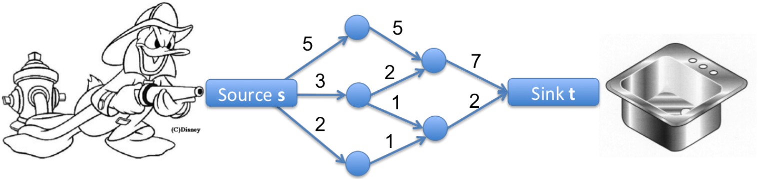
Algorithms: FLOWS AND CUTS

Alessandro Chiesa, Ola Svensson

School of Computer and Communication Sciences

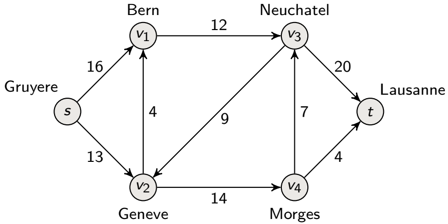
Lecture 16, 15.04.2025



FLOW NETWORKS

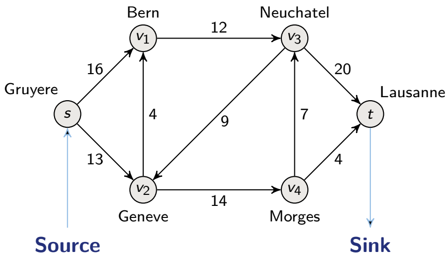
Flow Network

Transfer as much cheese as possible from Gruyere to Lausanne



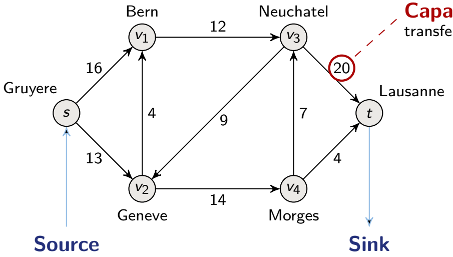
Flow Network

Transfer as much cheese as possible from Gruyere to Lausanne



Flow Network

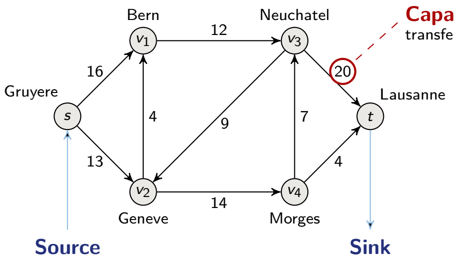
Transfer as much cheese as possible from Gruyere to Lausanne



Capacity: at most 20 cheeses can be transferred from Neuchatel to Lausanne

Flow Network

Transfer as much cheese as possible from Gruyere to Lausanne

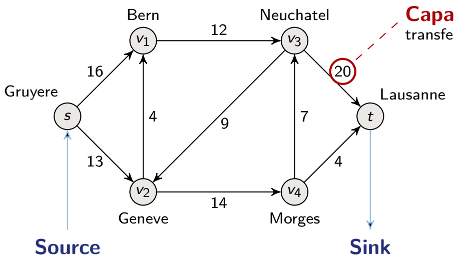


Capacity: at most 20 cheeses can be transferred from Neuchatel to Lausanne

- a graph to model flow through edges (pipes)

Flow Network

Transfer as much cheese as possible from Gruyere to Lausanne

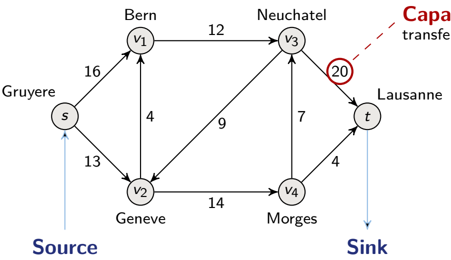


Capacity: at most 20 cheeses can be transferred from Neuchatel to Lausanne

- a graph to model flow through edges (pipes)
- each edge has a capacity an upper bound on the flow rate (pipes have different sizes)

Flow Network

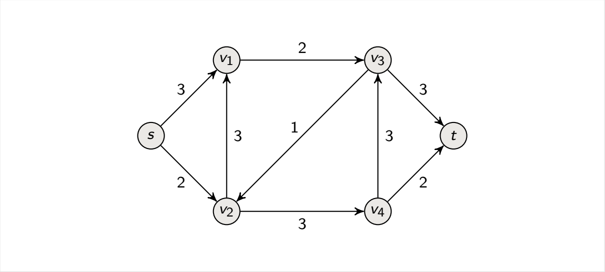
Transfer as much cheese as possible from Gruyere to Lausanne



