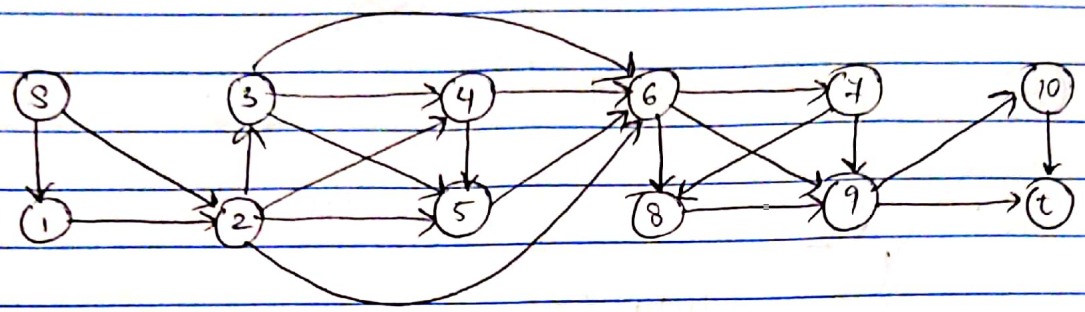
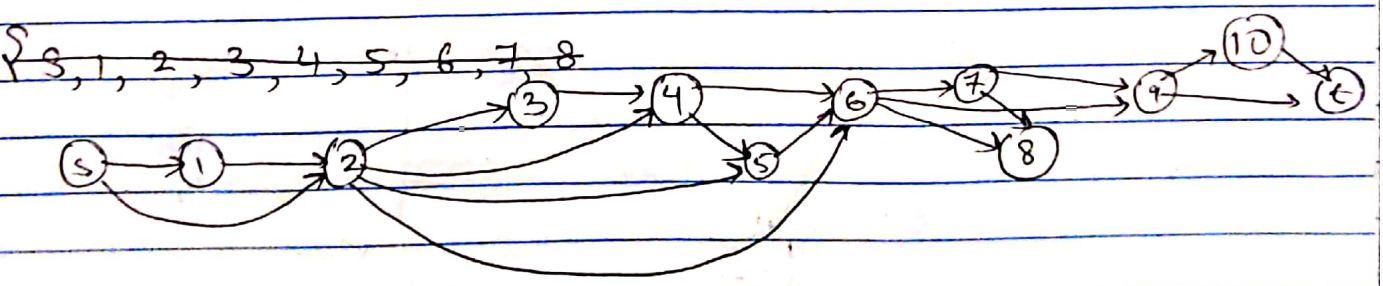
Q1) G is a DAG with source s and sink t.



For the above problem we compute pre and post of all nodes and sort them topologically ie by post.

Hence after topologically sorting the above graph we get



Let adj[] = Adjacentcy matrix of each vertices.

max - vertice:- store the furthest edge of the current vertex

Traverse through each vertex in array

- If current vertex = max vertex, then current value is the articulation point.

- Find the farthest edge of all the edges of the current vertex through the adjacency matrix. If the farthest edge is greater than max vertex, max vertex = farthest edge.

Time complexity :- $O(V+E) + O(V \log V + E)$

$\therefore T(n) = O(V \log V + E)$ //

(Q2) We can convert the above problem into max flow problem.

We create an artificial source which ~~converts~~ connects to m points in the grid

