Max Flow

Min Cut

Ford-Fulkerson Algorithm

(n vertices, m edges)

Start with 0 flow f.

Find P: Use BFS or DFS
$O(n+m) = O(m)$

while $\exists$ an s-t path P in G_f:

Update R_f:
≤ 2 modifications to each edge in P
$= O(m)$

Augment f with Bottleneck(P) flow

$O(m)$

return f.

Each loop iteration is $O(m)$.

Ford-Fulkerson Algorithm

Start with 0 flow f.

while $\exists$ an s-t path P in G_f:
    Augment f with Bottleneck(P) flow

return f.

Upper bound any flow by $C = \sum_{e \text{ out of } s} c_e$.

How many times do we execute the loop? $\boxed{\leq C \text{ iterations}}$

Assume all capacities are integers.

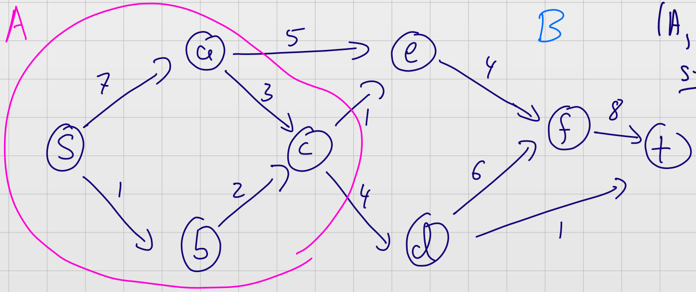
$\Rightarrow$ flow in FF is always integer-valued.

Each iteration adds flow (must take a forward edge out of s)

Runtime: $O(mC)$

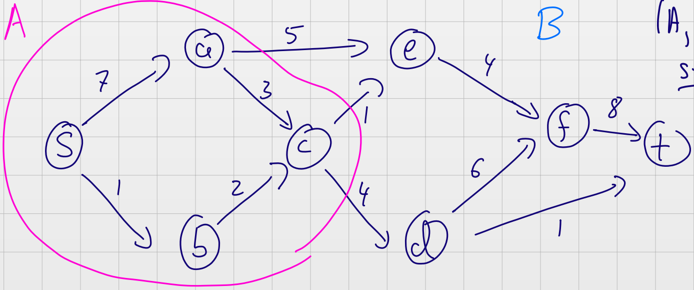
Ford-Fulkerson Algorithm

Why does FF return a maximal flow?



(A, B) is an **s-t cut** if

$s \in A$, $t \in B$ and A and B are a partition of V.



(A, B) is an **s-t cut** if $s \in A$, $t \in B$ and A and B are a partition of V.

The flow out of A is $f^{out}(A) - f^{in}(A)$

$$f^{out}(A) = \sum_{e \text{ out of } A} f(e) \qquad f^{in}(A) = \sum_{e \text{ into } A} f(e)$$

The flow into B is

$$f^{in}(B) - f^{out}(B)$$

Lemma Let (A,B) be an s-t cut. Then $v(f) = f^{out}(A) - f^{in}(A)$.

Proof.

$$f^{out}(A) - f^{in}(A) = \sum_{a \in A} f^{out}(a) - \sum_{a \in A} f^{in}(a)$$

$$= \sum_{a \in A \setminus \{s\}} \left(f^{out}(a) - f^{in}(a) \right) + f^{out}(s) - f^{in}(s)$$

$$= 0 + f^{out}(s)$$

$$= \sum_{e \text{ out of } s} f(e) = v(f).$$

Lemma 1 Let (A, B) be an s-t cut.
Then $v(f) = f^{out}(A) - f^{in}(A)$

Lemma 2 Let (A, B) be an s-t cut.
Then $v(f) \leq \sum_{e \text{ out of } A} c_e = C(A, B)$

Gives us a general upper bound on the value of any flow!

capacity of the cut

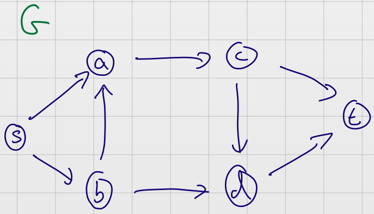
Proof: $v(f) \leq f^{out}(A) = \sum_{e \text{ out of } A} f(e) \leq \sum_{e \text{ out of } A} c_e = C(A, B)$

Lemma 1 Let (A, B) be an s-t cut.
Then $v(f) = f^{out}(A) - f^{in}(A)$

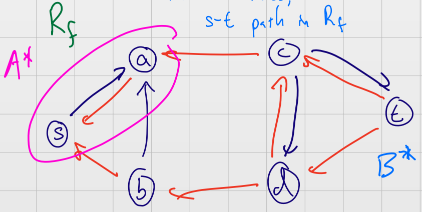
Lemma 2 Let (A, B) be an s-t cut.
Then $v(f) \leq \sum_{\substack{e \text{ out} \\ \text{of } A}} c_e = \underbrace{C(A, B)}_{\text{capacity of the cut}}$

Corollary If f is a flow and (A, B) is an s-t cut with $v(f) = C(A, B)$, then f is a maximal flow.

