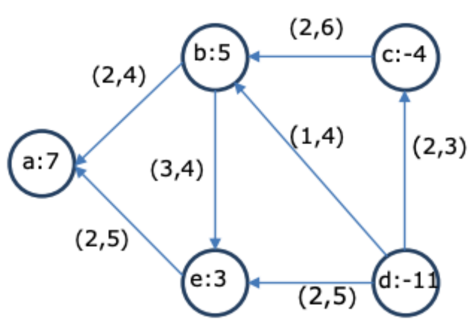
CSCI 570 - HW7 Isaac Delgado
4189182267

1. The computer science department course structure is represented as a directed acyclic graph G = (V, E) where the vertices correspond to courses and a directed edge (u, v) exists if and only if the course u is a prerequisite of the course v. By taking a course w, you gain a benefit of pw which could be a positive or negative number. Note, to take a course, you have to take all its prerequisites. Design an efficient algorithm that picks a subset S⊂V of courses such that the total benefit is maximized.

We can model this problem as a circulation graph where each course has a demand/supply associated with each of their vertices corresponding to the benefit gained from taking that course. We do not know what edge cost is between courses so we can assume their capacities are infinite. Next we can add two additional vertices *s* and *t.* We will connect all edges from *s* to all negative benefit courses (supply vertices) and transfer the supply to the capacity of these edges. Similarly we will connect all the positive benefit courses (demand vertices) to vertex *t* and transfer the demand to the capacity of these edges. We will use Ford-Fulkerson algorithm to find the min-cut. In this min-cut we will have *s* in one partition and *t* in the other. Since all positive benefited courses will be attached to vertex *s* then we know that this is the optimal solution, however to keep the topological ordering between prerequisites we well expand our partition to any neighboring vertices attached by an edge from the second partition to the first. We find the minimum of this cut in order to get the optimal solution.

2. In the network 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.



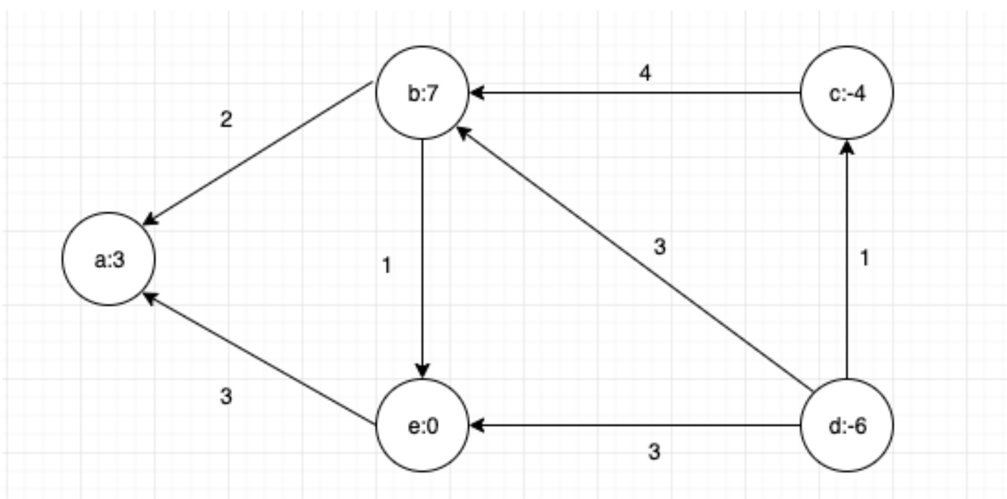
a) Turn the circulation with lower bounds problem into a circulation problem without lower bounds.

We first push a flow equal to the lower bound on each edge and thus the new capacity of the edge will be the ((upper bound) - (lower bound)). The capacities will change as well as we push the lower bound flow along an edge the new vertex demand will be (vertex demand - ((total flow-In) - (total flow-out)).

L(v) = ((total flow-In) - (total flow-out))