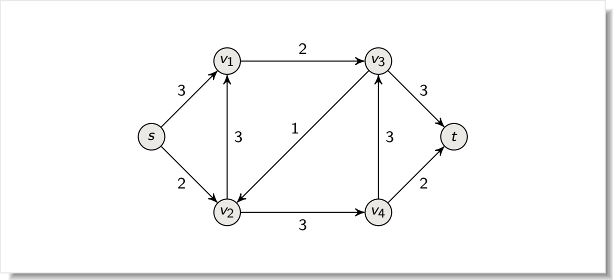
Capacity: at most 20 cheeses can be transferred from Neuchatel to Lausanne

- a graph to model flow through edges (pipes)
- each edge has a capacity an upper bound on the flow rate (pipes have different sizes)
- Want to maximize rate of flow from the source to the sink

Flow Network (formally)

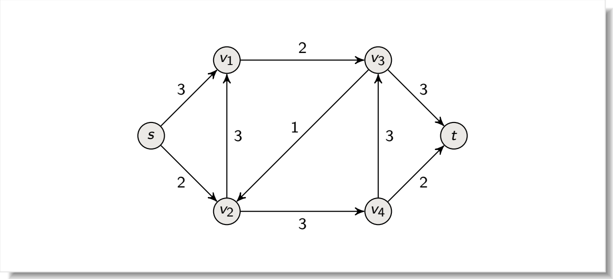


Flow Network (formally)



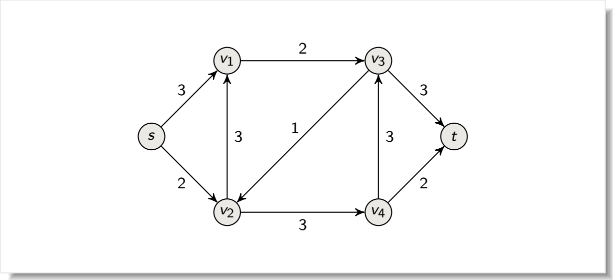
- Directed graph $G = (V, E)$

Flow Network (formally)



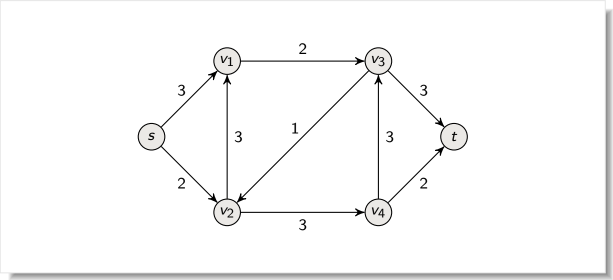
- Directed graph $G = (V, E)$
- Each edge (u, v) has a capacity $c(u, v) \geq 0$ ($c(u, v) = 0$ if $(u, v) \notin E$)

Flow Network (formally)



- Directed graph $G = (V, E)$
- Each edge (u, v) has a capacity $c(u, v) \geq 0$ $(c(u, v) = 0$ if $(u, v) \notin E)$
- Source s and sink t (flow goes from s to t)

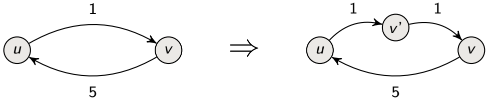
Flow Network (formally)



- Directed graph $G = (V, E)$
- Each edge (u, v) has a capacity $c(u, v) \geq 0$ ($c(u, v) = 0$ if $(u, v) \notin E$)
- Source s and sink t (flow goes from s to t)
- No antiparallel edges (assumed w.l.o.g. for simplicity)

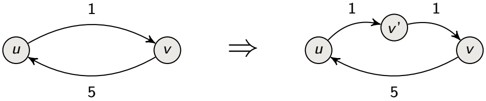
Why is “no antiparallel edges” w.l.o.g.?

Why is "no antiparallel edges" w.l.o.g.?



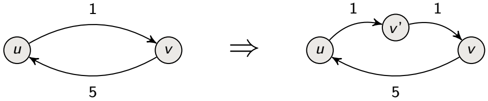
- If there are two parallel edges (u, v) and (v, u), choose one of them say (u, v)

Why is “no antiparallel edges” w.l.o.g.?



