29. Flow in Networks

Flow Network, Flow, Maximum Flow
Residual Capacity, Remainder Network, Augmenting path

Ford-Fulkerson Algorithm
Edmonds-Karp Algorithm

Cuts, Max-Flow Min-Cut Theorem

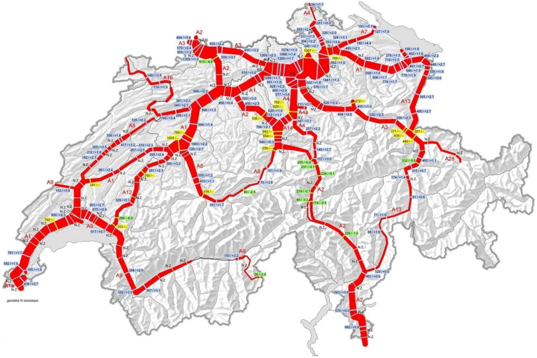
[Ottman/Widmayer, Kap. 9.7, 9.8.1], [Cormen et al, Kap. 26.1-26.3]

Slides redesigned by Manuela Fischer – thank you!

Maximum Traffic Flow

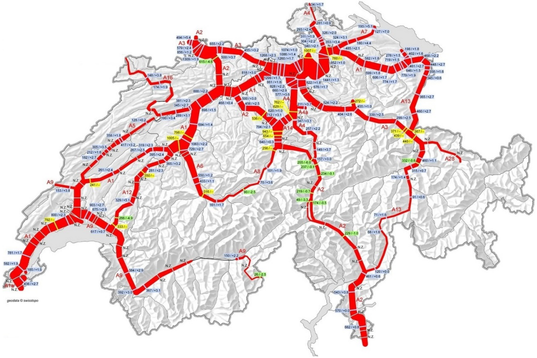
Maximum Traffic Flow

Given: Road Network with capacities



Maximum Traffic Flow

Given: Road Network with capacities

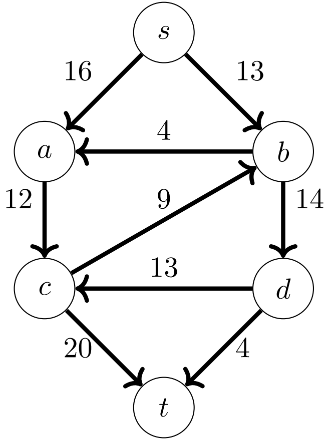


Wanted: Maximum traffic flow between Zurich and Geneva

Flow Network

Flow Network

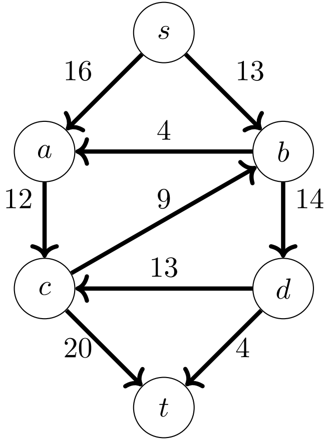
directed, weighted graph $G = (V, E, c)$ with capacities $c\colon E \to \mathbb{R}^{>0}$



Flow Network

directed, weighted graph $G = (V, E, c)$ with capacities $c\colon E \to \mathbb{R}^{>0}$

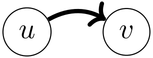
- without antiparallel edges:

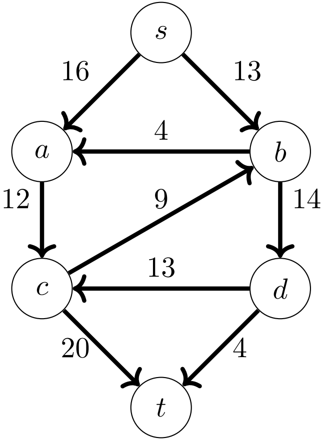


Flow Network

directed, weighted graph $G = (V, E, c)$ with capacities $c\colon E \to \mathbb{R}^{>0}$

- without antiparallel edges:
  $(u, v) \in E$

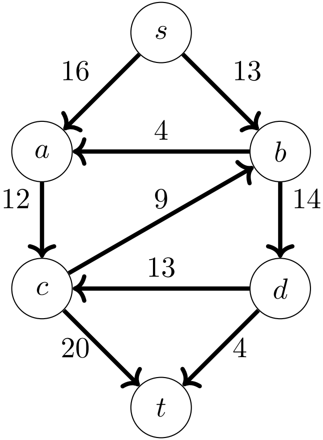




Flow Network

directed, weighted graph $G = (V, E, c)$ with capacities $c\colon E \to \mathbb{R}^{>0}$

- without antiparallel edges:
  $(u, v) \in E \;\Rightarrow\; (v, u) \notin E$

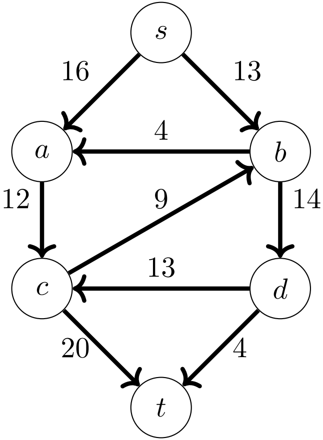


Flow Network

directed, weighted graph $G = (V, E, c)$ with capacities $c\colon E \to \mathbb{R}^{>0}$

- without antiparallel edges:
  $(u, v) \in E \;\Rightarrow\; (v, u) \notin E$

- source $s \in V$ without ingoing edges:

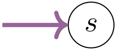


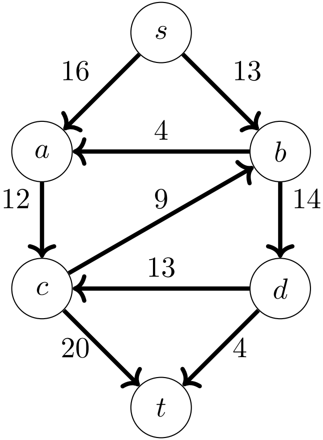
Flow Network

directed, weighted graph $G = (V, E, c)$ with capacities $c \colon E \to \mathbb{R}^{>0}$

- without antiparallel edges:
  $(u, v) \in E \;\Rightarrow\; (v, u) \notin E$

- source $s \in V$ without ingoing edges:
  $\forall v \in V : (v, s) \notin E$

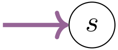




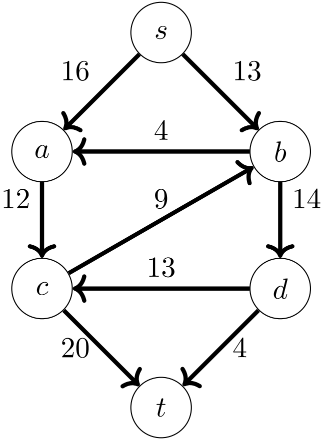
Flow Network

directed, weighted graph $G = (V, E, c)$ with capacities $c\colon E \to \mathbb{R}^{>0}$

- without antiparallel edges:
  $(u, v) \in E \;\Rightarrow\; (v, u) \notin E$
  

- source $s \in V$ without ingoing edges:
  $\forall v \in V : (v, s) \notin E$
  

- sink $t \in V$ without outgoing edges:

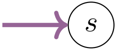


Flow Network

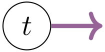
directed, weighted graph $G = (V, E, c)$ with capacities $c\colon E \to \mathbb{R}^{>0}$

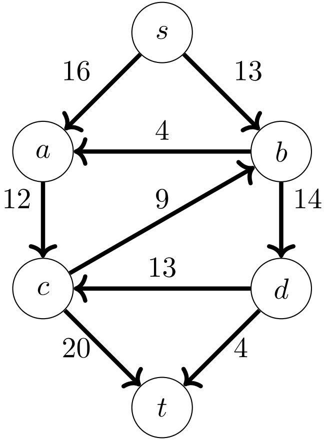
- without antiparallel edges:
  $(u, v) \in E \;\Rightarrow\; (v, u) \notin E$

- source $s \in V$ without ingoing edges:
  $\forall v \in V : (v, s) \notin E$



- sink $t \in V$ without outgoing edges:
  $\forall v \in V : (t, v) \notin E$





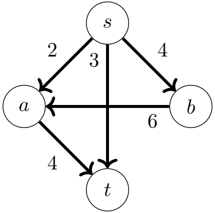
Quiz Flow Network

Quiz Flow Network

Which of the following graphs are flow networks?

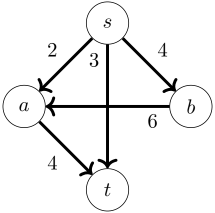
Quiz Flow Network

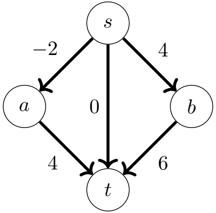
Which of the following graphs are flow networks?



Quiz Flow Network

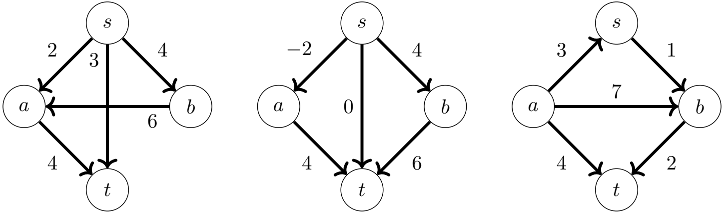
Which of the following graphs are flow networks?





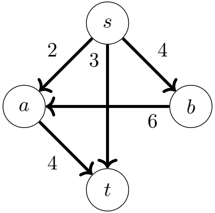
Quiz Flow Network

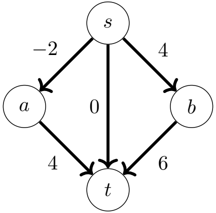
Which of the following graphs are flow networks?

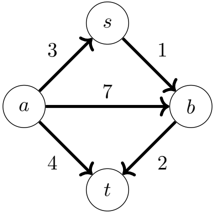


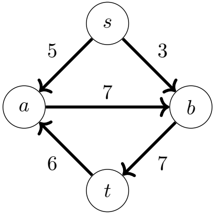
Quiz Flow Network

Which of the following graphs are flow networks?



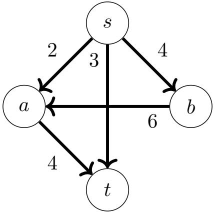


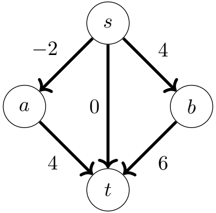


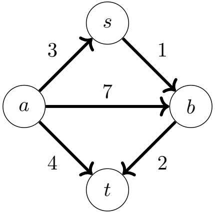


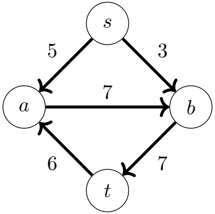
Quiz Flow Network

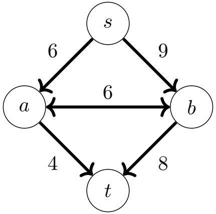
Which of the following graphs are flow networks?





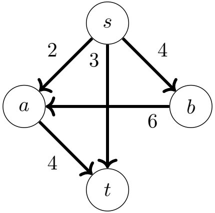


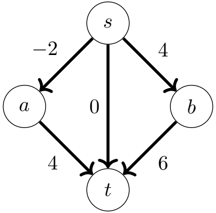


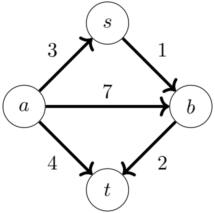


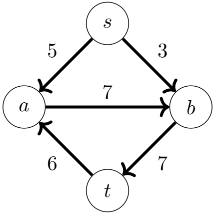
Quiz Flow Network

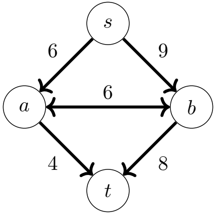
Which of the following graphs are flow networks?

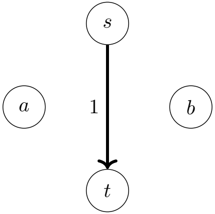






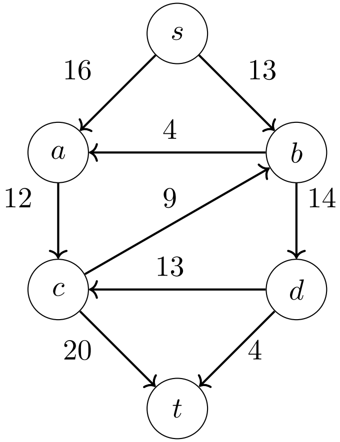






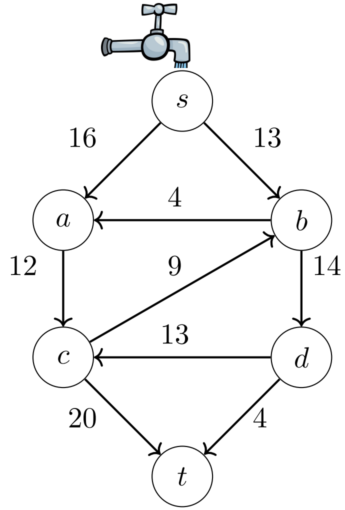
Flow in Flow Network

Flow in Flow Network



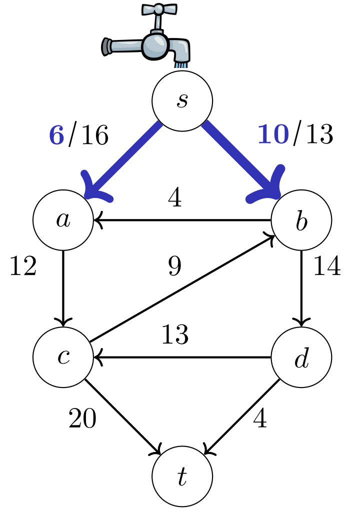
Flow in Flow Network

Flow is function $f : E \to \mathbb{R}^{\geq 0}$ such that



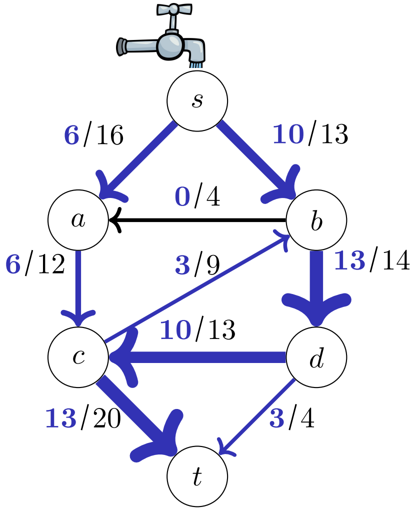
Flow in Flow Network

Flow is function $f : E \to \mathbb{R}^{\geq 0}$ such that



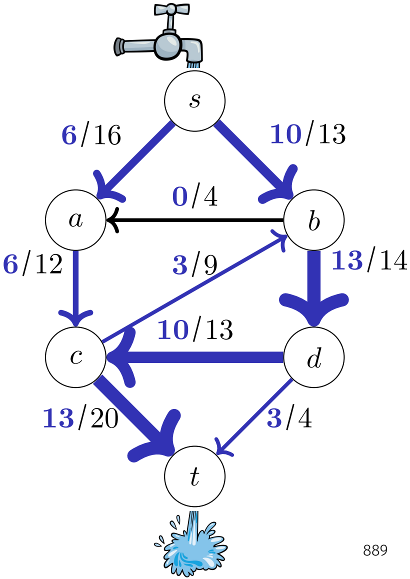
Flow in Flow Network

Flow is function $f: E \to \mathbb{R}^{\geq 0}$ such that



Flow in Flow Network

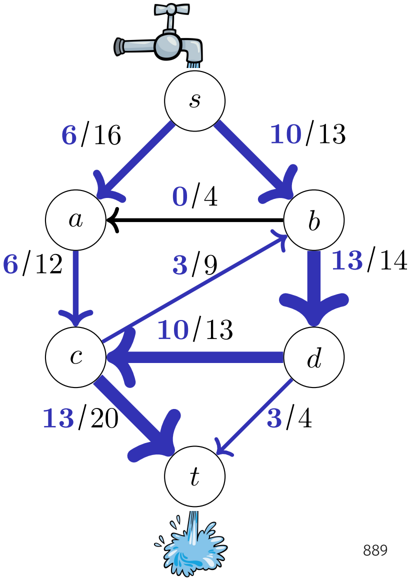
Flow is function $f: E \to \mathbb{R}^{\geq 0}$ such that



Flow in Flow Network

Flow is function $f \colon E \to \mathbb{R}^{\geq 0}$ such that

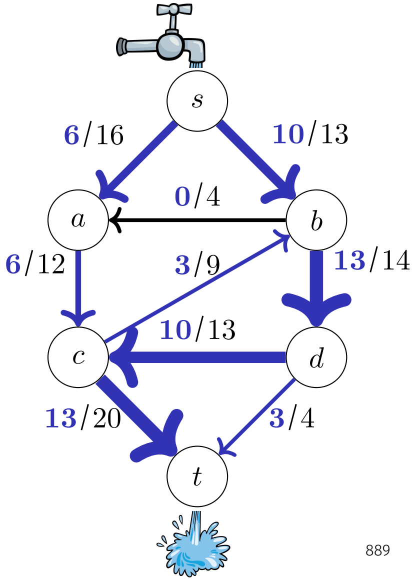
- **Bounded Capacity**: $\forall e \in E \colon f(e) \leq c(e)$



Flow in Flow Network

Flow is function $f\colon E \to \mathbb{R}^{\geq 0}$ such that

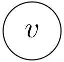
- **Bounded Capacity**: $\forall e \in E\colon\ f(e) \leq c(e)$
- **Conservation of flow**:

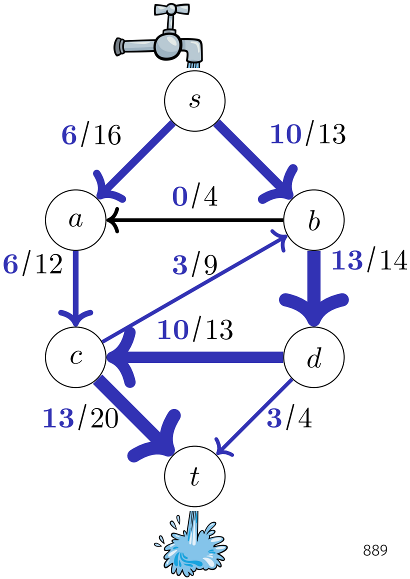


Flow in Flow Network

Flow is function $f\colon E \to \mathbb{R}^{\geq 0}$ such that

- **Bounded Capacity**: $\forall e \in E\colon\ f(e) \leq c(e)$
- **Conservation of flow**: $\forall v \in V \setminus \{s, t\}\colon$



