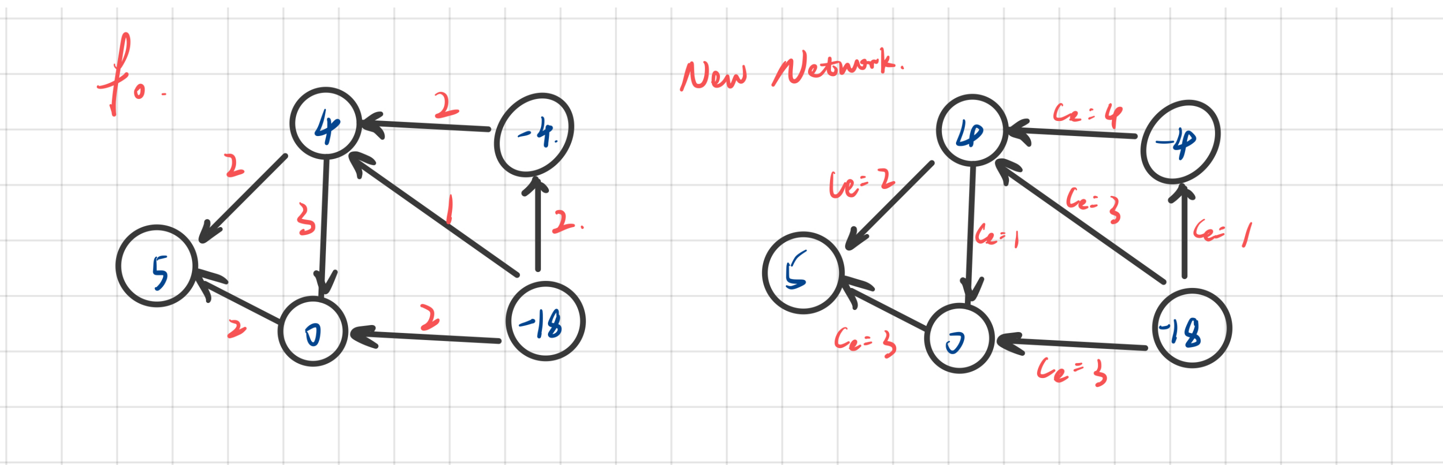
CSCI 570 - Homework 09

- **Author:** Boyang Xiao
- **Due Date:** Nov. 2nd 2022
- **USC id**: 3326-7302-74
- **Email**: boyangxi@usc.edu

