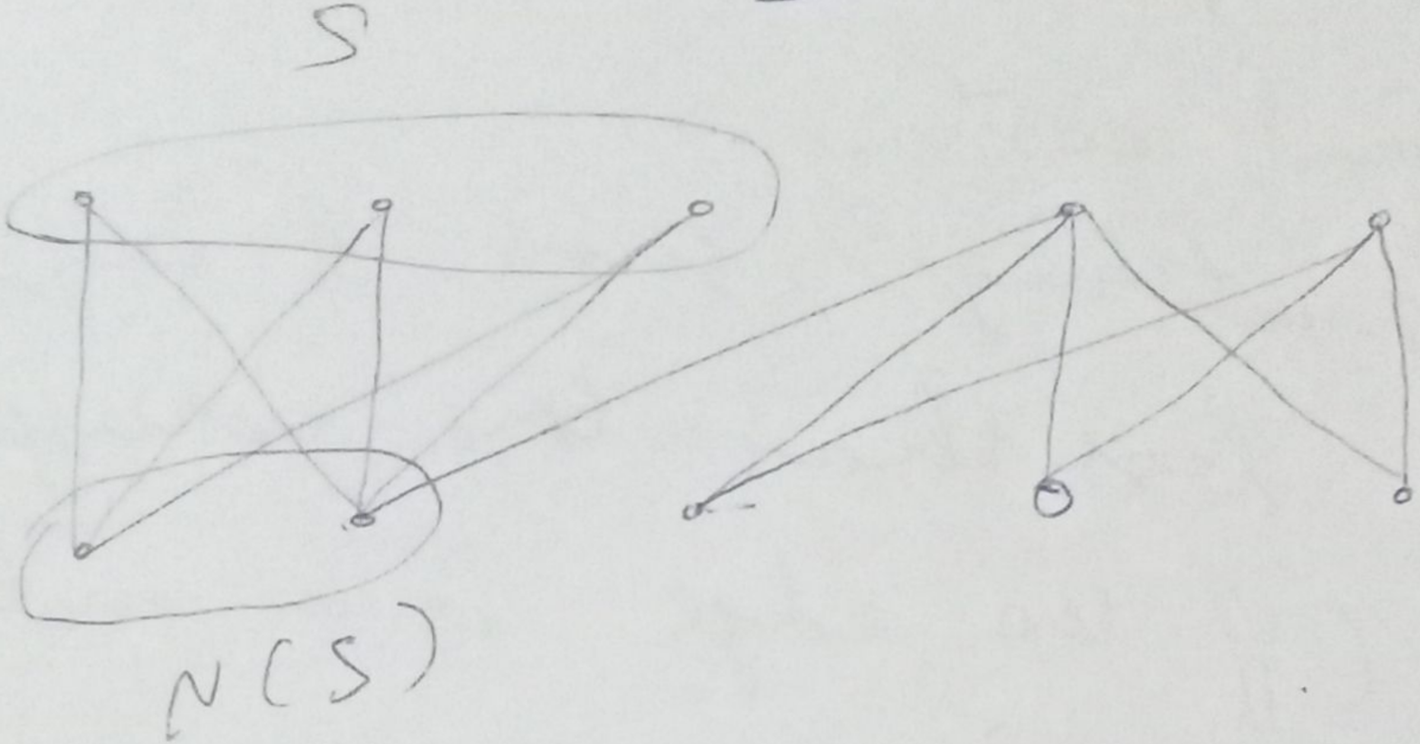
Oct 15, 2025

<u>LO</u>



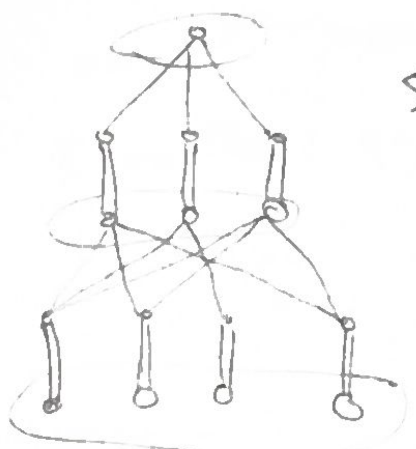
$N(S)$: neighbors set of S

$|S| > |N(S)| \Rightarrow$ No perfect matching otherway around,

G_1 is a bipartite graph with no perfect matching, then $\exists S \subset V$ $|S| > |N(S)|$

Take a max matching M that is not a perfect matching, then there's at least one unmatched vertex on both sides.

Take an unmatched vertex and do BFS.

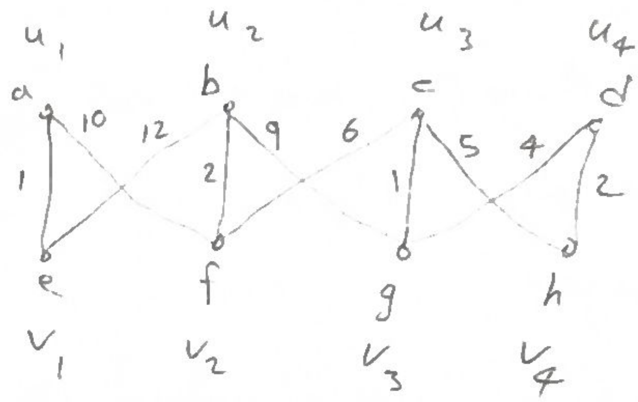


See pic

|| - in matching

| - Not in matching
Edge in graph.

~~Go~~ a
Formalize it!



dual $\not{q}$: $\max(u_1 + u_2 + u_3 + u_4 + v_1 + v_2 + v_3 + v_4)$

~~Suppose someone given dual optimal~~

Dual cost $\leq$ primal cost.

Because in the any matching, sum of any two vertices is less than the corresponding edge in matching (if the edge is in matching)

Dual optimal = Primal optimal

suppose someone given dual optimal, construct primal optimal, take only the edges whose constraints are tight.

That graph must ~~be a~~ ~~b~~ have a perfect matching
↑ check?

why? Because if there is no perfect matching then $\exists S \subset V$ $|S| > |N(S)|$, so dual objective can be increased $\Rightarrow$ given is not optimal

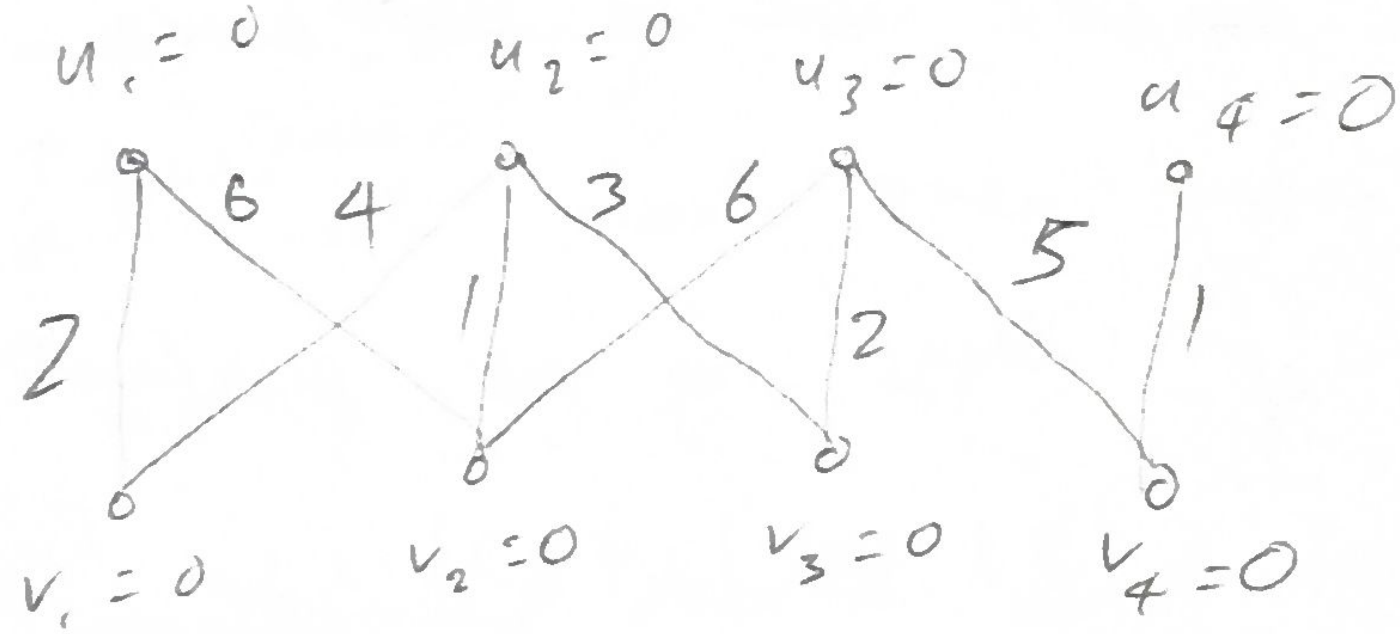
Formalize!

Oct 22, 2025

LO

Given a bipartite graph G, find minimum weight perfect matching

G



Dual: Give 0 weights to vertices

solve dual problem, construct a smaller graph with only the tight edges.

If the tight edge graph is not a perfect matching, change vertex weights u_S & v_S to increase no. of tight edges. How?

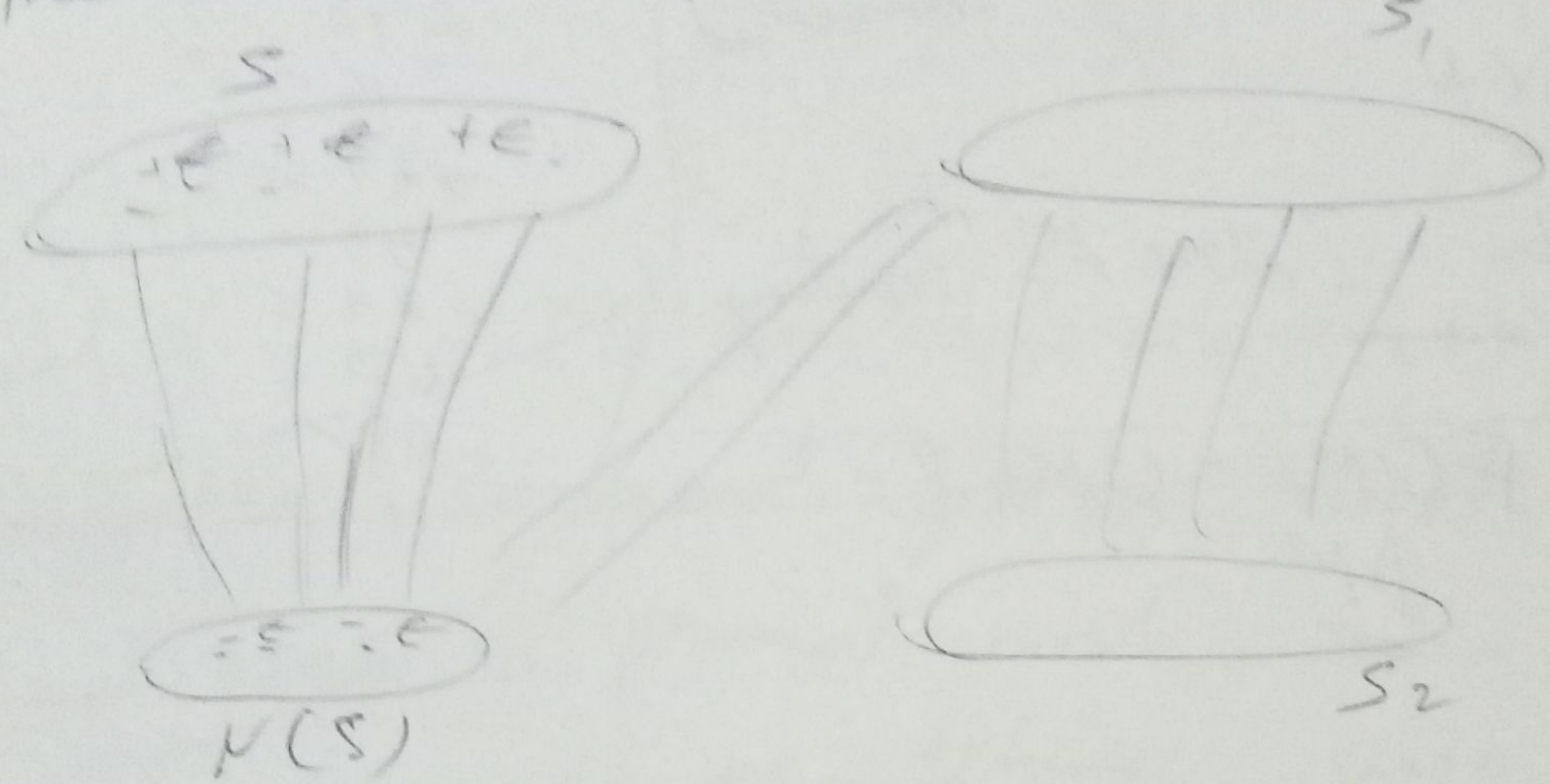
There exists $|S| > |N(S)|$

$S,$

graph with only the tight edges.

If the tight edge graph is not a perfect/maximum matching, change vertex weights u_S & v_S to increase no. of tight edges. How?

There exists $|S| > |N(S)|$



Add $+\epsilon$ to all in S

Add $-\epsilon$ to all in $N(S)$

In the process, tights btw S & $N(S)$ remain same, btw S_1 & S_2 remain same.

btw S_1 & $N(S)$ become untight and some from S become tight with S_2 so that maximum matching increases.

Total change is +ve because $|S| > |N(S)|$

$$\text{Change} = +\epsilon|S| - \epsilon|N(S)|$$
$$= \epsilon(|S| - |N(S)|)$$

Edges btw S & $N(S)$ and S_1 & S_2 remain in the smaller graph. Some edge btw S_1 & $N(S)$ will go away from smaller graph and some edge btw S & S_2 is added to smaller graph.

4th assignment:

Input: Bipartite graph with edge weights which has a maximum matching

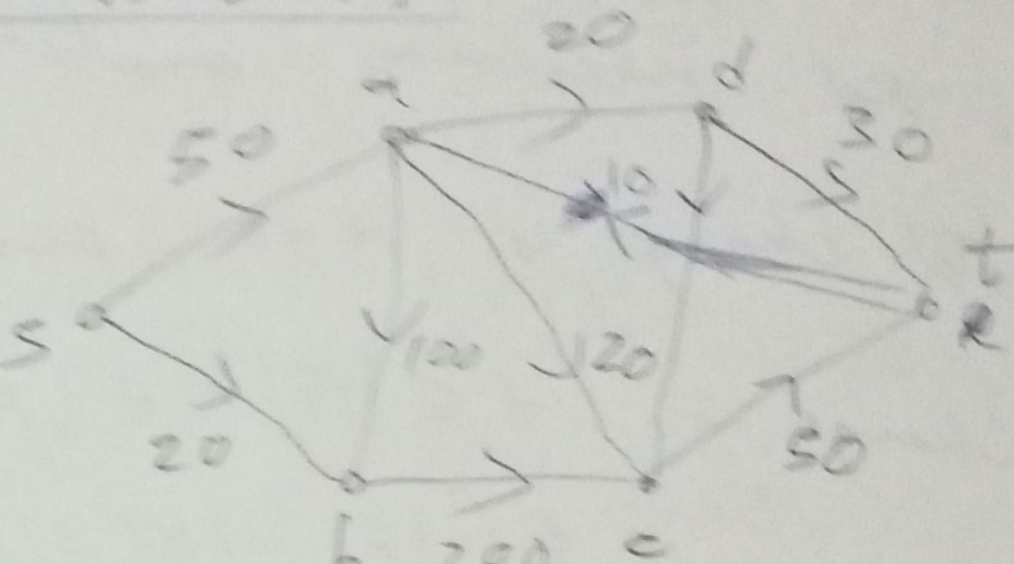
Output: Min. weight perfect

ANS.

Oct 28 2022

10

10:14 Network Flow Problem



Edges are pipes

S is source

t is destination

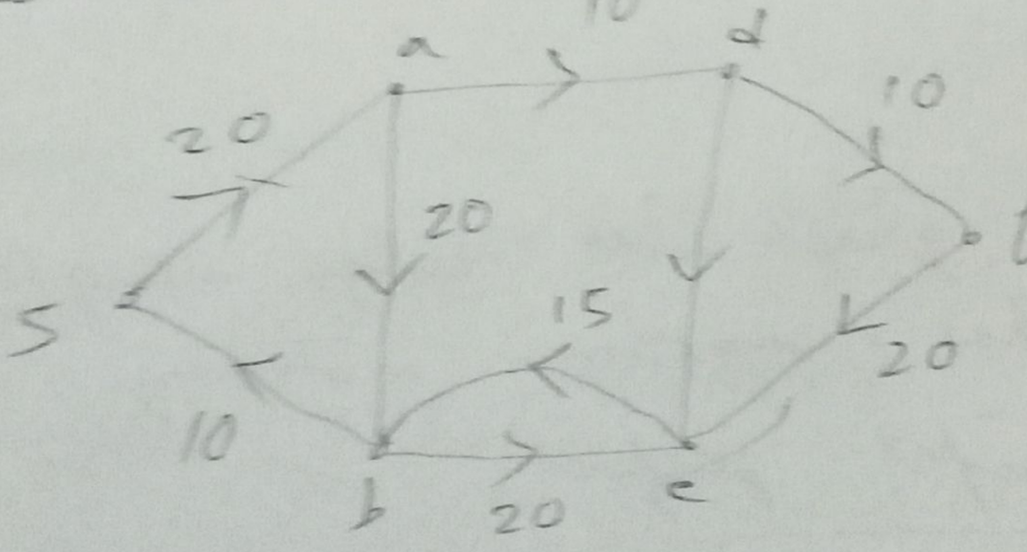
Other points are junctions.

Max flow in a pipe = edge weight (capacity)

" = " "

