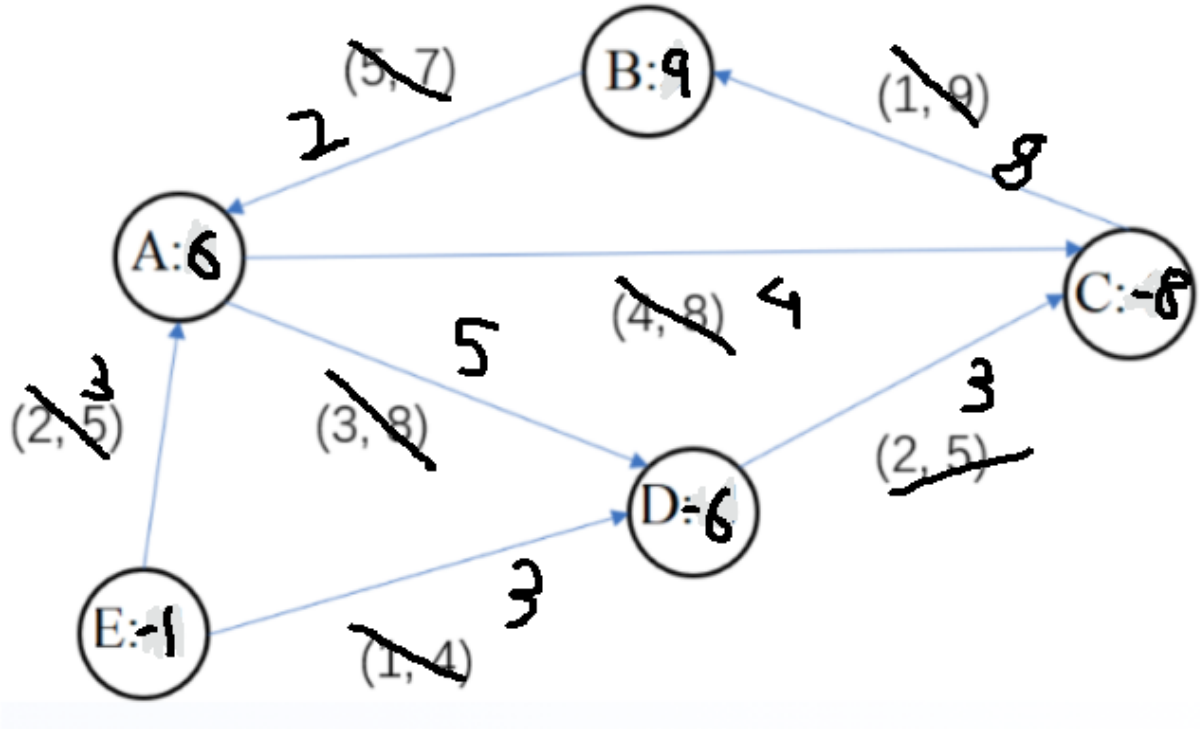
CSCI570 Homework 5

Name: Saunak Sahoo **USC ID: 3896400217**

Ans 1

a) Removing lower bounds:



New demands:

d(A) = 6 -2 -5 +4 +3 = 6

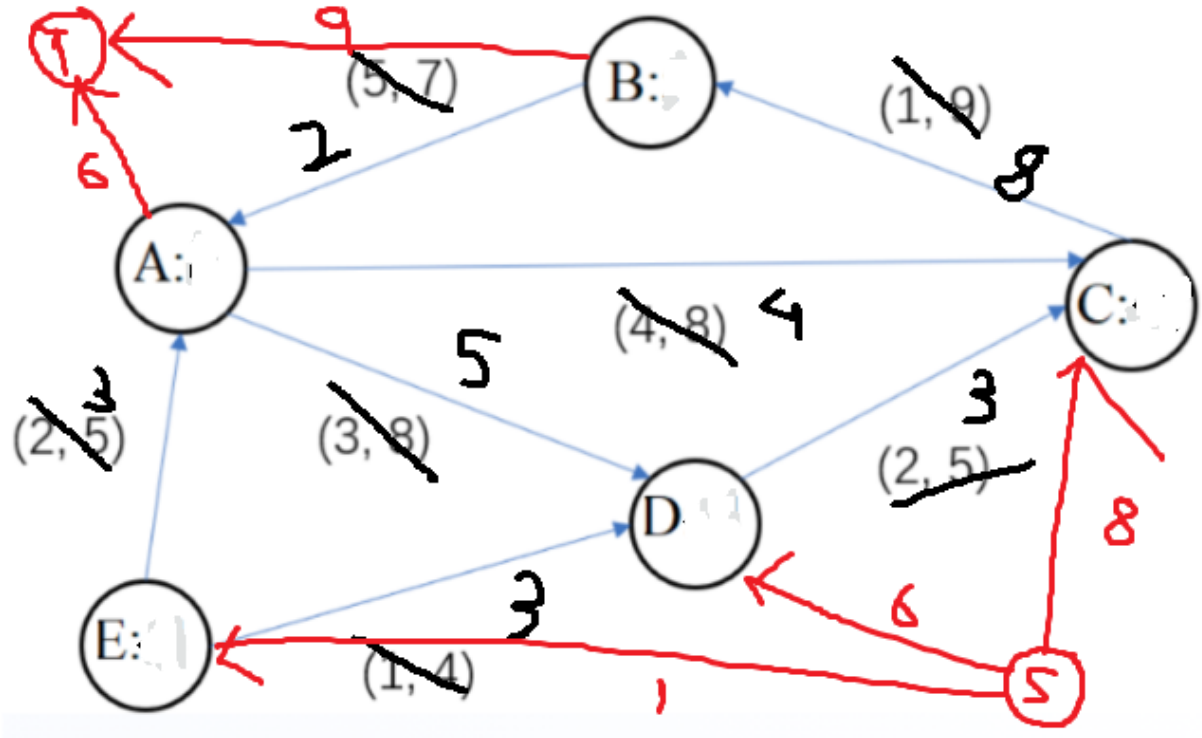
d(B) = 5 -1 +5 = 9

d(C) = -3 -2 -4 +1 = -8

d(D) = -4 -1 -3 +2 = -6

d(E) = -4 +2 +1 = -1

b) Solving circulation program:



Introduce Source S and Target T nodes. Connect Source node to vertices with demand<0 and Target node to vertices with demand>0

Now we can use Ford-Fulkerson algorithm to find the max-flow:

1. S-C-B-T, f = 8
2. S-E-A-T, f = 1

Thus max flow = 8+1 = 9

c) Checking if feasible circulation exists

Step 1: Verify sum of demands = 0

Sum of demands = 6 + 5 -4 -4 -3 = 0

Step 2: Verify max-flow in graphs with removed lower bounds = D (sum of capacities of edges with source or target)

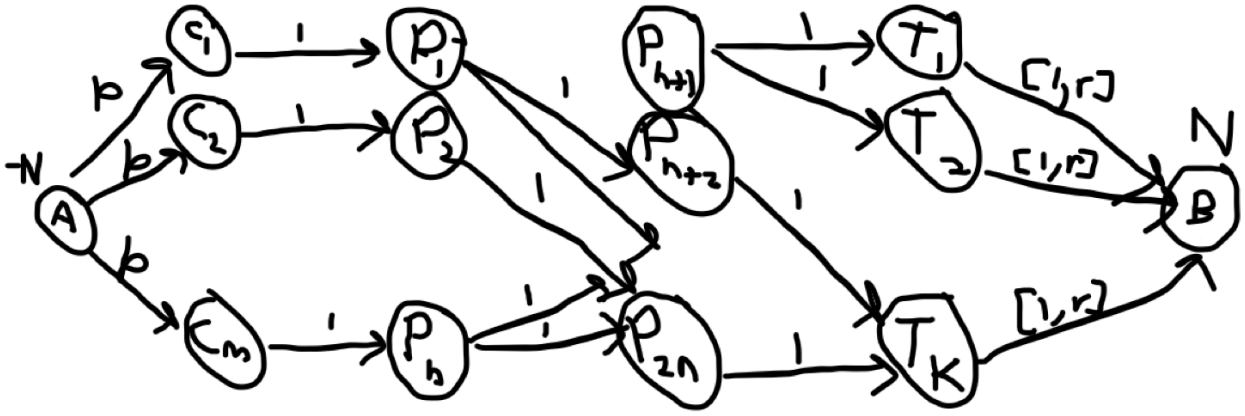
Max flow = 9

D (from source) = 8+6+1 = 15

Since max flow ≠ D, we do not have a feasible circulation

Ans 2 (old version)

