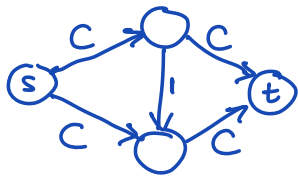


Max Flow: Ford-Fulkerson computes max flow

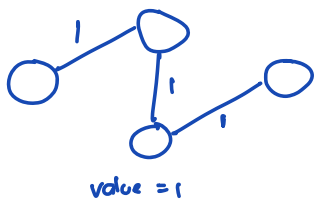
in time  $O(m \cdot C)$  where  $m = \# \text{ edges}$   
 $C = \text{capacity of max flow}$

- assuming integer capacities  $c_e \in \mathbb{N}$
- FF takes  $O(C)$  iterations (each  $O(m)$  time)
- in worst case, FF takes  $\Theta(C)$  iterations.

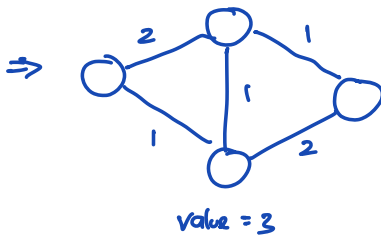
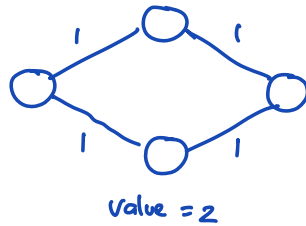


$C \gg 1$   
 max flow value =  $2C$

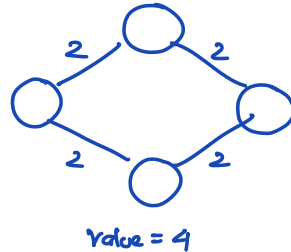
but FF can do:



$\Rightarrow$

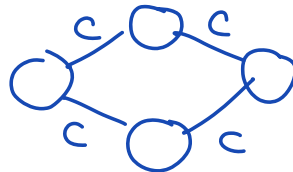


$\Rightarrow$



⋮

..

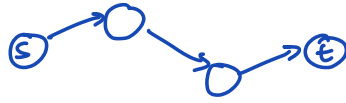


takes  $2C$  iterations.

But have good rules for choosing augmenting path:

- can use "scaling max flow" [§ 7.3]
  - $\Rightarrow$  takes  $O(m \cdot \log C)$  iterations
  - $\Rightarrow$  running time is  $O(m^2 \cdot \log C)$

- can use BFS to find augmenting path in  $G_f$



$$G = (V, E)$$

$m = \# \text{ edges in } G$

$n = \# \text{ vertices}$

$\Rightarrow$  Edmonds - Karp algorithm

$\Rightarrow$  takes  $O(m \cdot n)$  iterations

$\Rightarrow$  Running time is  $O(m^2 \cdot n)$

Notes:

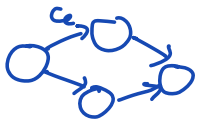
1)

Max - Flow Min - Cut Theorem

In every flow network, the max value of  $s-t$  flow is equal to min capacity of  $s-t$  cut:

$$\max_{\text{flow } f} \text{val}(f) = \min_{\text{cut}(A,B)} c(A,B)$$

- this holds even when  $c_e \in \mathbb{R}_+$

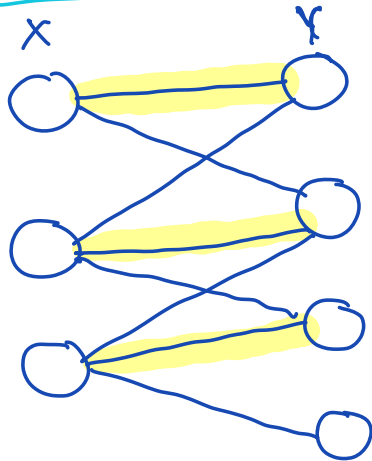


- 2) If all  $c_e \in \mathbb{N}$ , then there is a max flow with only integer values  $f(e) \in \mathbb{Z}$   
(eg. one returned by FF alg.)

