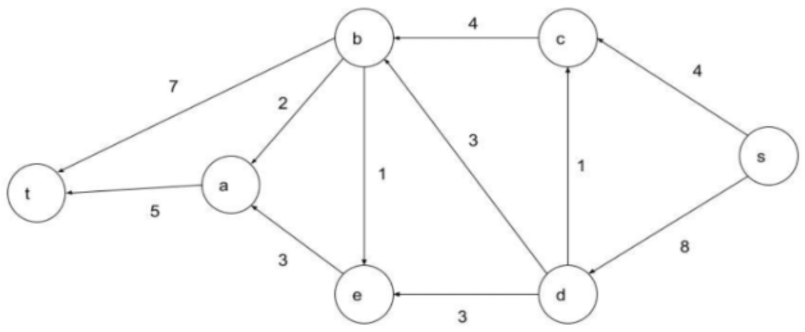


   Rubric:

   - 4 pts: If all edge values: (Upper_bound - Lower_bound) is correctly calculated
   - 4 pts: If all Node Values ($D' = D - (f_{in} - f_{out})$) is correctly calculated
   - -2 pts: If no steps explained

   (b) Reduce the Feasible Circulation problem obtained in part $a$ to a Maximum Flow problem in a Flow Network. (8 pt)

   Solution: The max flow problem is as follows:



Rubric:

- 2 pts: Add $S$ and $T$
- 2 pts: Connect $S$ with correct nodes
- 2 pts: Connect $T$ with correct nodes
- 2 pts: Give correct capacities to edges

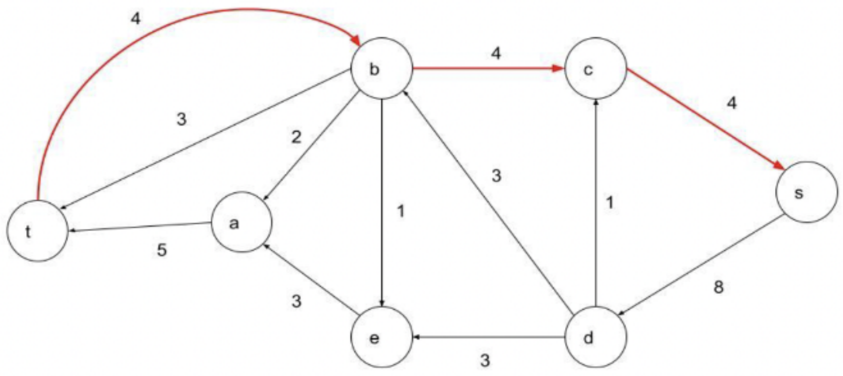
(c) Using the solution to the resulting Max Flow problem explain whether there is a Feasible Circulation in G. (4 pt)

Solution:

Candidate Solution 1:

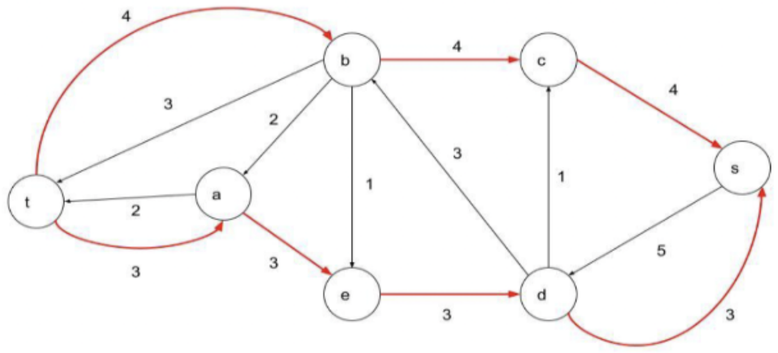
First Augmenting Path: $s$ -> $c$ -> $b$ -> $t$ with $flow = 4$

Residual Graph 1:

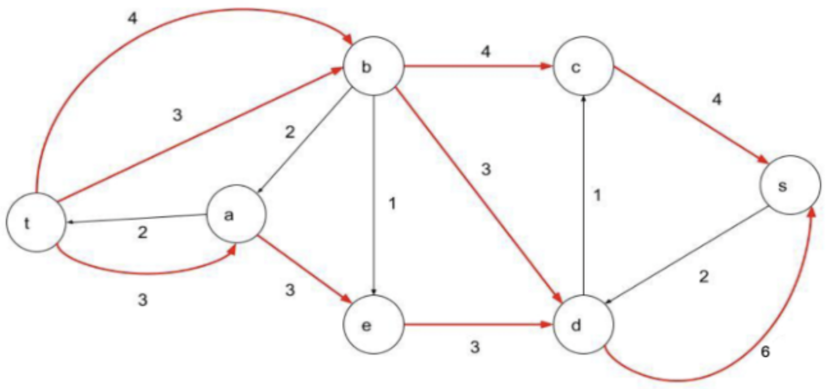


Second Augmenting Path: $s$ -> $d$ -> $e$ -> $a$ -> $t$ with $flow = 3$

Residual Graph 2:



Third Augmenting Path: $s$ -> $d$ -> $b$ -> $t$ with $flow = 3$
Residual Graph 3:



Candidate Solution 2:
First Augmenting Path: $s$ -> $c$ -> $b$ -> $t$ with $flow = 4$
Second Augmenting Path: $s$ -> $d$ -> $b$ -> $a$ -> $t$ with $flow = 2$
Third Augmenting Path: $s$ -> $d$ -> $b$ -> $t$ with $flow = 1$
Fourth Augmenting Path: $s$ -> $d$ -> $e$ -> $a$ -> $t$ with $flow = 3$

Candidate Solution 3:
First Augmenting Path: $s$ -> $d$ -> $e$ -> $a$ -> $t$ with $flow = 3$
Second Augmenting Path: $s$ -> $d$ -> $b$ -> $a$ -> $t$ with $flow = 2$
Third Augmenting Path: $s$ -> $d$ -> $c$ -> $b$ -> $t$ with $flow = 1$
Fourth Augmenting Path: $s$ -> $c$ -> $b$ -> $t$ with $flow = 3$
Fifth Augmenting Path: $s$ -> $d$ -> $b$ -> $t$ with $flow = 1$

Max Flow = 10

Since, the value of Max Flow is less than the total demand value $D = 12$, there is No Feasible solution in the circulation network, and therefore there is no feasible circulation in the circulation with lower bounds network.

Rubric:

- 2 pts: Finding correct Max-Flow and presenting their appropriate residual graphs according to the sequence of augmenting paths that the student has chosen
- 1 pt: Correctly concluding "No Feasible Circulation"
- 1 pt: Reasoning behind "No Feasible Circulation"

2. Determine if the following statements are true or false. Give reasoning for your answer. (12 pt)

   (a) There is a feasible circulation with demands $\{d_v\}$ if $\Sigma_v d_v = 0$. (3 pt)
   Solution: False, this is not the only condition for a feasible circulation but a necessary one. E.g., it should be possible for the flow to be adequately transferred by the edges of the network.

   (b) In a flow network, the value of flow from S to T can be higher than the maximum number of edge disjoint paths from S to T. (3 pt)
   Solution: True, consider the case when edges have capacity greater than 1.

   (c) There always exists a maximum flow without a cycle containing positive flow. (3 pt)
   Solution: True, we can always remove the flow from the cycle and still get the same $s - t$ flow.

