# Design and Analysis of Algorithms CS218M

## Network Flow Algorithms

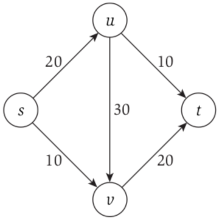
Paritosh Pandya

Indian Institute of Technology, Bombay

Autumn, 2022

# Optimal Network Flow Problem

A flow network is a directed graph $G = (V, E)$ where each edge $(u, v)$ has a non-negative integer capacity $c(u, v) \geq 0$. The graph has source vertex $s$ and sink vertext $t$.




- $s$ has no incoming edges. $t$ has no outgoing edges.
- For all internal nodes $v$ there is a path $s \rightsquigarrow v \rightsquigarrow t$.

# Flow

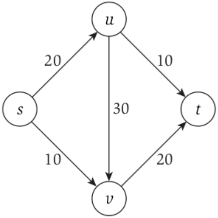
Given a flow network $(G, c)$, a flow is $f : E \rightarrow Z_0$ such that

- (capacity) $0 \leq f(u, v) \leq c(u, v)$.
- (conservation) For all internal nodes $v$ we have
  $$\Sigma_{e \text{ in to } v} f(e) \quad = \quad \Sigma_{e \text{ out of } v} f(e)$$

# Flow

Given a flow network $(G, c)$, a flow is $f : E \to Z_0$ such that

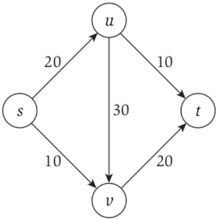
- (capacity) $0 \leq f(u, v) \leq c(u, v)$.
- (conservation) For all internal nodes $v$ we have
  $$\Sigma_{e \text{ in to } v} f(e) \quad = \quad \Sigma_{e \text{ out of } v} f(e)$$



# Flow

Given a flow network $(G, c)$, a flow is $f : E \rightarrow Z_0$ such that

- (capacity) $0 \leq f(u, v) \leq c(u, v)$.
- (conservation) For all internal nodes $v$ we have
  $$\Sigma_{e \text{ in to } v}\ f(e) \quad = \quad \Sigma_{e \text{ out of } v}\ f(e)$$

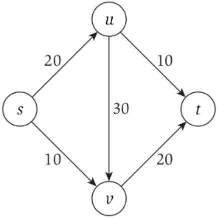


# Flow

Given a flow network $(G, c)$, a flow is $f : E \to Z_0$ such that

- (capacity) $0 \leq f(u, v) \leq c(u, v)$.
- (conservation) For all internal nodes $v$ we have

$$\Sigma_{e \text{ in to } v} f(e) = \Sigma_{e \text{ out of } v} f(e)$$

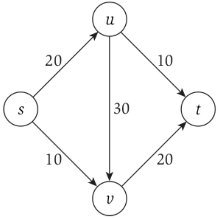



- Let $f^{out}(v) = \Sigma_{e \text{ out of } v} f(e)$ and $f^{in}(v) = \Sigma_{e \text{ in to } v} f(e)$.

# Flow

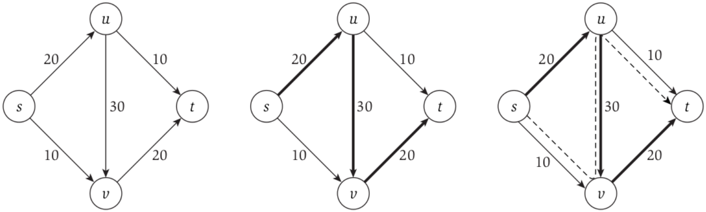
Given a flow network $(G, c)$, a flow is $f : E \to Z_0$ such that

- (capacity) $0 \leq f(u, v) \leq c(u, v)$.
- (conservation) For all internal nodes $v$ we have
$$\Sigma_{e \text{ in to } v} f(e) \quad = \quad \Sigma_{e \text{ out of } v} f(e)$$