Application: Bipartite Matching.

Let  $G = (V, E)$  undirected unweighted bipartite graph

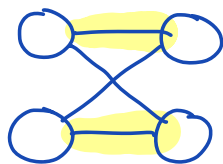
$$V = X \cup Y$$



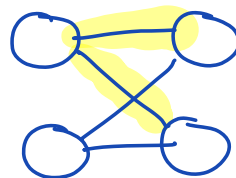
Goal: Find a matching in  $G$  with maximum size (# of edges)

Recall: a matching  $M$  is a collection of edges from  $E$   
s.t. no two edges in  $M$  have a common vertex.

eg.



matching



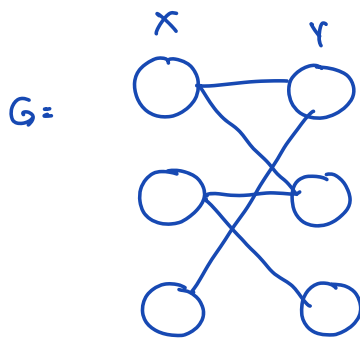
not a matching

Algorithm for Bipartite Matching:  $G = (V, E)$   $V = X \cup Y$

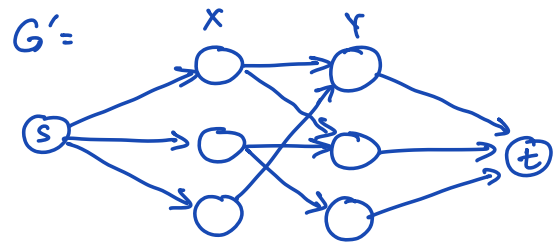
1) Construct flow network  $G' = (V', E')$  s.t.

$$V' = V \cup \{s, t\}$$

$$E' = \left\{ \begin{array}{l} \text{add edges } s \rightarrow x \quad \forall x \in X \quad \text{with capacity } 1, \\ \text{add edges } y \rightarrow t \quad \forall y \in Y \quad \text{with capacity } 1, \\ \text{add edges } x \rightarrow y \quad \forall \begin{matrix} (x,y) \in E \\ \uparrow \\ x \in X \quad y \in Y \end{matrix} \quad \text{with capacity } 1 \end{array} \right\}$$



$\rightsquigarrow$



all edges capacity  $C_e = 1$

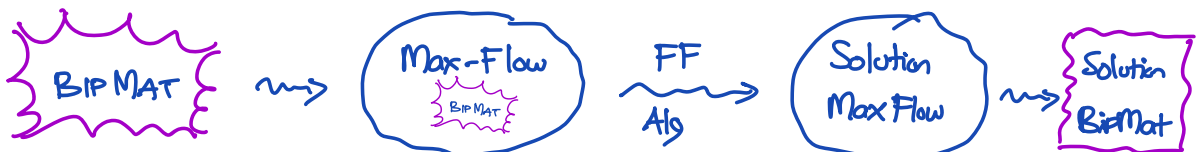
2) Find max  $s-t$  flow in  $G'$

Then: value of max flow in  $G' =$  size of max matching in  $G$

can find max matching in  $G$  from max flow in  $G'$

This is a reduction of Bipartite Matching to Max Flow

- given an algorithm for Max-Flow (e.g. FF)
- convert Bipartite Matching into an instance of Max-Flow
- apply algorithm  $A$  to solve max Flow
- extract solution to Bipartite Matching from output of  $A$



this shows: Max Flow is at least as hard as Bipartite Matching

$$\text{BIP MAT} \leq \text{MAX-FLOW}$$

Analysis of Algorithm:

1) Correctness: ✓

Claim: • Size of max matching in  $G$   
= value of max flow in  $G'$   
• The edges in max matching in  $G$   
= edges that carry flow in  $G'$

