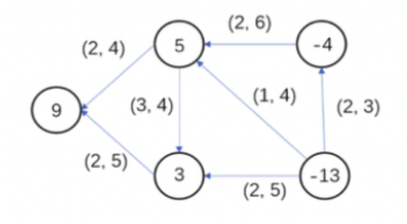
# CSCI 270 - Spring 2023 - HW9 Solutions

## Graded Problems

### Problem 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. (25 pt)

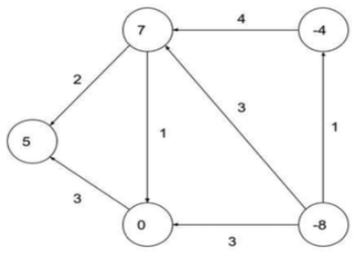


(a) Reduce the Feasible Circulation with Lower Bounds problem to a Feasible Circulation problem without lower bounds. (10 pt)

Solution: Steps to follow

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

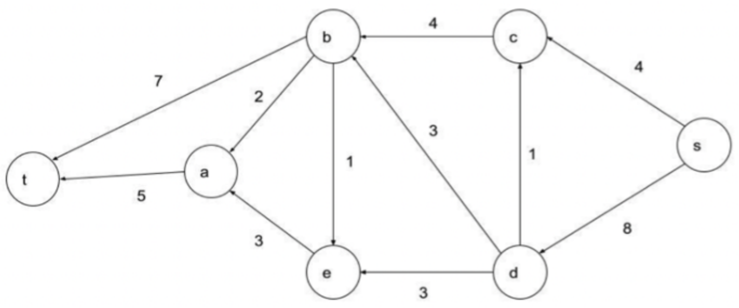
Resultant network:



Rubric
- 5 pts: If all edge values: (Upper Bound - Lower Bound) is correctly calculated
- 5 pts: If all Node Values ($D' = D - (f_{in} - f_{out})$ is correctly calculated
- -3 pts: If no steps explained

(b)Reduce the Feasible Circulation problem obtained in part a to a Maximum Flow problem in a Flow Network. (10 pt)

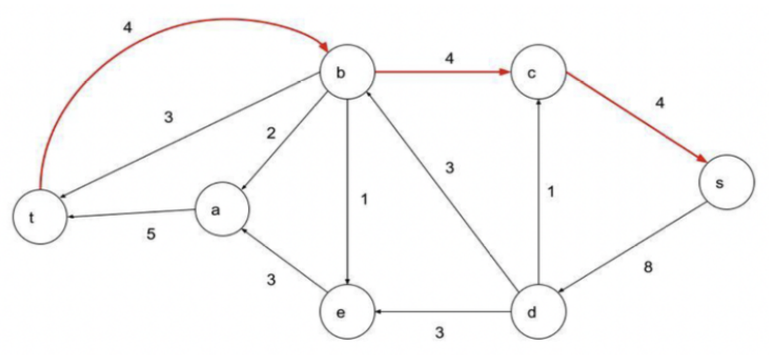
Solution: The max flow problem is as follows:



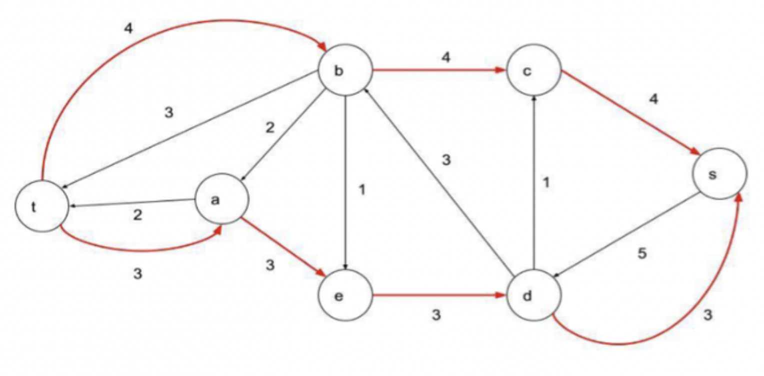
Rubric
- 2.5 pts: Add S and T
- 2.5 pts: Connect S with correct nodes
- 2.5 pts: Connect T with correct nodes
- 2.5 pts: Give correct capacities to edges

(c) Using the solution to the resulting Max Flow problem explain whether there is a Feasible Circulation in G. (5 pt)