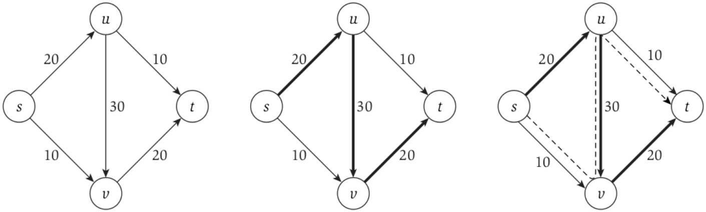



- Let $f^{out}(v) = \Sigma_{e \text{ out of } v} f(e)$ and $f^{in}(v) = \Sigma_{e \text{ in to } v} f(e)$.
- Define value of flow $f$ as $f^{out}(s)$.

# Finding Maximum Flow

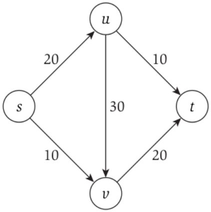


# Finding Maximum Flow




- Find a path $P = s \leadsto t$. Let $Bottleneck(P)$ be the smallest capacity on the path. Initial flow $f$ has value $Bottleneck(P)$.

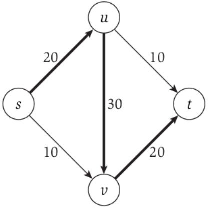
# Finding Maximum Flow

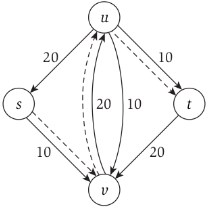


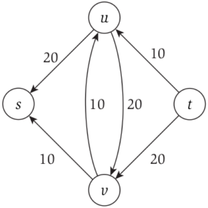

- Find a path $P = s \rightsquigarrow t$. Let $Bottleneck(P)$ be the smallest capacity on the path. Initial flow $f$ has value $Bottleneck(P)$.
- Given current flow $f$, find augmenting path $P = s \rightsquigarrow t$. Check that it is feasible and push $Bottleneck(P)$ additional flow to get revised flow $f'$.

# Residual Graph

Given flow $f$ for flow network $G, c$, define Residual graph $G_f$.

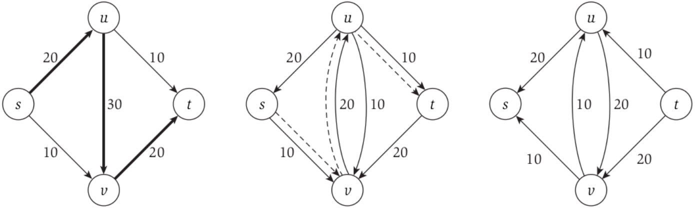


# Residual Graph

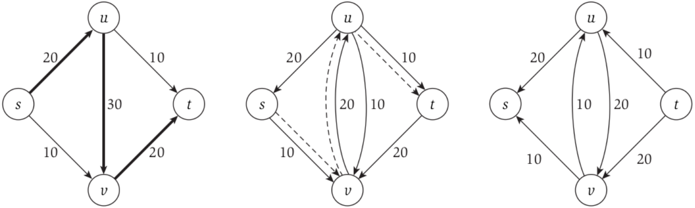
Given flow $f$ for flow network $G, c$, define Residual graph $G_f$.




- Forward edges: Edges $e$ with residual capacity $c(e) - f(e) > 0$.

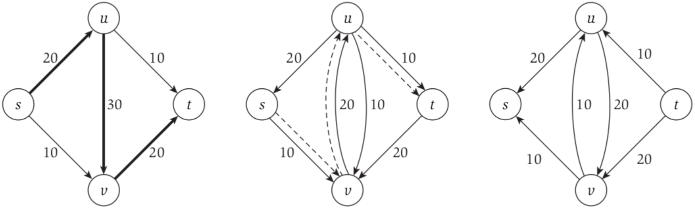
# Residual Graph

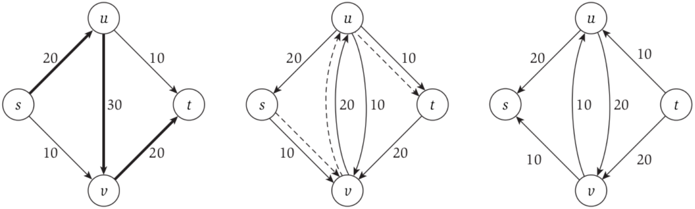
Given flow $f$ for flow network $G, c$, define Residual graph $G_f$.