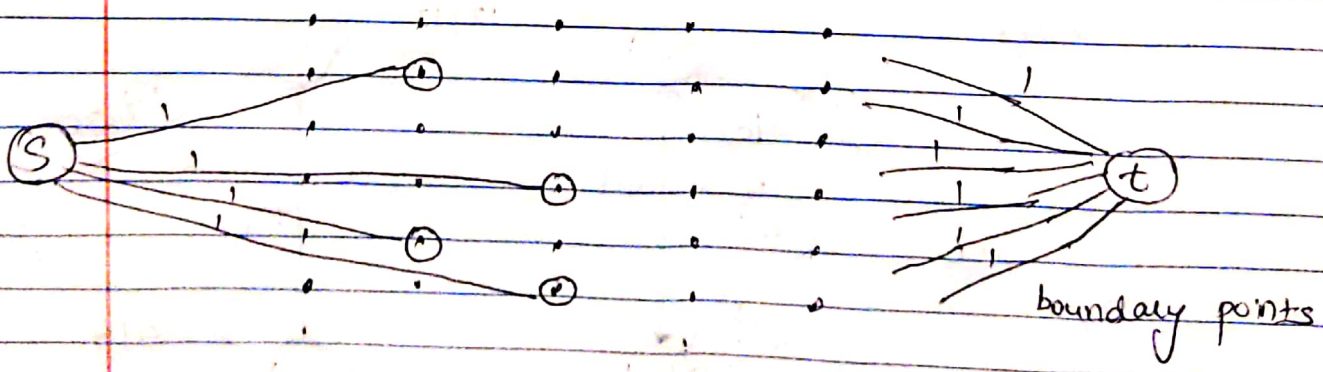
We create an artificial sink which connects to all the boundary points.

~~Since there is~~

Since the matrix is n×n; there are 4n-4 boundary points.

the edges connected to the starting points and the boundary points are of capacity 1.

thus the graph looks like



Thus it becomes a max flow problem.

If the max flow = m, then there is an escape in the original grid. If maxflow < m, then there is no escape in the original grid.

Time Complexity

To compute max-flow, $T(n) = O(VE^2)$

But no. of vertices = n^2

No of edges = $m + (4n - 4) + 2(2n^2 - 2n)$

$= n^2$

$\therefore T(n) = O(n^2 (n^2)^2)$

$= O(n^6)$

Q3) Given DAG graph $G(V, E)$, constructing another graph $G' = (V', E')$ such that :-

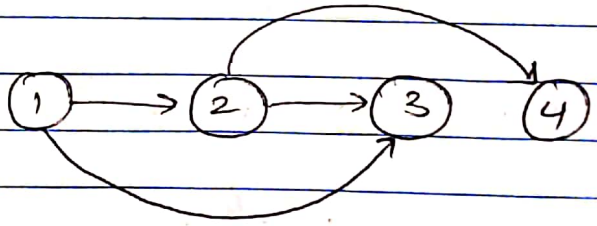
(1) G' contains 2 vertices V_{in} and root

(2) G' contains an undirected edge for every directed edge $u \rightarrow v \leftarrow G$.

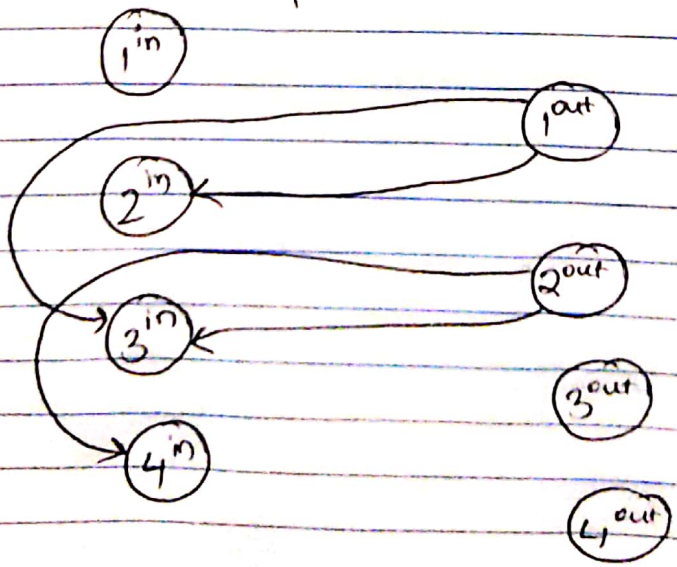
Both G and G' will be represented by the same adjacency matrix and G' is made of 2 independent disjoint vertex set made from in & out.

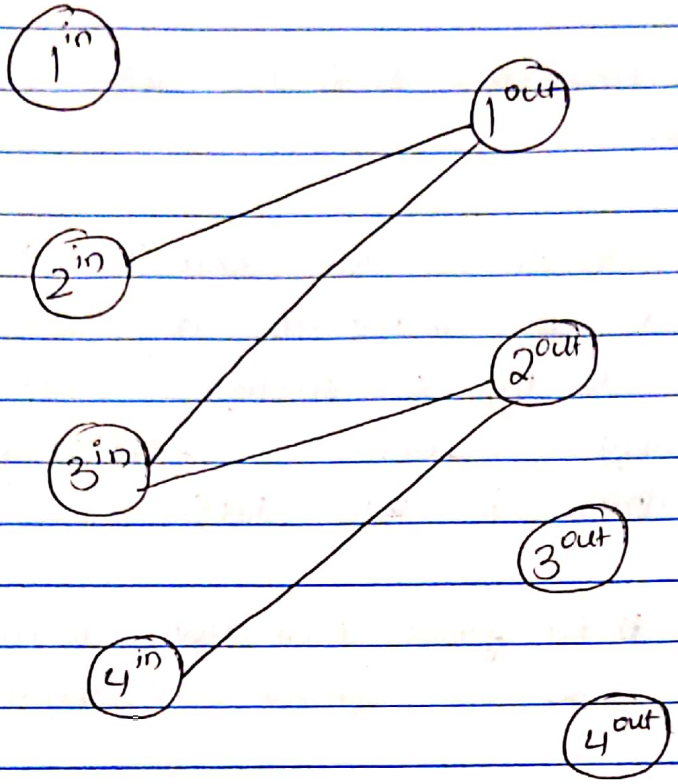
For Eg.

G :-



Bipartite Graph G':





Claim:- Graph G can be covered by k disjoint paths if and only if the new graph G' has a matching of size V - k

Suppose G has a disjoint path cover P with k paths, think of P has a subgraph of G. Every vertex in P has in degree of 0 or 1, it follows that P has exactly V-K edges.

A subset of 'm' of edges of 'G' is defined as follows

$$M: \{ v \longrightarrow v \in E \mid (v_{out}, v_{in}) \in M \}$$

By defination of disjoint path cover, every vertex of our Graph G has at most one incoming edge in P and atmost 1 outgoing edge in P

Thus P defines an disjoint path cover with V-K edges.

We conclude that P has exactly K paths, therefore we can find a minimum disjoint path covers in G by computing maximum matching in G' using ford fulkerson's max flow algorithm it will take O(VE) time

Our goal is to find minimum number of students so that class is filled to capacity

We find minimum disjoint paths that covers all vertices

With this we can find the maximum number of students required so that classes are filled at full capacity

Compute the minimum disjoint path cover of a by computing maximum matching of the bipartite graph G'

Then we compute all the paths disjoint path

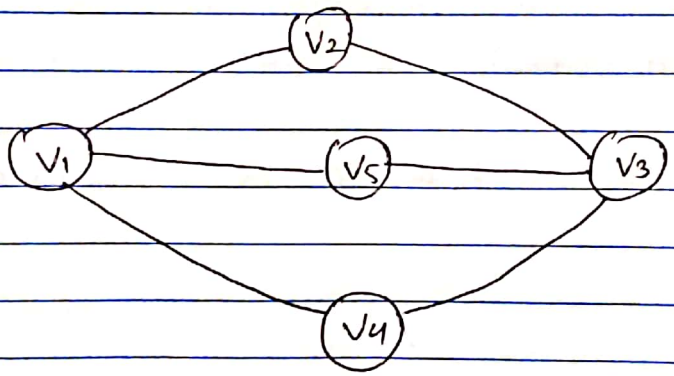
After this we compute the maximum req. capacity for each path and sum them up

$$T = O(V*E)$$

Assign 5

Q4) We can construct a graph connecting all the people who dislike each other. We need to maximize the people invited for the party, so minimize the people eliminated from the graph. Hence it is a vertex cover problem.

In vertex cover problem, we pick the minimum number of vertices which will include all the edges. Eg.



In this graph the vertices v_1 and v_3 cover all edges. Hence the vertix cover problem outputs v_1 and v_3.

Similarily consider the nodes as people and $u - v$ is the people who dislike each other. Hence we need to find the minimum number of people or nodes to eliminate.

To prove this problem is NP complete we need the prove 2 things.

(1) The problem is in NP.

(2) There is another problem in NP that can be reduced to this problem.

Proof of NP

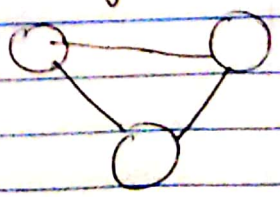
(1) If V' is the output of the vertex cover problem ~~then check~~ of length k then $V' < k$.

(2) ~~Check~~ If $V' = \{V_1, \ldots V_2\}$ then there exist atleast one path from u connecting V_1 or V_2 where u are the other nodes

Hence the vertex cover problem can be verified in polynomial time.

Clique problem of NP reduction

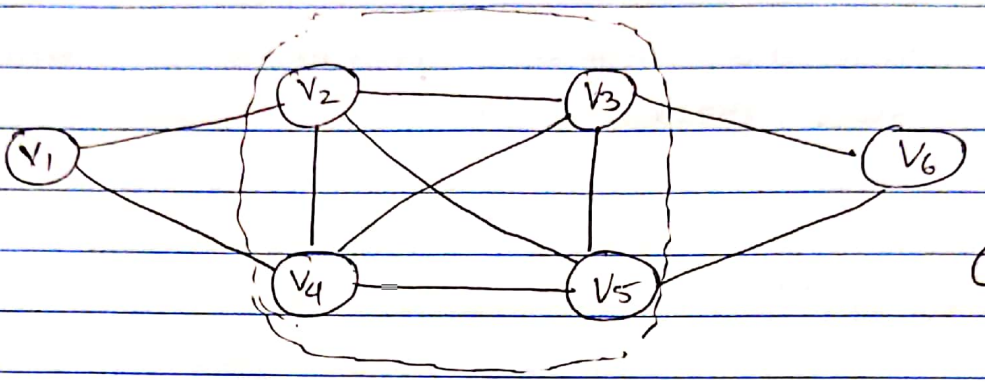
In clique problem every node must be connected to every other node. Eg 3 clique :-



It is in NP.

We can reduce clique problem to solve the above problem.

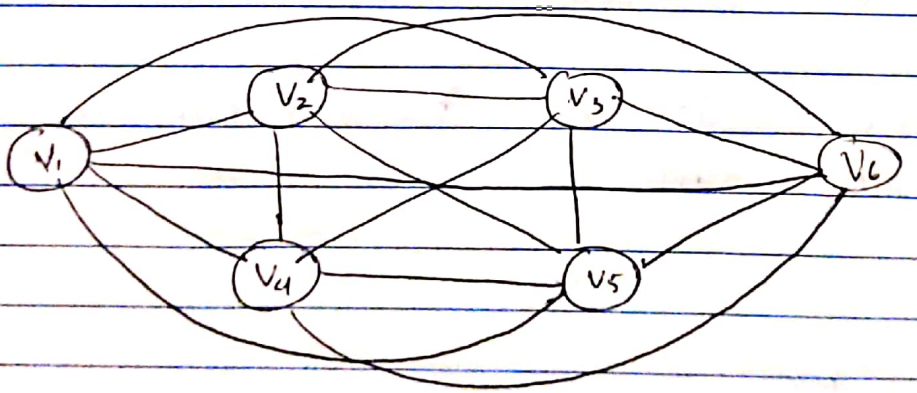
If the graph of the vertex cover problem is



G

We find the max clique; that is 4 in the above graph.