Problem 1

- **(a)**

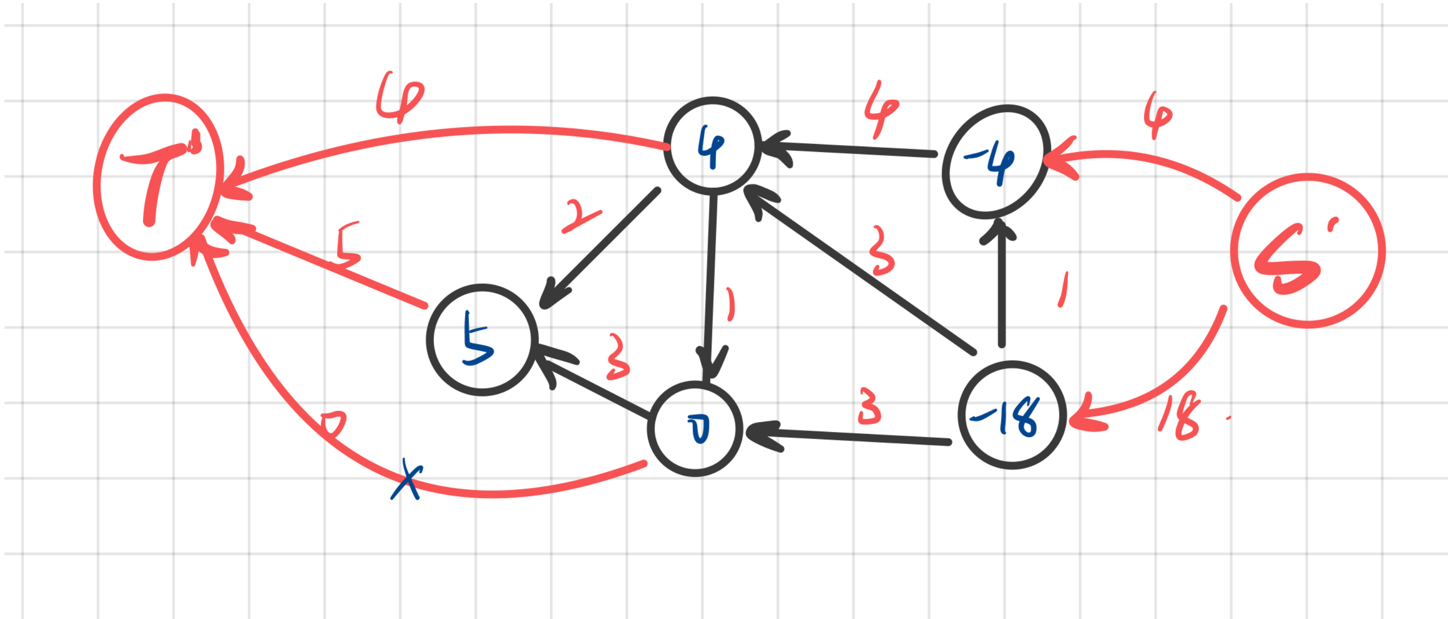


We first construct a flow network as shown in the figure on the left, where . And we construct a new network according to where:

- Each edge's new capacity is:
- Each node's demand:

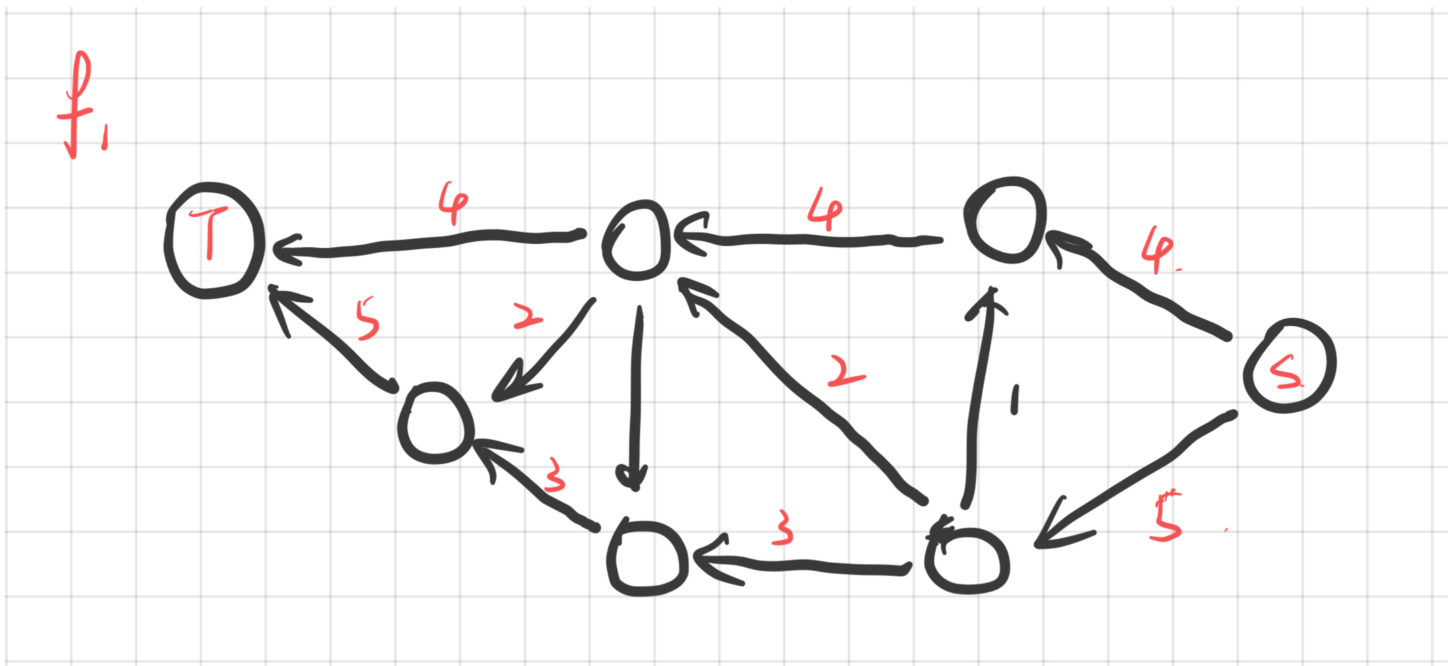
The new network can be treated as a feasible circulation problem without lower bounds.