- Forward edges: Edges $e$ with residual capacity $c(e) - f(e) > 0$.
- Backward edges: Reverse $\overline{e}$ of edges $e$ with $f(e) > 0$ allowing reverse flow upto $f(e)$.

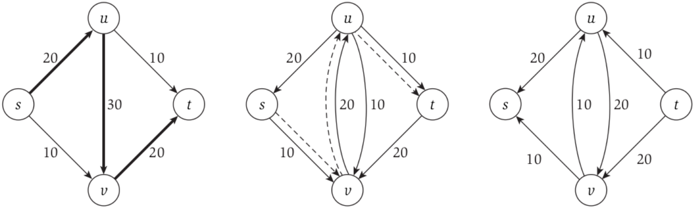
# Augmenting the Flow



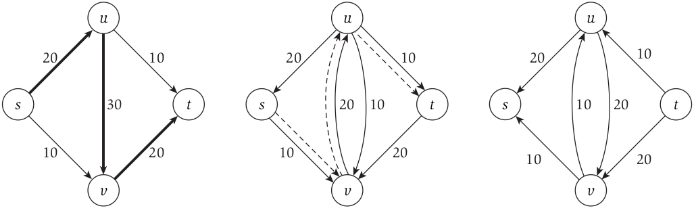



- Augmenting Path $P = s \rightsquigarrow t$ in residual $G_f$ with $b = Bottleneck(P, f)$ being the smallest capacity on $P$.

# Augmenting the Flow

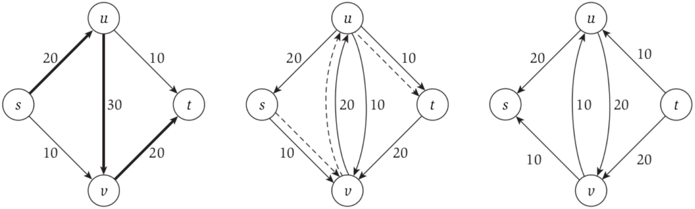



- Augmenting Path $P = s \rightsquigarrow t$ in residual $G_f$ with $b = Bottleneck(P, f)$ being the smallest capacity on $P$.
- Augmented Flow $f'$ is $f$ modified as follows:
  - for each forward edge $e$ on $P$, increase $f'(e) = f(e) + b$
  - for each backward edge $\overline{e}$ on $P$, decrease $f'(e) = f(e) - b$.




- Augmenting Path $P = s \rightsquigarrow t$ in residual $G_f$ with $b = Bottleneck(P, f)$ being the smallest capacity on $P$.
- Augmented Flow $f'$ is $f$ modified as follows:
  - for each forward edge $e$ on $P$, increase $f'(e) = f(e) + b$
  - for each backward edge $\overline{e}$ on $P$, decrease $f'(e) = f(e) - b$.
- Claim: $f'$ is a valid flow in $G, c$.