Step 1: Creating a Network Flow



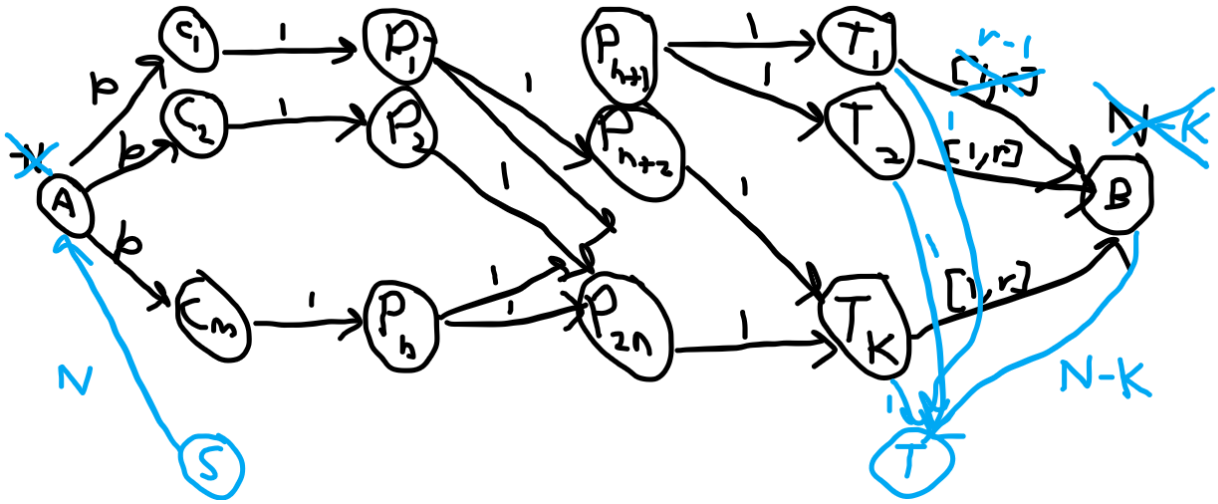
Create a network flow with courts represented as $c_1, c_2 ... c_m$, players of the top half represented as $p_1, p_2 ... p_n$, players of the bottom half represented as $p_{n+1}, p_{n+2} ... p_{2n}$ and time slots represented as $t_1, t_2 ... t_k$. Introduce additional nodes A and B. Assign a supply of -N on node A and a demand of N on node B respectively to represent the N matches that must take place between the top-half and bottom-half players. Connections between the nodes are made according to the following rules:

- Node A - Courts: Each court is connected to node A with capacity p to represent the fact that each court can host a maximum of p matches
- Courts - Top half players: A given court is connected to a player iff it is NOT a part of the m' courts that the player has requested to not play on. Each such edge has a capacity of 1 to represent the fact that each court can host only one top half player (who plays a bottom half player)
- Top half players - Bottom half players: All top half players are connected to players from the bottom half with capacity 1 to represent the fact that each top half player has to play 1 match with a bottom half player
- Bottom half players - Time slots: A given player is connected to a time slot iff it is part of the k' timeslots that they are okay to play in.
- Time slots - Node B: Each time slot is connected to the Node B with edge capacities having a lower bound of 1 and an upper bound of r to represent the fact that each

time slot has to have atleast one player playing, and at-max r players playing matches

Step 2: Making claim

We will reduce the circulation with demands and lower bounds problem to a network flow problem:



Note: Here we are considering the case that N>K ie. (N-K)≥0 so that a valid feasible schedule of matches exists

Claim: There exists a feasible schedule of matches iff max flow = N

Step 3: Proving claim in both directions

In forward direction →
To prove: Given a feasible schedule of matches, max flow = N

