# Design and Analysis of Algorithms
# CS218M
## Network Flow Algorithms

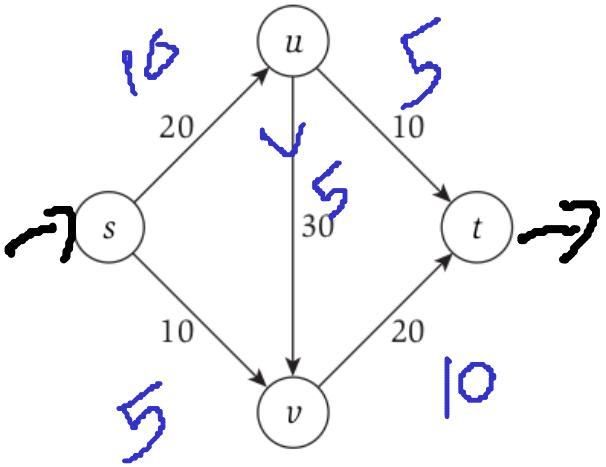
Paritosh Pandya

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 $f : E \to Z_0$ such that

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

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

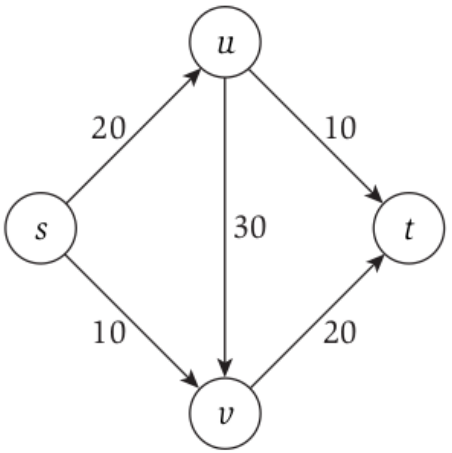
$$f^{in}(v) = f^{out}(v)$$

# 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

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

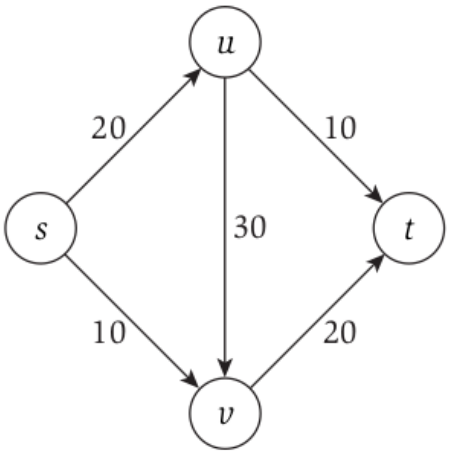


# Flow

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

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

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

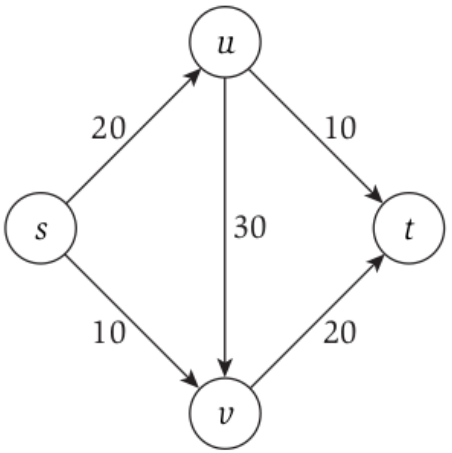


# Flow

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

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

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



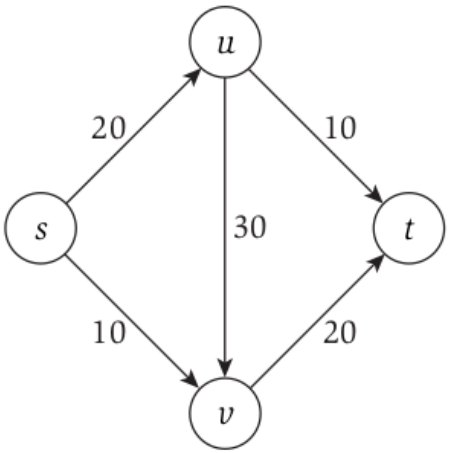

- Let $f^{out}(v) = \Sigma_{e \text{ out of } v} f(e)$ and $f^{in}(v) = \Sigma_{e \text{ in to } v} f(e)$.