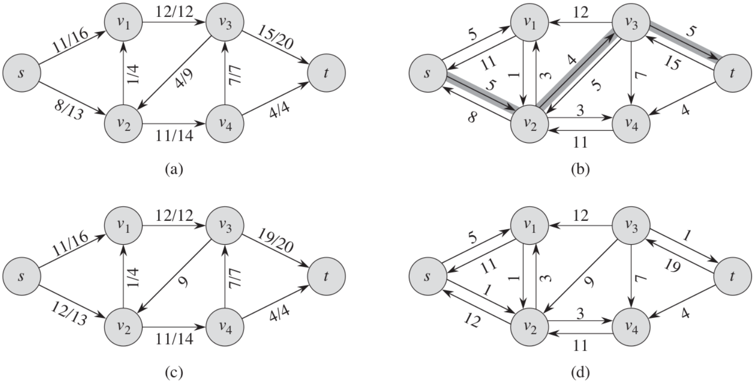
# Augmenting the Flow




- Augmenting Path $P = s \rightsquigarrow t$ in residual $G_f$ with $b = Bottleneck(P, f)$ being the smallest capacity on $P$.
- Augmented Flow $f'$ is $f$ modified as follows:
    - for each forward edge $e$ on $P$, increase $f'(e) = f(e) + b$
    - for each backward edge $\overline{e}$ on $P$, decrease $f'(e) = f(e) - b$.
- Claim: $f'$ is a valid flow in $G, c$.
- Augmentation $f'$ from $f$ can be computed in time $O(E)$.

```
Max-Flow
  Initially f(e) = 0 for all e in G
  While there is an s-t path in the residual graph G_f
    Let P be a simple s-t path in G_f
    f' = augment(f, P)
    Update f to be f'
    Update the residual graph G_f to be G_f'
  Endwhile
  Return f
```

# Example: Ford Fulkerson



# Max-flow Min-Cut Theorem

Given flow network $G, c$ and a valid flow $f$,

- partition $A, B$ of $V$ is an $s, t$-cut if $s \in A$ and $t \in B$.

# Max-flow Min-Cut Theorem

Given flow network $G, c$ and a valid flow $f$,

- partition $A, B$ of $V$ is an $s, t$-cut if $s \in A$ and $t \in B$.
- Let capacity $c(A, B) = \Sigma_{e \text{ out of } A}\ c(e)$.
  Clearly, flow value $f^{out}(s) = f^{out} A - f^{in}(A) \leq c(A, B)$.

# Max-flow Min-Cut Theorem

Given flow network $G, c$ and a valid flow $f$,

- partition $A, B$ of $V$ is an $s, t$-cut if $s \in A$ and $t \in B$.
- Let capacity $c(A, B) = \Sigma_{e \text{ out of } A}\ c(e)$.
  Clearly, flow value $f^{out}(s) = f^{out} A - f^{in}(A) \leq c(A, B)$.

**Theorem**

*If $f*$ is the flow such that there is no $s \rightsquigarrow t$ path in $G_f$ (i.e. $f*$ is returned by Ford-Fulkerson algorithm), then we can construct an $s, t$ cut $A*, B*$ such that $f* = c(A*, B*)$. Hence $f*$ is max flow and $A*, B*$ is min cut.*

Given flow network $G, c$ and a valid flow $f$,

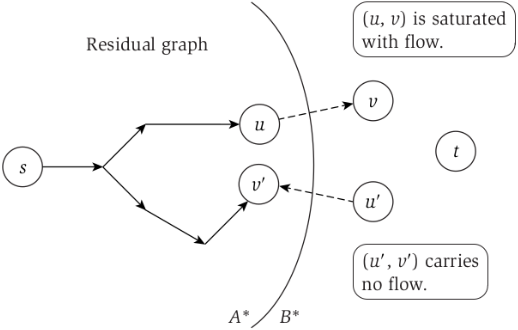
- partition $A, B$ of $V$ is an $s, t$-cut if $s \in A$ and $t \in B$.
- Let capacity $c(A, B) = \Sigma_{e \text{ out of } A}\ c(e)$.
  Clearly, flow value $f^{out}(s) = f^{out}A - f^{in}(A) \leq c(A, B)$.

**Theorem**

*If $f*$ is the flow such that there is no $s \rightsquigarrow t$ path in $G_f$ (i.e. $f*$ is returned by Ford-Fulkerson algorithm), then we can construct an $s, t$ cut $A*, B*$ such that $f* = c(A*, B*)$. Hence $f*$ is max flow and $A*, B*$ is min cut.*

Construction: Let $A*$ be all nodes $v$ s.t. $s \rightsquigarrow v$ in $G_f$. Let $B* = V - A*$.