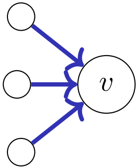


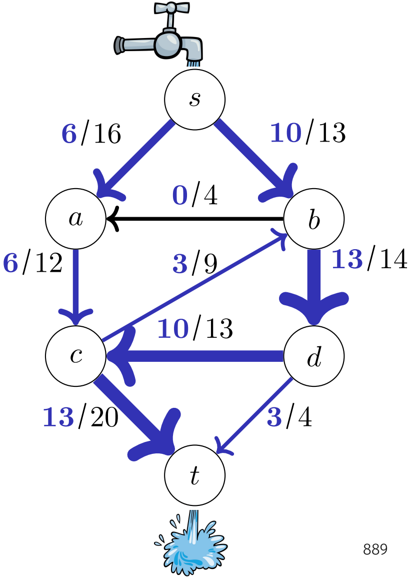
Flow in Flow Network

Flow is function $f\colon E \to \mathbb{R}^{\geq 0}$ such that

- **Bounded Capacity**: $\forall e \in E\colon\ f(e) \leq c(e)$
- **Conservation of flow**: $\forall v \in V \setminus \{s,t\}\colon$

$$\sum_{e \in E^-(v)} f(e)$$



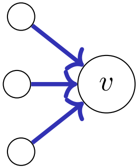


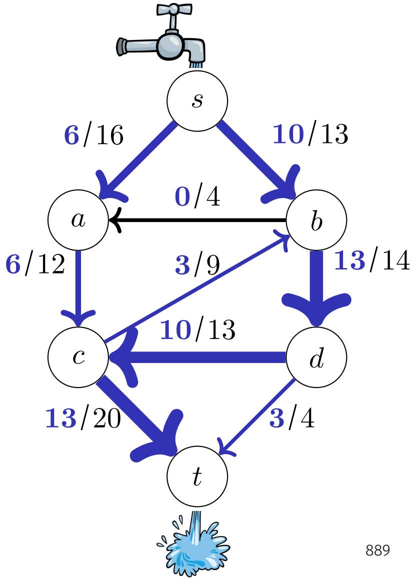
Flow in Flow Network

Flow is function $f\colon E \to \mathbb{R}^{\geq 0}$ such that

- **Bounded Capacity**: $\forall e \in E\colon\ f(e) \leq c(e)$
- **Conservation of flow**: $\forall v \in V \setminus \{s,t\}\colon$

$$\underbrace{\sum_{e \in E^-(v)} f(e)}_{=: f^-(v)}$$



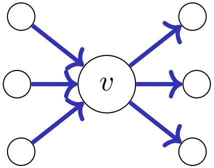


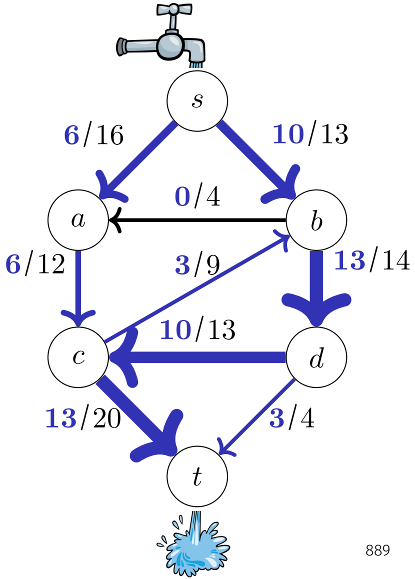
Flow in Flow Network

Flow is function $f\colon E \to \mathbb{R}^{\geq 0}$ such that

- **Bounded Capacity**: $\forall e \in E\colon\ f(e) \leq c(e)$
- **Conservation of flow**: $\forall v \in V \setminus \{s,t\}\colon$

$$\underbrace{\sum_{e\in E^-(v)} f(e)}_{=:f^-(v)} = \underbrace{\sum_{e\in E^+(v)} f(e)}_{=:f^+(v)}$$



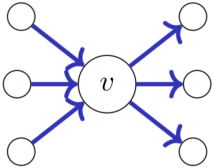


Flow in Flow Network

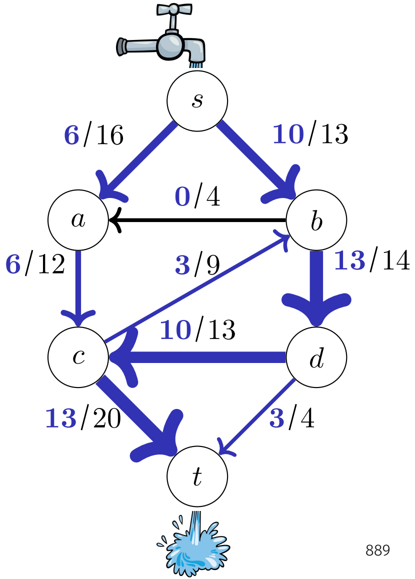
Flow is function $f\colon E \to \mathbb{R}^{\geq 0}$ such that

- **Bounded Capacity**: $\forall e \in E\colon\ f(e) \leq c(e)$
- **Conservation of flow**: $\forall v \in V \setminus \{s,t\}\colon$

$$\underbrace{\sum_{e \in E^-(v)} f(e)}_{=: f^-(v)} = \underbrace{\sum_{e \in E^+(v)} f(e)}_{=: f^+(v)}$$



Size of flow: $|f| := f^+(s) = f^-(t)$

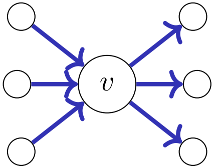


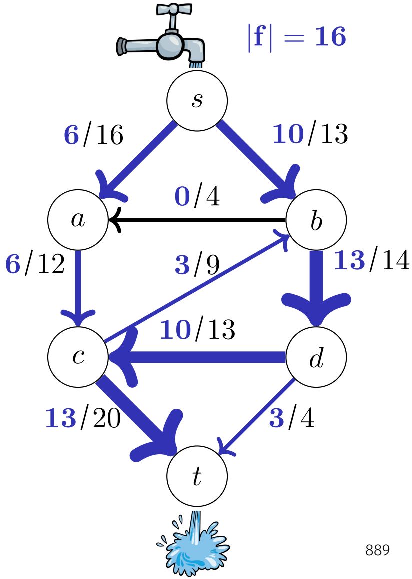
Flow in Flow Network

Flow is function $f\colon E \to \mathbb{R}^{\geq 0}$ such that

- **Bounded Capacity**: $\forall e \in E\colon\ f(e) \leq c(e)$
- **Conservation of flow**: $\forall v \in V \setminus \{s,t\}\colon$

$$\underbrace{\sum_{e \in E^-(v)} f(e)}_{=: f^-(v)} = \underbrace{\sum_{e \in E^+(v)} f(e)}_{=: f^+(v)}$$

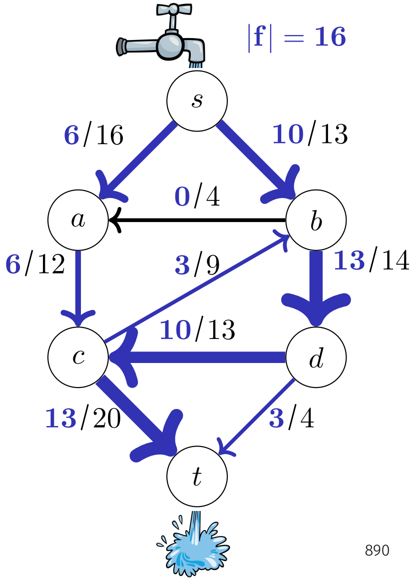




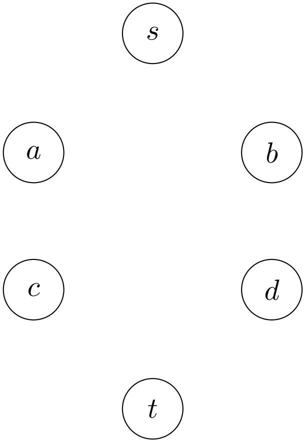
Size of flow: $|f| := f^+(s) = f^-(t)$

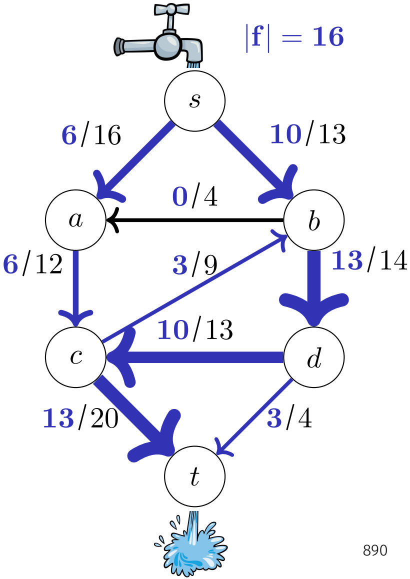
Intuition: Flow as set of paths $s \rightsquigarrow t$

Intuition: Flow as set of paths $s \rightsquigarrow t$

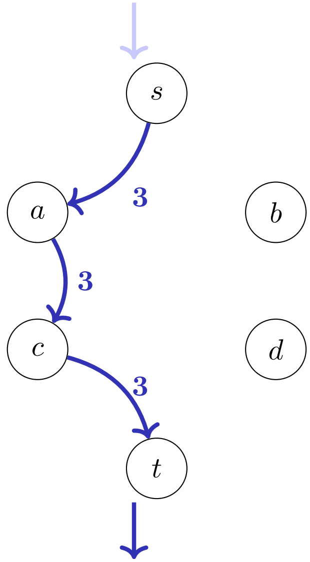


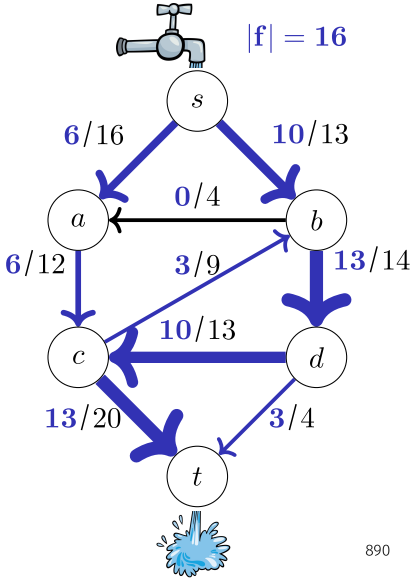
Intuition: Flow as set of paths $s \rightsquigarrow t$



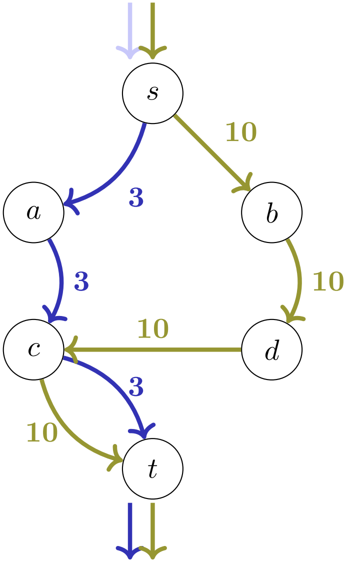


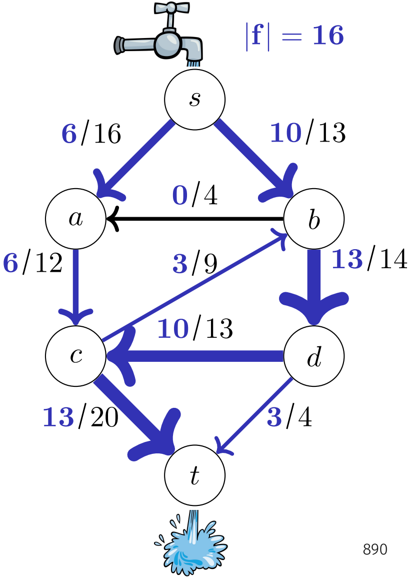
Intuition: Flow as set of paths $s \rightsquigarrow t$



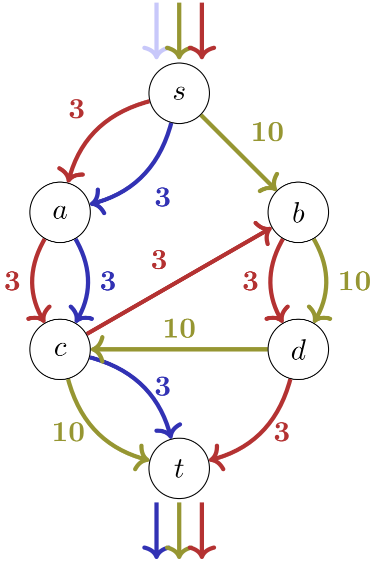


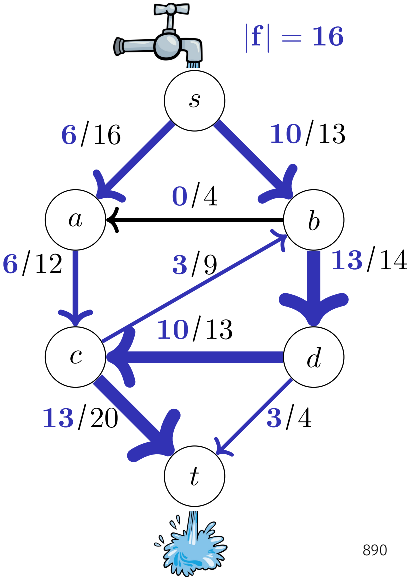
Intuition: Flow as set of paths $s \rightsquigarrow t$





Intuition: Flow as set of paths $s \rightsquigarrow t$





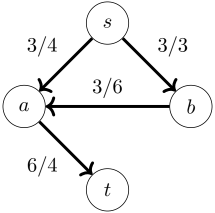
Quiz Flow

Quiz Flow

Which of the following are flows?

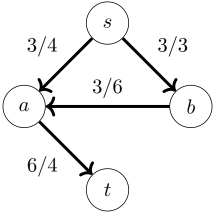
Quiz Flow

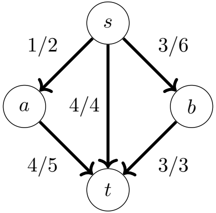
Which of the following are flows?



Quiz Flow

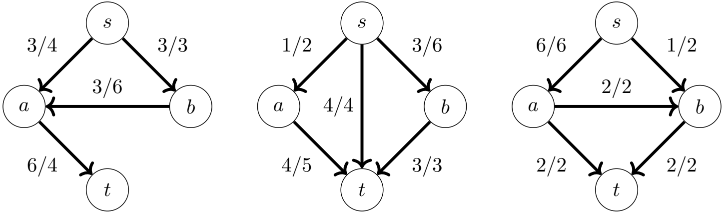
Which of the following are flows?





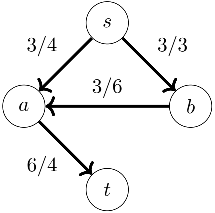
Quiz Flow

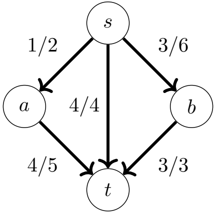
Which of the following are flows?

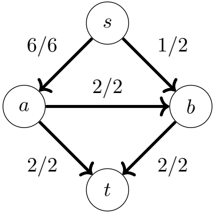


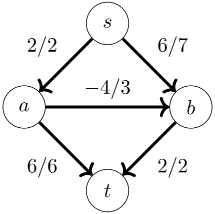
Quiz Flow

Which of the following are flows?



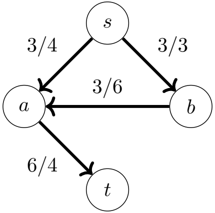


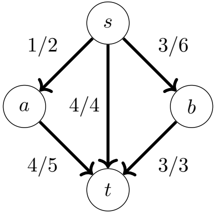


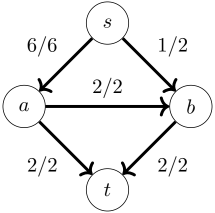


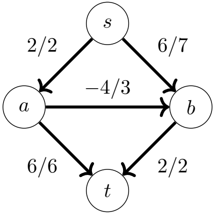
Quiz Flow

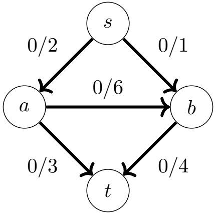
Which of the following are flows?





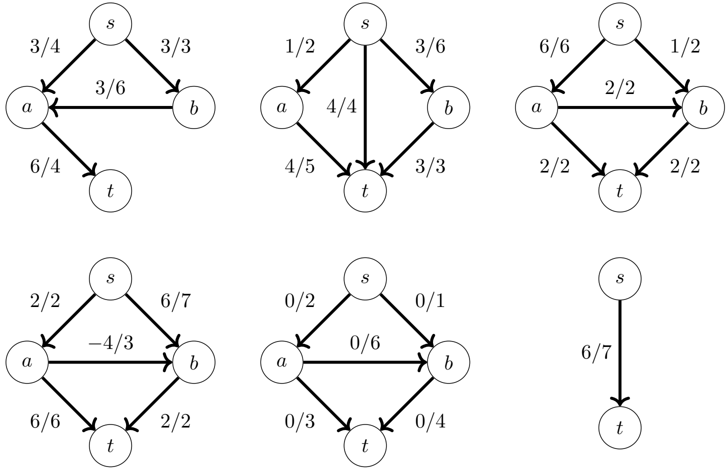






Quiz Flow

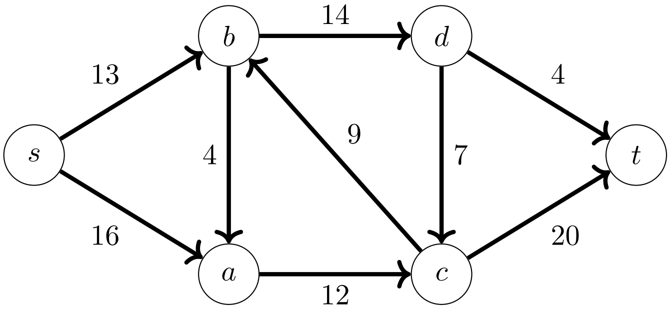
Which of the following are flows?



Maximal Flow

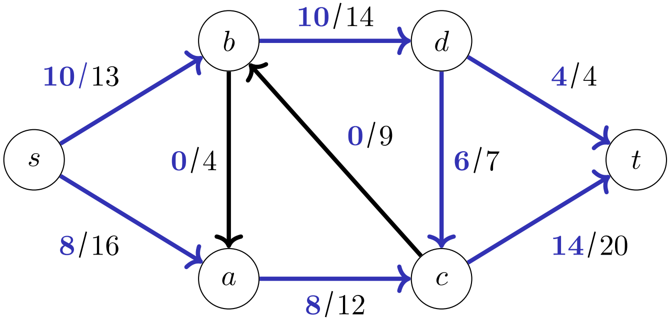
Maximal Flow

Given: Flow network: $G = (V, E, c)$, directed, positively weighted, without antiparallel edges, with source s and sink t



Maximal Flow

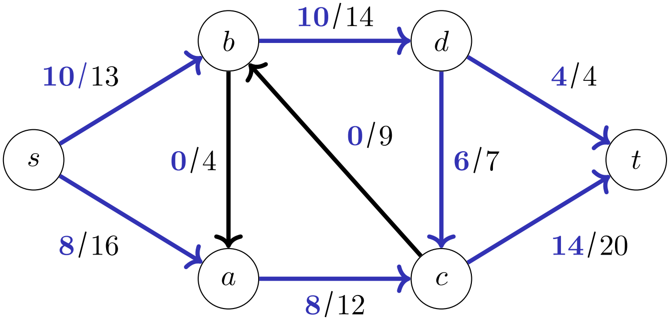
Given: Flow network: $G = (V, E, c)$, directed, positively weighted, without antiparallel edges, with source s and sink t



Wanted: Size $|f_{\max}|$ of the maximum flow in G

Maximal Flow

Given: Flow network: $G = (V, E, c)$, directed, positively weighted, without antiparallel edges, with source s and sink t



$\mathbf{18} = |\mathbf{f}| \leq |\mathbf{f}_{\max}| = \mathbf{23}$

Wanted: Size $|f_{\max}|$ of the maximum flow in G

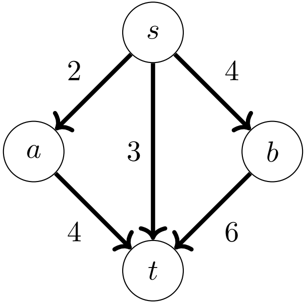
Quiz Maximum Flow

Quiz Maximum Flow

What is the maximum flow in the following flow network?

Quiz Maximum Flow

What is the maximum flow in the following flow network?



Greedy Algorithm?

Greedy Algorithm?

Residual capacity of an edge e**:** $r(e) := c(e) - f(e)$

Residual capacity of a path P**:** $\min_{e \in P} r(e)$

Greedy Algorithm?

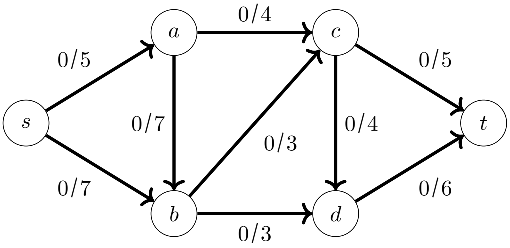
Residual capacity of an edge e: $r(e) := c(e) - f(e)$
Residual capacity of a path P: $\min_{e \in P} r(e)$

Greedy: Starting with $f(e) = 0$ for all $e \in E$, as long as there exists a path $s \rightsquigarrow t$ with remaining capacity $d > 0$, increase flow along this path by d.

Greedy Algorithm?

Residual capacity of an edge e**:** $r(e) := c(e) - f(e)$
Residual capacity of a path P**:** $\min_{e \in P} r(e)$

Greedy: Starting with $f(e) = 0$ for all $e \in E$, as long as there exists a path $s \rightsquigarrow t$ with remaining capacity $d > 0$, increase flow along this path by d.

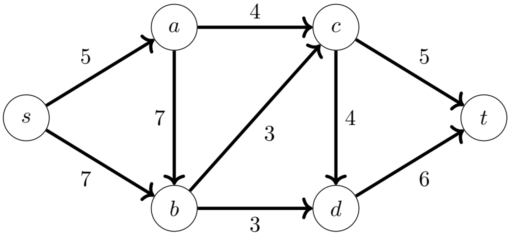


$|f| = 0$

Greedy Algorithm?

Residual capacity of an edge e: $r(e) := c(e) - f(e)$
Residual capacity of a path P: $\min_{e \in P} r(e)$

Greedy: Starting with $f(e) = 0$ for all $e \in E$, as long as there exists a path $s \rightsquigarrow t$ with remaining capacity $d > 0$, increase flow along this path by d.



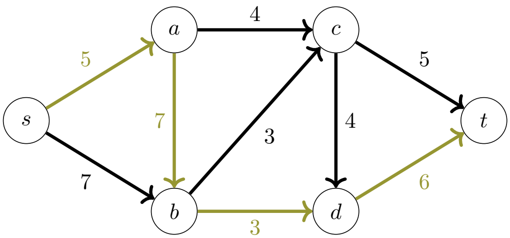
$$G_f^+ := (V, E, r := c - f)$$

$$|f| = 0$$

Greedy Algorithm?

Residual capacity of an edge e**:** $r(e) := c(e) - f(e)$
Residual capacity of a path P**:** $\min_{e \in P} r(e)$

Greedy: Starting with $f(e) = 0$ for all $e \in E$, as long as there exists a path $s \rightsquigarrow t$ with remaining capacity $d > 0$, increase flow along this path by d.



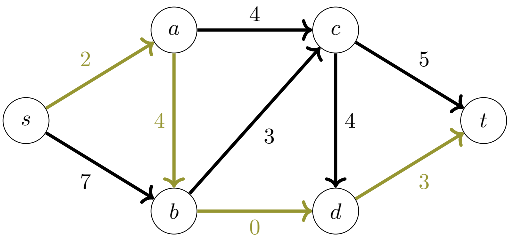
$G_f^+ := (V, E, r := c - f)$

$|f| = 0$

Greedy Algorithm?

Residual capacity of an edge e: $r(e) := c(e) - f(e)$
Residual capacity of a path P: $\min_{e \in P} r(e)$

Greedy: Starting with $f(e) = 0$ for all $e \in E$, as long as there exists a path $s \rightsquigarrow t$ with remaining capacity $d > 0$, increase flow along this path by d.



$G_f^+ := (V, E, r := c - f)$

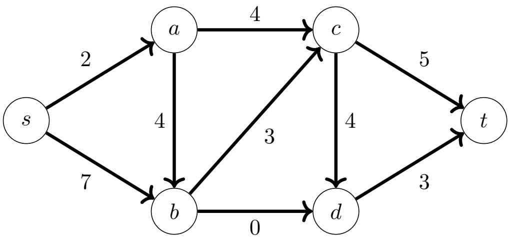
$|f| = 3$

$s \to a \to b \to d \to t$: 3

Greedy Algorithm?

Residual capacity of an edge e**:** $r(e) := c(e) - f(e)$
Residual capacity of a path P**:** $\min_{e \in P} r(e)$

Greedy: Starting with $f(e) = 0$ for all $e \in E$, as long as there exists a path $s \rightsquigarrow t$ with remaining capacity $d > 0$, increase flow along this path by d.



$G_f^+ := (V, E, r := c - f)$

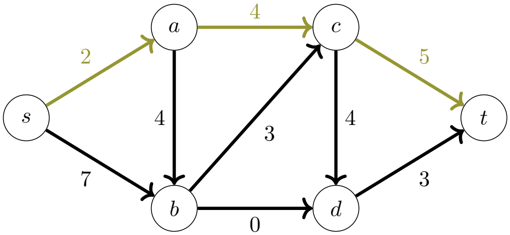
$|f| = 3$

$s \to a \to b \to d \to t$: 3

Greedy Algorithm?

Residual capacity of an edge e**:** $r(e) := c(e) - f(e)$
Residual capacity of a path P**:** $\min_{e \in P} r(e)$

Greedy: Starting with $f(e) = 0$ for all $e \in E$, as long as there exists a path $s \rightsquigarrow t$ with remaining capacity $d > 0$, increase flow along this path by d.



$G_f^+ := (V, E, r := c - f)$

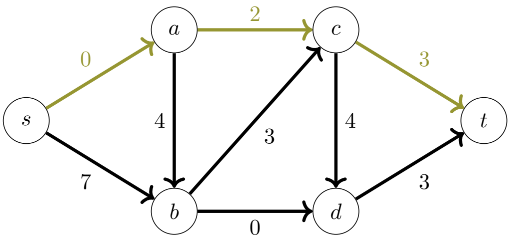
$|f| = 3$

$s \to a \to b \to d \to t$: 3

Greedy Algorithm?

Residual capacity of an edge e**:** $r(e) := c(e) - f(e)$
Residual capacity of a path P**:** $\min_{e \in P} r(e)$

Greedy: Starting with $f(e) = 0$ for all $e \in E$, as long as there exists a path $s \rightsquigarrow t$ with remaining capacity $d > 0$, increase flow along this path by d.



$G_f^+ := (V, E, r := c - f)$

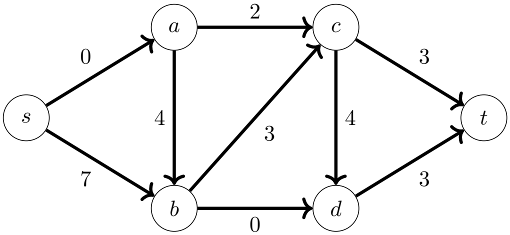
$|f| = 5$

$s \to a \to b \to d \to t$: 3
$s \to a \to c \to t$: 2

Greedy Algorithm?

Residual capacity of an edge e**:** $r(e) := c(e) - f(e)$
Residual capacity of a path P**:** $\min_{e \in P} r(e)$

Greedy: Starting with $f(e) = 0$ for all $e \in E$, as long as there exists a path $s \rightsquigarrow t$ with remaining capacity $d > 0$, increase flow along this path by d.



$G_f^+ := (V, E, r := c - f)$

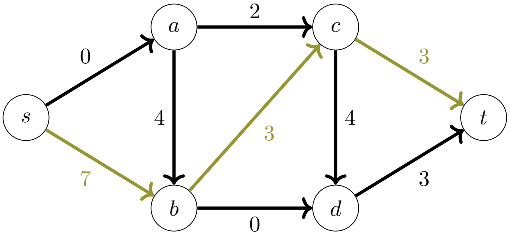
$|f| = 5$

$s \to a \to b \to d \to t$: 3
$s \to a \to c \to t$: 2

Greedy Algorithm?

Residual capacity of an edge e**:** $r(e) := c(e) - f(e)$
Residual capacity of a path P**:** $\min_{e \in P} r(e)$

Greedy: Starting with $f(e) = 0$ for all $e \in E$, as long as there exists a path $s \rightsquigarrow t$ with remaining capacity $d > 0$, increase flow along this path by d.



$G_f^+ := (V, E, r := c - f)$

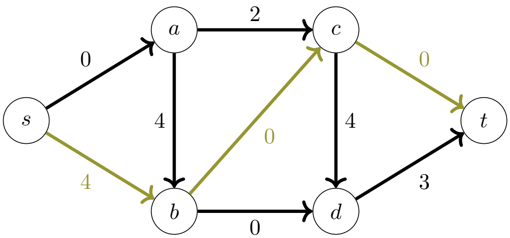
$|f| = 5$

$s \to a \to b \to d \to t$: 3
$s \to a \to c \to t$: 2

Greedy Algorithm?

Residual capacity of an edge e**:** $r(e) := c(e) - f(e)$
Residual capacity of a path P**:** $\min_{e \in P} r(e)$

Greedy: Starting with $f(e) = 0$ for all $e \in E$, as long as there exists a path $s \rightsquigarrow t$ with remaining capacity $d > 0$, increase flow along this path by d.



$G_f^+ := (V, E, r := c - f)$

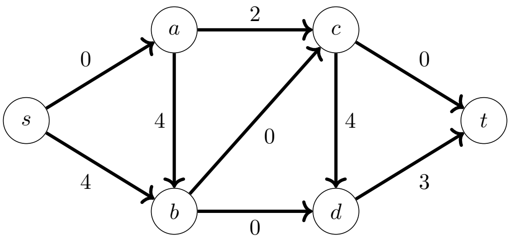
$|f| = 8$

$s \to a \to b \to d \to t$: 3
$s \to a \to c \to t$: 2
$s \to b \to c \to t$: 3

Greedy Algorithm?

Residual capacity of an edge e**:** $r(e) := c(e) - f(e)$
Residual capacity of a path P**:** $\min_{e \in P} r(e)$

Greedy: Starting with $f(e) = 0$ for all $e \in E$, as long as there exists a path $s \rightsquigarrow t$ with remaining capacity $d > 0$, increase flow along this path by d.



$G_f^+ := (V, E, r := c - f)$

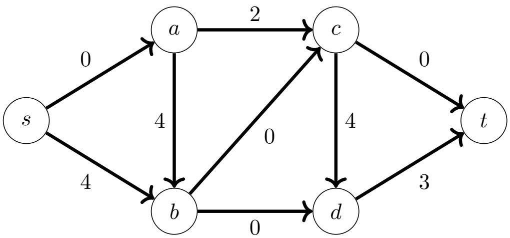
$|f| = 8$

$s \to a \to b \to d \to t$: 3
$s \to a \to c \to t$: 2
$s \to b \to c \to t$: 3

Greedy Algorithm?

Residual capacity of an edge e: $r(e) := c(e) - f(e)$
Residual capacity of a path P: $\min_{e \in P} r(e)$

Greedy: Starting with $f(e) = 0$ for all $e \in E$, as long as there exists a path $s \rightsquigarrow t$ with remaining capacity $d > 0$, increase flow along this path by d.



$G_f^+ := (V, E, r := c - f)$

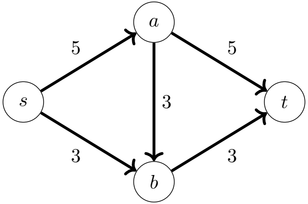
$|f| = 8$

$s \to a \to b \to d \to t$: 3
$s \to a \to c \to t$: 2
$s \to b \to c \to t$: 3

but $|f_{\max}| = 10$

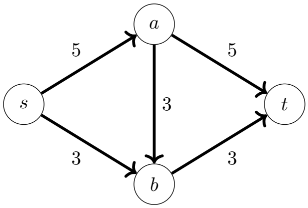
Problem with Greedy

Problem with Greedy

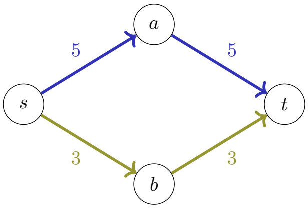


$$G = (V, E, c)$$

Problem with Greedy

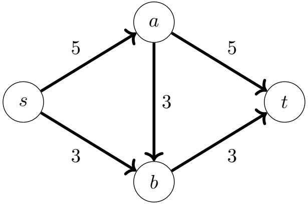


$$G = (V, E, c)$$

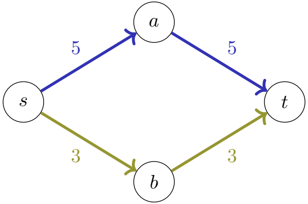


$$|f_{\max}| = 8$$

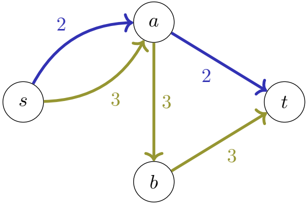
Problem with Greedy



$G = (V, E, c)$

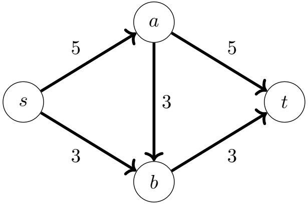


$|f_{\max}| = 8$

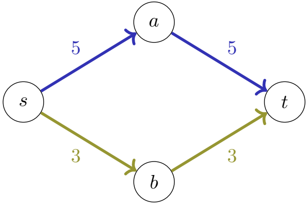


Greedy: $|f| = 5$

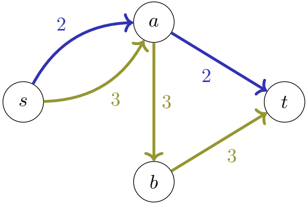
Problem with Greedy



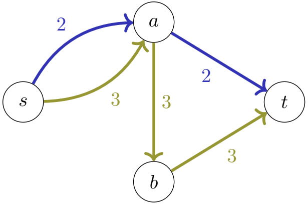
$G = (V, E, c)$



$|f_{\max}| = 8$

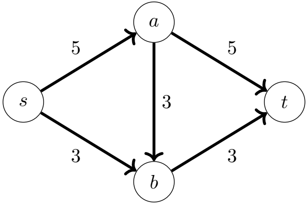


Greedy: $|f| = 5$

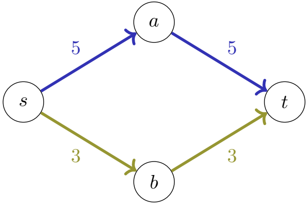


Redirection

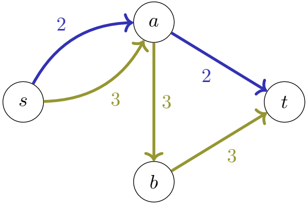
Problem with Greedy



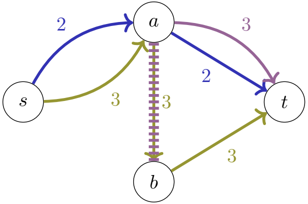
$G = (V, E, c)$



$|f_{\max}| = 8$

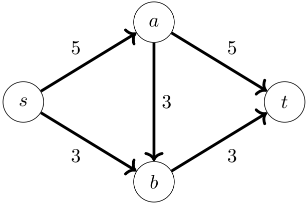


Greedy: $|f| = 5$

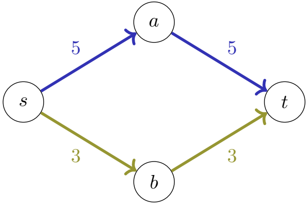


Redirection

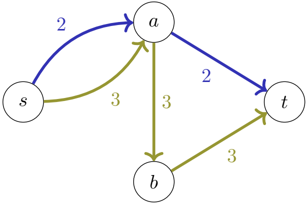
Problem with Greedy



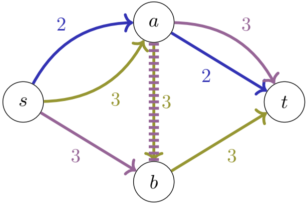
$G = (V, E, c)$



$|f_{\text{max}}| = 8$



Greedy: $|f| = 5$



Redirection

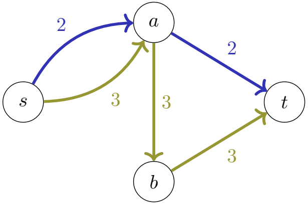
29.1 Flow Algorithms

Ford-Fulkerson Algorithm

Edmonds-Karp Algorithm

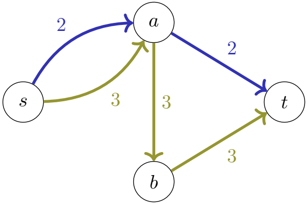
Redirection using flow decrement

Redirection using flow decrement

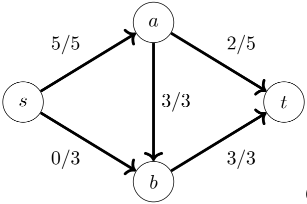


before

Redirection using flow decrement

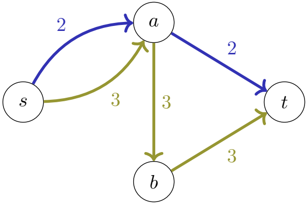


before

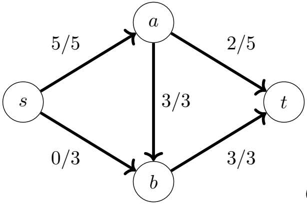


$G = (V, E, f/c)$

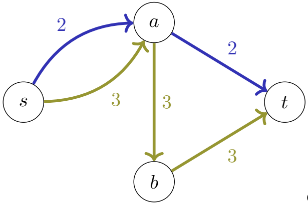
Redirection using flow decrement



before

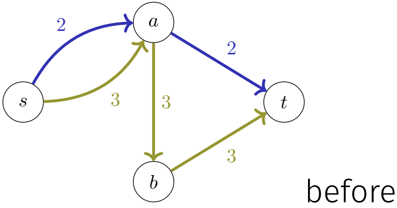


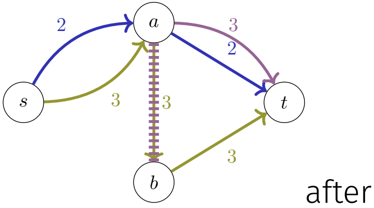
$G = (V, E, f/c)$

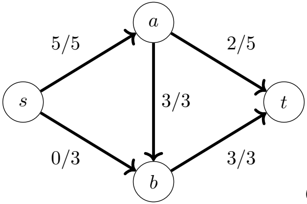


after

Redirection using flow decrement

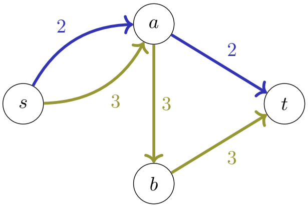




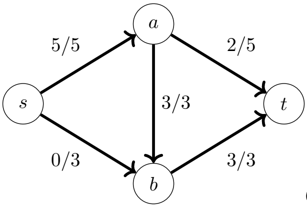


$G = (V, E, f/c)$

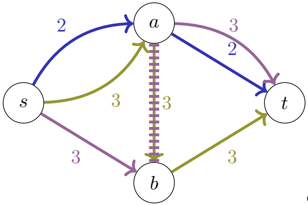
Redirection using flow decrement



before

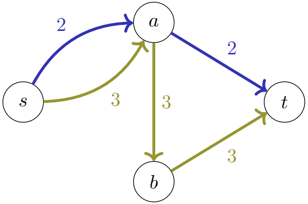


$G = (V, E, f/c)$

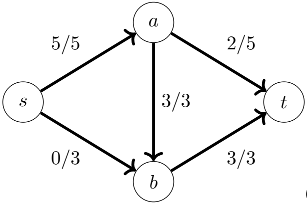


after

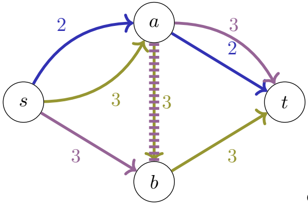
Redirection using flow decrement



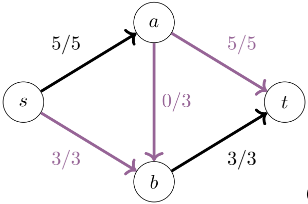
before



$G = (V, E, f/c)$

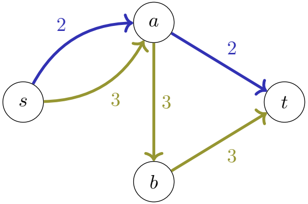


after

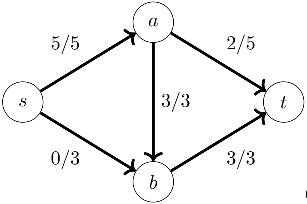


$G = (V, E, f'/c)$

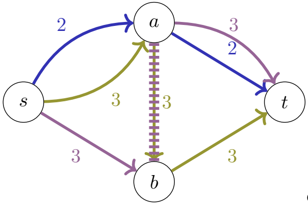
Redirection using flow decrement



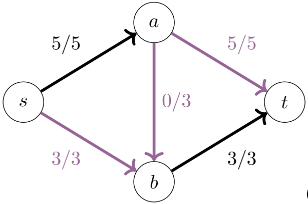
before



$G = (V, E, f/c)$



after

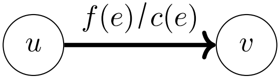


$G = (V, E, f'/c)$

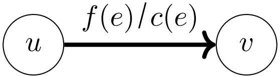
$\Rightarrow$ Umleitung entspricht Verringerung des Flusses durch Kante

Idea: Flow increments and decrements

Idea: Flow increments and decrements

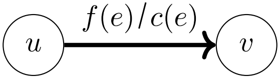


Idea: Flow increments and decrements



- **Increment:**
  flow through e can be increased by at most
  $r(e) := c(e) - f(e)$

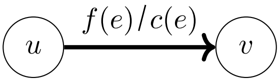
Idea: Flow increments and decrements



- **Increment:**
  flow through e can be increased by at most
  $r(e) := c(e) - f(e)$

$G_f^+ := (V, E, r := c - f)$

Idea: Flow increments and decrements

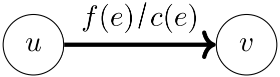


- **Increment:**
  flow through e can be increased by at most
  $r(e) := c(e) - f(e)$

$G_f^+ := (V, E, r := c - f)$

- **Decrement:**
  flow through e can be decreased by at most
  $f(e)$

Idea: Flow increments and decrements



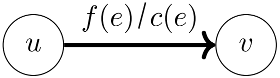
- **Increment:**
  flow through e can be increased by at most $r(e) := c(e) - f(e)$

$G_f^+ := (V, E, r := c - f)$

- **Decrement:**
  flow through e can be decreased by at most $f(e)$

  $\Rightarrow$ flow through $\overleftarrow{e}$ can be increased by at most $f(e)$

Idea: Flow increments and decrements



- **Increment:**
  flow through e can be increased by at most $r(e) := c(e) - f(e)$

$G_f^+ := (V, E, r := c - f)$

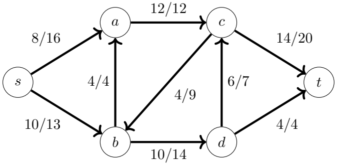
- **Decrement:**
  flow through e can be decreased by at most $f(e)$

  $\Rightarrow$ flow through $\overleftarrow{e}$ can be increased by at most $f(e)$

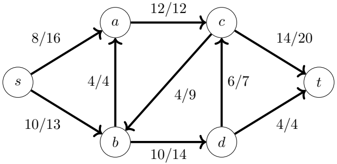
$G_f^- := (V, \overleftarrow{E}, f)$

Residual Network

Residual Network

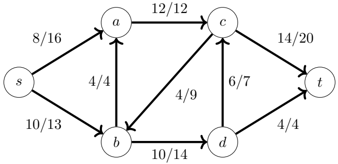


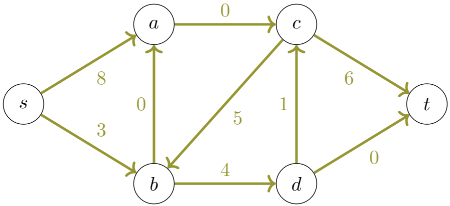
Residual Network



Residual network: $G_f := \mathbf{G_f^+} \cup \mathbf{G_f^-} = (V, E_f, c_f)$

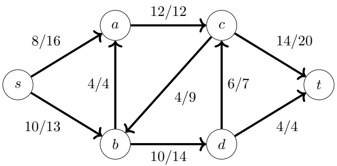
Residual Network

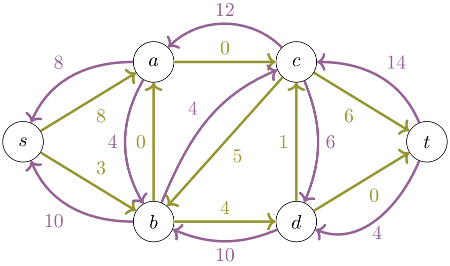




Residual network: $G_f := \mathbf{G_f^+} \cup \mathbf{G_f^-} = (V, E_f, c_f)$

Residual Network

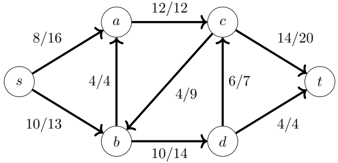


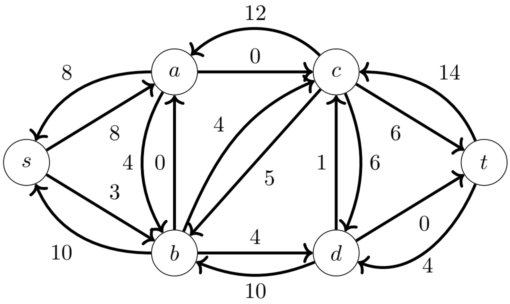


Residual network: $G_f := \mathbf{G_f^+} \cup \mathbf{G_f^-} = (V, E_f, c_f)$

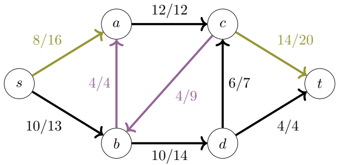
Ford-Fulkerson: Flow augmentation

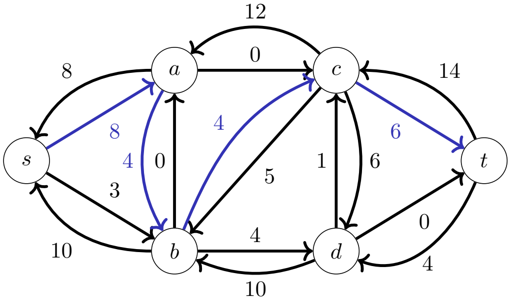
Ford-Fulkerson: Flow augmentation





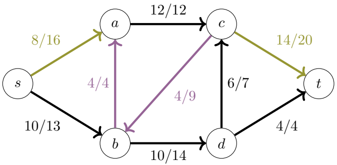
Ford-Fulkerson: Flow augmentation

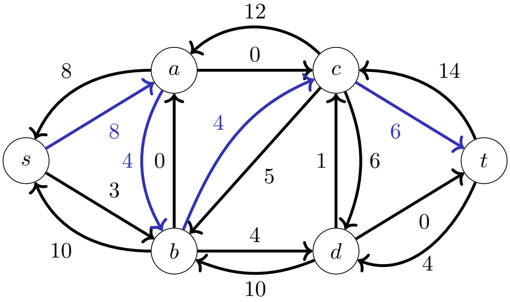




- **Augmenting Path:** Find a path $\mathbf{P}: \mathbf{s} \to \mathbf{t}$ with residual capacity $d > 0$ in G_f

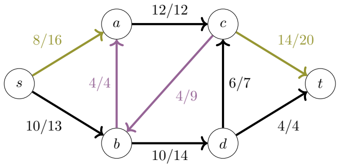
Ford-Fulkerson: Flow augmentation

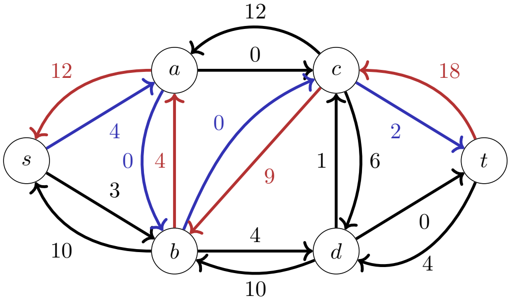




- **Augmenting Path:** Find a path $\mathbf{P} : \mathbf{s} \to \mathbf{t}$ with residual capacity $d > 0$ in G_f
- augment flow along this path for all $e \in P$ by d:

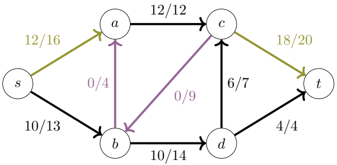
Ford-Fulkerson: Flow augmentation

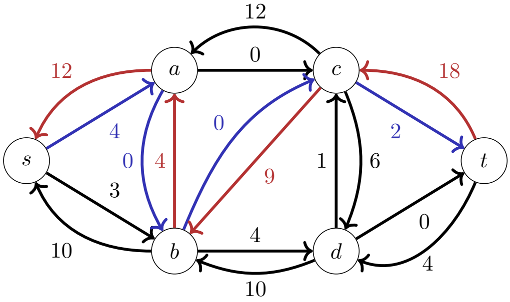




- **Augmenting Path:** Find a path $\mathbf{P}: \mathbf{s} \to \mathbf{t}$ with residual capacity $d > 0$ in G_f
- augment flow along this path for all $e \in P$ by d:
  - decrease residual capacity $\mathbf{c_f(e)}$ in G_f by d; increase $\mathbf{c_f}(\overleftarrow{e})$ by d

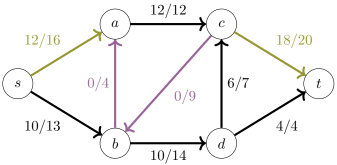
Ford-Fulkerson: Flow augmentation

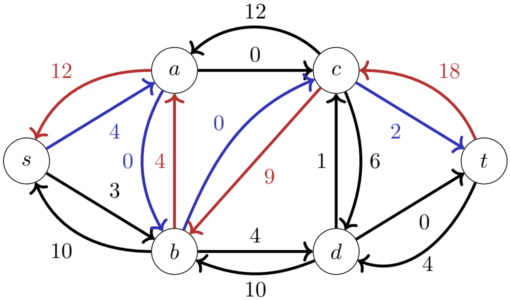




- **Augmenting Path:** Find a path $\mathbf{P} : \mathbf{s} \rightarrow \mathbf{t}$ with residual capacity $d > 0$ in G_f
- augment flow along this path for all $e \in P$ by d:
  - decrease residual capacity $\mathbf{c_f}(\mathbf{e})$ in G_f by d; increase $\mathbf{c_f}(\overleftarrow{e})$ by d
  - increase flow through $\mathbf{e} \in \mathbf{E}$ by d; decrease through $\overleftarrow{e} \in \mathbf{E}$

Ford-Fulkerson: Flow augmentation





- **Augmenting Path:** Find a path $\mathbf{P} : \mathbf{s} \to \mathbf{t}$ with residual capacity $d > 0$ in G_f
- augment flow along this path for all $e \in P$ by d:
  - decrease residual capacity $\mathbf{c_f(e)}$ in G_f by d; increase $\mathbf{c_f}(\overleftarrow{e})$ by d
  - increase flow through $\mathbf{e} \in \mathbf{E}$ by d; decrease through $\overleftarrow{e} \in \mathbf{E}$

Algorithm Ford-Fulkerson(G, s, t)

Input: Flow network $G = (V, E, c)$, source s, sink t
Output: Maximal flow f

Algorithm Ford-Fulkerson(G, s, t)

Input: Flow network $G = (V, E, c)$, source s, sink t
Output: Maximal flow f

for $e \in E$ **do**
  $f(e) \leftarrow 0$

Algorithm Ford-Fulkerson(G, s, t)

Input: Flow network $G = (V, E, c)$, source s, sink t
Output: Maximal flow f

for $e \in E$ **do**
  $f(e) \leftarrow 0$

while exists positive path $P\colon s \rightsquigarrow t$ in residual network $G_f = (V, E_f, c_f)$ **do**

Algorithm Ford-Fulkerson(G, s, t)

Input: Flow network $G = (V, E, c)$, source s, sink t
Output: Maximal flow f

for $e \in E$ **do**
  $f(e) \leftarrow 0$

while exists positive path $P\colon s \rightsquigarrow t$ in residual network $G_f = (V, E_f, c_f)$ **do**
  $d \leftarrow \min_{e \in P} c_f(e)$

Algorithm Ford-Fulkerson(G, s, t)

Input: Flow network $G = (V, E, c)$, source s, sink t
Output: Maximal flow f

for $e \in E$ **do**
  $f(e) \leftarrow 0$

while exists positive path $P\colon s \rightsquigarrow t$ in residual network $G_f = (V, E_f, c_f)$ **do**
  $d \leftarrow \min_{e \in P} c_f(e)$
  foreach $e \in P$ **do**

Algorithm Ford-Fulkerson(G, s, t)

Input: Flow network $G = (V, E, c)$, source s, sink t
Output: Maximal flow f

for $e \in E$ **do**
  $f(e) \leftarrow 0$

while exists positive path $P\colon s \rightsquigarrow t$ in residual network $G_f = (V, E_f, c_f)$ **do**
  $d \leftarrow \min_{e \in P} c_f(e)$
  foreach $e \in P$ **do**
    if $e \in E$ **then**
      $f(e) \leftarrow f(e) + d$

Algorithm Ford-Fulkerson(G, s, t)

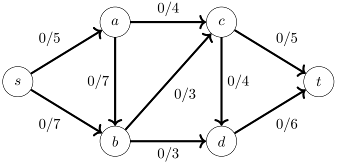
Input: Flow network $G = (V, E, c)$, source s, sink t
Output: Maximal flow f

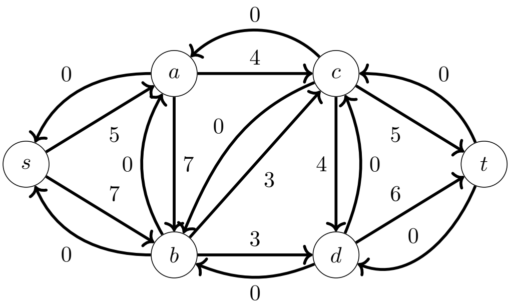
for $e \in E$ **do**
  $f(e) \leftarrow 0$

while exists positive path $P: s \rightsquigarrow t$ in residual network $G_f = (V, E_f, c_f)$ **do**
  $d \leftarrow \min_{e \in P} c_f(e)$
  foreach $e \in P$ **do**
    if $e \in E$ **then**
      $f(e) \leftarrow f(e) + d$
    else
      $f(\overleftarrow{e}) \leftarrow f(\overleftarrow{e}) - d$