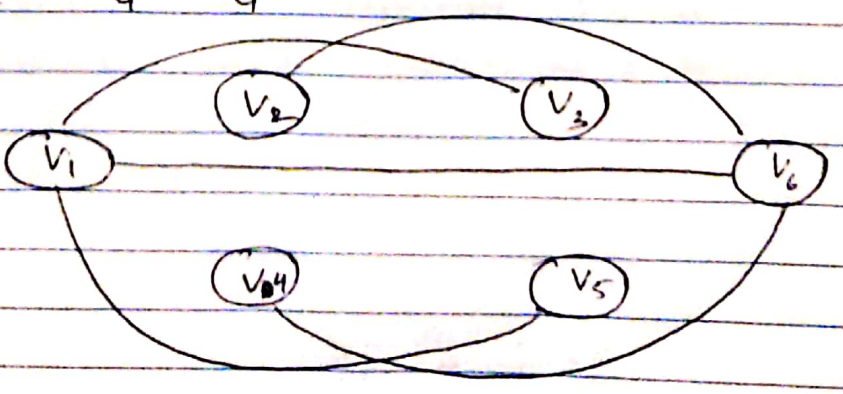
Looking at the number of nodes i.e. 6 construct a 6 clique graph.



G'

Let $G'' = G' - G$

=



For the above graph, the vertex cover = $\{v_1, v_6\}$
= 2

And result we obtained through clique =

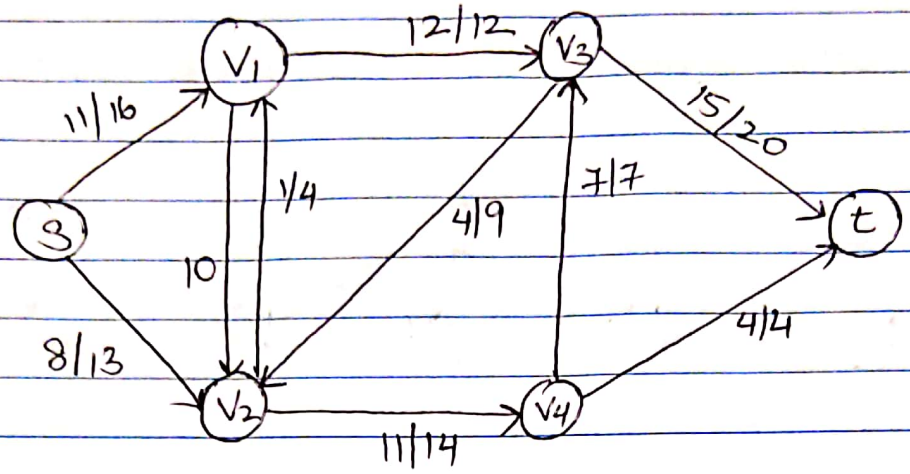
No. of Node - Max clique

$\Rightarrow 6 - 4 = 2$

Hence verified and reduced clique problem to our problem.

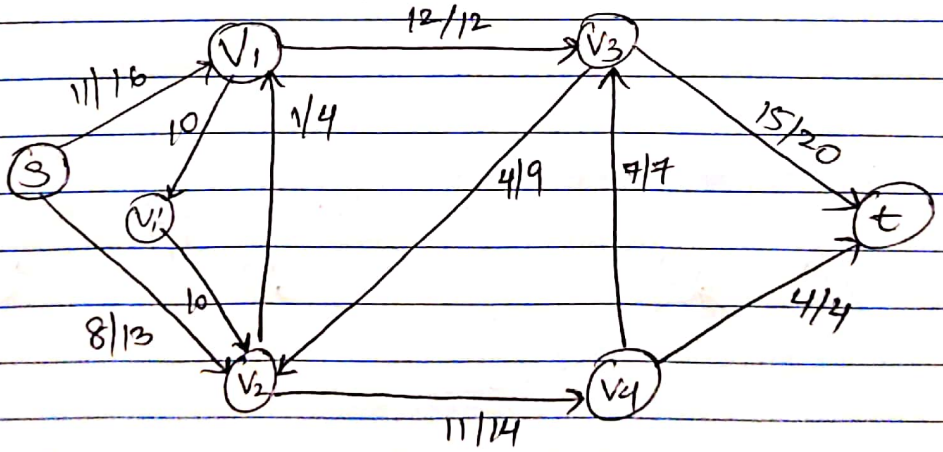
Hence our problem is in NP complete.