If there is a feasible schedule of matches then all N players from both halves of the draw will play matches against each other. This would mean N edges between the Top half players and Bottom half players would be saturated. Since the edge capacities is 1 for every edge, max flow = 1XN = N

In backward direction ←
To prove: Given the max flow = N, to find the actual assignments of players from the top-half and bottom-half that would play a match

To find the actual matches that would take place, we will look at the saturated edges between the players from the Top half and the Bottom half

Ans 3

Let:

x be the amount of Steel produced (in tons)

y be the amount of Brass produced (in tons)

z be the amount of Pewter produced (in tons)

a)

Objective function:

$$max(x + y + z)$$

subject to:

$$3x + y + 2z \leq 8$$

$$3x + 10y + 5z \leq 20$$

$$x \geq 0, y \geq 0, z \geq 0$$

b)

Objective function:

$$max(3x + 10y + 5z)$$

subject to:

$$3x + y + 2z \leq 8$$

$$3x + 10y + 5z \leq 20$$

$$x + z \geq 2$$

$$x \geq 0, y \geq 0, z \geq 0$$

Ans 4

Let:

x_1 be the amount of cement in tons shipped from city A to city C

x_2 be the amount of cement in tons shipped from city A to city D

y_1 be the amount of cement in tons shipped from city B to city C

y_2 be the amount of cement in tons shipped from city B to city D

Objective function:

$min(x_1 + 2x_2 + 3y_1 + 4y_2)$

subject to:

$x_1 + y_1 \geq 50$

$x_2 + y_2 \geq 60$

$x_1 + x_2 \leq 70$

$y_1 + y_2 \leq 80$

$x_1 \geq 0, x_2 \geq 0, y_1 \geq 0, y_2 \geq 0$

Ans 5

We need to add another equation to create the A matrix for standard linear form. We are given that

$x_4 \geq 0$

$\rightarrow -x_4 \leq 0$

We will use this as our 4th inequation (in addition to the inequations already given)

In normal form matrices are:

$$X = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & -3 & 4 & -1 \end{bmatrix}$$

$$A = \begin{bmatrix} 1 & -1 & -3 & 0 \\ 0 & 1 & 3 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$$

$$b = \begin{bmatrix} -1 & 5 & 1 & 0 \end{bmatrix}$$

Writing matrices for the dual problem would be:

$$Y = \begin{bmatrix} y_1 & y_2 & y_3 & y_4 \end{bmatrix}$$

$$A^T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ -3 & 3 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$$

Hence our dual becomes:

$$min(-y_1 + 5y_2 + y_3)$$

Subject to:

$$y_1 \geq 1$$

$$-y_1 + y_2 \geq -3$$

$$-3y_1 + 3y_2 + y_3 \geq 4$$

$$-y_4 \geq -1$$

Ans 6

Say the set of vertices in the graph $G(V, E)$ is

$$V = [v_1, v_2, ...v_n]$$

We shall assign $v_i = 1$ if a given vertex is part of the vertex cover, otherwise $v_i = 0$

So our linear program would be:

$$min(v_1 + v_2 + ...v_n)$$

Subject to:

$$v_j + v_k \geq 1, \forall e(j, k) \in E$$

$$v_i \in [0, 1]$$

Ans 7

Let us have our 3-SAT represented as $f(x_1, x_2, ...x_n)$ = $c_1 \wedge c_2 \wedge ...c_m$ where $x_1, x_2, ...x_n$ are the literals and $c_1, c_2, ..c_m$ are the clauses where each clause consists of at most 3 literals

Algorithm $A()$:

If $A(f(x_1, x_2, ...x_n))$ = 0 then the 3-SAT is not satisfiable. In this scenario no satisfying assignment exists and the algorithm terminates here

Else if $A(f(x_1, x_2, ...x_n))$ = 1, the satisfying assignments can be found in the following manner:

If $A(f(1, x_2, ...x_n))$ = 1, then we can set x_1 = 1 and we can be sure that $f_1(x_2, ...x_n)$:= $f(1, x_2, ...x_n)$ is satisfiable

Else if $A(f(1, x_2, ...x_n))$ = 0, then we can set x_1 = 0 and we can be sure that $f_1'(x_2, ...x_n)$:= $f(0, x_2, ...x_n)$ is satisfiable

In this manner we can continue iteratively by computing f or f'
(Eg. if f_1 is satisfiable - then in our next iteration we find assignment for x_2 by computing $A(f(1, x_3, ...x_n))$ and setting $x_2 = 1$ if it return 1 and $x_2 = 0$ otherwise and so on)

Time Complexity:

For every literal we will call the algorithm A() once. For each iteration, checking the value of the 3-SAT takes O(m) time

Thus, the time complexity of this algorithm to find assignments is polynomial time.