# Proof Idea



Residual graph
$(u, v)$ is saturated with flow.
$v$
$u$
$s$
$t$
$v'$
$u'$
$(u', v')$ carries no flow.
$A^*$
$B^*$

# Integrality of Maximal Flow

If flow network $G, c$ is such that $c(e)$ is non-negative integer for each $e$, then maximum flow $f*$ produced by Fork Fulkerson algorithm assigns integer flow value to each edge.

# Integrality of Maximal Flow

If flow network $G, c$ is such that $c(e)$ is non-negative integer for each $e$, then maximum flow $f*$ produced by Fork Fulkerson algorithm assigns integer flow value to each edge.

**Proof Idea**

At each iteration, the Ford-Fulkerson algorithm augments the flow with only integral value. Hence, flow value in each edge at each iteration is invariantly integral.

# Maximal Matching in Bipartate Graph

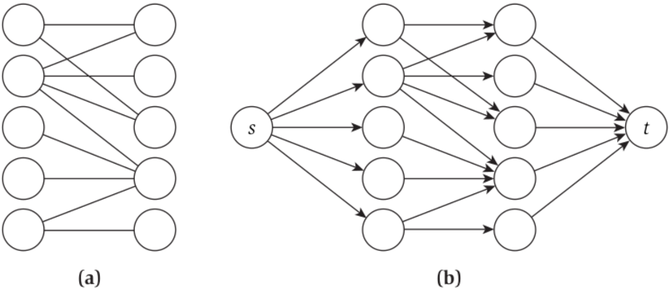



**Figure 7.9** (a) A bipartite graph. (b) The corresponding flow network, with all capacities equal to 1.

# Maximal Matching in Bipartate Graph

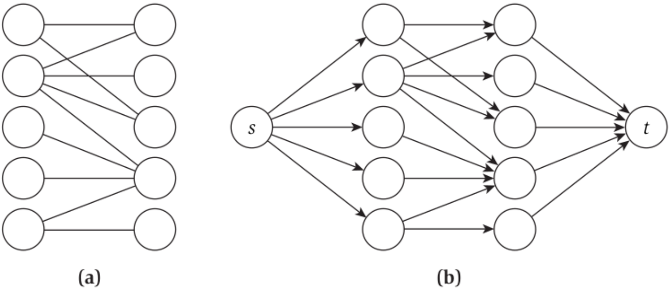



**Figure 7.9** (a) A bipartite graph. (b) The corresponding flow network, with all capacities equal to 1.

- Bipartate graph $(X; Y, E)$. Figure (a).

# Maximal Matching in Bipartate Graph

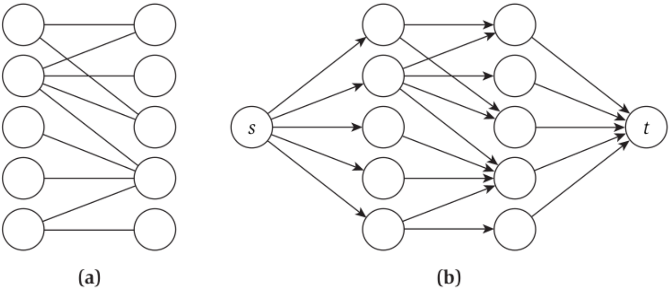



**Figure 7.9** (a) A bipartite graph. (b) The corresponding flow network, with all capacities equal to 1.

- Bipartate graph $(X; Y, E)$. Figure (a).
- Matching $M \subseteq E$ s.t. every $v \in X \cup Y$ occurs at most once in $M$.

# Maximal Matching in Bipartate Graph

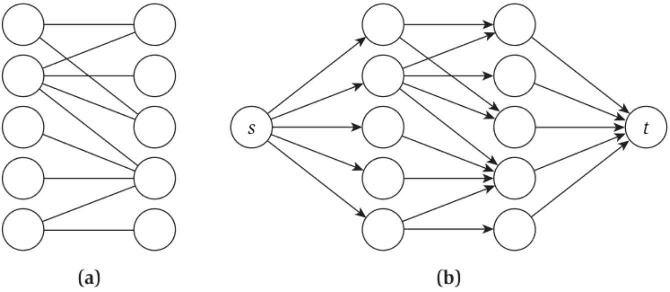



**Figure 7.9** (a) A bipartite graph. (b) The corresponding flow network, with all capacities equal to 1.

- Bipartate graph $(X; Y, E)$. Figure (a).
- Matching $M \subseteq E$ s.t. every $v \in X \cup Y$ occurs at most once in $M$.
- Maximal Matching.