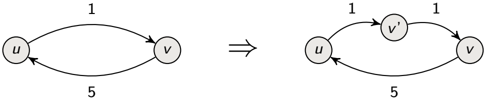
- If there are two parallel edges (u, v) and (v, u), choose one of them say (u, v)
- Create a new vertex v'

Why is “no antiparallel edges” w.l.o.g.?



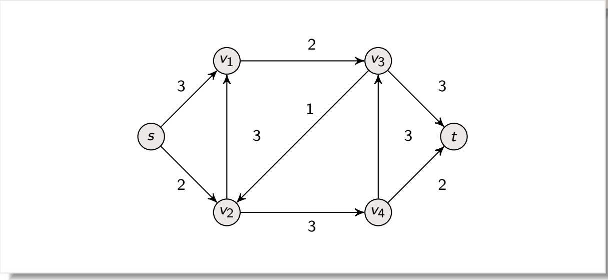
- If there are two parallel edges (u, v) and (v, u), choose one of them say (u, v)
- Create a new vertex v'
- Replace (u, v) by two new edges (u, v') and (v', v) with $c(u, v') = c(v', u) = c(u, v)$

Why is “no antiparallel edges” w.l.o.g.?



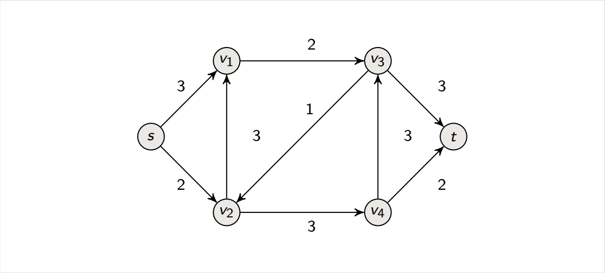
- If there are two parallel edges (u, v) and (v, u), choose one of them say (u, v)
- Create a new vertex v'
- Replace (u, v) by two new edges (u, v') and (v', v) with $c(u, v') = c(v', u) = c(u, v)$
- Repeat this $O(E)$ times to get an equivalent flow network with no antiparallel edges.

Definition of a flow



A flow is a function $f : V \times V \to \mathbb{R}$ satisfying:

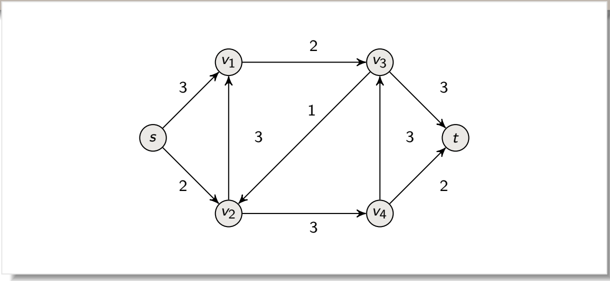
Definition of a flow



A flow is a function $f : V \times V \to \mathbb{R}$ satisfying:

Capacity constraint: For all $u, v \in V$: $0 \leq f(u,v) \leq c(u,v)$

Definition of a flow



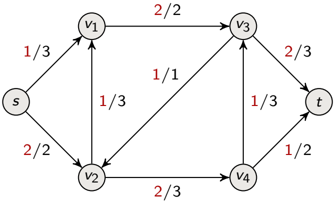
A flow is a function $f : V \times V \to \mathbb{R}$ satisfying:

Capacity constraint: For all $u, v \in V : 0 \leq f(u, v) \leq c(u, v)$

Flow conservation: For all $u \in V \setminus \{s, t\}$,

$$\underbrace{\sum_{v \in V} f(v, u)}_{\text{flow into } u} = \underbrace{\sum_{v \in V} f(u, v)}_{\text{flow out of } u}$$

Definition of a flow



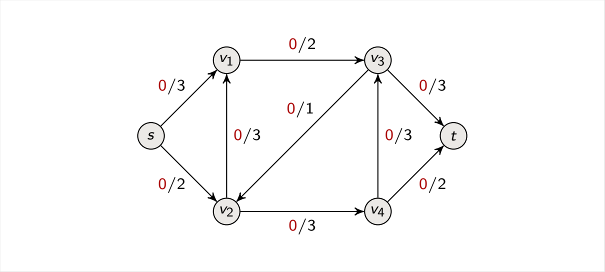
A flow is a function $f : V \times V \to \mathbb{R}$ satisfying:

Capacity constraint: For all $u, v \in V : 0 \leq f(u, v) \leq c(u, v)$

Flow conservation: For all $u \in V \setminus \{s, t\}$,

$$\underbrace{\sum_{v \in V} f(v, u)}_{\text{flow into } u} = \underbrace{\sum_{v \in V} f(u, v)}_{\text{flow out of } u}$$

Definition of a flow



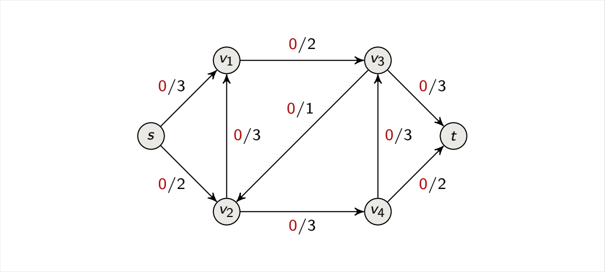
A flow is a function $f : V \times V \to \mathbb{R}$ satisfying:

Capacity constraint: For all $u, v \in V : 0 \leq f(u, v) \leq c(u, v)$

Flow conservation: For all $u \in V \setminus \{s, t\}$,

$$\underbrace{\sum_{v \in V} f(v, u)}_{\text{flow into } u} = \underbrace{\sum_{v \in V} f(u, v)}_{\text{flow out of } u}$$

Value of a flow

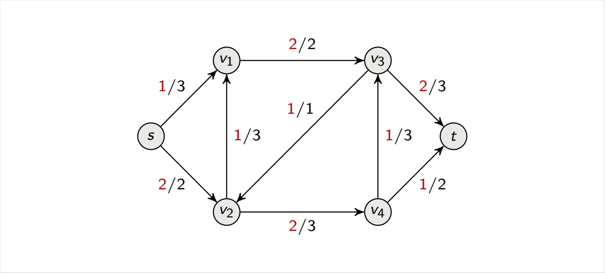


Value of a flow $f = |f|$

$$= \sum_{v \in V} f(s, v) - \sum_{v \in V} f(v, s)$$

$=$ flow out of source $-$ flow into source

Value of a flow

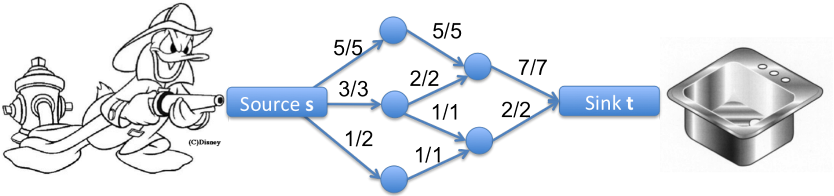


Value of a flow $f = |f|$

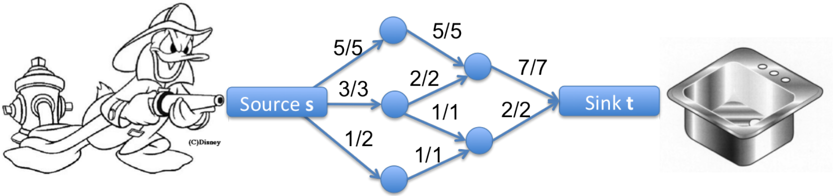
$$= \sum_{v \in V} f(s,v) - \sum_{v \in V} f(v,s)$$

$=$ flow out of source $-$ flow into source

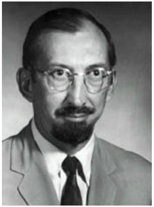
What’s the value of this flow?



What’s the value of this flow? 9



L. R. Ford, Jr. (1927-)



D, R, Fulkerson (1924-1976)

MAXIMUM-FLOW PROBLEM

Ford-Fulkerson Method

The Ford-Fulkerson Method'54

```
FORD-FULKERSON-METHOD($G, s, t$):

Initialize flow $f$ to 0
while exists an augmenting path $p$ in the residual network $G_f$
    augment flow $f$ along $p$
return $f$
```

The Ford-Fulkerson Method'54

```
FORD-FULKERSON-METHOD($G, s, t$):

1. Initialize flow $f$ to 0
2. while exists an augmenting path $p$ in the residual network $G_f$
3.         augment flow $f$ along $p$
4. return $f$
```

The Ford-Fulkerson Method'54

```
FORD-FULKERSON-METHOD($G, s, t$):

Initialize flow $f$ to 0
while exists an augmenting path $p$ in the residual network $G_f$
    augment flow $f$ along $p$
return $f$
```