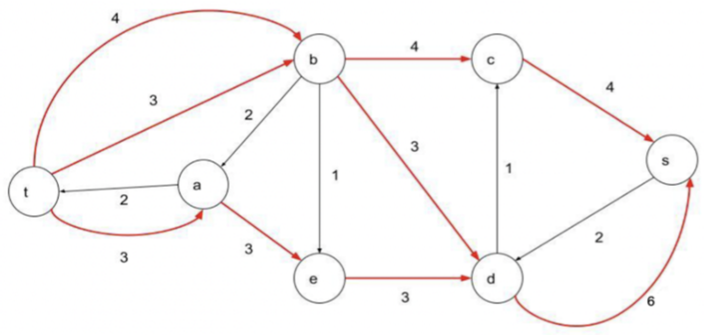
Solution:
Candidate Solution 1:
First Augmenting Path: $s \to c \to b \to t$ with flow $= 4$
Residual Graph 1:



Second Augmenting Path: $s \to d \to e \to a \to t$ with flow =3
Residual Graph 2:



Third Augmenting Path: $s \to d \to b \to t$ with flow $= 3$
Residual Graph 3:



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

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

## Problem 2

Solve Kleinberg and Tardos, Chapter 7, Exercise 31. (25 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

- 11 pts: Correct Construction of Circulation Flow graph.
- -2 pts: For each incorrect edge weight/node.
- 6 pts: Proving that a valid solution of the flow problem corresponds to a valid solution for the given problem.
- 6 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.

## Problem 3

At a dinner party, there are $n$ families $\{a_1, a_2, ..., a_n\}$ and m tables $\{b_1, b_2, ..., b_m\}$. The $i$th family $a_i$ has $g_i$ members and the $j$th table $b_j$ has $h_j$ seats. Everyone is interested in making new friends and the dinner party planner wants to seat people such that no two members of the same family are seated in the same table. Design an algorithm that decides if there exists a seating assignment such that everyone is seated and no two members of the same family are seated at the same table. (25 pt)

Solution:
Construct the following network $G = (V, E)$. For every family introduce a vertex and for every table introduce a vertex. Let $a_i$ denote the vertex corresponding to the $i$th family and let $b_j$ denote the vertex corresponding to the jth table. From every family vertex $a_i$ to every table vertex $b_j$, add an edge $(a_i, b_j)$ of capacity 1. Add two more vertices $s$ and $t$. To every family vertex $a_i$ add an edge $(s, a_i)$ of capacity $g_i$. From every table vertex $b_j$ add an edge $(b_j, t)$ of capacity $h_j$.

Claim: There exists a valid seating if and only if the value of max flow from s to t in the above network equals $g_1 + g_2 + ... + g_n$.

Proof of Claim: Assume there exists a valid seating, that is a seating where every one is seated and no two members in a family are seated at a table. We construct a flow f to the network as follows. If a member of the $i$th family is seated at the $j$th table in the seating assignment, then assign a flow of 1 to the edge $(a_i, b_j)$. Else assign a flow of 0 to the edge $(a_i, b_j)$. The edge $(s, a_i)$ is assigned a flow equaling the number of members in the $i$th family that are seated (which since the seating is valid equals $g_i$). Likewise the edge $(b_j, t)$ is assigned a flow equaling the number of seats taken in the table $b_j$ (which since the seating is valid is at most $h_j$). Clearly the assignment is valid since by construction the capacity and conservation constrains are satisfied. Further, the value of the flow equals $g_1 + g_2 + ... + g_n$.

Conversely, assume that the value of the max $s - t$ flow equals $g_1 + g_2 + ... + g_n$. Since the capacities are integers, by the correctness of the Ford-Fulkerson algorithm, there exists a maxflow (call $f$) such that the flow assigned to every edge is an integer. In particular, every edge between the family vertices and table vertices has a flow of either 0 or 1 (since these edges are of capacity 1). Construct a seating assignment as follows, seat a person of the $i$th family at the $j$th table if and only if $f(a_i, b_j)$ is 1. By construction at most one member of a family is seated at a table. Since the value of $f$ equals the capacity of the cut $(\{s\}, V - \{s\})$, every edge out of $s$ is saturated. Thus by flow conservation at $a_i$, for every $a_i$ the number of edges out of $a_i$ with a flow of 1 is $g_i$. Thus in the seating assignment, every one is seated. Further, since the flow $f(b_j, t)$ out of $b_j$ is at most $h_j$, at most $h_j$ persons are seated at table $b_j$. Thus we have a valid seating.

Remark. You can solve the problem using "circulation" as well.