   (d) If you have non integer edge capacities, then you cannot have an integer max flow. (3 pt)
   Solution: False, consider a graph with source $s$, sink $t$ and two nodes $a$ and $b$. Let's say there is an edge from $s$ to both $a$ and $b$ with capacity 0.5 and from both $a$ and $b$ to $t$ with capacity 0.5, then the max flow for this graph is 1, which is an integer.
   Rubric [For Every Question]:

- 1 pt: Correct choice
- 2 pt: Valid reasoning for the answer

3. Solve Kleinberg and Tardos, Chapter 7, Exercise 7. (10 pt)

Solution: We build the following flow network. There is a node $v_i$ for each client $i$, a node $w_j$ for each base station $j$, and an edge $(vi, wj)$ of capacity 1 if client $i$ is within range of base station $j$. We then connect a super-source $s$ to each of the client nodes by an edge of capacity 1, and we connect each of the base station nodes to a super-sink $t$ by an edge of capacity $L$. We claim that there is a feasible way to connect all clients to base stations if and only if there is an $s$–$t$ flow of value $n$. If there is a feasible connection, then we send one unit of flow from $s$ to $t$ along each of the paths $s, vi, wj, t$, where client $i$ is connected to base station $j$. This does not violate the capacity conditions, in particular on the edges $(wj, t)$, due to the load constraints. Conversely, if there is a flow of value $n$, then there is one with integer values. We connect client $i$ to base station $j$ if the edge $(vi, wj)$ carries one unit of flow, and we observe that the capacity condition ensures that no base station is overloaded. The time complexity is the time required to solve a max-flow problem on a graph with $O(n+k)$ nodes and $O(nk)$ edges. This can be made to run in polynomial time with any algorithm that solves the max flow problem in polynomial time.

Rubric:

- 4 pts: Correct Construction of graph.
- 2 pts: Proving that a valid solution of the flow problem corresponds to a valid solution for the given problem.
- 2 pts: Proving that a valid solution for the given problem corresponds to a valid solution of the flow problem.
- 2 pts: For polynomial time complexity

4. Consider a case where there are $n$ tasks, with time requirements $r_1, r_2, ..., r_n$ hours. On the project team, there are $k$ people with time availabilities $a_1, a_2, ..., a_k$. For each task $i$ and person $j$, you are told if person $j$ has the skills to do task $i$. You are to decide if the tasks can be split up among the people so that all tasks get done. People only execute tasks they are qualified for, and no one exceeds his time availability. Remember that you can split up one task between multiple qualified people. Now, in addition, there are group constraints. For instance, even if each of you and your two teammates can in principle spend 4 hours on the project, you may have decided that between the three of you, you only want to spend 10 hours. Formally, we assume that there are $m$ sets $S_j \subseteq \{1, ..., k\}$ of people, with set constraints $t_j$ . Then, any valid solution must ensure, in addition to the previous constraints, that the combined work of all people in $S_j$ does not exceed $t_j$, for all $j$. Note that one person may belong to at most one constraint set. Give an algorithm with running time polynomial in $n, m, k$ for this problem, under the assumption that all the $S_j$ are disjoint, and prove that your algorithm is correct. (20 pt)

Solution: We will have one node $u_h$ for each task $h$, one node $v_i$ for each person i, and one node $w_j$ for each constraint set $S_j$. In addition, there is a source $s$ and a sink $t$. The source connects to each node $u_h$ with capacity $r_h$. Each $u_h$ connects to each node $v_i$ such that person $i$ is able to do task $h$, with infinite capacity. If a person $i$ is in no constraint set, node $v_i$ connects to the sink $t$ with capacity $a_i$. Otherwise, it connects to the node $w_j$ for the constraint set $S_j$ with $i \in S_j$, with capacity $a_i$. (Notice that because the constraint sets are disjoint, each person only connects to one set.) Finally, each node $w_j$ connects to the sink $t$ with capacity $t_j$.