# Maximal Matching in Bipartate Graph

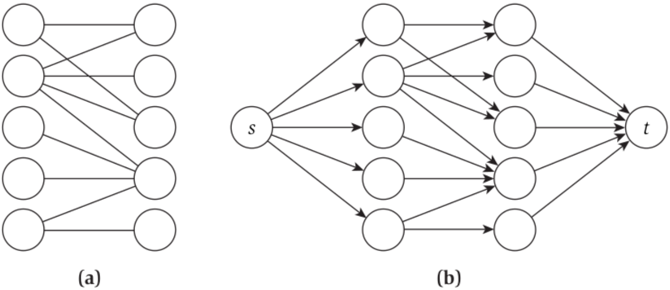



**Figure 7.9** (a) A bipartite graph. (b) The corresponding flow network, with all capacities equal to 1.

- Bipartate graph $(X; Y, E)$. Figure (a).
- Matching $M \subseteq E$ s.t. every $v \in X \cup Y$ occurs at most once in $M$.
- Maximal Matching.
- Perfect Matching: Every $v \in X \cup Y$ occurs exactly once in $M$

# Illustration

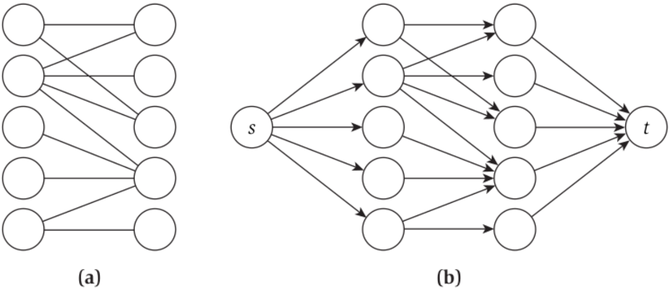



**Figure 7.9** (a) A bipartite graph. (b) The corresponding flow network, with all capacities equal to 1.

# Reduction to Flow Network

Given bipartate graph $(X; Y, E)$, we can construct a flow graph $(G', c)$ with every edge $e$ having capacity $c(e) = 1$. See figure.

Given bipartate graph $(X; Y, E)$, we can construct a flow graph $(G', c)$ with every edge $e$ having capacity $c(e) = 1$. See figure.

**Theorem**

*Let $(X; Y, E)$ be a bipartate graph and $G', c$ be the related flow network.*

Given bipartate graph $(X; Y, E)$, we can construct a flow graph $(G', c)$ with every edge $e$ having capacity $c(e) = 1$. See figure.

**Theorem**

*Let $(X; Y, E)$ be a bipartate graph and $G', c$ be the related flow network.*

- *If $M$ is a matching, then correspoding flow $f_M$ obtained by assigning $f_M(e) = 1$ if $e \in M$ and $f_M(e) = 0$ otherwise is a valid integral flow of $G', c$.*

# Reduction to Flow Network

Given bipartite graph $(X; Y, E)$, we can construct a flow graph $(G', c)$ with every edge $e$ having capacity $c(e) = 1$. See figure.

**Theorem**

*Let $(X; Y, E)$ be a bipartite graph and $G', c$ be the related flow network.*

- *If $M$ is a matching, then correspoding flow $f_M$ obtained by assigning $f_M(e) = 1$ if $e \in M$ and $f_M(e) = 0$ otherwise is a valid integral flow of $G', c$.*
- *If $f$ is a valid integral flow then corresponding subset of edges $M_f$ between $X, Y$ having flow value 1 forms a matching with $|M_f| = v(f)$.*

Given bipartate graph $(X; Y, E)$, we can construct a flow graph $(G', c)$ with every edge $e$ having capacity $c(e) = 1$. See figure.