One way is to modify the above network by adding (for every $a_i$) a lower bound of $g_i$ to the edge $(s, a_i)$ and adding an edge $(t, s)$ of infinite capacity. Then the modified network has a valid circulation if and only if there is a valid seating.

Another way is to remove s and its incident edge from the above network and add at t a demand of $g_1 + g_2 + .. + g_n$ and at each $a_i$ a demand of $-g_j$. The modified network has a circulation if and only if there is a valid seating.

Rubric
Construction (11pt):
- 5 pt: Constructing the correct Family to Table Graph
- 3 pt: Constructing the correct Source and Sink
- 3 pt: Assign right capacity to each edge

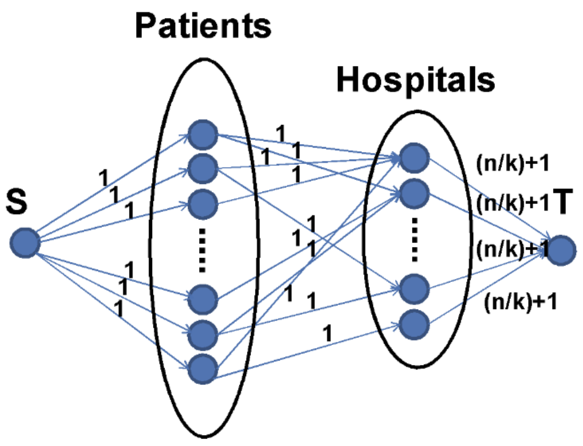
Proof (14pt):
- 7 pt: Prove if there is an answer in the flow, there is an answer to the question
- 7 pt: Prove if there is an answer in the question, there is an answer to the flow

## Problem 4

Due to large-scale flooding in a region, paramedics have identified a set of $n$ injured people distributed across the region who need to be rushed to hospitals. There are $k$ hospitals in the region, and each of the n people needs to be brought to a hospital that is within a half-hour's drive to their current location. (So different patients will be able to be served by different hospitals depending upon the patients' locations.) However, overloading one hospital with too many patients at the same time is undesirable, so we would like to distribute the patients as evenly as possible across all the hospitals. So the paramedics (or a centralized service advising the paramedics) would like to work out whether they can choose a hospital for each of the injured people in such a way that each hospital receives at most $(n/k + 1)$ patients. (25 pt)

(a) Describe a procedure that takes the given information about the patients' locations (hence specifying which hospital each patient could go to) and determines whether a balanced allocation of patients is possible (i.e.each hospital receives at most $(n/k+1)$ patients). (11pt)

Solution:



**If the hospital is within the patient half-hour's drive we connect them**

(b) Provide proof of correctness for your procedure. (10pt)

Solution:
Each unit flow from $S$ to $T$ is equivalent to assigning a patient to a hospital with the

following restriction:

Each hospital get less than $(n/k) + 1$ patients. Each patient will be assign to only one hospital which is located in the half-hour's drive to the patient

We use the ford-Fulkerson algorithm to find the maximum flow. The maximum flow is the maximum of this assignment and if we can assign all patients (max flow $= n$) we can do the balance allocation.

(c) What is the asymptotic running time of your procedure (in terms of $n$ and $k$)?(4pt)

Solution:
We use the Ford-Fulkerson algorithm to solve this problem and in for this algorithm the running time is $O(Cm)$ in which the $C$ is the maximum possible flow and m is the number of edges. Thus, $C = n$ and $m = n + nk + k$ and the complexity will be $O(n(n + k + nk)) = O(n^2k)$.

Rubric
a)
• 5 pt: Constructing the correct Patients to Hospital Graph
• 3 pt: Constructing the correct Source and Sink
• 3 pt: Assign right capacity to each edge
b)
• 5 pt: Prove if there is an answer in the flow, there is an answer to the question
• 5 pt: Prove if there is an answer in the question, there is an answer to the flow
c)
• 2 pt: Correct FF algorithm run time
• 2 pt: Correct number of edges

## Practice Problems

### Problem 5

Solve Kleinberg and Tardos, Chapter 7, Exercise 7.

Solution:
We build the following flow network. There is a node vi for each client $i$, a node $w_j$ for each base station $j$, and an edge $(v_i, w_j)$ 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 $s, v_i, w_j, t$, where client $i$ is connected to base station j. This does not violate the capacity conditions, in particular on the edges $(w_j, t)$, 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 $(v_i, w_j)$ 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