Name: Sindhuja Yerramalla
UUID: U00839259.
email : Syrrmlla@memphis.edu

HomeWork - 4, Algorithms and Problem Solving

Note :- I have referred textbook, lecture notes and so many random websites like chegg & stack overflow and also worked with few of my classmates. ~~& previous workload~~ done!!

Problem 1 :- Problem 8.9 from DPV.

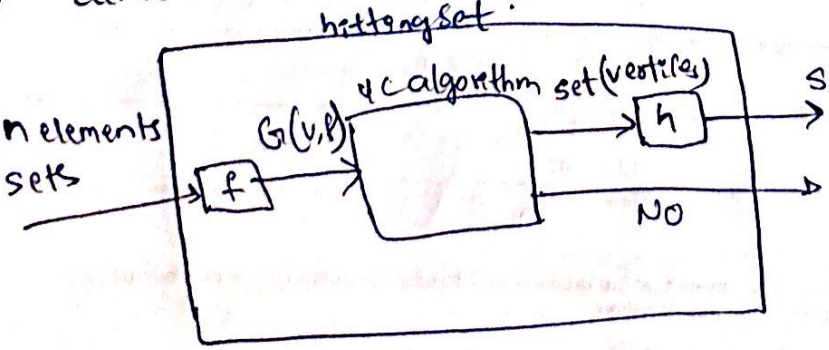
To prove Hitting set problem with a budget 'b' is an Np-complete., we should prove that it is both Np and Np-hard problem.

To verify whether given solution is appropriate answer or not for this problem, we need to check whether any element present in set 'H' is also present in the given sets $S_1, S_2, \ldots S_n$. If there is atleast one element from 'H' presents in all given sets and total elements in set H is less than or equal to 'b'. So, we will take one element will all elements in 'n' sets until we found atleast one intersection. Let's assume each of 'n' sets have 'm' no of elements, (m will be proportional to b). So, total operations need will be $O(n \cdot m \cdot b)$. $\Rightarrow$ Time comp $= O(nb^2)$

As it is verifiable in polynomial time. This is Np-problem

Reduction to vertex cover:

In vertex cover we take a set of vertices & check if these vertices touch all the edges (cover all the edges) - here we can take hitting set elements as vertices of vertex cover. The edges u, v are set of vertices $\{u, v\}$. This set can be taken as set of elements in hitting set problem.



∴ we reduced hitting set to vc (which is Np-hard)

So, we can say that hitting set problem is NP-Complete

Problem 2 : Problem 8.20 from D.P.V.

For any given undirected graph $G(V, E)$, we can validate whether any given dominating set is a correct solution or not with budget 'b'. we take the dominating set, D and find whether the size of this set is atmost b or not. If it is beyond the budget we can conclude it as a wrong solution, else we traverse through every vertex, $v \in V$ in graph and validates its existence in dominating set or its adjacency with any vertex in dominating set, all this verification can be done in polynomial time, so it is a NP class problem.

Reducing to Np hard :

In order to prove Dominating set as NP-hard we need to reduce to a known NP-hard problem. (Vertex cover).

For sending instances to vertex cover, ne need to modify graph. $G(V, E)$ to $G'(V', E')$.