# Flow

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

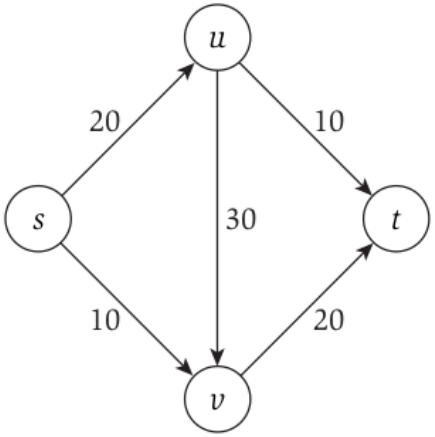
- (capacity) $0 \leq f(u, v) \leq c(u, v)$.
- (conservation) For all internal nodes $v$ we have

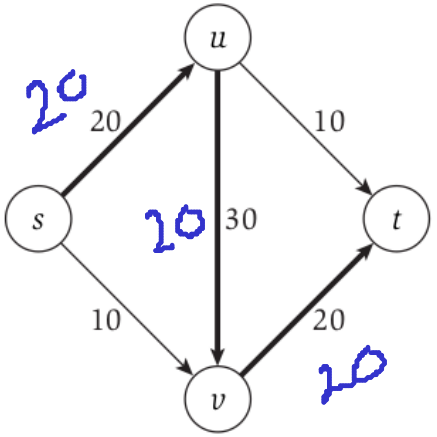
$$\Sigma_{e \text{ in to } v} f(e) \quad = \quad \Sigma_{e \text{ out of } v} f(e)$$

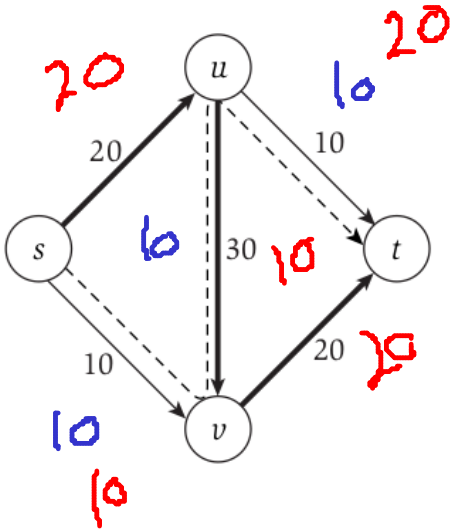



- Let $f^{out}(v) = \Sigma_{e \text{ out of } v} f(e)$ and $f^{in}(v) = \Sigma_{e \text{ in to } v} f(e)$.
- Define value of flow $f$ as $f^{out}(s)$.

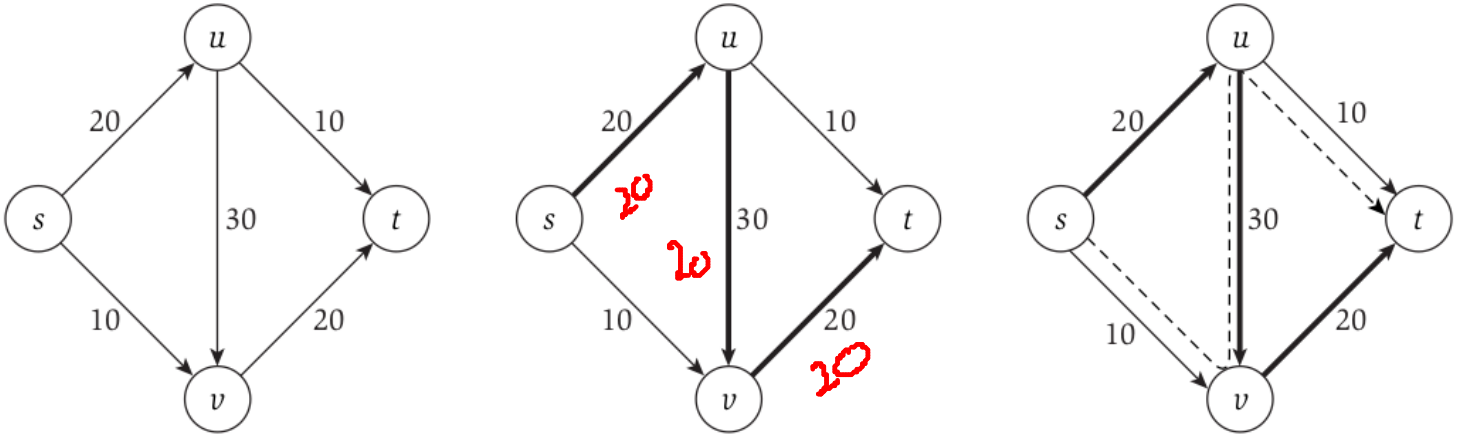
# Finding Maximum Flow



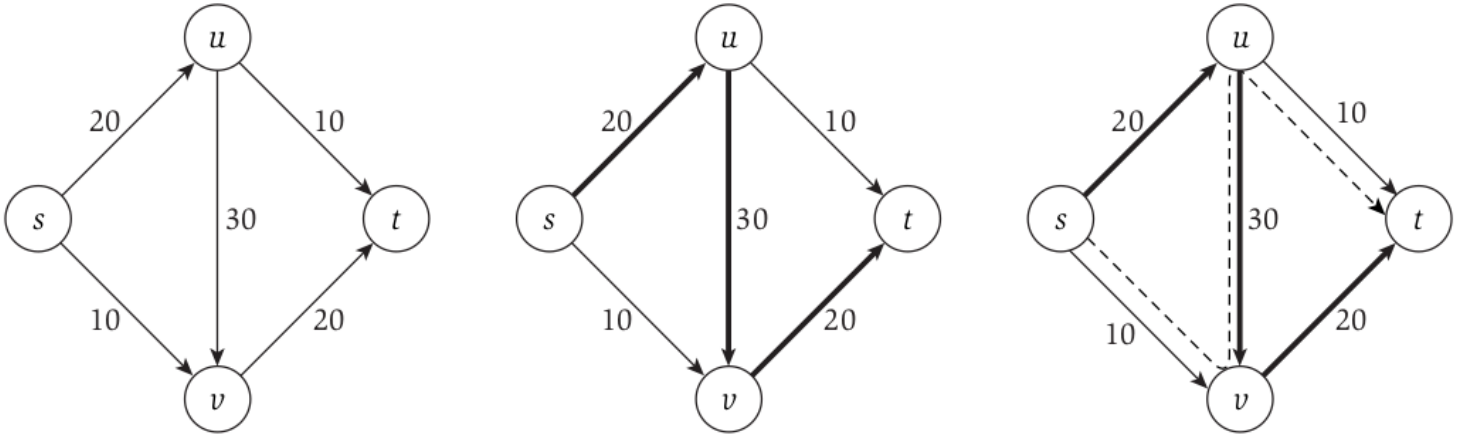



# Finding Maximum Flow




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

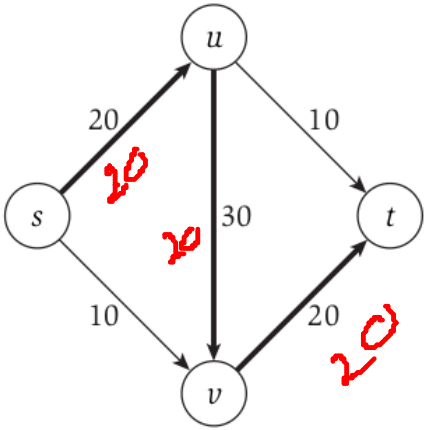
# Finding Maximum Flow

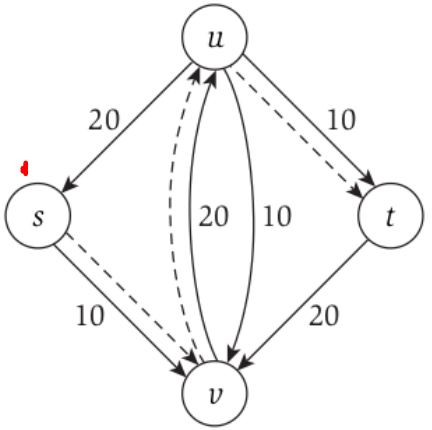


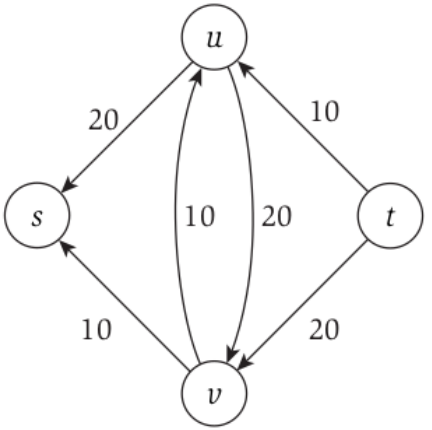

- Find a path $P = s \rightsquigarrow t$. Let *Bottleneck*$(P)$ be the smallest capacity on the path. Initial flow $f$ has value *Bottleneck*$(P)$.
- Given current flow $f$, find augmenting path $P = s \rightsquigarrow t$. Check that it is feasible and push *Bottleneck*$(P)$ additional flow to get revised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

# Residual Graph

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

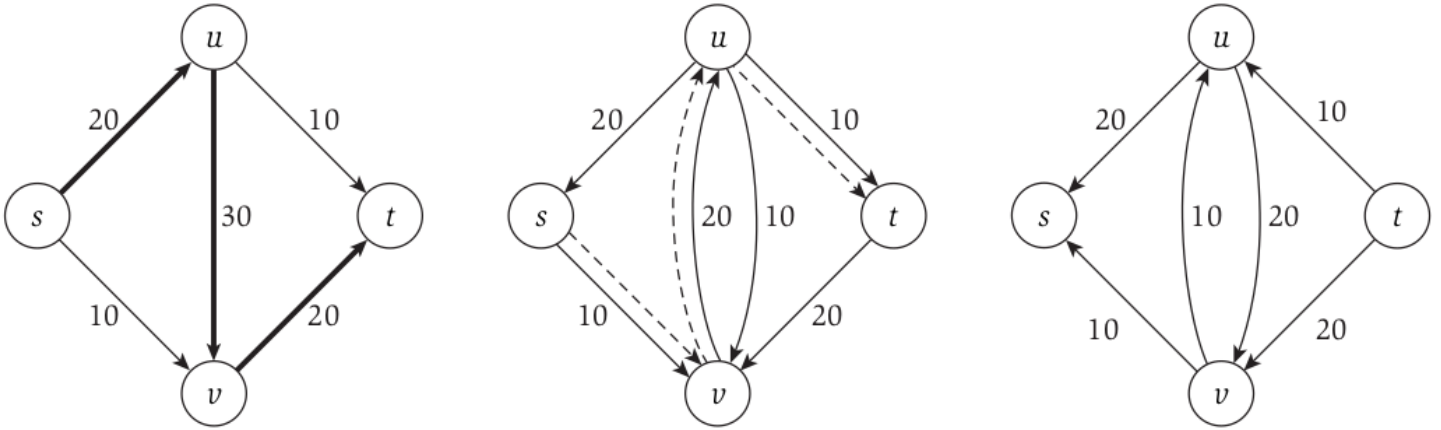


# Residual Graph

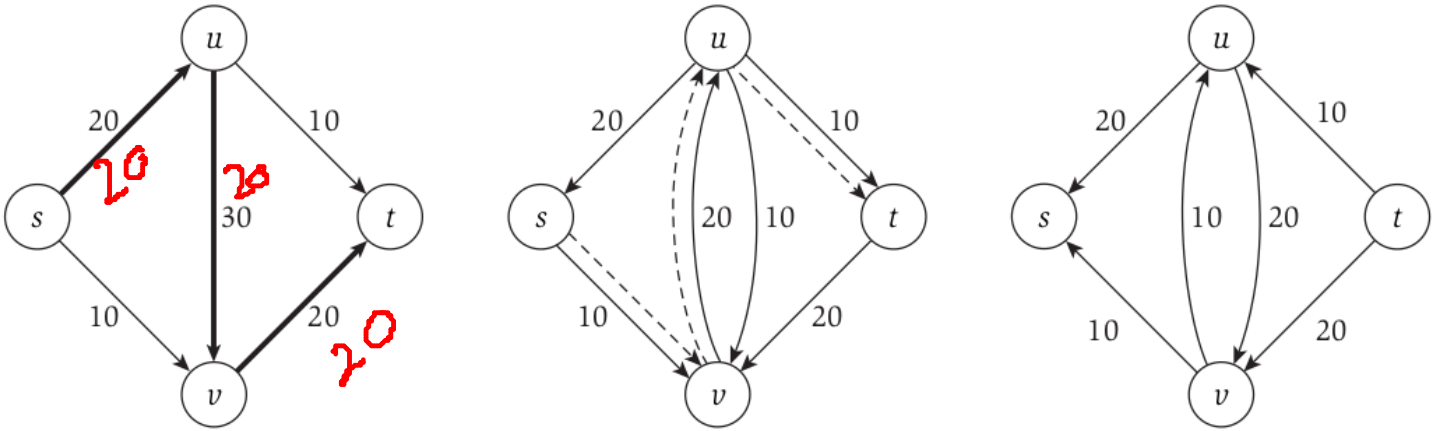
Given flow $f$ for flow network $G, c$, define Residual graph $G_f$.




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

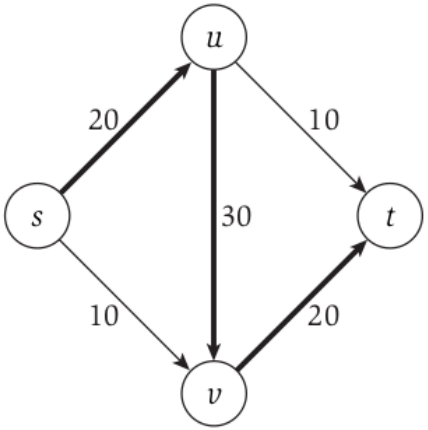
# Residual Graph

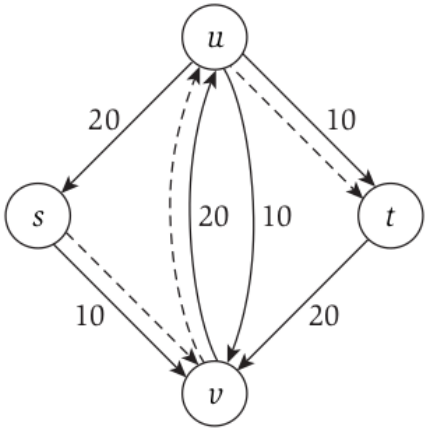
Given flow $f$ for flow network $G, c$, define Residual graph $G_f$.

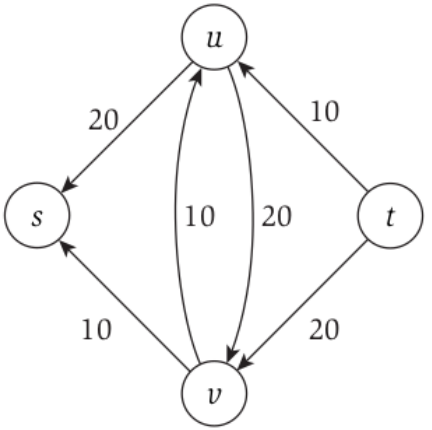



- Forward edges: Edges $e$ with residual capacity $c(e) - f(e) > 0$.
- Backward edges: Reverse $\overline{e}$ of edges $e$ with $f(e) > 0$ allowing reverse flow upto $f(e)$.

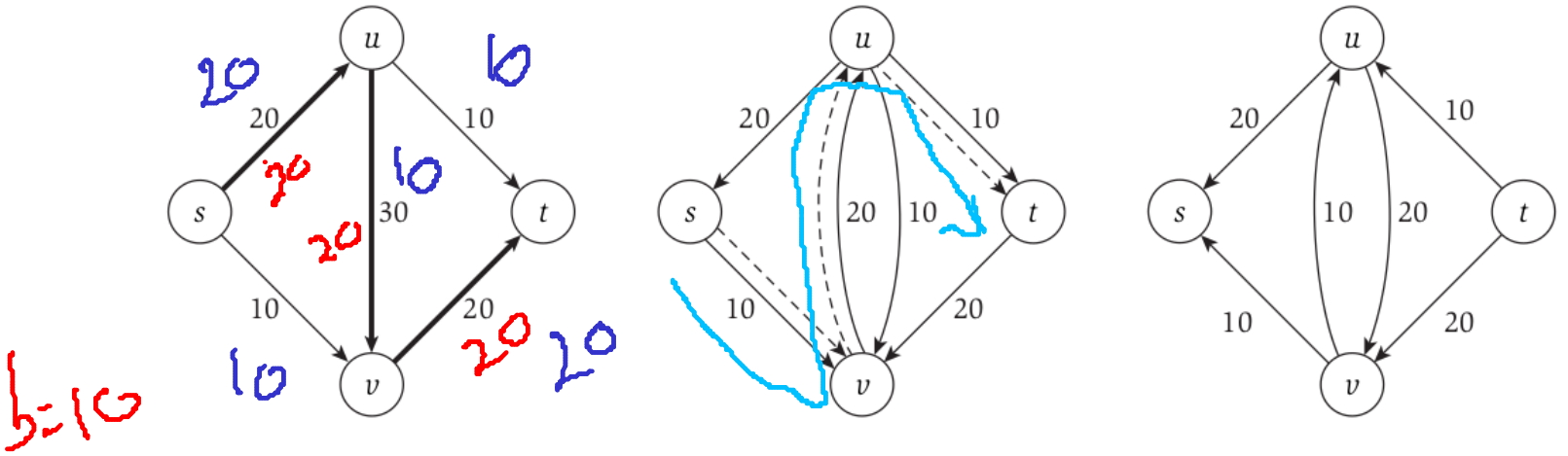
# Augmenting the Flow



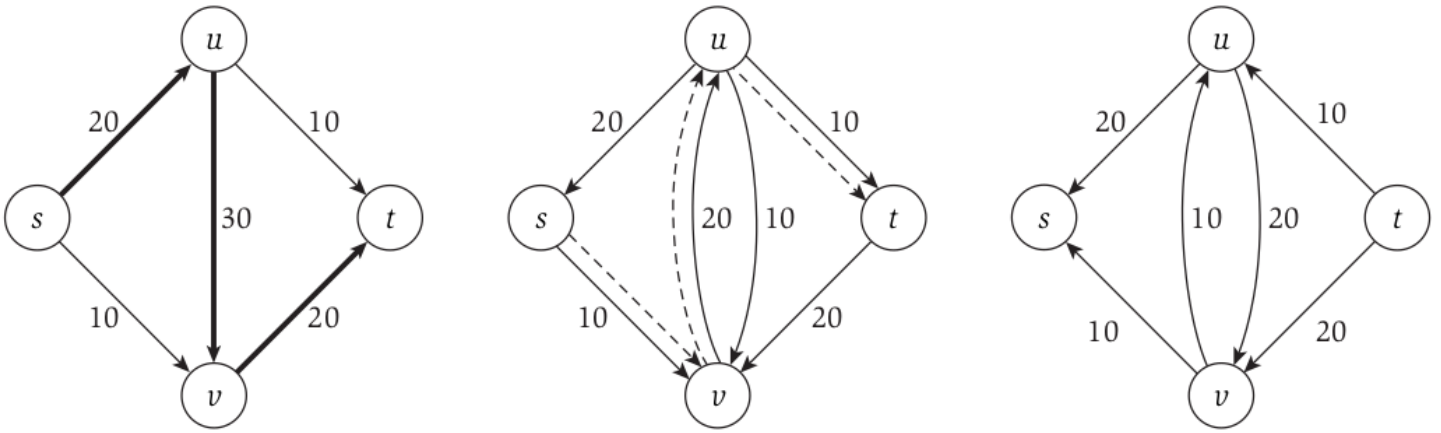



# Augmenting the Flow



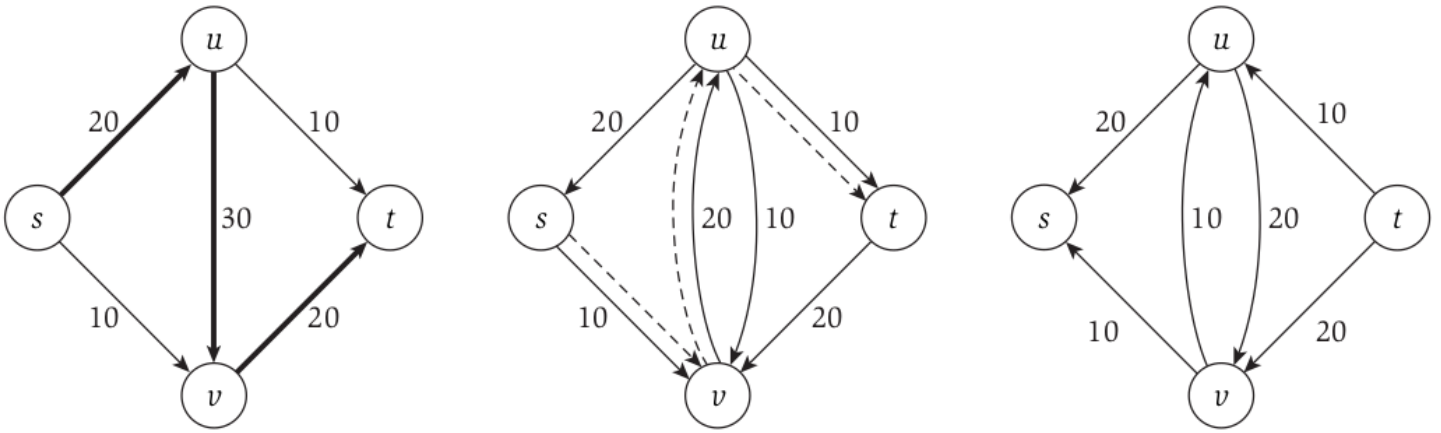

- Augmenting Path $P = s \rightsquigarrow t$ in residual $G_f$ with $b = Bottleneck(P, f)$ being the smallest capacity on $P$.