Basic idea:

The Ford-Fulkerson Method'54

```
FORD-FULKERSON-METHOD($G, s, t$):

Initialize flow $f$ to 0
while exists an augmenting path $p$ in the residual network $G_f$
    augment flow $f$ along $p$
return $f$
```

Basic idea:

- As long as there is a path from source to sink, with available capacity on all edges in the path

The Ford-Fulkerson Method'54

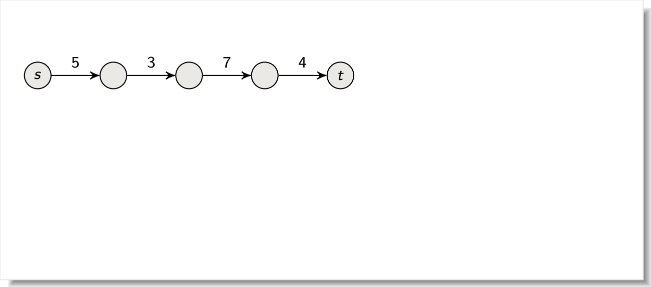
```
FORD-FULKERSON-METHOD($G, s, t$):

Initialize flow $f$ to 0
while exists an augmenting path $p$ in the residual network $G_f$
        augment flow $f$ along $p$
return $f$
```

Basic idea:

- As long as there is a path from source to sink, with available capacity on all edges in the path
- send flow along one of these paths and then we find another path and so on

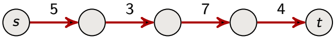
Applying the basic idea to examples

- As long as there is a path from source to sink, with available capacity on all edges in the path
- send flow along one of these paths and then we find another path and so on



Applying the basic idea to examples

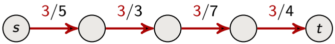
- As long as there is a path from source to sink, with available capacity on all edges in the path
- send flow along one of these paths and then we find another path and so on



Exists a path **p** from s to t with remaining capacity
$\Rightarrow$ Push flow on **p**

Applying the basic idea to examples

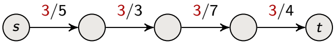
- As long as there is a path from source to sink, with available capacity on all edges in the path
- send flow along one of these paths and then we find another path and so on



Exists a path **p** from s to t with remaining capacity
$\Rightarrow$ Push flow on **p**

Applying the basic idea to examples

- As long as there is a path from source to sink, with available capacity on all edges in the path
- send flow along one of these paths and then we find another path and so on

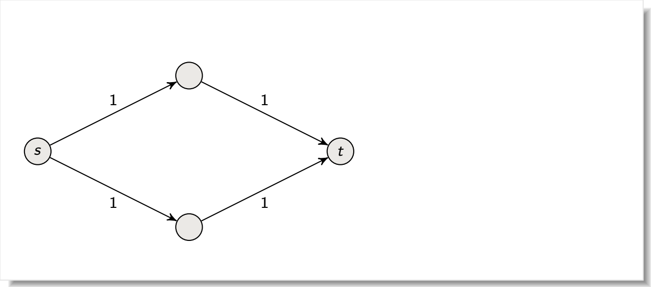


No path from s to t
with remaining capacity

and the flow is maximum

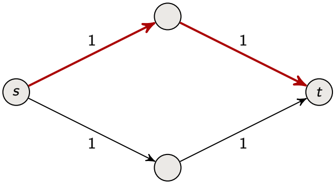
Applying the basic idea to examples

- As long as there is a path from source to sink, with available capacity on all edges in the path
- send flow along one of these paths and then we find another path and so on



Applying the basic idea to examples

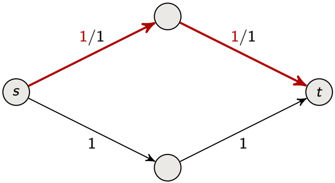
- As long as there is a path from source to sink, with available capacity on all edges in the path
- send flow along one of these paths and then we find another path and so on



Exists a path **p** from s to t with remaining capacity
$\Rightarrow$ Push flow on **p**

Applying the basic idea to examples

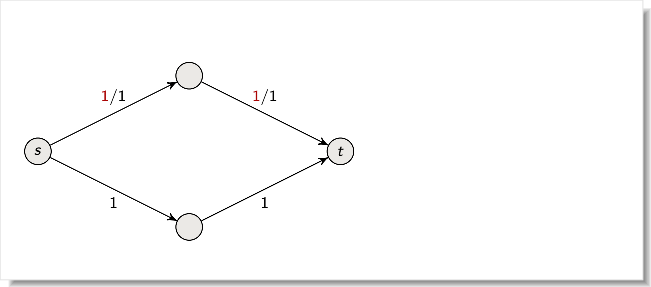
- As long as there is a path from source to sink, with available capacity on all edges in the path
- send flow along one of these paths and then we find another path and so on