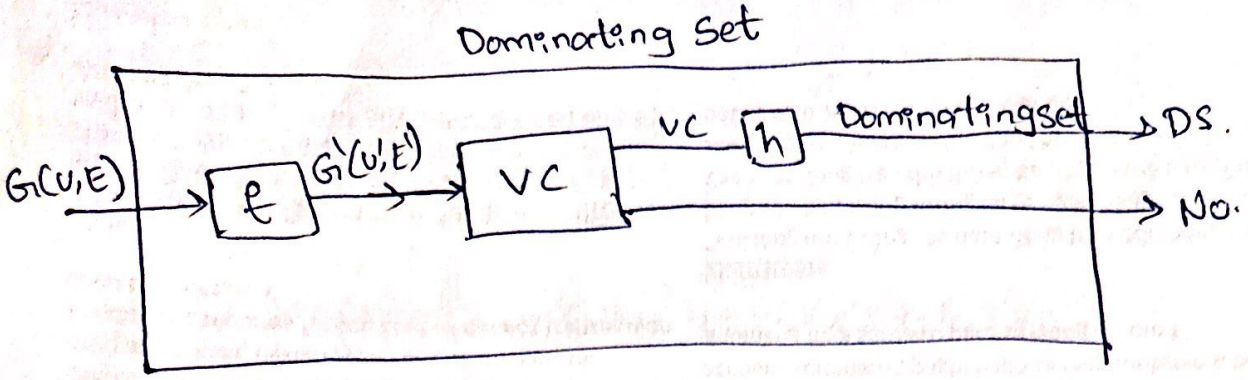
$V' \longrightarrow V \cup \{(u,v)$ for every edge $\{u,v\}\}$

$E' \longrightarrow E \cup \{(u,uv), (uv,v)$ for every new vertex $(uv)\}$

we give G' as input to VC to find the vertex cover set.

So the VC obtained VC has both old vertices & new vertices For every new vertex added to VC it leaves the adjacent vertex of its vertices, for example. u, v, uv

to cover all edges we take u, uv (or) uv, v in VC. This leaves the adjacent of u or v. Then the solution for Dominating set is obtained by removing all the new vertices from VC.



f → adds both vertices & edges h → removes vertices & edges

both f, h runs in polynomial time.

∴ we reduced Dominating set to NP-hard (VC) & proved that DS is NP-hard.

⇒ Dominating set is Np-Complete (Np & Np hard) //.

Problem 3 :-

Given a problem to find appropriate vector 'x' of size. nx1 which satisfies the inequality $A \cdot x \leq b$ where A is a matrix of order mxn and b is a vector of order mx1, and if we are given with vector 'x' as a solution we can complete the verification of solution in a non-deterministic polynomial time. So, this is a Np-class.

We can reduce the Np-Complete 3SAT problem to inequality constraint ILP problem in following way. For every clause in the given problem, we could turn it into a matrix A, for 'm' clauses with 3 literals, we can accomodate of matrix of order mx3 by having a each row for each clause in given CNF. Each row of matrix will be represented in a way that '1' will be placed for literal

linear program ..

while '-1' will be used for negation of literal. vector 'b' of order m*1 will have all values as '3'. values for literals $x_1, x_2, \ldots x_n$ will be '1' for false and '-1' for true

so, contraints will be $x_i = \{-1, 1\}$

so, CNF: $(\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3) \wedge (x_1 \vee \bar{x}_2 \vee x_3) \wedge (\bar{x}_1 \vee x_2 \vee x_3)$

$$\underset{A}{\begin{bmatrix} -1 & -1 & -1 \\ 1 & -1 & 1 \\ -1 & 1 & 1 \end{bmatrix}} \underset{x}{\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}} \underset{\leq}{\leq} \underset{b}{\begin{bmatrix} 3 \\ 3 \\ 3 \end{bmatrix}}$$

This CNF will be satisfied for x_1 – True
x_2 – false
x_3 – True.

∴ vector x will be $\begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix}$

If we use this vector 'x' in above $Ax \leq b$ we will get $1 \leq 3 \rightarrow 1^{st}$ clause