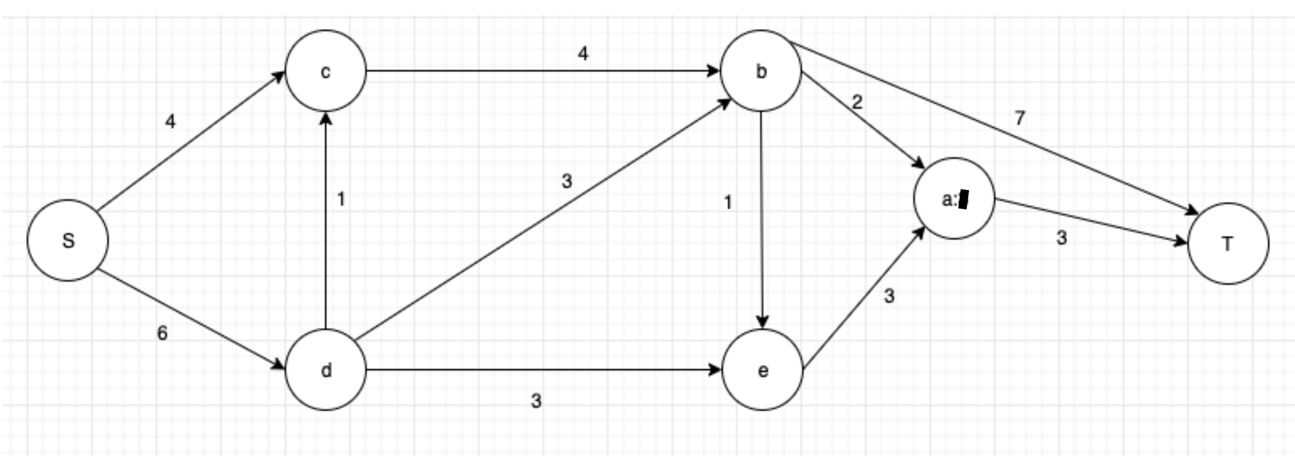
c'(b, a) = c(b, a) - l(b, a) = 4 - 2 = 2
c'(e, a) = c(e, a) - l(e, a) = 5 - 2 = 3
c'(b, e) = c(b, e) - l(b, e) = 4 - 3 = 1
c'(d, e) = c(d, e) - l(d, e) = 5 - 2 = 3
c'(d, b) = c(d, b) - l(d, b) = 4 - 1 = 3
c'(d, c) = c(d, c) - l(d, c) = 3 - 2 = 1
c'(c, b) = c(c, b) - l(c, b) = 6 - 2 = 4

d'(a) = d(a) - L(a) = 7 - (4 - 0) = 3
d'(b) = d(b) - L(b) = 5 - (3 - 5) = 7
d'(c) = d(c) - L(c) = -4 - (2 - 2) = -4
d'(d) = d(d) - L(d) = -11 - (0 - 5) = -6
d'(e) = d(e) - L(e) = 3 - (5 - 2) = 0



b) Turn the circulation with demands problem into the maximum flow problem.

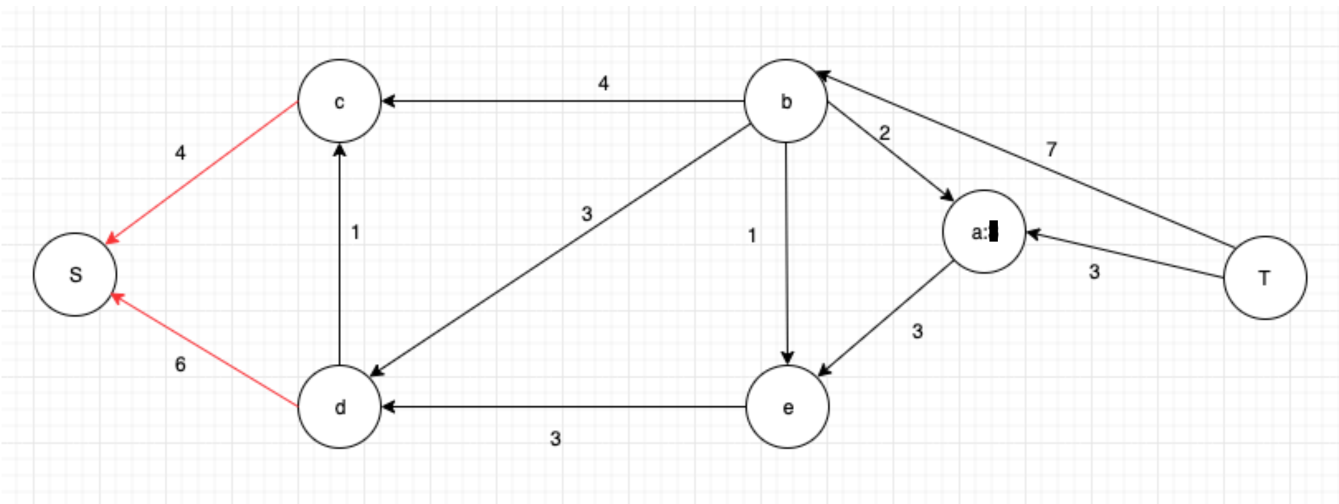
What we do is create a new graph with two additional vertices *s* and *t.* We then create an edge from *s* to all negative demand (supply) vertices and convert them to positive and make them the capacity for those edges. Similarly, create edges from all positive demands to the *t* and make their demands the the edge capacities as well. I flipped the graph to so we can has *s* on the left and *t* on the right



Isaac Delgado
4189182267

c) Does a feasible circulation exist? Explain your answer.