# Augmenting the Flow



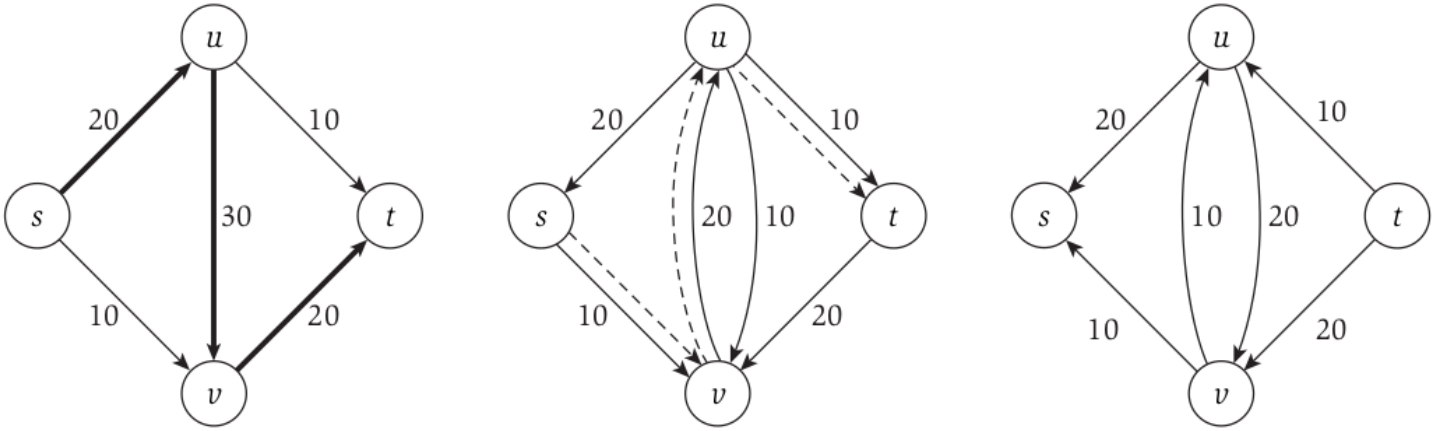

- Augmenting Path $P = s \rightsquigarrow t$ in residual $G_f$ with $b = Bottleneck(P, f)$ being the smallest capacity on $P$.
- Augmented Flow $f'$ is $f$ modified as follows:
  - for each forward edge $e$ on $P$, increase $f'(e) = f(e) + b$
  - for each backward edge $\overline{e}$ on $P$, decrease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