- **(b)**



We create a super Source node and super Sink node as shown in the figure above. With all capacity denoted, we can treate this as a Max flow problem now.

- **(c)**



The figure above is the max flow we get from part (b). Since the value of this max flow:
, we can tell that **there is no feasible circulation in G**

Problem 2

First we construct a graph in this way:

- Each box is represented by two nodes: and there is an edge between with lower bound 1 and capacity 1(means that this box must be counted into the result).
- We go over each pair of boxes and find out which box can be nested into the other box. For example, if box can be nested into box , then we add an edge between and , with capacity 1 and lower bound 0.
- We create a super source node and it is conneted with every with an edge of capacity 1 and lower bound 0.
- We also create a super sink node and it is connected with every with an edge of capacity 1 and lower bound 0.

Then, we claim that if there is an arrangement with boxes visible in the room, then the total demand of this circulation network should be . So we can add an demand of on the source node and a demand of on the sink node .

Since we have to find the minimized visible boxed in the arrangement, we start from to find if there is a feasible circulation in this network. If there is not, we make and try again, until we find an feasible circulation. The current value should be the minimized number of visible boxed in the room.

Problem 3

We construct a graph in this way:

- Each family is represented by a node with a demand value that (family members).
- Each table is also represented by a node with a demand value that (seats number).
- We connect every family node to every table node with an edge of capacity 1. (which means each family can only have one member assigned to this specific table)

Then we find if there is a feasible circulation existing in this network. If there is, then we can make this arrangement, and the flow of each edge denote the assignments. Or if there is not, then we can say that we can't find such arrangement.

*PS: I think my algorithm works only if the number of seats is equal to the number of guests: . If not, we can mildly change it into a max flow problem which is very similar.

Problem 4

(a) We construct a graph in this way:

- Each patient is represented by a node and each hospital is represented by a node .
- We iterate every pair of and if the patient has less than half-hour's drive to the hospital, we add an edge between them with capacity 1.
- We create a Source node and connect to every patient with an edge of capacity 1.
- We create a Sink node and connect every hospital to with an edge of capacity .

Then we can find the max flow in this graph. If the value of this flow , we can say that this balanced allocation is possible, and the flow of each edge between the patients and hospitals denote the specific allocations. If not, then this kind of allocation does not exist.

(b)

First, each edge between has a capacity 1, which means one patient can be allocated at most once. And each edge between has a capacity , which means that each hospital can only accommodate at most patients.

Then, each edge between stands for the possible allocation for patient to the hospital .

If we find a max flow that has a value of , it means that each edge between has a flow of 1, so each patient is allocated to a specific hospital. In the mean time, each edge between has a capacity constraint, so the number of patients in each hospital can never exceed this value.

Therefore, we can prove this algorithm is correct.

(c)

The runtime complexity for this algorithm should be: