Network Flows

February 3, 2017 1:59 PM

(are a subset of linear programming)

Lemma: (from last time)
For any flow f and any cut (S,T),
$\text{size}(f) \leq \text{cap}(S,T)$

① flow across cut$(S,T) \leq \text{cap}(S,T)$

② flow across cut(S,T) =
flow across cut$(S-v,\ T+v)$

shift a vertex from S to T

source vertex

③ flow across cut$(\{s\},\ V-\{s\}) = \text{size}(f)$

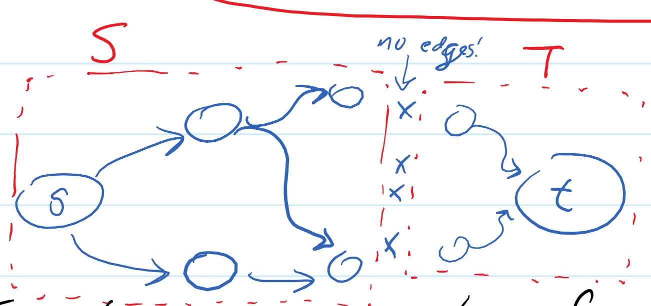
Now:

Correctness of Ford-Fulkerson

Theorem: If residual network G_f has no augmenting path, then f is a max size flow.

Proof:

e.g. G_f:
(RESIDUAL!)



T is unreachable from S now!

Let S be the set of vertices reachable

by directed path in G_f.
Let $T = V - S$.

All edges from S to T are not in the residual network, so those edges must be saturated! (Full. No capacity left.)
$\Rightarrow$ flow across cut $(S,T) = cap(S,T)$!

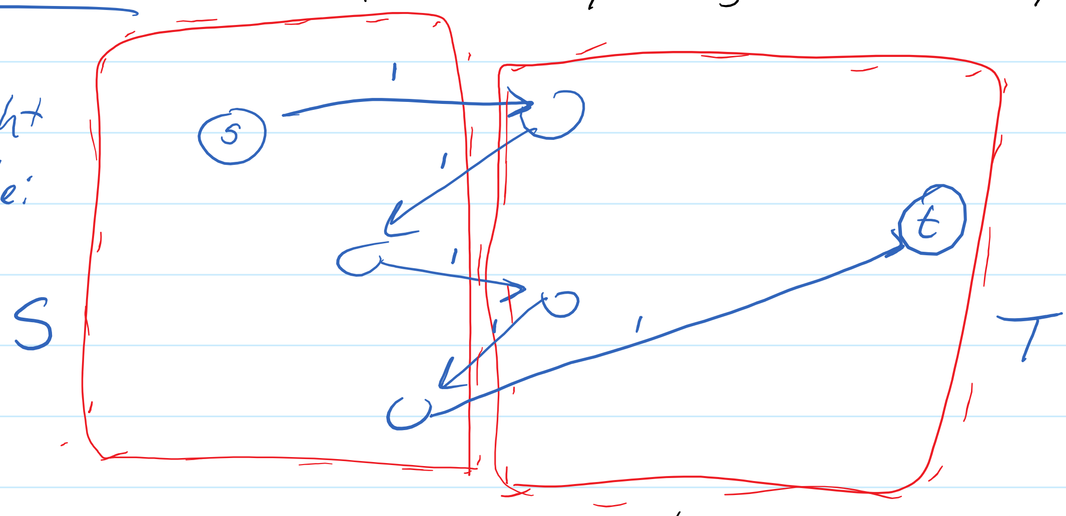
So we know $size(f) = f(S,T) = cap(S,T)$
because $f(u,v) = c(u,v)$
for all $u \in S$, $v \in T$.
and size of any flow $\leq cap(S,T)$

which leads us to....

Max Flow - Min Cut Theorem

size of max-flow = capacity of min capacity cut

Slight aside:



From $S \to T$, this cut has capacity 3, and flow 1. (i.e. $(S \to T) - (T \to S)$).