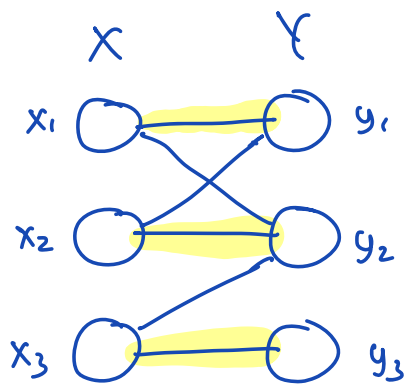
2) Running time:

Claim: FF can be used to solve Bipartite Matching  
in  $O(m \cdot n)$

Proof of Correctness:

1) WTS: matching of size  $k \Rightarrow$  flow of value  $k$

Suppose  $G$  has matching of size  $k$



$$M = \{e_1, \dots, e_k\} \subset E$$

$$e_1 = (x_{i_1}, y_{i_1})$$

$$e_2 = (x_{i_2}, y_{i_2})$$

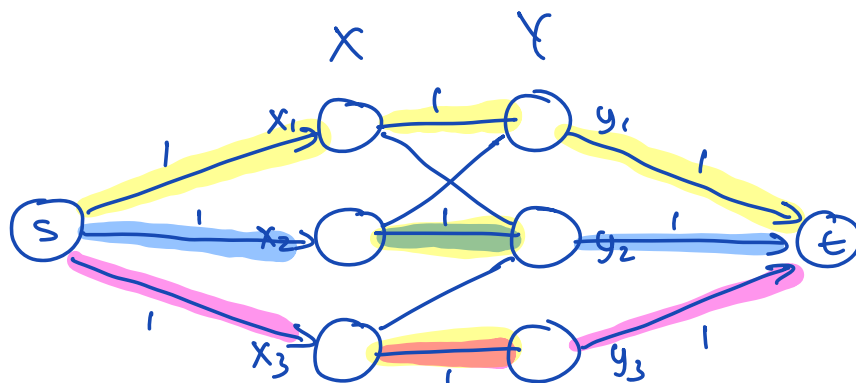
$\vdots$

$$e_k = (x_{i_k}, y_{i_k})$$

then in  $G'$ , can define flow:

$\forall j = 1, \dots, k$ : push flow  $s \rightarrow x_{i_j} \xrightarrow{e_j} y_{i_j} \rightarrow t$

$f(e) = 1$  along edges of path  $\curvearrowright$



this gives a flow (sat. capacity & conservation)  
of value =  $k$

2) WTS: max-flow in  $G'$  of value  $k \Rightarrow$  matching in  $G$  of size  $k$ .

- Since  $G'$  has integer capacities, then there exists a max flow  $f$  with integer values  $f(e) \in \mathbb{Z}$  (by FF discussion)

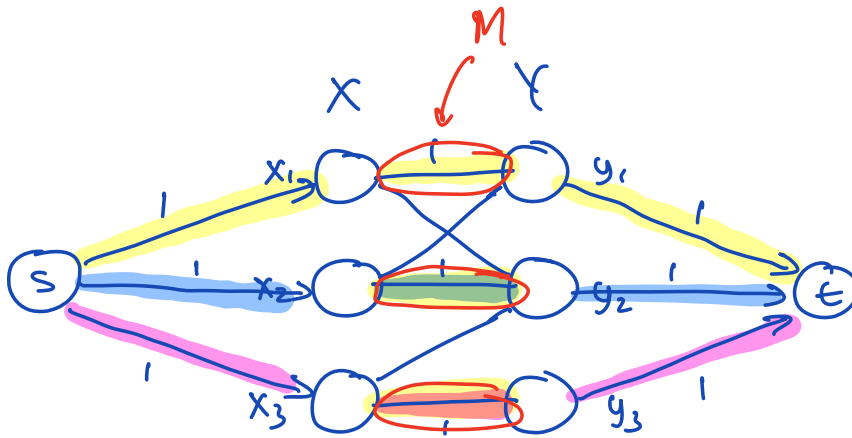
• Take such a max flow  $f$  in  $G'$ ,  $\text{val}(f) = k$