Q5)



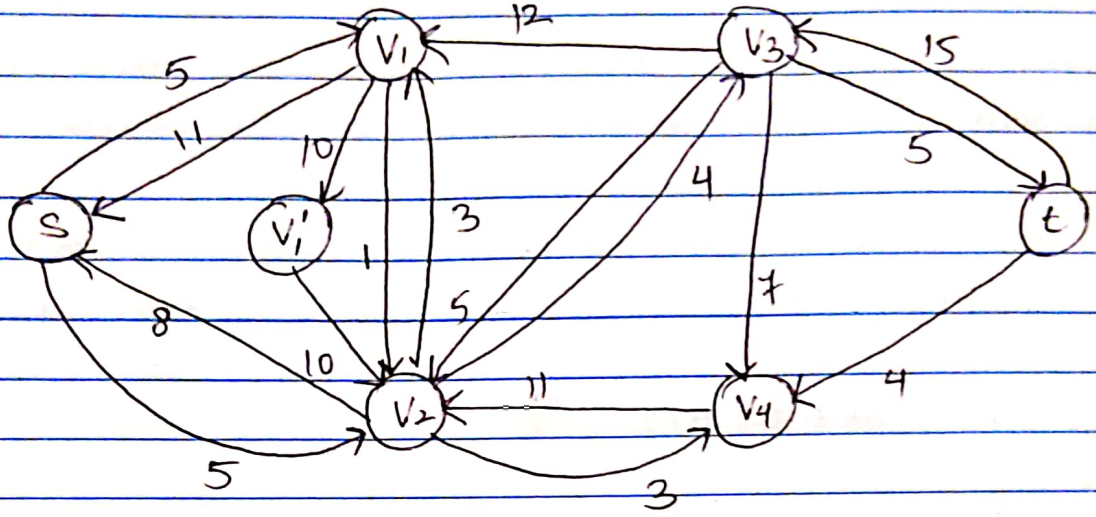
In the above graph we add another node between V_1 and V_2 since we cannot have a reverse flow on the same node

∴ Graph:-



Max flow in the above graph = 19 //

Residual Network:-



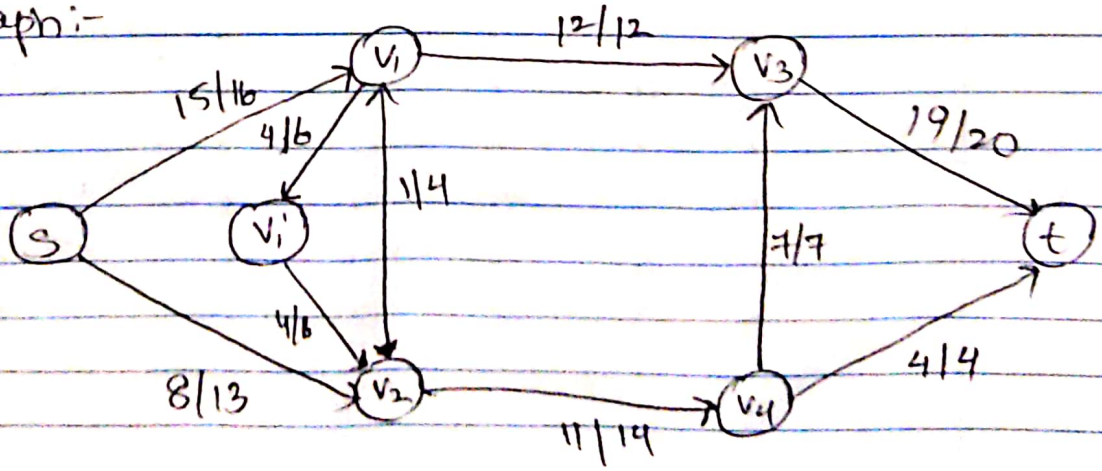
taking $s - v_1 - v_1' - v_2 - v_3 - t$ as augmented path

the bottleneck is $\min\{5, 10, 10, 4, 5\}$

$= 4$

∴ 4 can be added through this augmented path

Graph:-



∴ New max flow = $19 + 4 = 23$