**Theorem**

*Let $(X; Y, E)$ be a bipartate graph and $G', c$ be the related flow network.*

- *If $M$ is a matching, then correspoding flow $f_M$ obtained by assigning $f_M(e) = 1$ if $e \in M$ and $f_M(e) = 0$ otherwise is a valid integral flow of $G', c$.*
- *If $f$ is a valid integral flow then corresponding subset of edges $M_f$ between $X, Y$ having flow value 1 forms a matching with $|M_f| = v(f)$.*

**Corollary**

*If $f*$ is a maximal integral flow then $M_{f*}$ is a maximal matching.*

# Hall's Theorem

A bipartite graph $G = (X; Y, E)$ with $|X| = |Y| = n$

- has a perfect matching

  if and only iff

- for all $A \subseteq X$ we have $|A| \leq |E(A)|$.

A bipartite graph $G = (X; Y, E)$ with $|X| = |Y| = n$

- has a perfect matching

  if and only iff

- for all $A \subseteq X$ we have $|A| \leq |E(A)|$.

**Proof**

- $G$ has perfect matching iff corresponding flow graph $G', c$ hnas a maximal flow $f*$ of value $n$.

# Hall's Theorem

A bipartite graph $G = (X; Y, E)$ with $|X| = |Y| = n$

- has a perfect matching

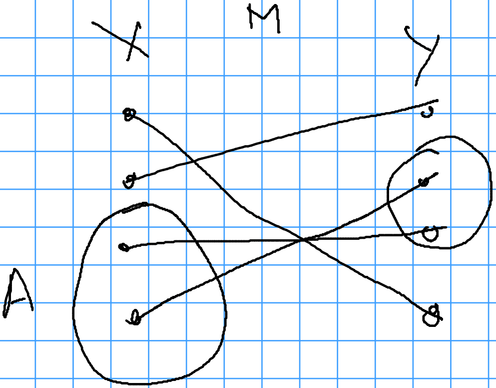
  if and only iff

- for all $A \subseteq X$ we have $|A| \leq |E(A)|$.

$C_1 \Uparrow C_2.$ $\neg C_1 \Downarrow \neg C_2$

## Proof

- $G$ has perfect matching iff corresponding flow graph $G', c$ hnas a maximal flow $f*$ of value $n$.
- If $G$ has a perfect matching $M$ then every node $x$ in $X$ is uniquely paired to a node $y$ in $Y$ via an $M$ edge. Hence, for any $A$, we have $|A| = |M(A)| \leq |E(A)|$. (Because, $M(A) \subseteq E(A)$). Thus, RHS.




$M(A)$

$$|A| = |M(A)| \le |E(A)|$$

$$M \subseteq E$$

# Hall's Theorem

A bipartite graph $G = (X; Y, E)$ with $|X| = |Y| = n$

- has a perfect matching

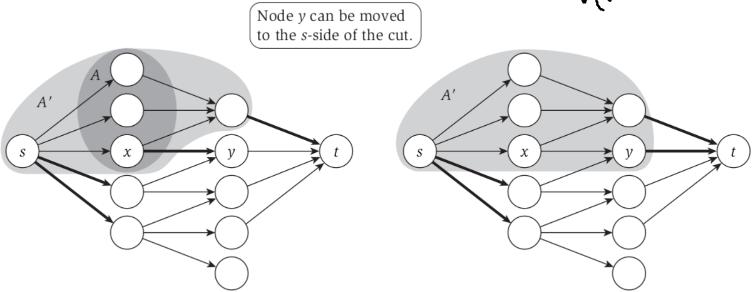
  if and only iff

- for all $A \subseteq X$ we have $|A| \leq |E(A)|$.