Continued.

Continued.

Proof: use proof of correctness of Ford-Fulkerson.

Integrality Theorem:

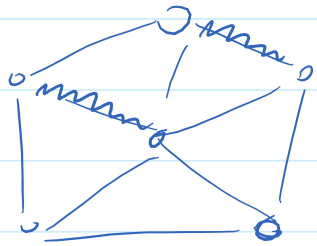
If all capacities are integers then there exists a max flow such that every edge has integer flow.

Proof is by induction on number of augmentations of Ford-Fulkerson.

Maximum Matching in a Bipartite Graph

A matching in a graph is a set of edges in a graph such that no two segments in the matching have a common endpoint.

eg. A matching: (non maximal)



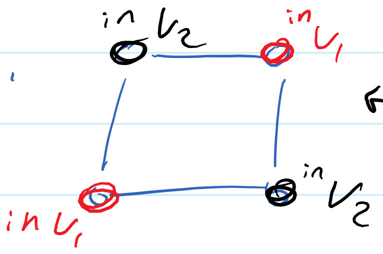
~~~ matching

(The maximal for these 6 vertices is 3 edges! Add the bottom edge to make it so.)

## Bipartite Graph:
~~~

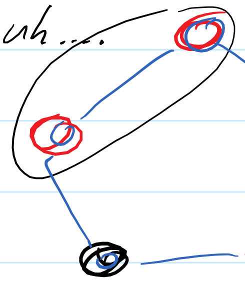
Vertices can be partitioned in sets V_1 & V_2 so that for all edges, one endpoints is in V_1 & the other is in V_2.

eg.



← works!

uh....



FAILS!

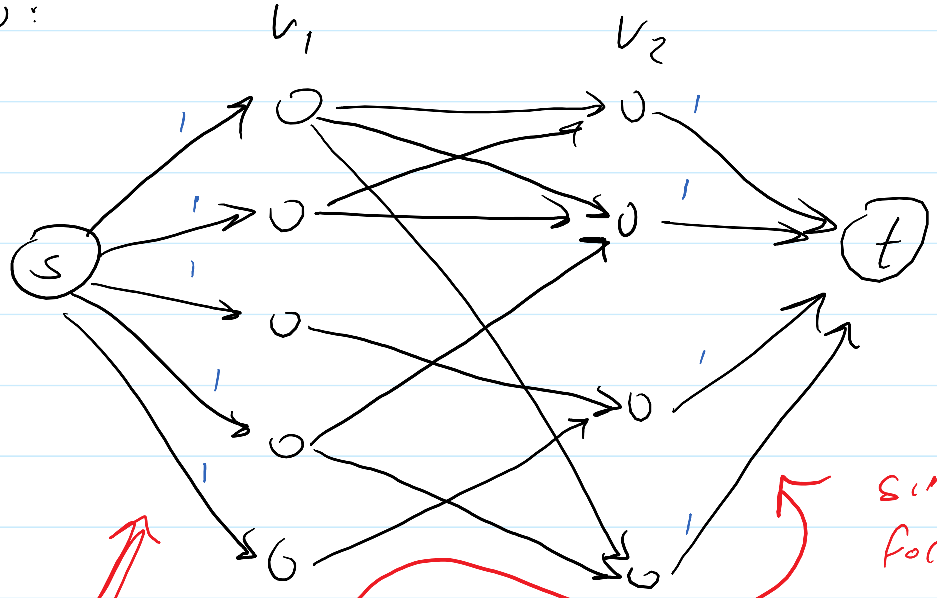
(all odd cycles fail)

→ This is often an <u>assignment problem</u>:

eg. assigning workers to jobs, stable marriage, etc.

So we can solve Maximum Matching in a bipartite graph with network flow:

eg.



similarly for each $v \in V_2$.

weights of 1 on each

weights don't matter in middle,

weights ...
mean each
$v \in V$, can
only be chosen
once!

weights dont matter in middle,
so pick whatever.