Incoming flow = Outgoing flow

Junctions cannot store, create or destroy flow

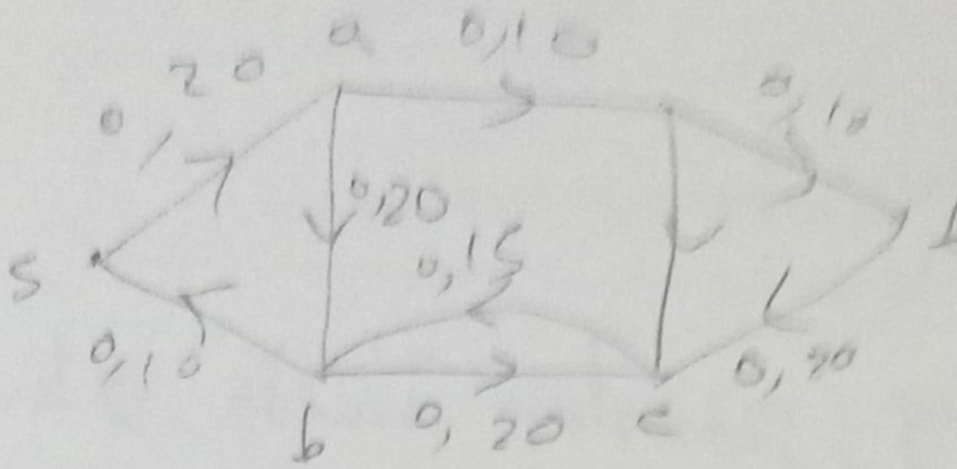


Since intermediate vertices do not change the flow,

From S: Outgoing - incoming = From t: Incoming - outgoing

can optimize anyone

t,d

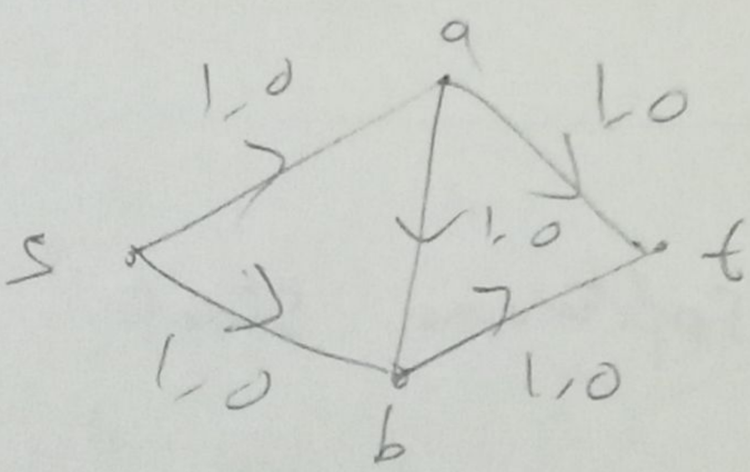


0, 20 meaning
flow = 0
capacity = 20
can write 20, 0 due
upto notation

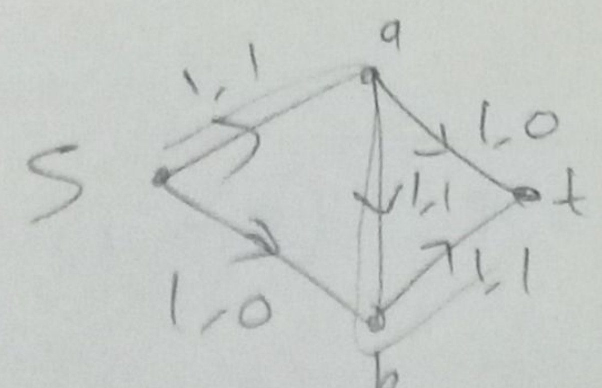
choose path s - a - b - t adds flow 10

Any path will not work

because

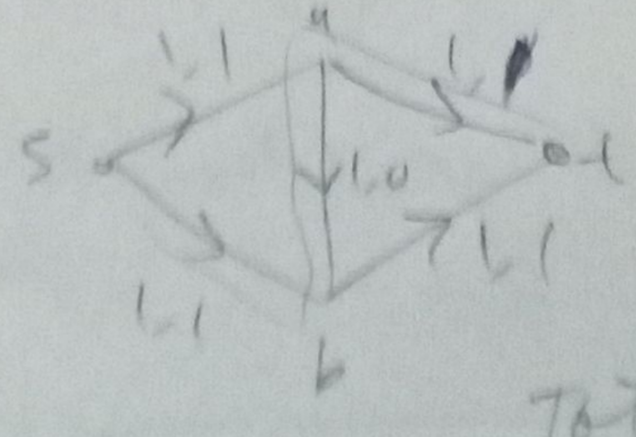


If we choose s - a - b - t,
total flow = 1
But optimum = 2



?

To fix:
when traverse along edge, increase flow
when traverse opposite to edge, decrease flow



s - b - a - t

Total flow = 2