$-3 \leq 3 \rightarrow 2^{nd}$ clause

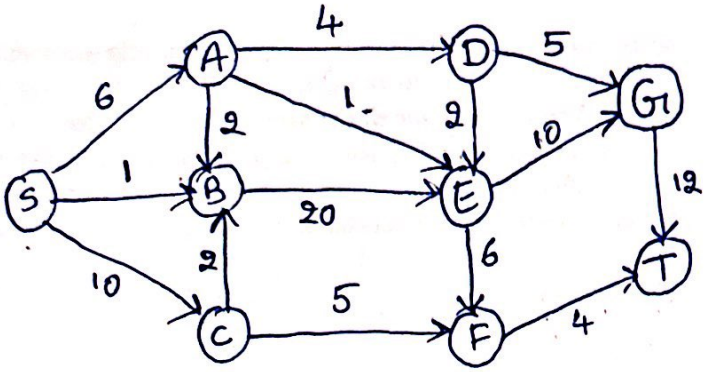
$-1 \leq 3 \rightarrow 3^{rd}$ clause

so, if we solve this ILP $Ax \leq b$, we can get answer for 3SAT problem also. conversly, if we have solution for 3SAT, there will be solution for this ILP $Ax \leq b$ problem. This ILP can be reduced from Np-complete 3SAT problem in polynomial time; so this ILP is Np-Hard Problem.

∴ ILP is both NP & NPhard problem it is Np-complete problem.

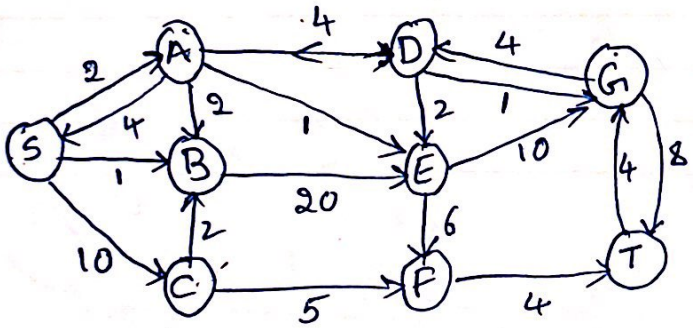
Problem 4 :- Problem 7.10 from D.P.V.