\* since  $C_e = 1 \forall e \in G'$ :

$$0 \leq f(e) \leq C_e = 1$$

\* since  $f(e) \in \mathbb{Z}$ , then:  $f(e) = \{0, 1\}$

Define:  $M = \{e \in E' : e = (x_i, y_j), f(e) = 1\}$

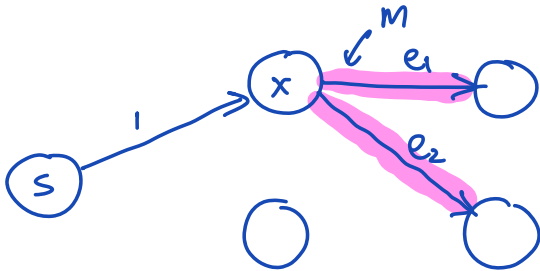


Claim:  $M$  is a matching in  $G$  of size  $k$ .

Proof of claim:

1) each vertex in  $X$  is connected to at most one edge in  $M$ .

• Suppose  $\exists x \in X$  s.t.  $M$  uses  $x$  at least twice, on  $e_1, e_2$



by definition

$$\left. \begin{array}{l} f(e_1) = 1 \\ f(e_2) = 1 \end{array} \right\} f^{\text{out}}(x) \geq 2$$

• but  $x$  has only 1 incoming edge (from  $s$ )

$$\text{so } f^{\text{in}}(x) \leq 1$$

• this contradicts conservation:  $f^{\text{in}}(x) = f^{\text{out}}(x)$ .

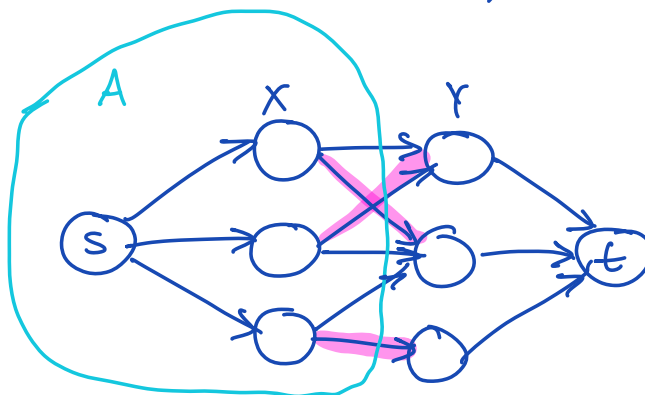
2) each vertex in  $Y$  is connected to at most one edge in  $M$ .

• by similar argument.

so (1) + (2) =  $M$  is a matching.

3) To show size of  $M = k$ :

Consider cut  $(A, B)$ ,  $A = \{s\} \cup X$



then:  $val(f) = \underbrace{f^{out}(A)}_{=|M|} - \underbrace{f^{in}(A)}_{=0}$  [from last time]

$\therefore k = val(f) = |M|$

□

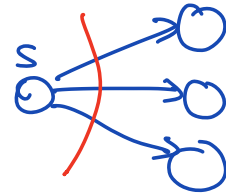
So we have shown correctness:

value of max-flow in  $G'$   
 $=$  size of max-matching in  $G$ .

For running time:

Note:  $C =$  value of max flow

$$\leq \sum_{\substack{e=(s,x): \\ x \in X}} c_e$$



$\therefore C \leq |X| \leq n$

in Alg for Bipartite Matching:

- 1) Convert  $G \rightarrow G'$  [ $O(m+n) = O(m)$ ]
- 2) Find max-flow in  $G'$  [FF:  $O(m \cdot C) = O(m \cdot n)$ ]
- 3) Extract matching in  $G'$  [ $O(n)$ ]

$\therefore$  running time is  $O(m \cdot n)$  to solve Bipartite Matching



using naive  $O(m \cdot C)$

- if use scaling max-flow =

$$O(m^2 \cdot \log C) = O(m^2 \cdot \log n) \quad (\text{slower})$$

- if use Edmonds-Karp =

$$O(m^2 \cdot n) \quad (\text{slower})$$