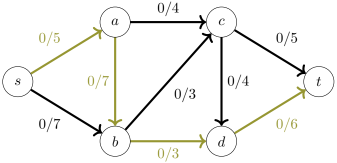
Example Ford-Fulkerson

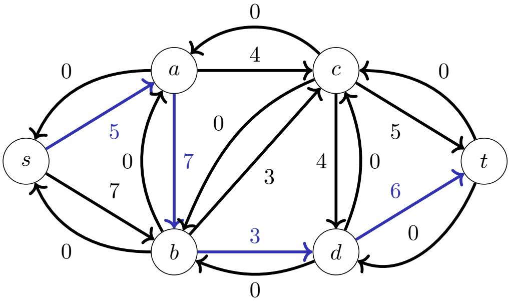
Example Ford-Fulkerson



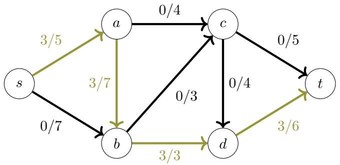


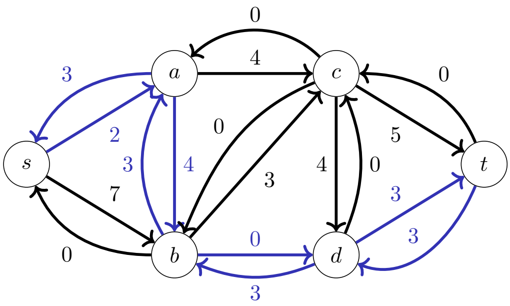
Example Ford-Fulkerson



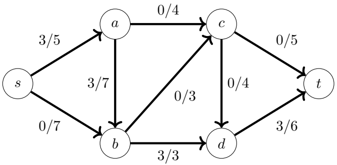


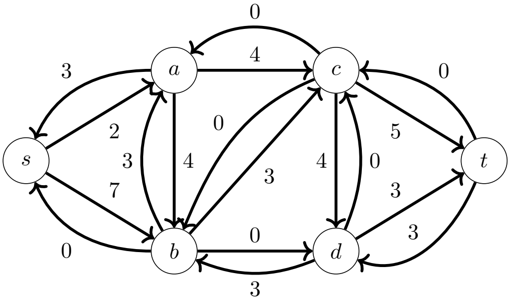
Example Ford-Fulkerson



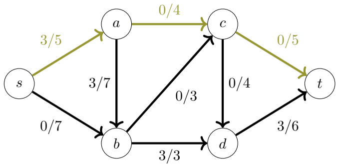


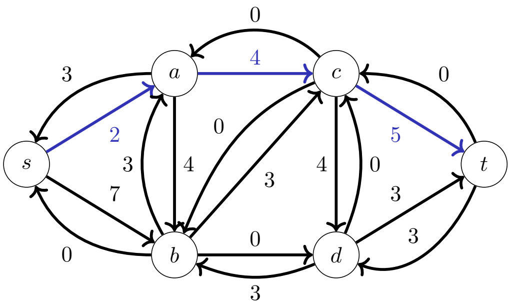
Example Ford-Fulkerson



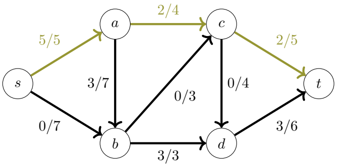


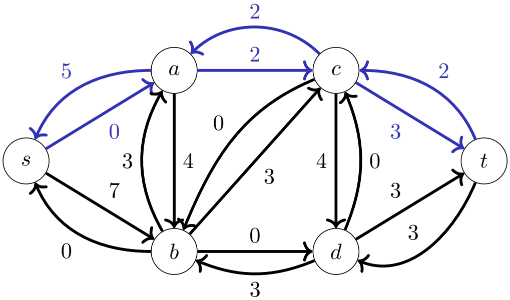
Example Ford-Fulkerson



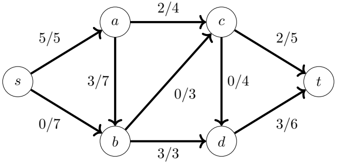


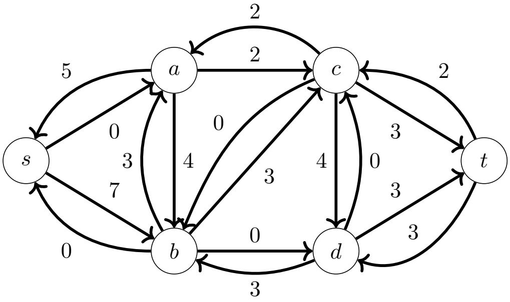
Example Ford-Fulkerson



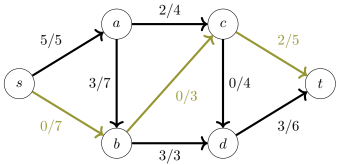


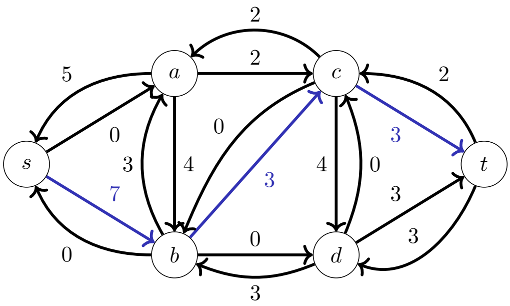
Example Ford-Fulkerson



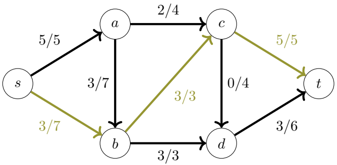


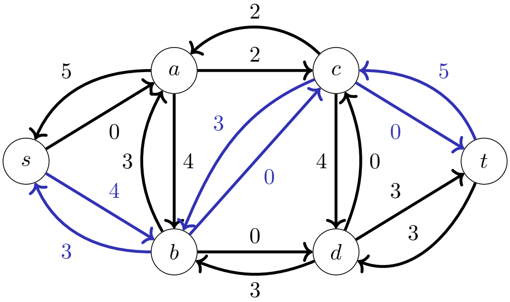
Example Ford-Fulkerson



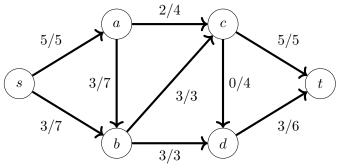


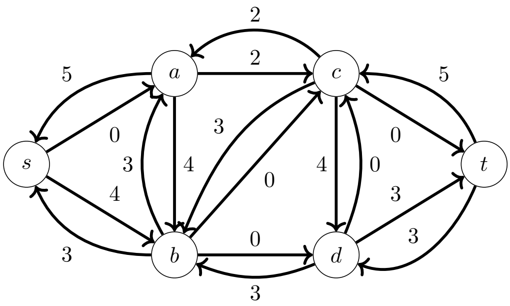
Example Ford-Fulkerson



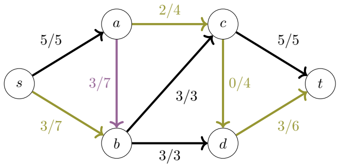


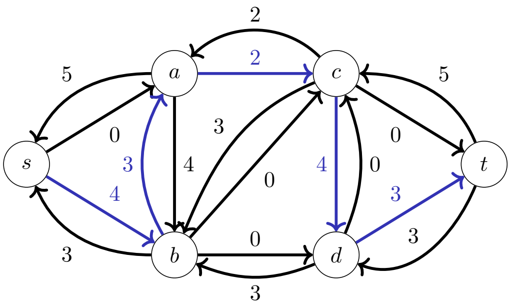
Example Ford-Fulkerson



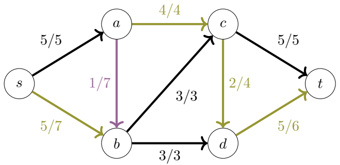


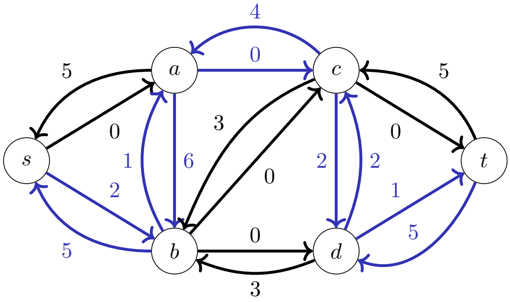
Example Ford-Fulkerson



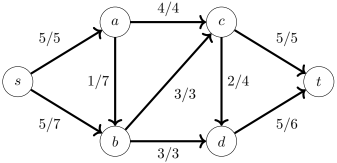


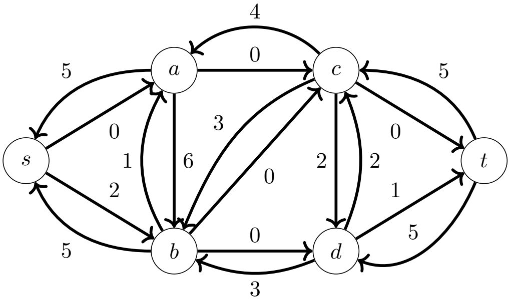
Example Ford-Fulkerson



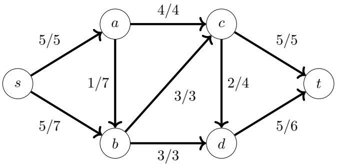


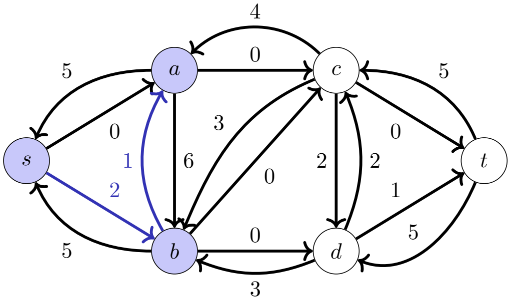
Example Ford-Fulkerson





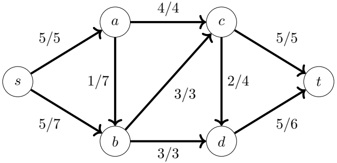
Example Ford-Fulkerson

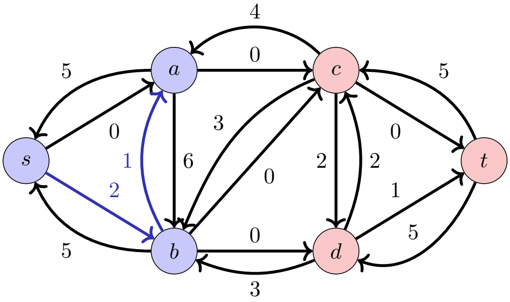




nodes reachable from s

Example Ford-Fulkerson

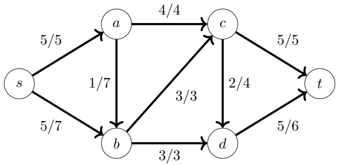


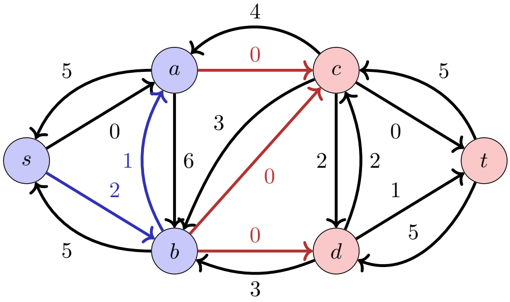


nodes reachable from s

nodes not reachable from s

Example Ford-Fulkerson



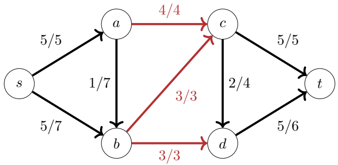


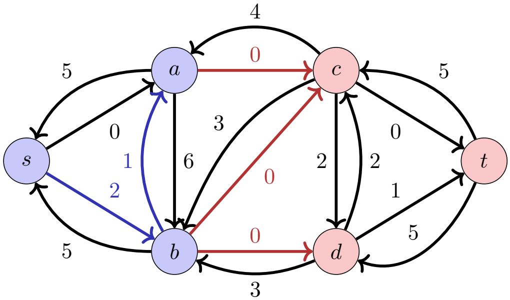
nodes reachable from s

nodes not reachable from s

all outgoing edges have residual capacity 0 in G_f

Example Ford-Fulkerson





nodes reachable from s

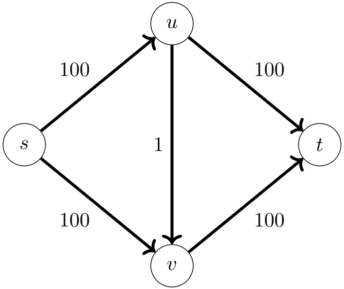
nodes not reachable from s

all outgoing edges have residual capacity 0 in G_f

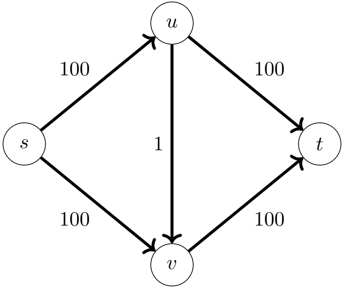
$\Rightarrow$ flow fully exhausts capacity on these edges!

Quiz Ford-Fulkerson

Quiz Ford-Fulkerson



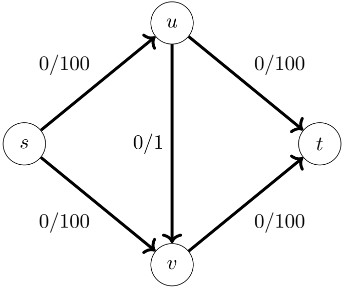
Quiz Ford-Fulkerson



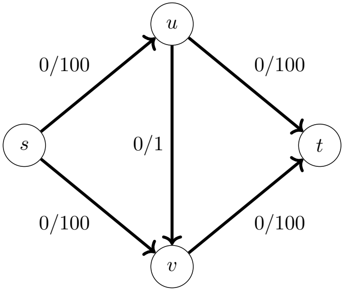
How many iterations does Ford-Fulkerson need in the worst case?

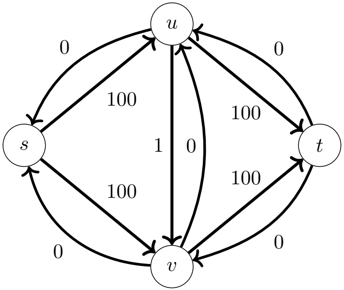
Solution

Solution

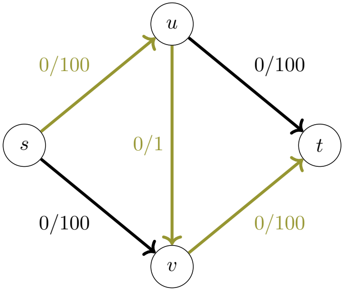


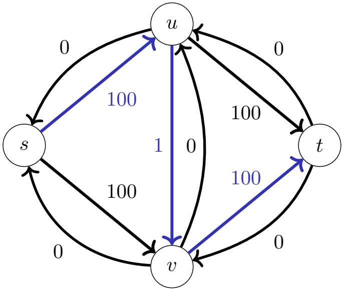
Solution



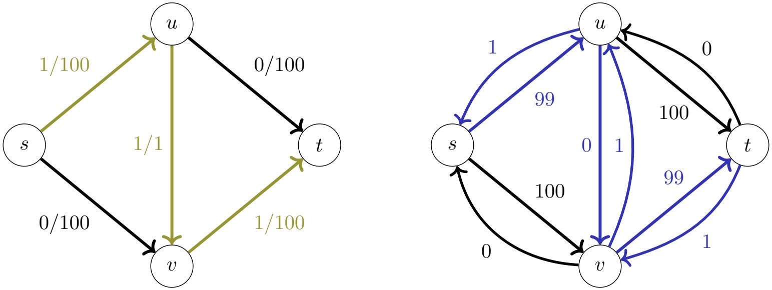


Solution

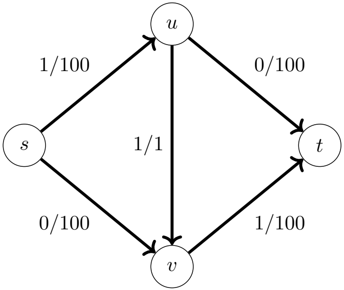


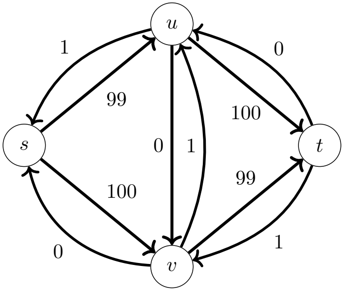


Solution

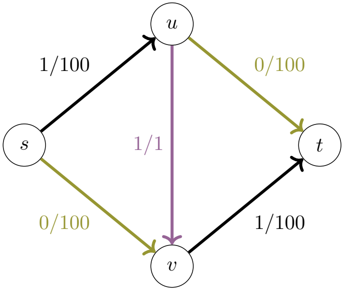


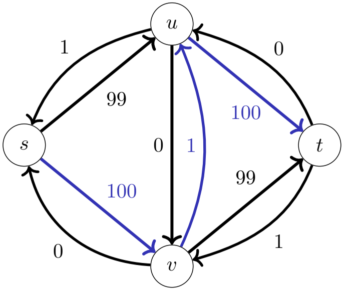
Solution



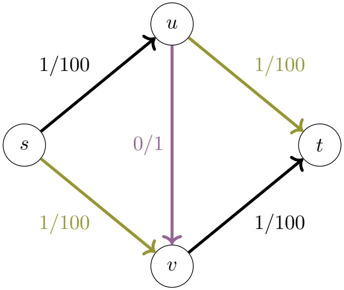


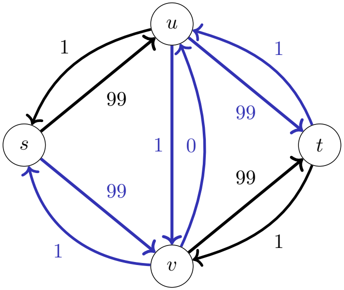
Solution



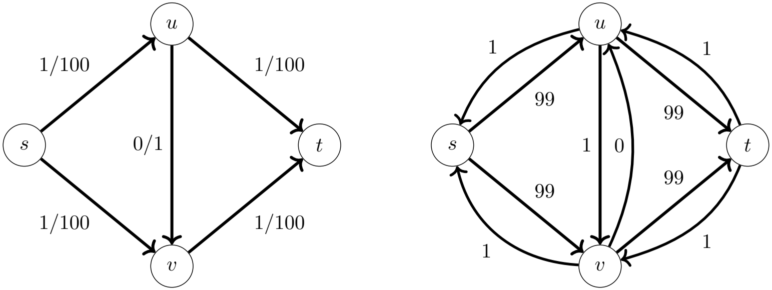


Solution

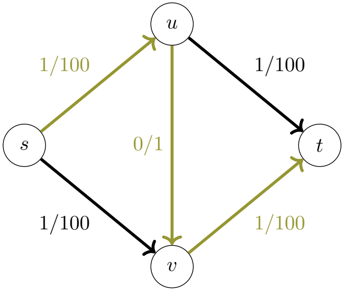


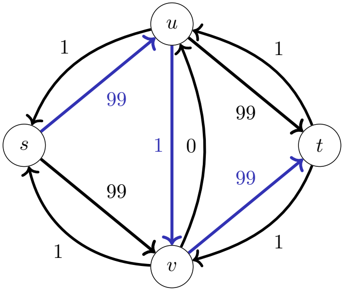


Solution

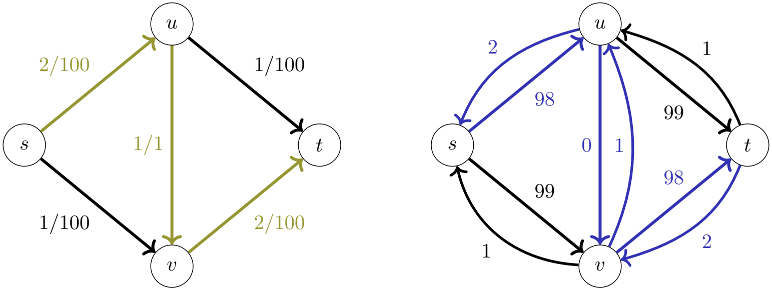


Solution

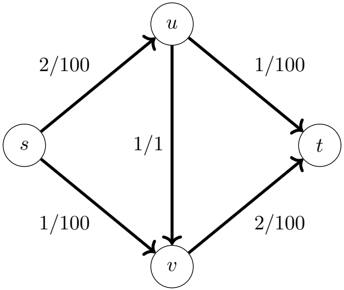


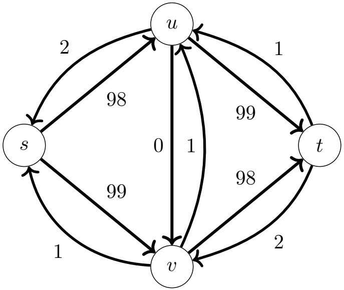


Solution

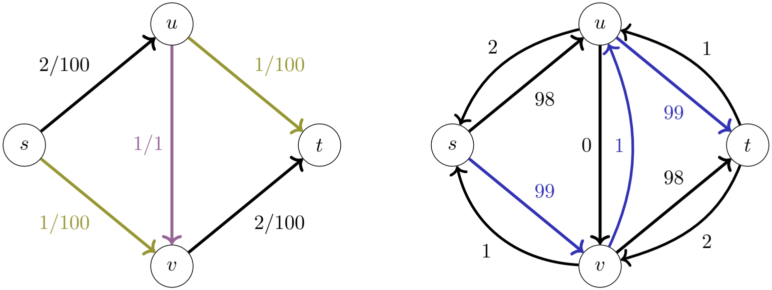


Solution

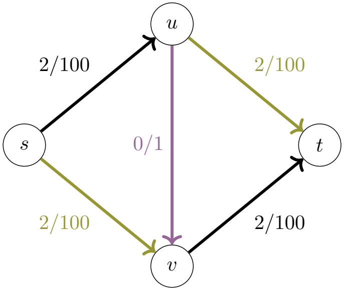


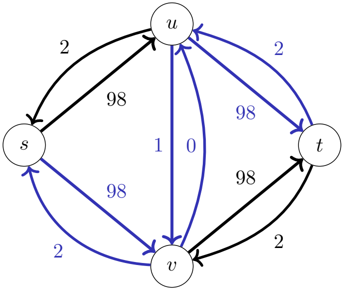


Solution

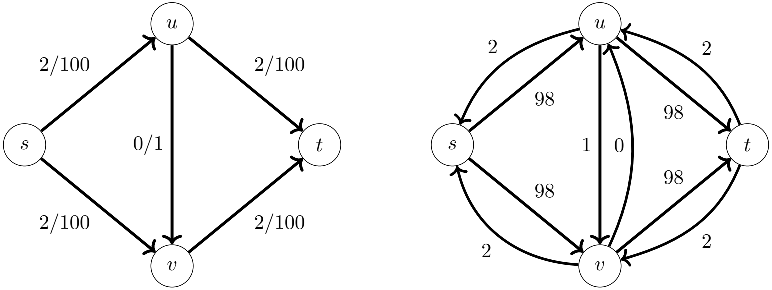


Solution

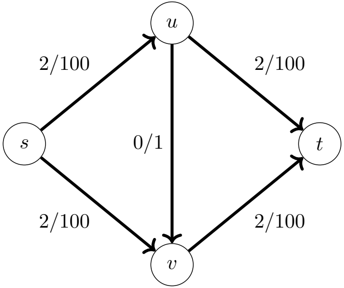


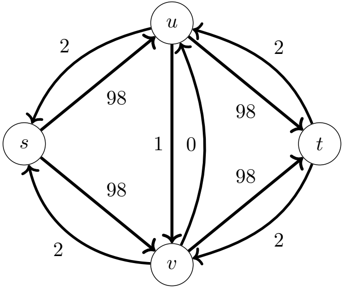


Solution



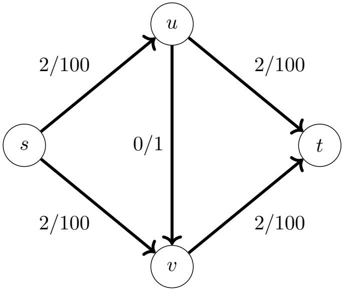
Solution

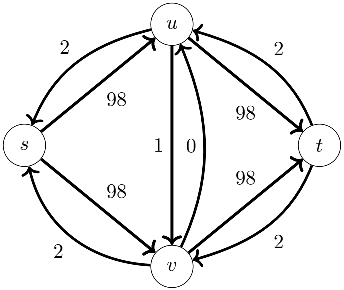




After i iterations: $|f| = i$

Solution

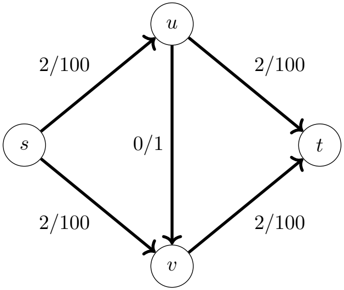


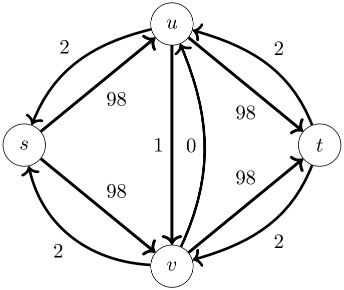


After i iterations: $|f| = i$

$\Rightarrow$ in total $|f_{\max}|$ iterations

Solution





After i iterations: $|f| = i$

$\Rightarrow$ in total $|f_{\max}| = 200$ iterations

Running Time Analysis of Ford-Fulkerson

Running Time Analysis of Ford-Fulkerson

Running time of each iteration:

Running Time Analysis of Ford-Fulkerson

Running time of each iteration: search of an augmenting path $s \rightsquigarrow t$

Running Time Analysis of Ford-Fulkerson

Running time of each iteration: search of an augmenting path $s \rightsquigarrow t$

$\Rightarrow$ BFS or DFS: $\mathcal{O}(|V| + |E|) = \mathcal{O}(|E|)$

Running Time Analysis of Ford-Fulkerson

Running time of each iteration: search of an augmenting path $s \rightsquigarrow t$

$\Rightarrow$ BFS or DFS: $\mathcal{O}(|V| + |E|) = \mathcal{O}(|E|)$

($|V| \leq |E|$, because all non-reachable nodes can be ignored.)

Running Time Analysis of Ford-Fulkerson

Running time of each iteration: search of an augmenting path $s \rightsquigarrow t$

$\Rightarrow$ BFS or DFS: $\mathcal{O}(|V| + |E|) = \mathcal{O}(|E|)$

($|V| \leq |E|$, because all non-reachable nodes can be ignored.)

Number of iterations:

Running Time Analysis of Ford-Fulkerson

Running time of each iteration: search of an augmenting path $s \rightsquigarrow t$

$\Rightarrow$ BFS or DFS: $\mathcal{O}(|V| + |E|) = \mathcal{O}(|E|)$

($|V| \leq |E|$, because all non-reachable nodes can be ignored.)

Number of iterations:

In every step, the size of the flow increases by $d > 0$.

Running Time Analysis of Ford-Fulkerson

Running time of each iteration: search of an augmenting path $s \rightsquigarrow t$

$\Rightarrow$ BFS or DFS: $\mathcal{O}(|V| + |E|) = \mathcal{O}(|E|)$

($|V| \leq |E|$, because all non-reachable nodes can be ignored.)

Number of iterations:

In every step, the size of the flow increases by $d > 0$.

integer capacities

Running Time Analysis of Ford-Fulkerson

Running time of each iteration: search of an augmenting path $s \rightsquigarrow t$

$\Rightarrow$ BFS or DFS: $\mathcal{O}(|V| + |E|) = \mathcal{O}(|E|)$

($|V| \leq |E|$, because all non-reachable nodes can be ignored.)

Number of iterations:

In every step, the size of the flow increases by $d > 0$.

integer capacities $\Rightarrow$ increment by ≥ 1

Running Time Analysis of Ford-Fulkerson

Running time of each iteration: search of an augmenting path $s \rightsquigarrow t$

$\Rightarrow$ BFS or DFS: $\mathcal{O}(|V| + |E|) = \mathcal{O}(|E|)$

($|V| \le |E|$, because all non-reachable nodes can be ignored.)

Number of iterations:

In every step, the size of the flow increases by $d > 0$.

integer capacities $\Rightarrow$ increment by ≥ 1 $\Rightarrow$ at most $|f_{\max}|$ iterations

Running Time Analysis of Ford-Fulkerson

Running time of each iteration: search of an augmenting path $s \rightsquigarrow t$

$\Rightarrow$ BFS or DFS: $\mathcal{O}(|V| + |E|) = \mathcal{O}(|E|)$

($|V| \leq |E|$, because all non-reachable nodes can be ignored.)

Number of iterations:

In every step, the size of the flow increases by $d > 0$.

integer capacities $\Rightarrow$ increment by $\geq 1 \Rightarrow$ at most $|f_{\max}|$ iterations

$\Rightarrow \mathcal{O}(|f_{\max}| \cdot |E|)$ for flow networks $G = (V, E, c)$ with $c\colon E \rightarrow \mathbb{N}^{\geq 1}$

Running Time Analysis of Ford-Fulkerson

Running time of each iteration: search of an augmenting path $s \rightsquigarrow t$

$\Rightarrow$ BFS or DFS: $\mathcal{O}(|V| + |E|) = \mathcal{O}(|E|)$

($|V| \leq |E|$, because all non-reachable nodes can be ignored.)

Number of iterations:

In every step, the size of the flow increases by $d > 0$.

integer capacities $\Rightarrow$ increment by $\geq 1 \Rightarrow$ at most $|f_{\max}|$ iterations

$\Rightarrow \mathcal{O}(|f_{\max}| \cdot |E|)$ for flow networks $G = (V, E, c)$ with $c \colon E \to \mathbb{N}^{\geq 1}$

Edmonds-Karp Algorithm: (Variant of Ford-Fulkerson)

Running Time Analysis of Ford-Fulkerson

Running time of each iteration: search of an augmenting path $s \rightsquigarrow t$

$\Rightarrow$ BFS or DFS: $\mathcal{O}(|V| + |E|) = \mathcal{O}(|E|)$

($|V| \leq |E|$, because all non-reachable nodes can be ignored.)

Number of iterations:

In every step, the size of the flow increases by $d > 0$.

integer capacities $\Rightarrow$ increment by $\geq 1 \Rightarrow$ at most $|f_{\max}|$ iterations

$\Rightarrow \mathcal{O}(|f_{\max}| \cdot |E|)$ for flow networks $G = (V, E, c)$ with $c\colon E \to \mathbb{N}^{\geq 1}$

Edmonds-Karp Algorithm: (Variant of Ford-Fulkerson)

shortest augmenting path (number of edges)

Running Time Analysis of Ford-Fulkerson

Running time of each iteration: search of an augmenting path $s \rightsquigarrow t$

$\Rightarrow$ BFS or DFS: $\mathcal{O}(|V| + |E|) = \mathcal{O}(|E|)$

($|V| \leq |E|$, because all non-reachable nodes can be ignored.)

Number of iterations:

In every step, the size of the flow increases by $d > 0$.

integer capacities $\Rightarrow$ increment by $\geq 1 \Rightarrow$ at most $|f_{\max}|$ iterations

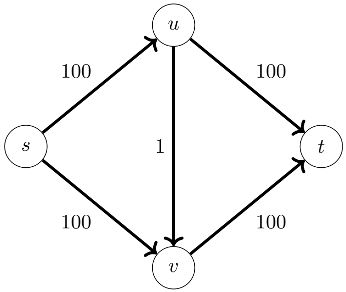
$\Rightarrow \mathcal{O}(|f_{\max}| \cdot |E|)$ for flow networks $G = (V, E, c)$ with $c\colon E \to \mathbb{N}^{\geq 1}$

Edmonds-Karp Algorithm: (Variant of Ford-Fulkerson)

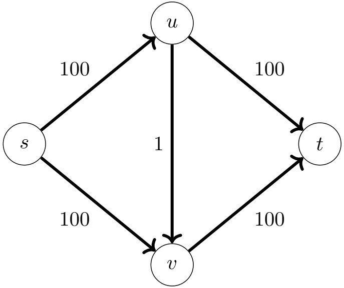
shortest augmenting path (number of edges) $\Rightarrow \mathcal{O}(|V| \cdot |E|^2)$ (without explanation)

Quiz Edmonds-Karp

Quiz Edmonds-Karp



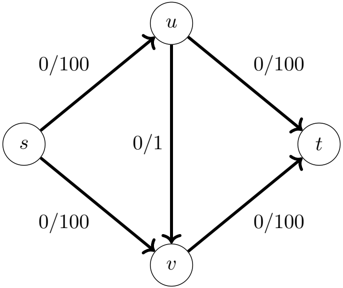
Quiz Edmonds-Karp



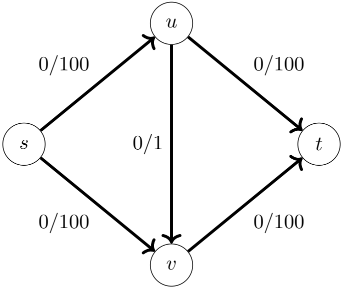
How many iterations does Edmonds-Karp need in the worst case?

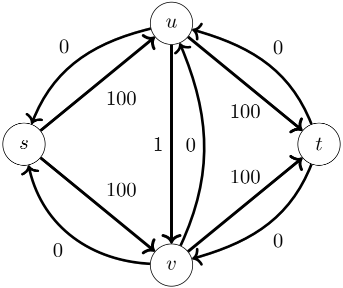
Solution

Solution

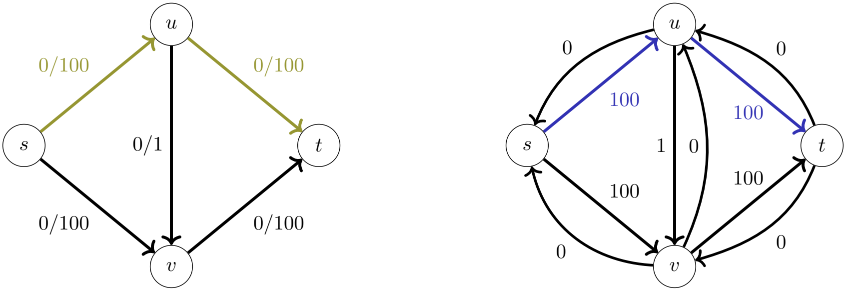


Solution

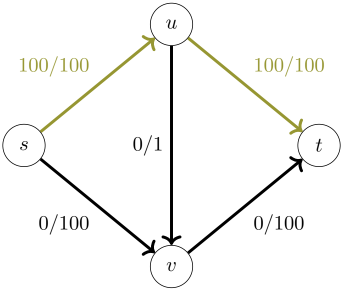


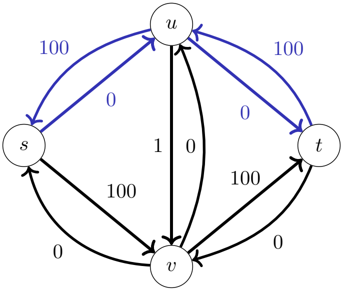


Solution

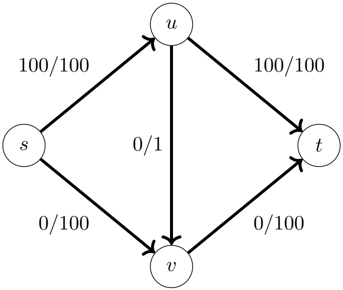


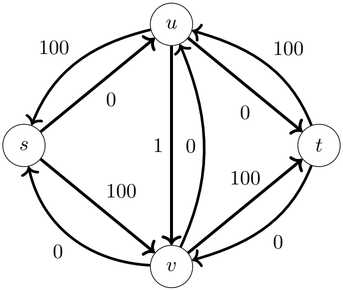
Solution



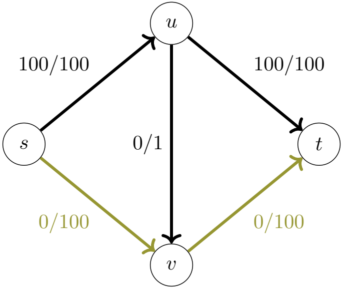


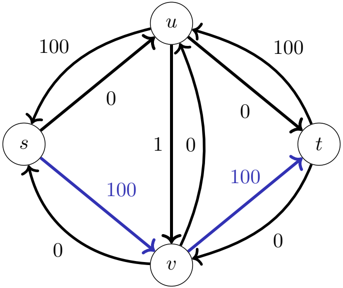
Solution



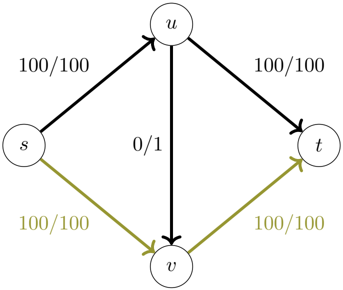


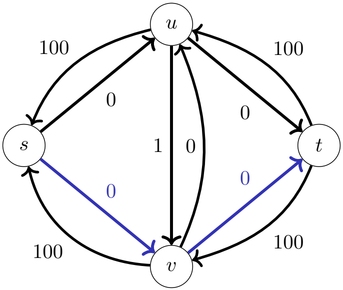
Solution



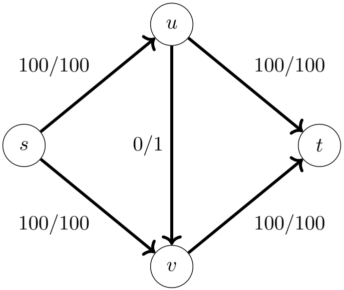


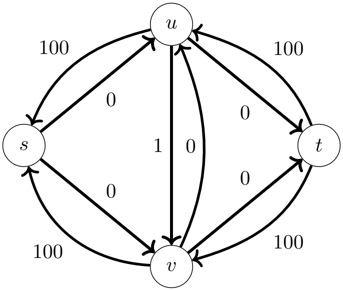
Solution


u
s
t
v
100/100
100/100
0/1
100/100
100/100


u
s
t
v
100
100
0
0
1
0
0
0
100
100

Solution



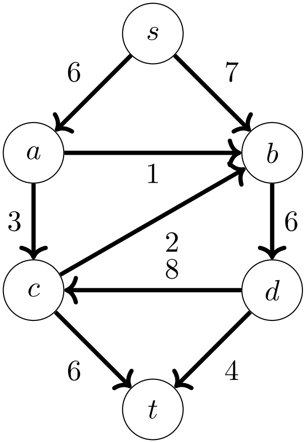


Termination after 2 iterations!

29.2 Max-Flow Min-Cut

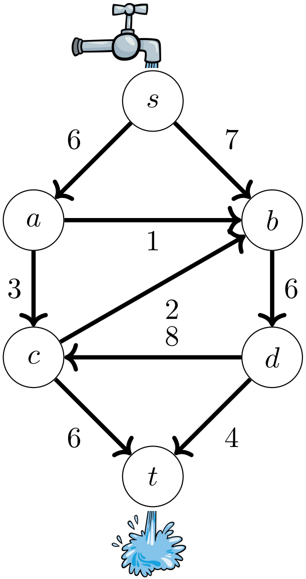
Flows and Cuts: Bottleneck Intuition

Flows and Cuts: Bottleneck Intuition



Flows and Cuts: Bottleneck Intuition

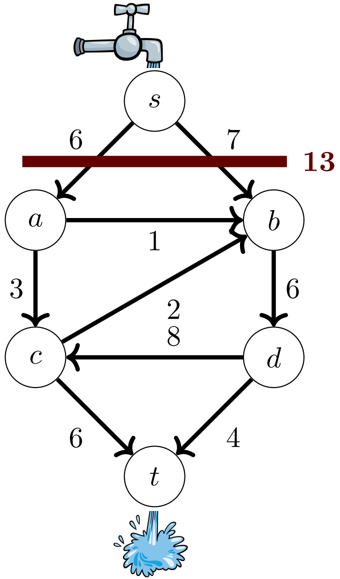
Upper bounds on size of flow:



Flows and Cuts: Bottleneck Intuition

Upper bounds on size of flow:

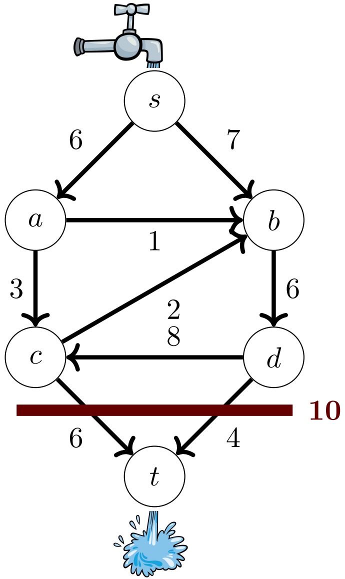
- what can flow out of s: $c^+(s)$



Flows and Cuts: Bottleneck Intuition

Upper bounds on size of flow:

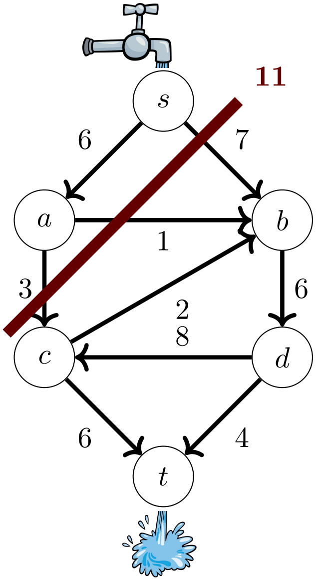
- what can flow out of s: $c^+(s)$
- what can flow into t: $c^-(t)$



Flows and Cuts: Bottleneck Intuition

Upper bounds on size of flow:

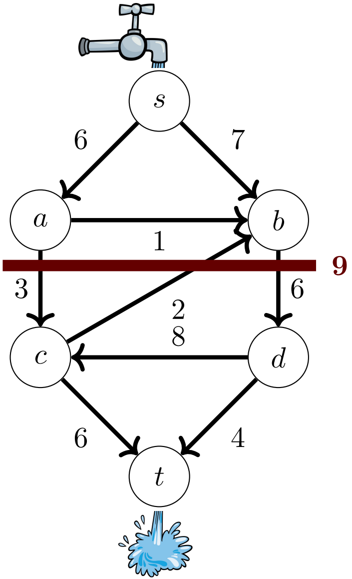
- what can flow out of s: $c^+(s)$
- what can flow into t: $c^-(t)$
- what can flow through arbitrary cut



Flows and Cuts: Bottleneck Intuition

Upper bounds on size of flow:

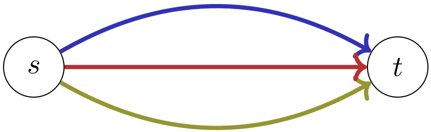
- what can flow out of s: $c^+(s)$
- what can flow into t: $c^-(t)$
- what can flow through arbitrary cut
- what can flow through bottleneck: $c_{\min}$

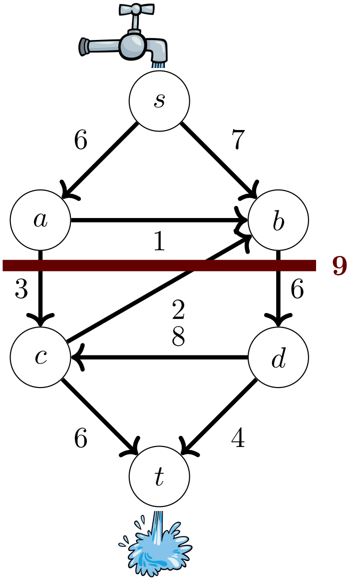


Flows and Cuts: Bottleneck Intuition

Upper bounds on size of flow:

- what can flow out of s: $c^+(s)$
- what can flow into t: $c^-(t)$
- what can flow through arbitrary cut
- what can flow through bottleneck: $c_{\min}$

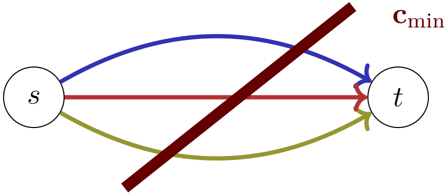


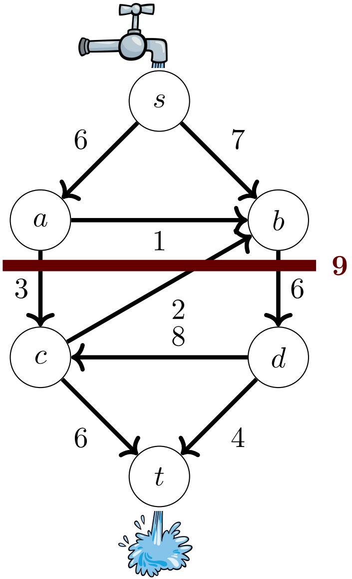


Flows and Cuts: Bottleneck Intuition

Upper bounds on size of flow:

- what can flow out of s: $c^+(s)$
- what can flow into t: $c^-(t)$
- what can flow through arbitrary cut
- what can flow through bottleneck: $c_{\min}$

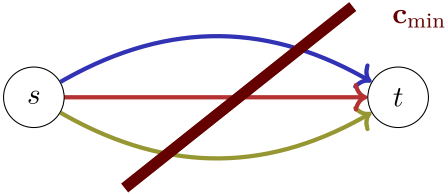




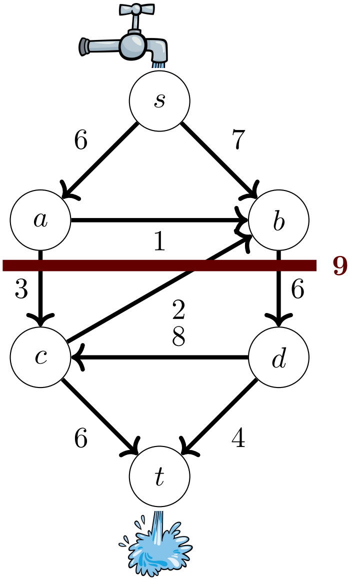
Flows and Cuts: Bottleneck Intuition

Upper bounds on size of flow:

- what can flow out of s: $c^+(s)$
- what can flow into t: $c^-(t)$
- what can flow through arbitrary cut
- what can flow through bottleneck: $c_{\min}$



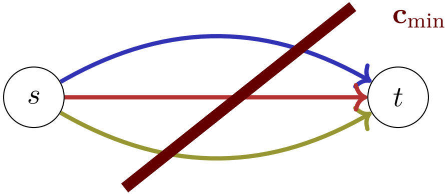
$\Rightarrow$ flow $|f| \leq$ bottleneck



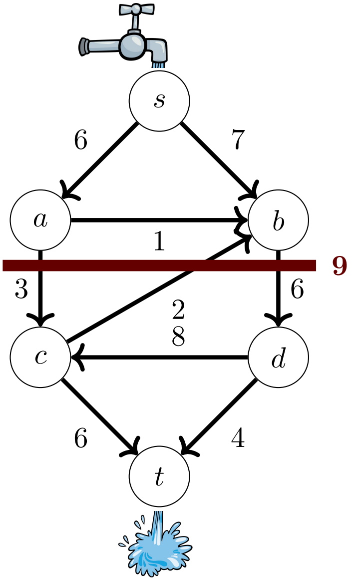
Flows and Cuts: Bottleneck Intuition

Upper bounds on size of flow:

- what can flow out of s: $c^+(s)$
- what can flow into t: $c^-(t)$
- what can flow through arbitrary cut
- what can flow through bottleneck: $c_{\min}$

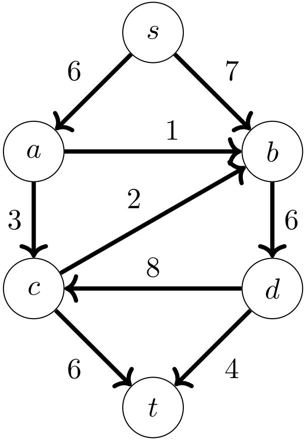


$\Rightarrow$ flow $|f| \leq$ bottleneck
$\Rightarrow$ maximum flow $\leq$ bottleneck



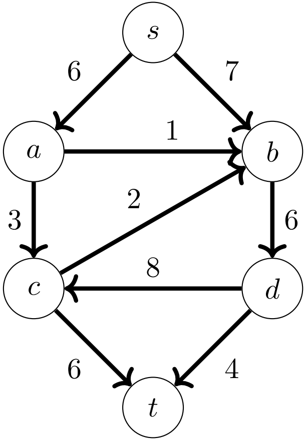
Cut

Cut



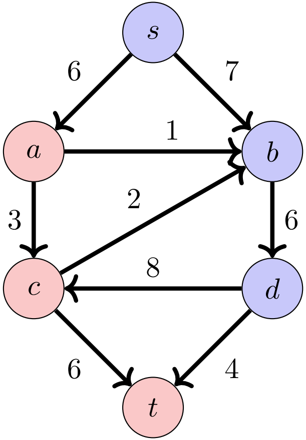
Cut

(s,t)**-Cut** of graph $G = (V, E, c)$:



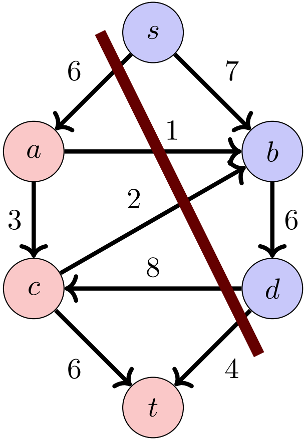
Cut

(s,t)**-Cut** of graph $G = (V, E, c)$: Partition $(\mathbf{S}, \mathbf{T})$ of V such that $s \in S, t \in T$



Cut

(s,t)**-Cut** of graph $G = (V, E, c)$: Partition $(\mathbf{S}, \mathbf{T})$ of V such that $s \in S, t \in T$

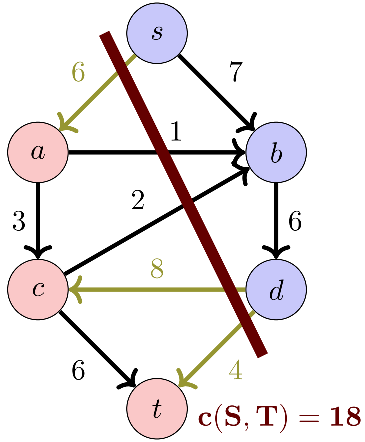


Cut

(s,t)**-Cut** of graph $G = (V, E, c)$: Partition $(\mathbf{S}, \mathbf{T})$ of V such that $s \in S, t \in T$

Size of cut:

$c(S,T) := \sum_{\mathbf{e}:\, \mathbf{S} \to \mathbf{T}} \mathbf{c}(\mathbf{e})$

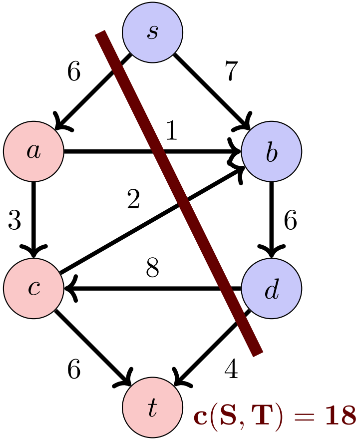


Cut

(s,t)**-Cut** of graph $G = (V, E, c)$: Partition $(\mathbf{S}, \mathbf{T})$ of V such that $s \in S, t \in T$

Size of cut:

$c(S,T) := \sum_{\mathbf{e:\, S \to T}} \mathbf{c(e)}$

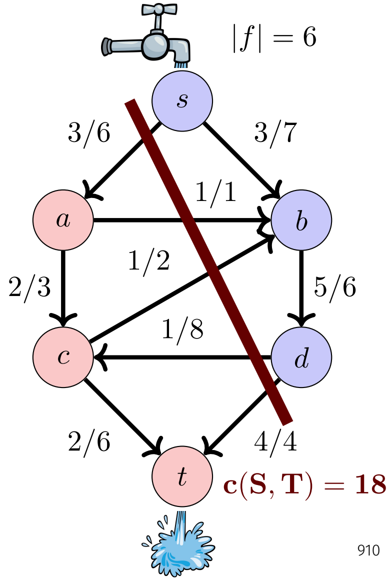


Cut

(s,t)**-Cut** of graph $G = (V, E, c)$: Partition $(\mathbf{S}, \mathbf{T})$ of V such that $s \in S, t \in T$

Size of cut:
$c(S,T) := \sum_{\mathbf{e}:\, \mathbf{S} \to \mathbf{T}} \mathbf{c}(\mathbf{e})$

Flow through cut of flow network:

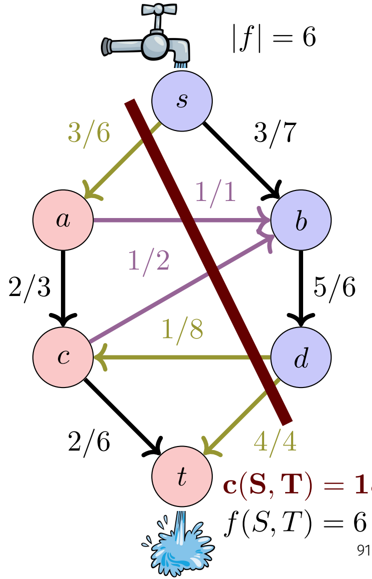


Cut

(s,t)**-Cut** of graph $G = (V, E, c)$: Partition $(\mathbf{S}, \mathbf{T})$ of V such that $s \in S, t \in T$

Size of cut:
$c(S,T) := \sum_{\mathbf{e}:\, \mathbf{S}\to\mathbf{T}} \mathbf{c}(\mathbf{e})$

Flow through cut of flow network:
$f(S,T) := \sum_{\mathbf{e}:\, \mathbf{S}\to\mathbf{T}} \mathbf{f}(\mathbf{e}) - \sum_{\mathbf{e}:\, \mathbf{T}\to\mathbf{S}} \mathbf{f}(\mathbf{e})$



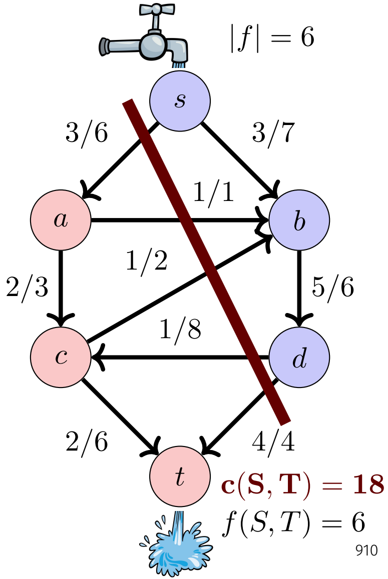
$\mathbf{c(S,T) = 18}$
$f(S,T) = 6$

Cut

(s,t)**-Cut** of graph $G = (V, E, c)$: Partition $(\mathbf{S}, \mathbf{T})$ of V such that $s \in S, t \in T$

Size of cut:
$c(S,T) := \sum_{\mathbf{e}:\, \mathbf{S} \to \mathbf{T}} \mathbf{c(e)}$

Flow through cut of flow network:
$f(S,T) := \sum_{\mathbf{e}:\, \mathbf{S} \to \mathbf{T}} \mathbf{f(e)} - \sum_{\mathbf{e}:\, \mathbf{T} \to \mathbf{S}} \mathbf{f(e)}$



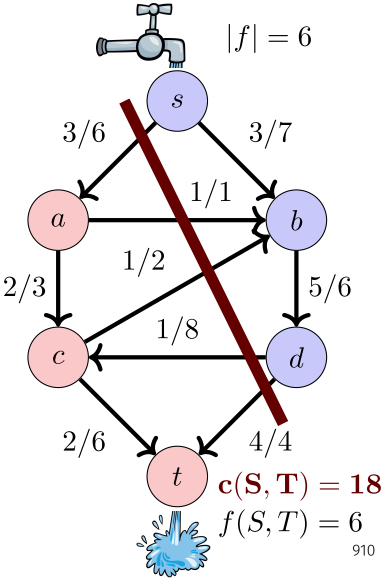
Cut

(s, t)**-Cut** of graph $G = (V, E, c)$: Partition $(\mathbf{S}, \mathbf{T})$ of V such that $s \in S, t \in T$

Size of cut:
$c(S, T) := \sum_{\mathbf{e:\, S \to T}} \mathbf{c(e)}$

Flow through cut of flow network:
$f(S, T) := \sum_{\mathbf{e:\, S \to T}} \mathbf{f(e)} - \sum_{\mathbf{e:\, T \to S}} \mathbf{f(e)}$

Observation:
$\forall f, S, T\colon |f| = f(S, T) \leq c(S, T)$



Cut

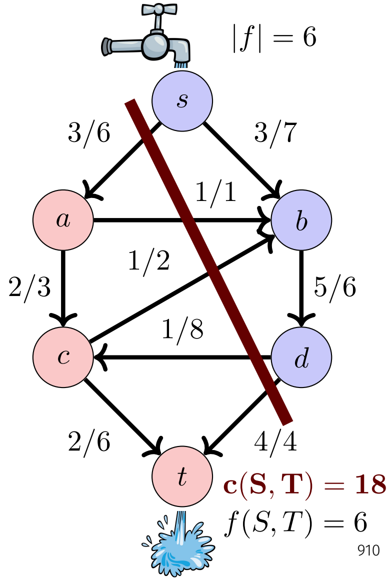
(s,t)**-Cut** of graph $G = (V, E, c)$: Partition $(\mathbf{S}, \mathbf{T})$ of V such that $s \in S, t \in T$

Size of cut:
$c(S,T) := \sum_{\mathbf{e}:\, \mathbf{S} \to \mathbf{T}} \mathbf{c}(\mathbf{e})$

Flow through cut of flow network:
$f(S,T) := \sum_{\mathbf{e}:\, \mathbf{S} \to \mathbf{T}} \mathbf{f}(\mathbf{e}) - \sum_{\mathbf{e}:\, \mathbf{T} \to \mathbf{S}} \mathbf{f}(\mathbf{e})$

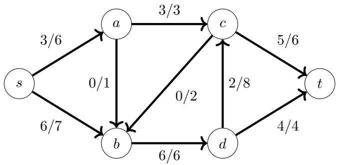
Observation:
$\forall f, S, T\colon |f| = f(S,T) \leq c(S,T)$

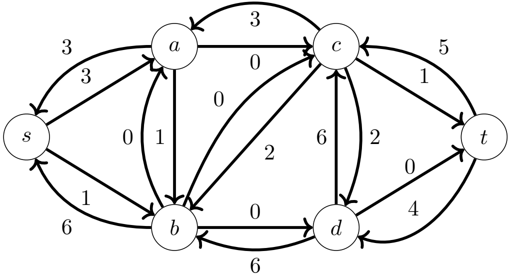
$\Rightarrow |f_{\max}| \leq c_{\min}$



Maximum Flow and Minimum Cut

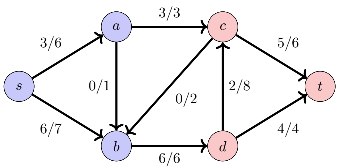
Maximum Flow and Minimum Cut

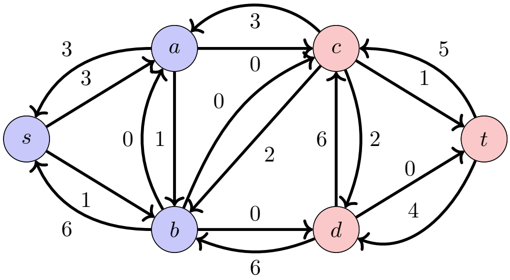




after termination of Ford-Fulkerson/Edmonds-Karp:

Maximum Flow and Minimum Cut

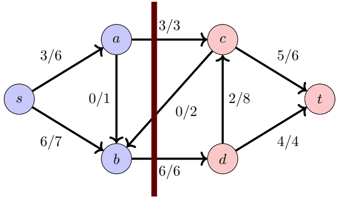


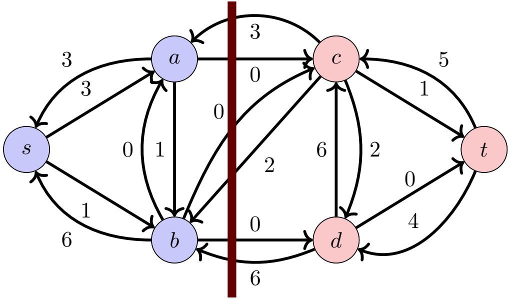


after termination of Ford-Fulkerson/Edmonds-Karp:

- reachable from s, $\mathbf{T} \subseteq \mathbf{V}$ nodes not reachable from s

Maximum Flow and Minimum Cut

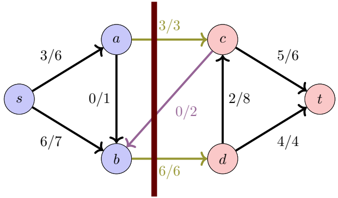


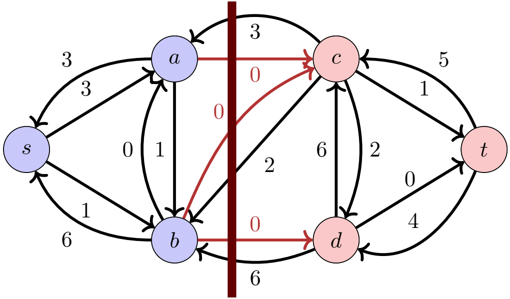


after termination of Ford-Fulkerson/Edmonds-Karp:

- reachable from s, $\mathbf{T} \subseteq \mathbf{V}$ nodes not reachable from s $\Rightarrow$ **Cut** $(\mathbf{S}, \mathbf{T})$

Maximum Flow and Minimum Cut

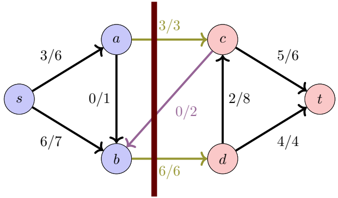


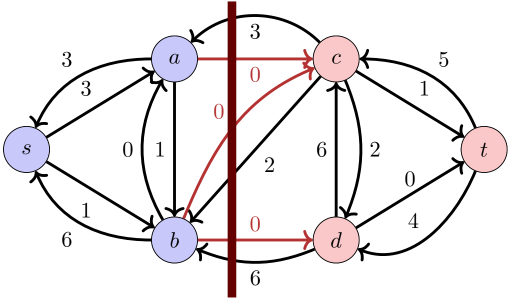


after termination of Ford-Fulkerson/Edmonds-Karp:

- reachable from s, $\mathbf{T} \subseteq \mathbf{V}$ nodes not reachable from $s \Rightarrow$ **Cut** $(\mathbf{S}, \mathbf{T})$
- all outgoing edges e have remaining capacity 0 in G_f

Maximum Flow and Minimum Cut

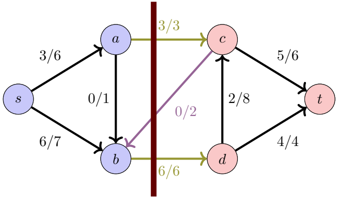


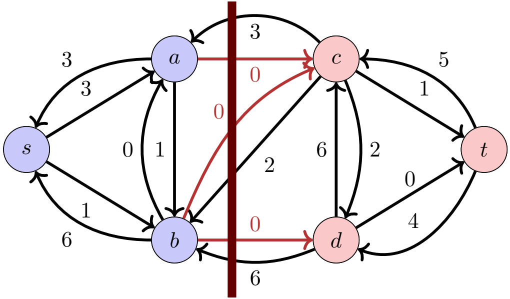


after termination of Ford-Fulkerson/Edmonds-Karp:

- reachable from s, $\mathbf{T} \subseteq \mathbf{V}$ nodes not reachable from $s \Rightarrow$ **Cut** $(\mathbf{S}, \mathbf{T})$
- all outgoing edges e have remaining capacity 0 in G_f
- $f(S,T) = \sum_{\mathbf{e}:\, \mathbf{S} \to \mathbf{T}} \mathbf{f}(\mathbf{e}) - \sum_{\mathbf{e}:\, \mathbf{T} \to \mathbf{S}} \mathbf{f}(\mathbf{e})$

Maximum Flow and Minimum Cut

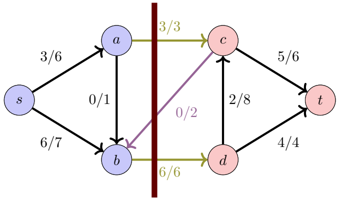


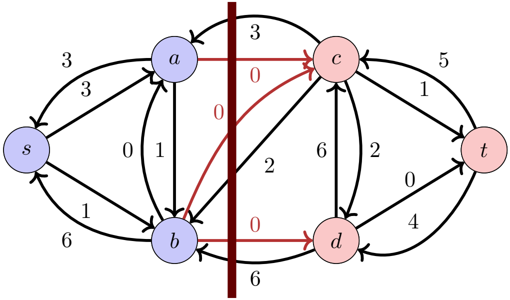


after termination of Ford-Fulkerson/Edmonds-Karp:

- reachable from s, $\mathbf{T} \subseteq \mathbf{V}$ nodes not reachable from $s \Rightarrow$ **Cut** $(\mathbf{S}, \mathbf{T})$
- all outgoing edges e have remaining capacity 0 in G_f
- $f(S,T) = \sum_{\mathbf{e}:\, \mathbf{S} \to \mathbf{T}} \mathbf{f}(\mathbf{e}) - \sum_{\mathbf{e}:\, \mathbf{T} \to \mathbf{S}} \mathbf{f}(\mathbf{e}) = \sum_{\mathbf{e}:\, \mathbf{S} \to \mathbf{T}} \mathbf{c}(\mathbf{e})$

Maximum Flow and Minimum Cut

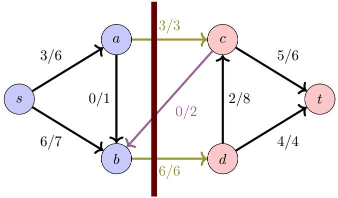


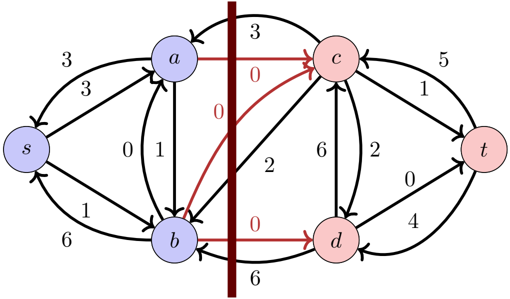


after termination of Ford-Fulkerson/Edmonds-Karp:

- reachable from s, $\mathbf{T} \subseteq \mathbf{V}$ nodes not reachable from $s \Rightarrow$ **Cut** $(\mathbf{S}, \mathbf{T})$
- all outgoing edges e have remaining capacity 0 in G_f
- $f(S,T) = \sum_{\mathbf{e:\, S \to T}} \mathbf{f}(\mathbf{e}) - \sum_{\mathbf{e:\, T \to S}} \mathbf{f}(\mathbf{e}) = \sum_{\mathbf{e:\, S \to T}} \mathbf{c}(\mathbf{e}) = \mathbf{c}(\mathbf{S}, \mathbf{T})$

Maximum Flow and Minimum Cut





after termination of Ford-Fulkerson/Edmonds-Karp:

- reachable from s, $\mathbf{T} \subseteq \mathbf{V}$ nodes not reachable from $s \Rightarrow$ **Cut** $(\mathbf{S}, \mathbf{T})$
- all outgoing edges e have remaining capacity 0 in G_f
- $f(S,T) = \sum_{\mathbf{e}:\,\mathbf{S}\to\mathbf{T}} \mathbf{f}(\mathbf{e}) - \sum_{\mathbf{e}:\,\mathbf{T}\to\mathbf{S}} \mathbf{f}(\mathbf{e}) = \sum_{\mathbf{e}:\,\mathbf{S}\to\mathbf{T}} \mathbf{c}(\mathbf{e}) = \mathbf{c}(\mathbf{S}, \mathbf{T})$
  $\Rightarrow |f_{\max}| = c_{\min}$

Max-Flow Min-Cut Theorem

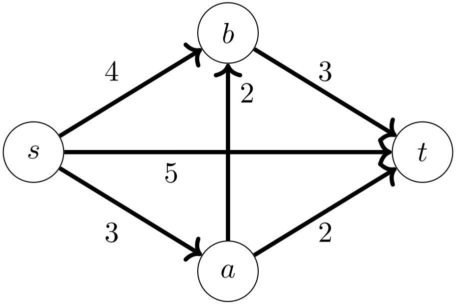
Max-Flow Min-Cut Theorem

For a flow f in a flow network $G = (V, E, c)$ with source s and sink t, the following statements are equivalent:

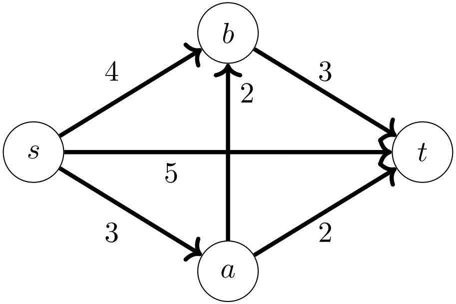
1. f is a maximum flow in G
2. The residual network G_f does not provide any augmenting paths
3. $|f| = c(S, T)$ for a cut (S, T) of G.

Quiz

Quiz

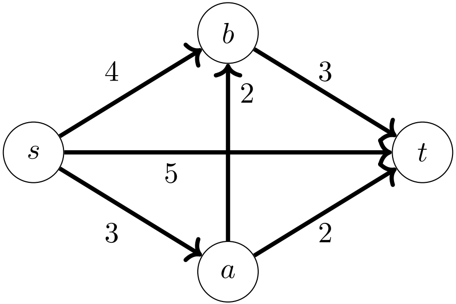


Quiz



What is the minimum cut?

Quiz



What is the minimum cut?

What is the maximum flow?

Application Examples

Application Examples

- Maximum Rate:
  - water in sewage system
  - cars in traffic

Application Examples

- Maximum Rate:
  - water in sewage system
  - cars in traffic
  - current in electrical networks

Application Examples

- Maximum Rate:
  - water in sewage system
  - cars in traffic
  - current in electrical networks
  - components on conveyors

Application Examples

- Maximum Rate:
  - water in sewage system
  - cars in traffic
  - current in electrical networks
  - components on conveyors
  - information flow in communication networks

Application Examples

- Maximum Rate:
  - water in sewage system
  - cars in traffic
  - current in electrical networks
  - components on conveyors
  - information flow in communication networks
- Scheduling

Application Examples

- Maximum Rate:
  - water in sewage system
  - cars in traffic
  - current in electrical networks
  - components on conveyors
  - information flow in communication networks
- Scheduling
- Bipartite Matching

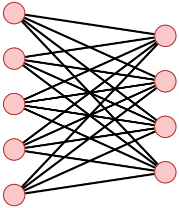
Application Examples

- Maximum Rate:
  - water in sewage system
  - cars in traffic
  - current in electrical networks
  - components on conveyors
  - information flow in communication networks
- Scheduling
- Bipartite Matching
- Image Segmentation

29.4 Maximales Bipartites Matching

Notation

A graph where V can be partitioned into disjoint sets U and W such that each $e \in E$ provides a node in U and a node in W is called **bipartite**.

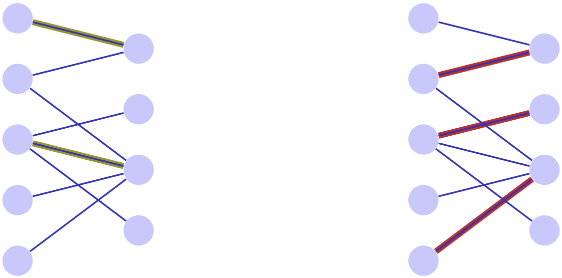


Application: maximal bipartite matching

Given: bipartite undirected graph $G = (V, E)$.

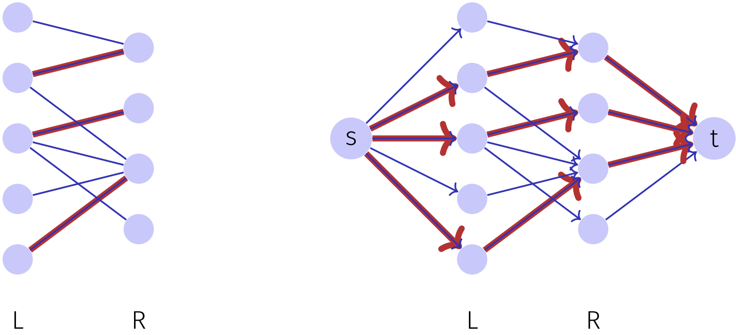
Matching M: $M \subseteq E$ such that $|\{m \in M : v \in m\}| \leq 1$ for all $v \in V$.

Maximal Matching M: Matching M, such that $|M| \geq |M'|$ for each matching M'.



Corresponding flow network

Construct a flow network that corresponds to the partition L, R of a bipartite graph with source s and sink t, with directed edges from s to L, from L to R and from R to t. Each edge has capacity 1.



Summary

Summary

- Definitions: flow networks, flow, cut

Summary

- Definitions: flow networks, flow, cut
- Concepts: Redirection, remainder network, augmenting path

Summary

- Definitions: flow networks, flow, cut
- Concepts: Redirection, remainder network, augmenting path
- Algorithms

Summary

- Definitions: flow networks, flow, cut
- Concepts: Redirection, remainder network, augmenting path
- Algorithms
  - Greedy: incorrect!

Summary

- Definitions: flow networks, flow, cut
- Concepts: Redirection, remainder network, augmenting path
- Algorithms
  - Greedy: incorrect!
  - Ford-Fulkerson: $\mathcal{O}(|f_{\max}| \cdot |E|)$
    Greedy augmenting paths in remainder network

Summary

- Definitions: flow networks, flow, cut
- Concepts: Redirection, remainder network, augmenting path
- Algorithms
  - Greedy: incorrect!
  - Ford-Fulkerson: $\mathcal{O}(|f_{\max}| \cdot |E|)$
    Greedy augmenting paths in remainder network
  - Edmonds-Karp: $\mathcal{O}(|V| \cdot |E|^2)$
    Ford-Fulkerson with shortest augmenting paths (number of edges)

Summary

- Definitions: flow networks, flow, cut
- Concepts: Redirection, remainder network, augmenting path
- Algorithms
  - Greedy: incorrect!
  - Ford-Fulkerson: $\mathcal{O}(|f_{\max}| \cdot |E|)$
    Greedy augmenting paths in remainder network
  - Edmonds-Karp: $\mathcal{O}(|V| \cdot |E|^2)$
    Ford-Fulkerson with shortest augmenting paths (number of edges)
- Max Flow = Min Cut

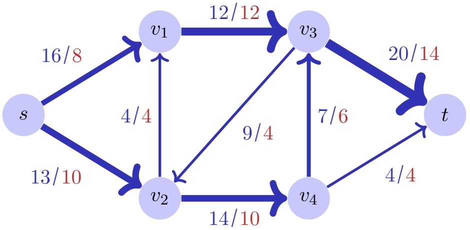
29.5 Appendix: Some Formal Things

Flow: Formulation with Skew Symmetry

A **Flow** $f : V \times V \to \mathbb{R}$ fulfills the following conditions:

- **Bounded Capacity**:
  For all $u, v \in V$: $f(u, v) \leq c(u, v)$.
- **Skew Symmetry**:
  For all $u, v \in V$: $f(u, v) = -f(v, u)$.
- **Conservation of flow**:
  For all $u \in V \setminus \{s, t\}$:

$$\sum_{v \in V} f(u, v) = 0.$$



Value of the flow:
$|f| = \sum_{v \in V} f(s, v)$.
Here $|f| = 18$.

Cuts

- **Capacity** of an (s,t)–-cut: $c(S,T) = \sum_{v \in S, v' \in T} c(v,v')$
- **Minimal cut**: cut with minimal capacity.
- **Flow over the cut**: $f(S,T) = \sum_{v \in S, v' \in T} f(v,v')$

Generally: Let $U, U' \subseteq V$

$$f(U,U') := \sum_{\substack{u \in U \\ u' \in U'}} f(u,u'), \qquad f(u,U') := f(\{u\},U')$$

Then

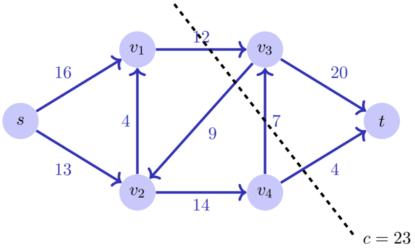
- $|f| = f(s,V)$
- $f(U,U) = 0$
- $f(U,U') = -f(U',U)$
- $f(X \cup Y, Z) = f(X,Z) + f(Y,Z)$, if $X \cap Y = \emptyset$.
- $f(R,V) = 0$ if $R \cap \{s,t\} = \emptyset$. [flow conversation!]

How large can a flow possibly be?

$$f(S,T) = f(S,V) - \underbrace{f(S,S)}_{0} = f(S,V)$$

$$= f(s,V) + f(\underbrace{S - \{s\}}_{\not\ni t, \not\ni s}, V) = |f|.$$

$$\Rightarrow |f| \leq \sum_{v \in S, v' \in T} c(v,v') = c(S,T)$$



Rest Network

Rest network G_f provided by the edges with positive rest capacity:

$$
\begin{aligned}
G_f &:= (V, E_f, c_f) \\
c_f(u,v) &:= c(u,v) - f(u,v) \quad \forall u,v \in V \\
E_f &:= \{(u,v) \in V \times V | c_f(u,v) > 0\}
\end{aligned}
$$

- Increase of the flow along some edge possible, when flow can be increased along the edge,i.e. if $f(u,v) < c(u,v)$. Rest capacity $c_f(u,v) = c(u,v) - f(u,v) > 0$.
- Increase of flow **against the direction** of the edge possible, if flow can be reduced along the edge, i.e. if $f(u,v) > 0$. Rest capacity $c_f(v,u) = f(u,v) > 0$.

The increased flow is a flow

Theorem 32

Let $G = (V, E, c)$ be a flow network with source s and sink t and f a flow in G. Let G_f be the corresponding rest networks and let f' be a flow in G_f. Then $f \oplus f'$ with

$$(f \oplus f')(u, v) = f(u, v) + f'(u, v)$$

defines a flow in G with value $|f| + |f'|$.

Proof

$f \oplus f'$ defines a flow in G:

- capacity limit

$$(f \oplus f')(u,v) = f(u,v) + \underbrace{f'(u,v)}_{\leq c(u,v) - f(u,v)} \leq c(u,v)$$

- skew symmetry

$$(f \oplus f')(u,v) = -f(v,u) + -f'(v,u) = -(f \oplus f')(v,u)$$

- flow conservation $u \in V - \{s,t\}$:

$$\sum_{v \in V} (f \oplus f')(u,v) = \sum_{v \in V} f(u,v) + \sum_{v \in V} f'(u,v) = 0$$

Proof

Value of $f \oplus f'$

$$\begin{aligned}
|f \oplus f'| &= (f \oplus f')(s, V) \\
&= \sum_{u \in V} f(s, u) + f'(s, u) \\
&= f(s, V) + f'(s, V) \\
&= |f| + |f'|
\end{aligned}$$

$\blacksquare$

Augmenting Paths

expansion path p: simple path from s to t in the rest network G_f.

Rest capacity $c_f(p) = \min\{c_f(u,v) : (u,v) \text{ edge in } p\}$

Theorem 33

The mapping $f_p : V \times V \to \mathbb{R}$,

$$f_p(u,v) = \begin{cases} c_f(p) & \text{if } (u,v) \text{ edge in } p \\ -c_f(p) & \text{if } (v,u) \text{ edge in } p \\ 0 & \text{otherwise} \end{cases}$$

provides a flow in G_f *with value* $|f_p| = c_f(p) > 0$.

f_p is a flow (easy to show). there is one and only one $u \in V$ with $(s,u) \in p$.
Thus $|f_p| = \sum_{v \in V} f_p(s,v) = f_p(s,u) = c_f(p)$.

Max-Flow Min-Cut Theorem

Theorem 34

Let f be a flow in a flow network $G = (V, E, c)$ with source s and sink t. The following statementsa are equivalent:

1. *f is a maximal flow in G*
2. *The rest network G_f does not provide any expansion paths*
3. *It holds that $|f| = c(S, T)$ for a cut (S, T) of G.*

Proof

- $(3) \Rightarrow (1)$:
  It holds that $|f| \leq c(S, T)$ for all cuts S, T. From $|f| = c(S, T)$ it follows that $|f|$ is maximal.
- $(1) \Rightarrow (2)$:
  f maximal Flow in G. Assumption: G_f has some expansion path $|f \oplus f_p| = |f| + |f_p| > |f|$. Contradiction.

Proof $(2) \Rightarrow (3)$

Assumption: G_f has no expansion path

Define $S = \{v \in V : \text{ there is a path } s \rightsquigarrow v \text{ in } G_f\}$.

$(S, T) := (S, V \setminus S)$ is a cut: $s \in S, t \in T$.

Let $u \in S$ and $v \in T$. Then $c_f(u, v) = 0$, also $c_f(u, v) = c(u, v) - f(u, v) = 0$.

Somit $f(u, v) = c(u, v)$.

Thus

$$|f| = f(S, T) = \sum_{u \in S} \sum_{v \in T} f(u, v) = \sum_{u \in S} \sum_{v \in T} c(u, v) = C(S, T).$$

■

Edmonds-Karp Algorithm

Theorem 35

When the Edmonds-Karp algorithm is applied to some integer valued flow network $G = (V, E)$ with source s and sink t then the number of flow increases applied by the algorithm is in $\mathcal{O}(|V| \cdot |E|)$.

$\Rightarrow$ *Overal asymptotic runtime:* $\mathcal{O}(|V| \cdot |E|^2)$

[Without proof]

Edmonds-Karp Algorithmus

Theorem 36

Wenn der Edmonds-Karp Algorithmus auf Flussnetzwerk $G = (V, E)$ *mit Quelle* s *und Senke* t *angewendet wird, dann wächst für jeden Knoten* $v \in V \setminus \{s, t\}$ *die Distanz* $\delta_f(s, v)$ *des kürzesten Pfades von* s *nach* v *im Restnetzwerk* G_f *monoton mit jeder Flusserhöhung.*

Beweis

Annahme: Distanz $\delta_f(s,v)$ wird bei Flusserhöhung $f \to f'$ kleiner für ein v:
$\delta_f(s,v) < \delta_{f'}(s,v)$
Sei $p = s \rightsquigarrow u \to v$ kürzester Pfad von s nach v in $G_{f'}$, so dass $(u,v) \in E_{f'}$ und $\delta_{f'}(s,u) = \delta_{f'}(s,v) - 1$. Es gilt $\delta_{f'}(s,u) \geq \delta_f(s,u)$.
Wenn $(u,v) \in E_f$: $\delta_f s, v \leq \delta_f(s,u) + 1 \leq \delta_{f'}(s,u) + 1 = \delta_{f'}(s,v)$ Widerspruch.
Also $(u,v) \notin E_f$.

Integer number theorem

Theorem 37

If the capacities of a flow network are integers, then the maximal flow generated by the Ford-Fulkerson method provides integer numbers for each $f(u,v)$, $u,v \in V$.

[without proof]

Consequence: Ford-Fulkerson generates for a flow network that corresponds to a bipartite graph a maximal matching $M = \{(u,v) : f(u,v) = 1\}$.