Given network,



Path 1 S - A - D - G - T , min capacity = 4

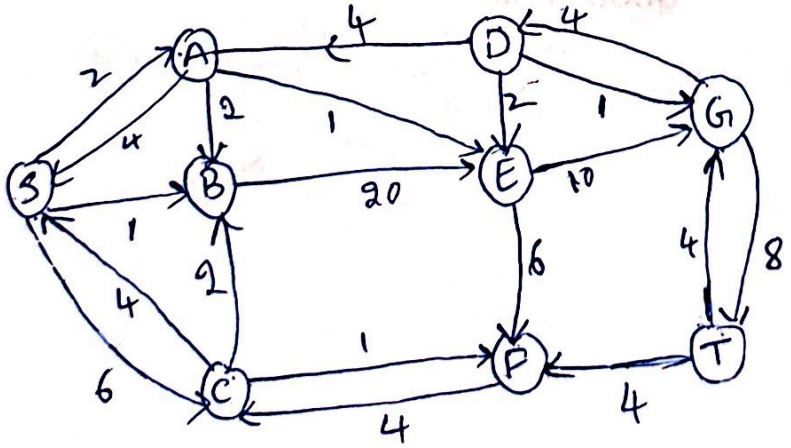
Residual network for path 1 is



$f_{SA} = f_{AD} = f_{DG} = f_{GT} = 4$

Path 2 S - C - F - T , min capacity = 4

Residual network for path 2 is

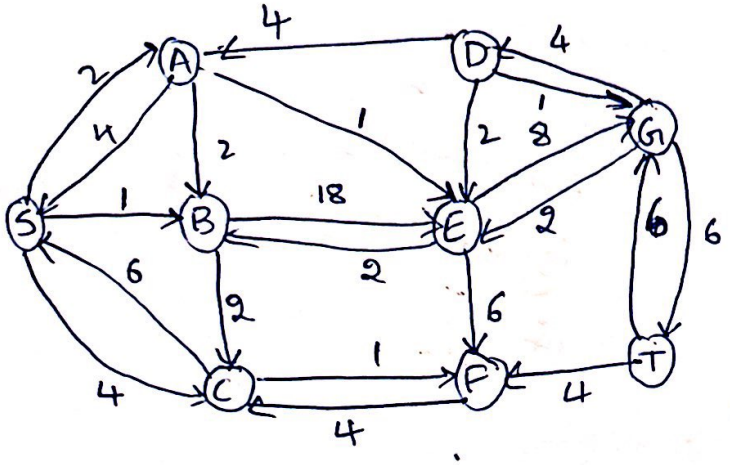


$f_{SA} = f_{AD} = f_{DG} = f_{GT} = 4$

$f_{SC} = f_{CF} = f_{FT} = 4$

Path 3 $S-C-B-E-G-T$, min capacity = 2

Residual network for Path 3 is



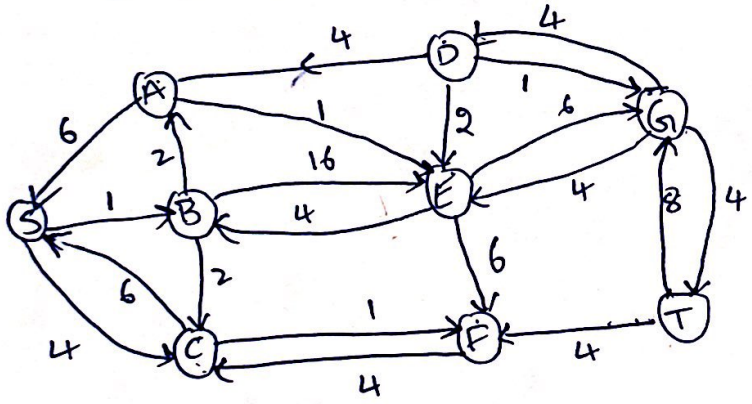
$f_{SA} = f_{AD} = f_{DG} = 4$

$f_{GT} = 6$

$f_{SC} = 6$, $f_{CF} = f_{FT} = 4$,

$F_{CB} = F_{BE} = F_{EG} = 2$

Path 4 $S-A-B-E-G-T$, min capacity = 2



$f_{SA} = 6$, $f_{AD} = F_{BG} = 4$

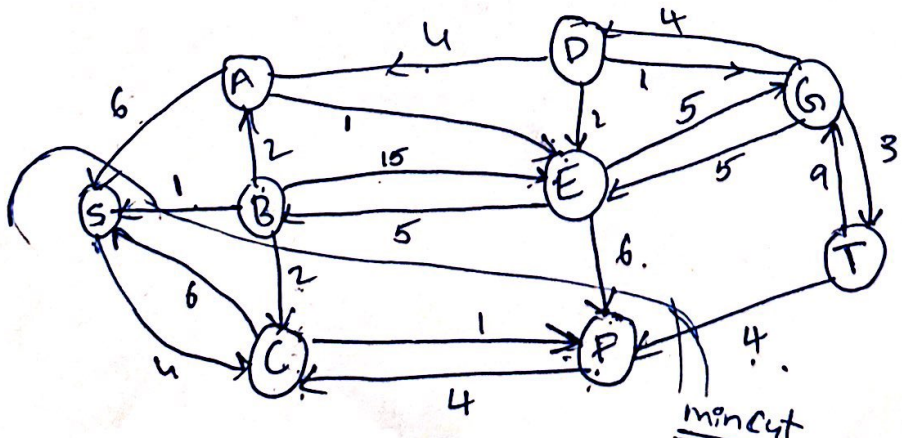
$F_{GT} = 8$

$F_{SC} = 6$, $F_{CF} = F_{FT} = 4$,

$F_{CB} = 2$, $f_{AB} = 2$,

$F_{BE} = 4$, $F_{EG} = 4$

Path 5 $S-B-E-G-T$, min capacity = 1



$f_{SA} = 6$ $f_A = F_{DG} = 4$

$F_{GT} = 9$, $f_{SB} = 1$