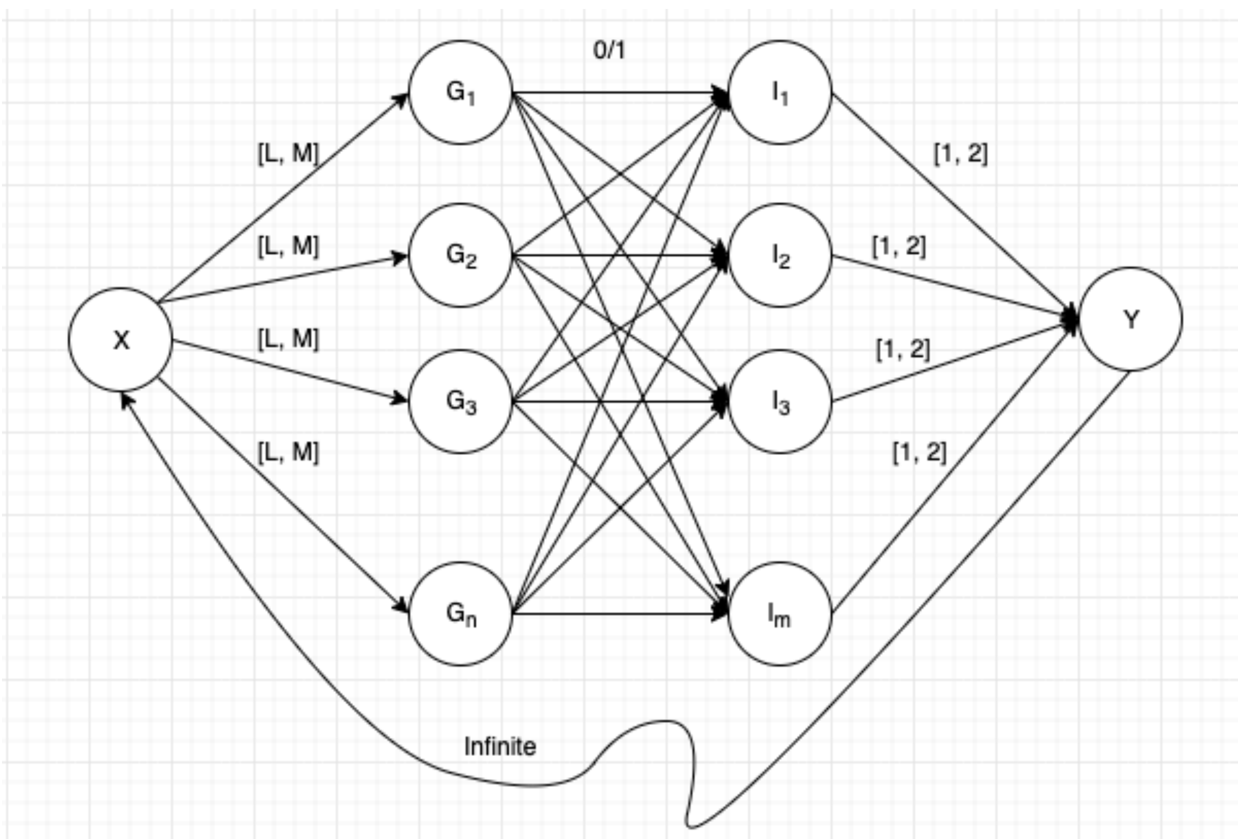
We can run Ford-Fulkerson Algorithm and get a possible residual graph like below. In this case, and any residual graph case after running Ford-Fulkerson Algorithm we saturate all edges from *s*. Since all edges from *s* are attached to vertices of supply and all are saturated then there is a feasible solution for the original circulation graph above.



Isaac Delgado
4189182267

3. There is a precious diamond that is on display in a museum at m disjoint time intervals. There are n security guards who can be deployed to protect the precious diamond. Each guard has a list of intervals for which he/she is available to be deployed. Each guard can be deployed to at most M time slots and has to be deployed to at least L time slots. Design an algorithm that decides if there is a deployment of guards to intervals such that each interval has either one or two guards deployed.