We claim that this network has an s-t flow of value at least $\Sigma_h r_h$ if and only if the tasks can be divided between the people based on their availability. For the forward proof, assume that there is such a flow. For each person $i$, assign her/him to do as many units of work on task $h$ as the flow from $u_h$ to $v_i$. As the flow value is at least $\Sigma_h r_h$, the flow saturates all the edges out of the source (the total capacity out of the source is only $\Sigma_h r_h$ ), and by flow conservation, each job is fully assigned to people. Also, because the capacity on the (unique) edges out of $v_i$ is $a_i$, no person does more than $a_i$ units of work. Moreover, because the only way to send on the flow into $v_i$ is to the node $w_j$ for nodes $i \in S_j$, by the capacity constraint on the edge $(w_j, t)$, the total work done by people in $S_j$ is at most $t_j$. Conversely, if we have an assignment that has person $i$ doing $x_i h$ units of work on task $h$, meeting all constraints, then we send $x_i h$ units of flow along the path $s - u_h - v_i - t$ (if person $i$ is in no constraint sets), or along $s - u_h - v_i - w_j - t$, if person $i$ is in constraint set $S_j$. This clearly satisfies conservation and non-negativity. The flow along each edge $(s, u_h)$ is exactly $r_h$, because that is the total amount of work assigned on job $h$. The flow along the edge $(v_i, t)$ or $(v_i, s_j)$ is at most $a_i$, because that is the maximum amount of work assigned to person $i$. And the total flow along $(w_j, t)$ is at most $t_j$, because each constraint was satisfied by the assignment. So we have exhibited an $s - t$ flow of total value at least $\Sigma_h r_h$ .

The running is the time required to solve a max-flow problem on a graph with $O(n+k+m)$ nodes and $O(nk+m)$ edges. This can be made to run in polynomial time with any algorithm that solves the max flow problem in polynomial time.

Rubric:

- 10 pts: Correct Construction of Network Flow graph.
- $-2$ pts: For each incorrect edge weight/node.
- 4 pts: Proving that a valid solution of the flow problem corresponds to a valid solution for the given problem.
- 4 pts: Proving that a valid solution for the given problem corresponds to a valid solution of the flow problem.
- 2 pts: for polynomial time complexity.

5. A company has $n$ different teams. Each team $i$ has $t_i$ members. In order to improve inter-team communication, the company has decided to organize $m$ events. Each event $j$ can have at most $e_j$ attendees. To encourage attendees to communicate with other teams, the company has decided that each event should not have more than two attendees from the same team. Also, there are certain events which need attendees from some specific teams. More specifically, there are $k$ constraints of the form $(a, b)$ which means that the company needs at least one attendee from team $a$ for event $b$. The company wants every employee to attend an event. The company also wants every event to be full. Give an efficient solution to assign employees to events, such that all of the company's requirements are fulfilled. (20 pt)

Solution:

Solution: We reduce the given problem to a circulation flow with lower bound problem where units of flow correspond to employees getting assigned to an event. We construct a circulation problem in the following way :

- For each team $i$, we have a node $u_i$ and for each event $j$, we have a node $v_j$.
- Each node $u_i$ has a supply of $t_i$ and each node $v_j$ has a demand of $e_j$.
- Each node $u_i$ is connected to each node $v_j$ through an edge with capacity 2.
- For every constraint $(a, b)$, the edge $u_a$ to $v_b$ will have a lower bound of 1.

We claim the following: There is a valid employee-event assignment (fulfilling all of the company's requirements) if and only if there is a feasible circulation in G.

Forward Proof: Firstly, let's say there is a valid employee-event mapping. Then, for every employee from department $i$ assigned to an event $j$, there will be a flow of 1 going from node $u_i$ to $v_j$. Since, the employee-event mapping needs to follow the $k$ constraints given by the company, there will be at least one flow from $u_a$ to $v_b$ for each given constraint $(a, b)$. Hence, the lower bound constraints of the graph is fulfilled. Since, all the events need to be full and every employee needs to be assigned an event, we know that $\Sigma_i t_i = \Sigma_j e_j$ and hence, $\Sigma_i s_i = \Sigma_j d_j$ (i.e. supply and demand are equal). Since, all the employees need to be assigned to an event there will be a flow of $\Sigma_i t_i$ from supply nodes to demand nodes and hence, there will be a valid solution for the circulation problem.

Backward Proof: Let's say there is a feasible circulation for the given graph G. Each flow from $u_i$ to $v_j$ will represent a member from team $i$ assigned to event $j$. Since, there is a capacity of 2 for each edge, any event will not have more than 2 members from the same team. Also, for each constraint $(a, b)$, there is a lower bound of 1 on the edge $a$ to $b$, and hence all of the constraints will be satisfied. Since there is a feasible circulation, all the supply is being flowed to the demands, which means that every employee is assigned to an event and every event is full. Hence, if there is a feasible circulation, then there exists a valid employee-events mapping which satisfies all of the company's requirements.

Rubric:

- 10 pts: Correct Construction of Circulation Flow graph.
- $-2$ pts: For each incorrect edge weight/node.
- 4 pts: Proving that a valid solution of the flow problem corresponds to a valid solution for the given problem.
- 4 pts: Proving that a valid solution for the given problem corresponds to a valid solution of the flow problem.
- 2 pts: for polynomial time complexity.

6. Solve Kleinberg and Tardos, Chapter 7, Exercise 31. (20 pt)

Solution: We reduce the given problem to a circulation flow with lower bound problem where units of flow correspond to sets of boxes nested inside one visible box. We construct the following graph G:

- For each box $i$, G has two nodes $u_i$ and $v_i$ and an edge between them that corresponds to this box. This edge $(u_i, v_i)$ has a lower bound of 1 and a capacity of 1. (Each box is exactly in one set of boxes nested one in another.)
- For each pair of boxes, $i$ and $j$, if box $j$ can nest inside box $i$, there is an edge $(v_i, u_j)$ with a lower bound of 0 and a capacity of 1. (One can store box j inside i.)

- G also has a source node $s$, with demand $-k$ (corresponding to the back room where boxes are stored) and a sink node $t$, with demand $k$ (corresponding to nothing inside empty boxes).
- For each box $i$, G has an edge $(s, u_i)$ with a lower bound of 0 and a capacity of 1. (Any box can be visible.)
- For each box $j$, G has an edge $(v_j, t)$ with a lower bound of 0 and a capacity of 1. (Any box can be empty.)

We claim the following: There is a nesting arrangement with $k$ visible boxes if and only of there is a feasible circulation in G with demand $-k$ in the source node s and demand $k$ in the sink $t$.

Proof: First, suppose there is a nesting arrangement with $k$ visible boxes. Each sequence of nested boxes inside one visible box $i_1, i_2, ..., i_n$ defines a path from $s$ to $t$: $(s, u_{i_1}, v_{i_1}, u_{i_2}, v_{i_2}, ..., u_{i_n}, v_{i_n}, t)$ Therefore we have $k$ paths from $s$ to $t$. The circulation corresponding to all these paths satisfy all demands, capacity and lower bound. Conversely, consider a feasible circulation in our network. Without lost of generality, assume that this circulation has integer flow values. There are exactly $k$ edges going to $t$ that carry one unit of flow. Consider one of such edges $(v_i, t)$. We know that $(u_i, v_i)$ has one unit of flow. Therefore, there is a unique edge into $u_i$ that carries one unit of flow. If this edge is of the kind $(v_j, u_i)$ then put box $i$ inside $j$ and continue with box $j$. If this edge of the kind $(s, u_i)$, then put the box $i$ in the back room. This box became visible. Continuing in this way we pack all boxes into $k$ visible ones. So we can answer the question whether there is a nesting arrangement with exactly $k$ visible boxes. Now to find the minimum possible number of visible boxes we answer this question for k = 1, 2, 3, and so on, until we find a positive answer. The maximum number of this iteration is n, therefore the algorithm is polynomial since we can find a feasible circulation in polynomial time.

Alternative Solution - We can also optimize the above solution by doing a binary search on $k$, reducing the number of iterations to $log(k)$.

Rubric:

- 10 pts: Correct Construction of Circulation Flow graph.
- $-2$ pts: For each incorrect edge weight/node.
- 4 pts: Proving that a valid solution of the flow problem corresponds to a valid solution for the given problem.
- 4 pts: Proving that a valid solution for the given problem corresponds to a valid solution of the flow problem.
- 2 pts: for polynomial time complexity.
- No points to be deducted for not suggesting the binary search solution.