$f_{SC} = 6$, $F_{CF} = f_{FT} = 4$,

$F_{CB} = F_{AB} = 2$,

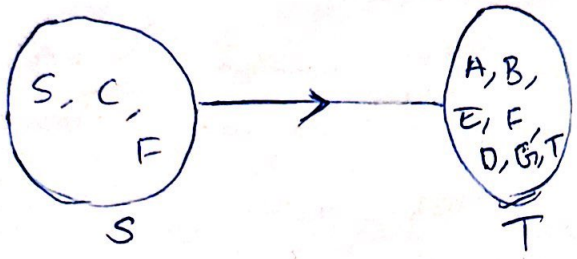
$F_{BE} = F_{EG} = 5$,

There are no more paths possible from S to T

∴ The max flow of the network $|f|$ is 13 ($f_{SA} + f_{SB} + f_{SC} = 6+6+1 = 13$).

The vertices that are reachbiable from 's' are.
'C' & 'F'

∴



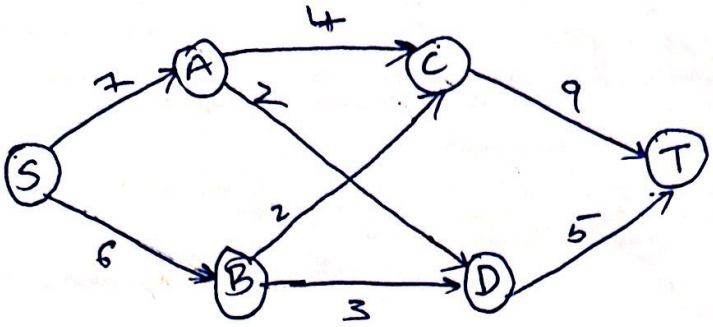
mincut (S, T)

The matching cut is {S, C F} & {A, B, D, E, G, T}

Problem 5 : 7 to 17 from D.P.V.

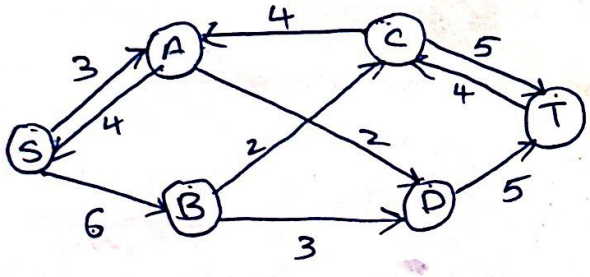
Given Network.

a)



Path 1 . S - A - C - T . min capacity = 4

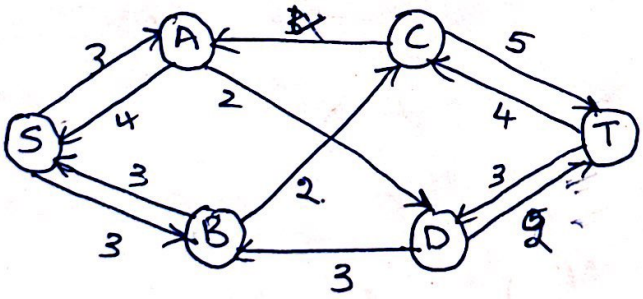
Residual network after Path 1 is



$f_{SA} = f_{AC} = f_{CT} = 4$

Path 2, S - B - D - T , min capacity = 3

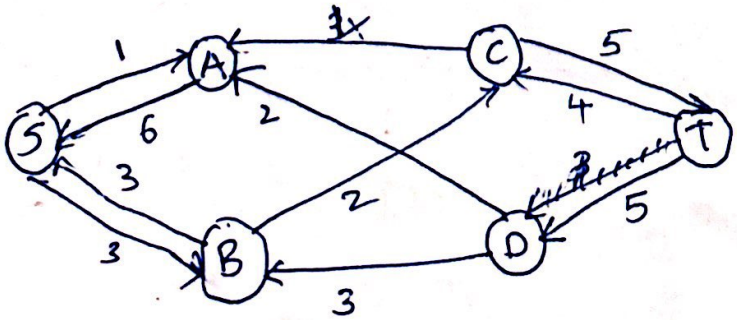
Residual network after path 2 is



$f_{SA} = f_{AC} = f_{CT} = 4$

$f_{SB} = f_{BD} = f_{DT} = 3.$

Path 3, S - A - D - T , min capacity = 2.