Using a circulation graph we will have m time intervals and n security guards as individual vertices. Each guard is connected by and edge to every time interval. The capacity of these edges will either be 1 or 0 depending if that specific guard is available for that time interval or not available respectively. We will add an additional vertex X and create an edge from X to every guard. Each capacity for these edges from X will have a lower bound capacity of L and an upper capacity of M. Next we create a vertex Y and create edges from every time interval to Y. The capacity of every each from time intervals to Y will have a lower bound of 1 and upper bound of 2. Lastly we will create an edge from Y to X that will have an infinite capacity, our lower and upper bounds will take care of the total number of guards restriction. The graph will look similar to the following:



Isaac Delgado
4189182267

4. The Canine Products company has two dogfood products, Frisky Pup and Husky Hounds, that are made from a blend of two raw materials, cereal and meat. 1 pound of cereal and 1.5 pounds of meat are needed to make a package of Frisky Pup and it sells for $7 a package. 2 pounds of cereal and 1 pound of meat are needed to make a package of Husky Hound and it sells for $6 a package. Raw cereal costs $1 per pound and raw meat costs $2 per pound. It also costs $1.40 to package the Frisky Pup and $.60 to package the Husky Hound. A total of 240,000 pounds of cereal and 180,000 pounds of meat are available per month. The only production bottleneck is that the factory can only package 110,000 bags of Frisky Pup per month. Management would like to maximize profit. Formulate the problem as a linear program.

Cost of Producing Frisky Pup:
Packaging = $1.40
*Cereal = (1 * 1) = $1*
*Meat = (1.5 * 2) = $3*
Total Cost = $5.40

Cost of Producing Husky Hounds:
Packaging = $0.60
*Cereal = (2 * 1) = $2*
*Meat = (1 * 2) = $2*
Total Cost = $4.60

Total profit for Frisky Pup = Selling price - (Cost of Producing Frisky Pup)
= 7 - 5.40
= 1.60

Total profit for Husky Hounds = Selling price - (Cost of Producing Frisky Pup)
= 6 - 4.60
= 1.40

Let x_1 be Frisky Pup and x_2 be Husky Hounds. Thus, our objective function is:

$$\text{Max}(1.60x_1 + 1.40x_2)$$

Total amount of cereal, $(x_1 + 2x_2)$ cannot exceed 240000lbs:

$$x_1 + 2x_2 \leq 240000$$

Total amount of meat, $(1.5x_1 + x_2)$ cannot exceed 180000lbs:

$$1.5x_1 + x_2 \leq 180000$$

Frisky Pup packages cannot exceed 110000 packages:

$$x_1 \leq 110000$$

Lastly, both Frisky Pup and Husky Hounds number of packages cannot be negative:

$$x_1, x_2 \geq 0$$

Therefore our linear program is;

$$\text{Max}(1.60x_1 + 1.40x_2)$$
$$x_1 + 2x_2 \leq 240000$$
$$1.5x_1 + x_2 \leq 180000$$
$$x_1 \leq 110000$$
$$x_1, x_2 \geq 0$$

Isaac Delgado
4189182267

5. Consider the following linear program:

$Max(3x_1 + 2x_2 + x_3)$
$x_1 - x_2 + x_3 \leq 4$
$2x_1 + x_2 + 3x_3 \leq 6$
$-x_1 + 2x_3 = 3$
$x_1 + x_2 + x_3 \leq 8$
$x_1, x_2, x_3 \geq 0$

Write the dual problem.

Our dual problem will be the form
$Max(b^T y)$
$A^T \geq c$
$y \geq 0$

We first convert our linear program to standard inequality form by fixing $-x_1 + 2x_3 = 3$.
$-x_1 + 2x_3 = 3$ -> $-x_1 + 2x_3 \leq 3$ & $x_1 - 2x_3 \leq -3$

Our linear program is now
$Max(3x_1 + 2x_2 + x_3)$
$x_1 - x_2 + x_3 \leq 4$
$2x_1 + x_2 + 3x_3 \leq 6$
$-x_1 + 2x_3 \leq 3$
$x_1 - 2x_3 \leq -3$
$x_1 + x_2 + x_3 \leq 8$
$x_1, x_2, x_3 \geq 0$

Now we can convert it to the dual problem:
$Max(4y_1 + 6y_2 + 3y_3 - 3y_4 + 8_y)$
$y_1 + 2y_2 - y_3 + y_4 + y_5 \geq 3$
$-y_1 + y_2 + 0y_3 + 0y_4 + y_5 \geq 2$
$y_1 + 3y_2 + 2y_3 - 2y_4 + y_5 \geq 1$