Idea: Find a cut (A^*, B^*) using FF's residual graph and show that the output flow f achieves $v(f) = C(A^*, B^*)$.



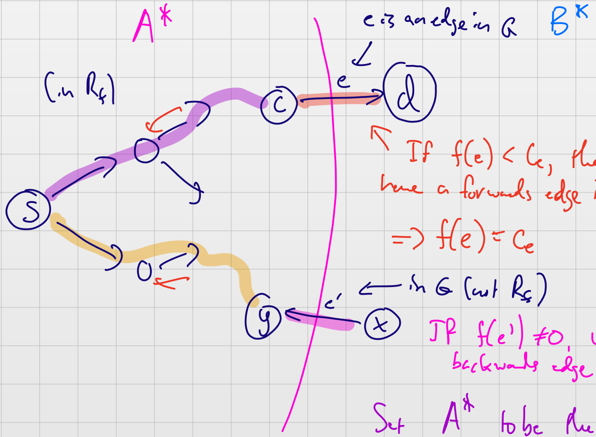
After FF finishes, there is no s-t path in R_f



Set A^* to be the set of all nodes v with an s-v path in R_f.

$s \in A^*$, $t \notin A^* \Rightarrow t \in B^*$

$\Rightarrow (A^*, B^*)$ is an s-t cut.

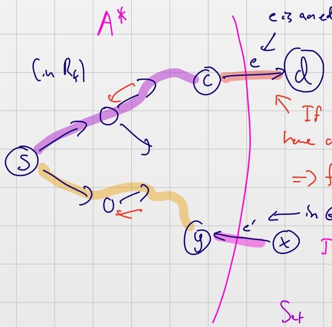


If $f(e) < c_e$, then e would have a forwards edge in R_f.

$\Rightarrow f(e) = c_e$

If $f(e') \neq 0$, we would have a backwards edge $\Rightarrow f(e') = 0$.

Set A^* to be the set of all nodes v with an s-v path in R_f.



e is an edge in G

B^*

If $f(e) < c_e$, then e would have a forwards edge in R_f.

$\Rightarrow f(e) = c_e$

← in G (not R_f)

If $f(e') \neq 0$, we would have a backwards edge $\Rightarrow f(e') = 0$.

Set A^* to be the set of all nodes v with an s–v path in R_f.

$$v(f) = f^{out}(A^*) - f^{in}(A^*)$$

$$= \sum_{e \text{ out of } A^*} f(e) - \sum_{e \text{ into } A^*} f(e)$$

$$= \sum_{e \text{ out of } A} c_e - 0$$

$$= C(A,B)$$

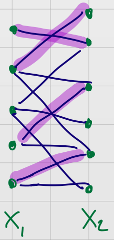
Shown: The output flow f has $v(f) = C(A,B)$

$\Rightarrow$ f is maximal.

Bipartite Matching Problem

Have a bipartite graph $G=(V,E)$.

<u>Bipartite</u>: $V = X_1 \sqcup X_2$, every edge goes from X_1 to X_2.

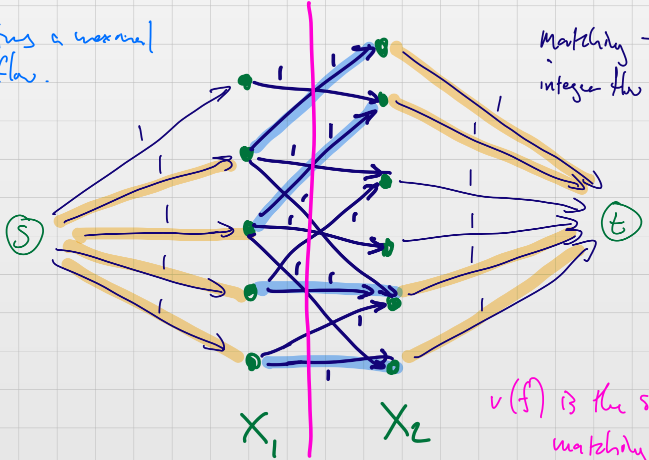


A match is a subset $M \subseteq E$ such that no vertex is adjacent to more than 1 edge in M

FF returns a maximal integer flow.

matching → flow

integer flow → matching



v(f) is the size of the matching.