## Proof

- $G$ has perfect matching iff corresponding flow graph $G', c$ hnas a maximal flow $f*$ of value $n$.
- If $G$ has a perfect matching $M$ then every node $x$ in $X$ is uniquely paired to a node $y$ in $Y$ via an $M$ edge. Hence, for any $A$, we have $|A| = |M(A)| \leq |E(A)|$. (Because, $M(A) \subseteq E(A)$). Thus, RHS.
- Conversely, if $G$ does not have perfect matching then, $v(f*) < n$. We show that for some $A \subseteq X$ we have $|A| > |E(A)|$.

$c(A,B) \leq n$ $v(f^*) < n$




(a) (b)

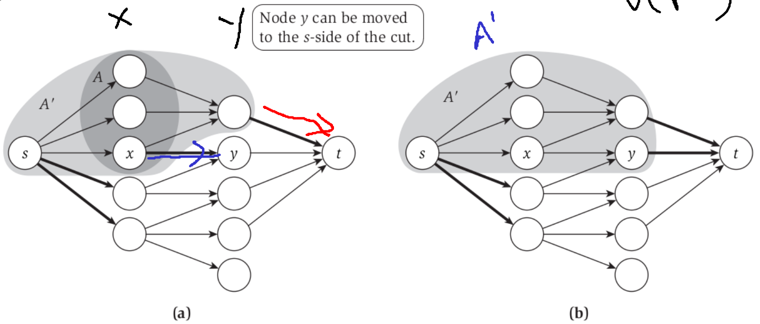
A,B

x

Y

A'

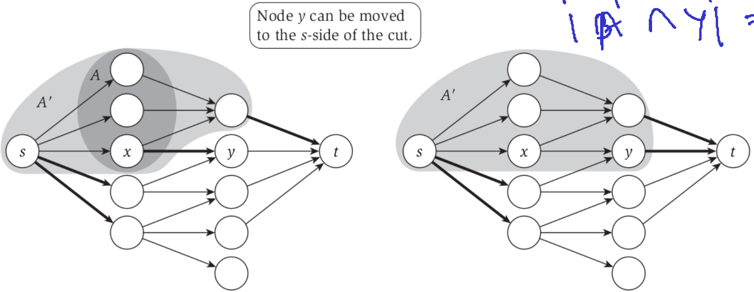
v(f*) < n




- There exists a min-cut $(A, B)$ of capacity less than $n$ with edges in $s- > X$, in $X- > Y$ and in $Y- > t$.

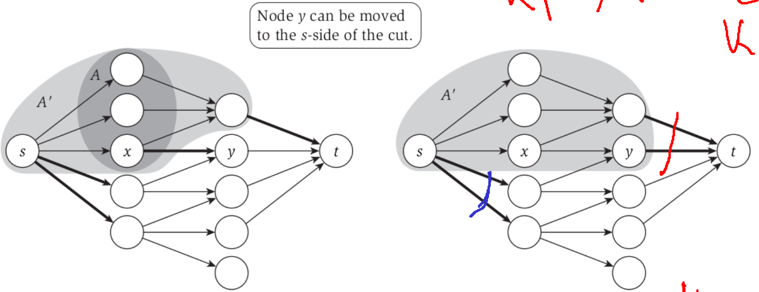
Let $|A' \cap X| = k_1$

$|A' \cap Y| = k_2$




(a) (b)

- There exists a min-cut $(A, B)$ of capacity less than $n$ with edges in $s->X$, in $X->Y$ and in $Y->t$.
- We can transform this to cut $(A', B')$ of same capacity with edges only in $s->X$ and $Y->t$. There are less than $n$ edges.




(a) (b)

$K_1 = X \cap A'$ & $K_2 = Y \cap A'$

$K_2 = E(K_1)$

- There exists a min-cut $(A, B)$ of capacity less than $n$ with edges in $s->X$, in $X->Y$ and in $Y->t$.
- We can transform this to cut $(A', B')$ of same capacity with edges only in $s->X$ and $Y->t$. There are less than $n$ edges.
- If $k_1 = |X \cap A'|$ and $k_2 = |Y\cancel{cap}\cancel{B}'|$ $\cap A'$, then $(n - k_1 + k_2) < n$ giving $k_2 < k_1$.