# Augmenting 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 $G, c$.

# 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 $O(E)$.

# Ford-Fulkerson Algorithm (1956) for Max-Flow

f.n. $(G, c)$

$f^*$ maximal

$f^*$

**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 $G_f$

$f' = \text{augment}(f, P)$

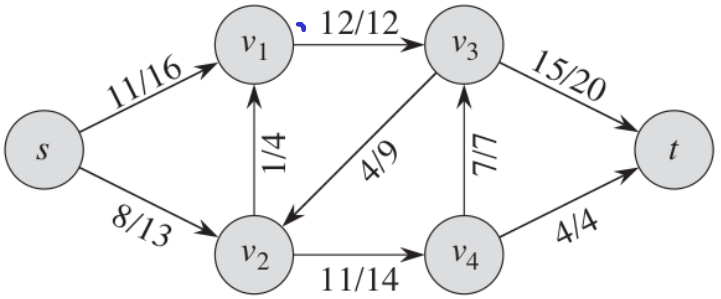
Update $f$ to be $f'$

Update the residual graph $G_f$ to be $G_{f'}$

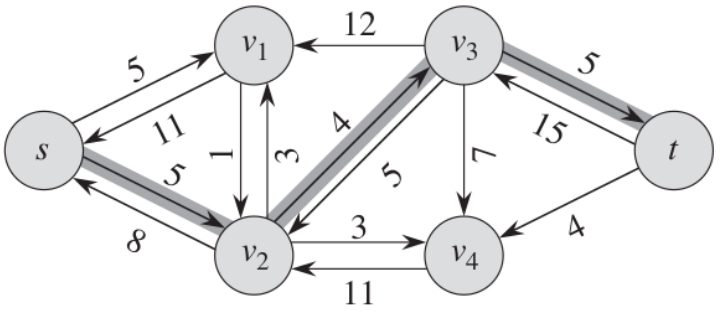
Endwhile

Return $f$

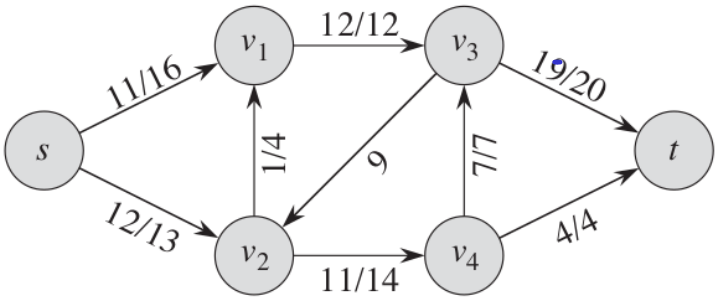
# Example: Ford Fulkerson



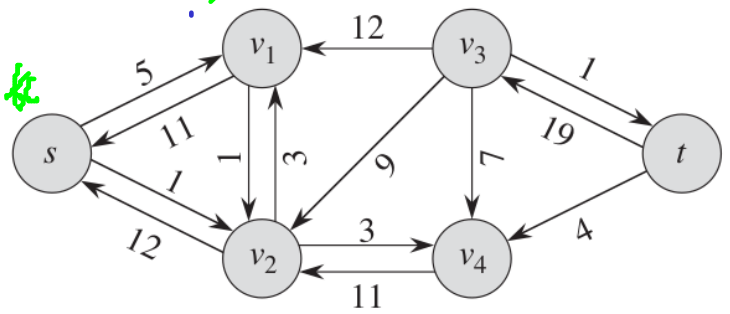

(a)




(b)




(c)




(d)