Ans 8

Step 1: Show that problem is in NP

Given a graph with colored nodes, we can verify in polynomial time that:

- For each edge (u,v) the color of node u is different from the color of node v
- At most 5 colors are used in the graph

So we can conclude that the problem is in NP

Step 2: Show that problem is in NP-hard

Using reduction we will show **3-Coloring $\leq_p$ 5-Coloring**

For any arbitrary graph G, we shall construct a new graph G' by adding 2 new nodes X and Y. The nodes are connected to each other as well as to all existing nodes of G

Claim: The graph G has a valid 3-coloring solution iff the graph G' has a valid 5-coloring solution

Proof in → direction:
For the graph G with a valid 3-coloring solution, we add 2 new colors to the X and Y nodes in graph G'. So with the additional 2 new colors, G' has a total of 3+2 = 5 colors and hence satisfies the 5-coloring solution

Proof in ← direction:
If we are given a graph with a 5-coloring solution for G', we have nodes X and Y that are connected to each other and to all other vertices of original graph G. So these 2 nodes have to be assigned different colors from all other colors of the graph G. Removing these 2 nodes will give us our original graph G with a 3-coloring solution

Thus, in this manner we can perform the reduction in polynomial time. Since we have proven that the problem is NP and NP-Hard, 5-coloring problem is also NP-Complete.

Ans 9

Step 1: Show that problem is in NP

Given a solution to the longest path problem, we can trace through the path and count the number of edges visited in polynomial time. We can compare this number with k and hence verify the solution

So we can conclude that the problem is in NP

Step 2: Show that problem is in NP-hard

Using reduction we will show **Hamiltonian path $\leq_p$ Longest path**

We know that the Hamiltonian path problem attempts to visit every vertex once. To transform it to the Longest path problem we can record the edges being visited while running the Hamiltonian path problem. We can then compare the number of edges visited to k

Claim: The Hamiltonian path visits $\geq k$ nodes iff the Longest path goes through $\geq k$ edges

Proof in → direction:
If we are given a Hamiltonian path which visits k nodes, it means that along the path we will visit all of the edges which satisfies the Longest path problem

Proof in ← direction:
If we are given a Longest path with k edges, it would have visited all the vertices which satisfies the Hamiltonian path

Thus, in this manner we can perform the reduction in polynomial time. Since we have proven that the problem is NP and NP-Hard, Longest path problem is NP-Complete.

Ans 10

Step 1: Show that problem is in NP

Given a solution to the problem of taking courses at USC, we can verify in polynomial time that:

- Number of courses taken ≥ K
- Courses taken don’t have any overlapping time intervals

So we can conclude that the problem is in NP

Step 2: Show that problem is in NP-hard

Using reduction we will show **Independent Set** $\leq_p$ **Choosing courses**

We shall construct a graph with each node representing a given course time interval. The nodes are connected to each other wherever the time intervals are intersecting

Claim: The graph contains an independent set of size atleast K iff there is a possibility to take atleast K courses

Proof in → direction:
Say we have an independent set in the graph whose size is greater than or equal to K. This means that in our independent set we have atleast K nodes that are not joined by an edge. This also means that we have atleast K courses with non-overlapping time intervals. And in this case our algorithm will output Yes

Proof in ← direction:

If we are given a set of atleast K courses that has been chosen, then it follows that the nodes corresponding to these courses would not be connected to each other. Hence, these nodes would form an independent set of atleast size K

Thus, in this manner we can perform the reduction in polynomial time. Since we have proven that the problem is NP and NP-Hard, Choosing Courses problem is NP-Complete