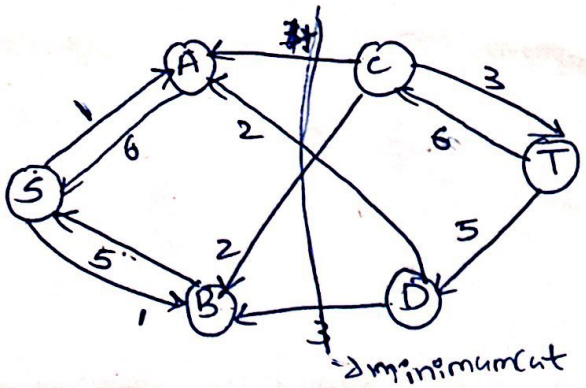
$f_{SA} = 6$, $f_{AC} = f_{CT} = 4$

$f_{SB} = f_{BD} = 3$ $f_{DT} = 5$

$f_{AD} = 2.$

(3) Path 4, S - B - C - T, min capacity = 2

(3)



$f_{SA} = 6$, $f_{AC} = 4$

$f_{CT} = 6$, $f_{BD} = 3$,

$f_{DT} = 5$, $f_{AD} = 2$,

$f_{SB} = 5$, $f_{BC} = 2$,

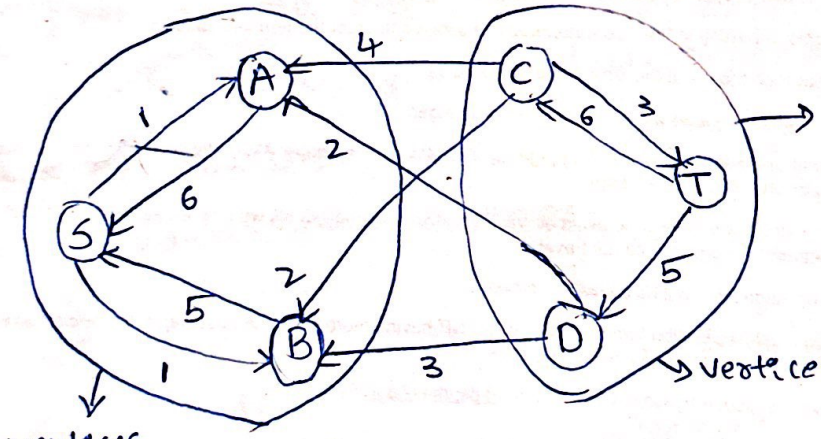
There are no more paths from S → T

Maxflow of the given network $|f| = 6 + 5 = 11$ //.

$$|f| = 11$$

The minimum cut is between $\{S, A, B\}$ & $\{C, D, T\}$

b)



c) In the given network, bottleneck edges are,

A → C

A → D

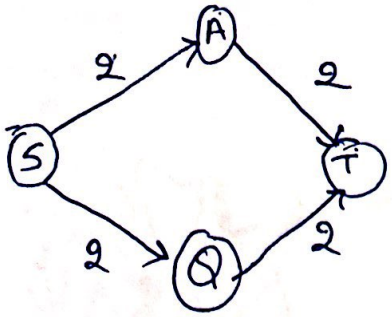
B → C

B → D

D → T

The edges that are not present in G' (residual network) but they are in original network are bottleneck edges.

d)



In this network, there are 4 nodes & 4 edges, All the edges are of same capacity (2).

∴ There are no bottleneck edges in this network.

e) We can use Ford-Fulkerson algorithm to get residual graph and then examine it for bottleneck edges.

Algorithm:

FordFulkerson(G, S, T)

1. Set the flow in all edges to 0
2. Repeat search if an s-t pathPexists
3. Increase the flow from s to T along the path
4. Update residual network graph.

   return. G'

5. Compare the edges in input G & residual G'.
6. Missing edges in G' are bottleneck edges.