# 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.*

# Max-flow Min-Cut Theorem

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

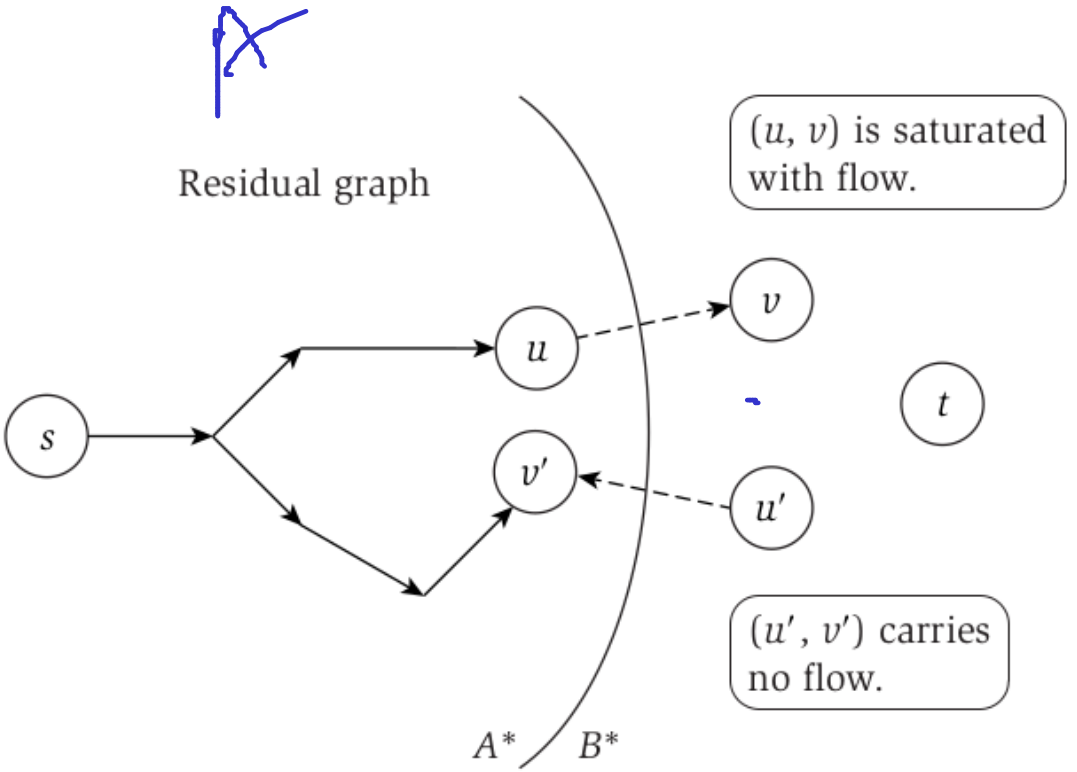
- partition $A, B$ of $V$ is an $s, t$-cut if $s \in A$ and $t \in B$.
- Let capacity $c(A, B) = \Sigma_{e \text{ out of } A}\, c(e)$.
  Clearly, flow value $f^{out}(s) = f^{out}A - f^{in}(A) \leq c(A, B)$.

**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.*

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.
$v$
$u$
$t$
$s$
$v'$
$u'$
$(u', v')$ carries no flow.
$A^*$
$B^*$

# Maximal Matching in Bipartate Graph

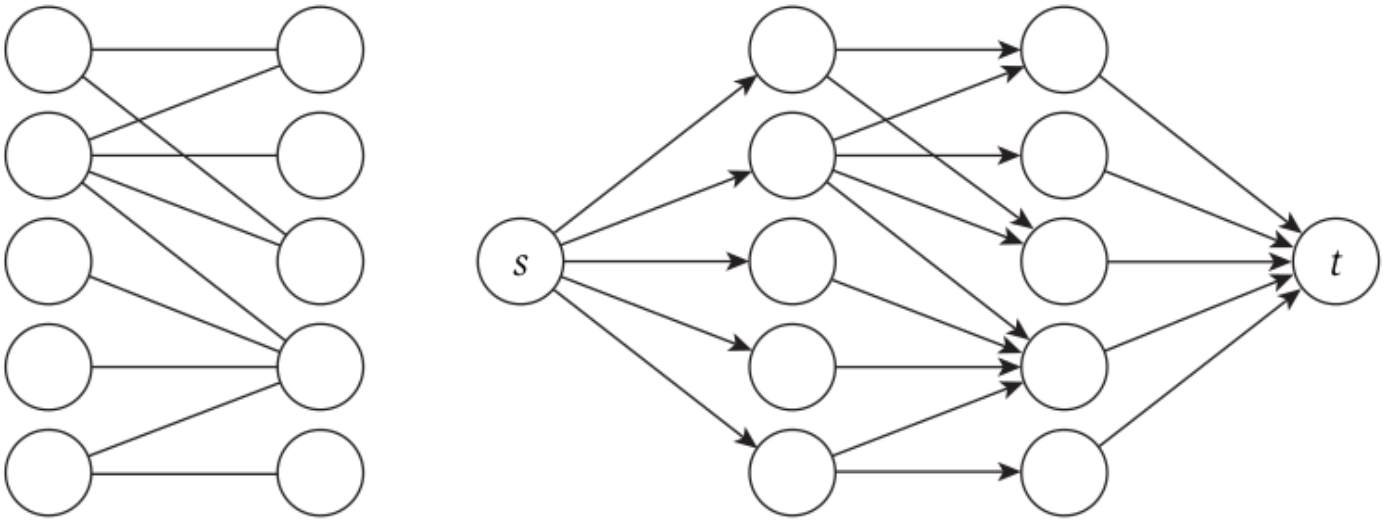



**Figure 7.9** (a) A bipartite graph. (b) The corresponding flow network, with all capacities equal to 1.

- Matching $M \subseteq E$ s.t. every $v \in X \cup Y$ occurs at most once in $M$.
- Maximal Matching. Perfect Matching.

# Hall's Theorem

A bipartate graph $(X; Y, E)$ with $|X| = |Y|$ has

- either a perfect matching
- or there exists $A \subseteq X$ s.t. $|A| > |E(A)|$.