Exists a path **p** from s to t with remaining capacity
$\Rightarrow$ Push flow on **p**

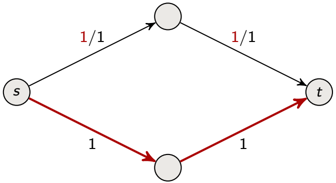
Applying the basic idea to examples

- As long as there is a path from source to sink, with available capacity on all edges in the path
- send flow along one of these paths and then we find another path and so on



Applying the basic idea to examples

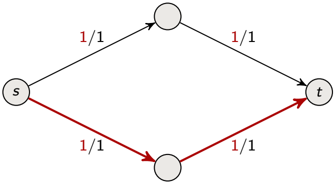
- As long as there is a path from source to sink, with available capacity on all edges in the path
- send flow along one of these paths and then we find another path and so on



Exists a path **p** from *s* to *t* with remaining capacity
$\Rightarrow$ Push flow on **p**

Applying the basic idea to examples

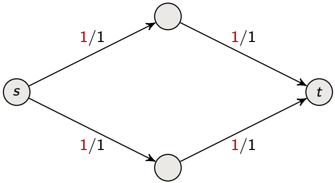
- As long as there is a path from source to sink, with available capacity on all edges in the path
- send flow along one of these paths and then we find another path and so on



Exists a path **p** from s to t with remaining capacity
$\Rightarrow$ Push flow on **p**

Applying the basic idea to examples

- As long as there is a path from source to sink, with available capacity on all edges in the path
- send flow along one of these paths and then we find another path and so on

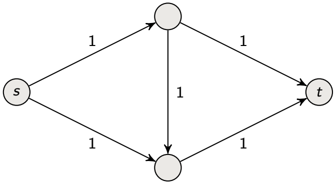


No path from s to t
with remaining capacity

and the flow is maximum

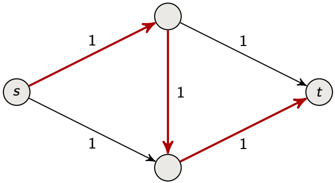
Applying the basic idea to examples

- As long as there is a path from source to sink, with available capacity on all edges in the path
- send flow along one of these paths and then we find another path and so on



Applying the basic idea to examples

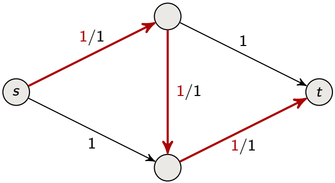
- As long as there is a path from source to sink, with available capacity on all edges in the path
- send flow along one of these paths and then we find another path and so on



Exists a path **p** from *s* to *t* with remaining capacity
⇒ Push flow on **p**

Applying the basic idea to examples

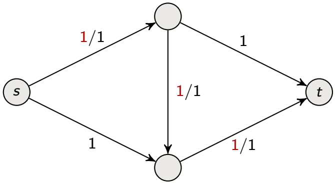
- As long as there is a path from source to sink, with available capacity on all edges in the path
- send flow along one of these paths and then we find another path and so on



Exists a path **p** from *s* to *t* with remaining capacity
⇒ Push flow on **p**

Applying the basic idea to examples

- As long as there is a path from source to sink, with available capacity on all edges in the path
- send flow along one of these paths and then we find another path and so on

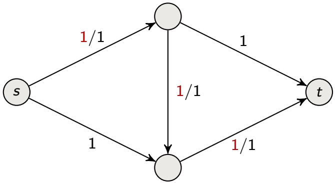


No path from s to t
with remaining capacity

but the flow is **not maximum**

Applying the basic idea to examples

- As long as there is a path from source to sink, with available capacity on all edges in the path
- send flow along one of these paths and then we find another path and so on



No path from *s* to *t*
with remaining capacity

but the flow is **not maximum**

☹

What went wrong? How can we fix it?

The Ford-Fulkerson Method'54

```
FORD-FULKERSON-METHOD($G, s, t$):

Initialize flow $f$ to 0
while exists an augmenting path $p$ in the residual network $G_f$
    augment flow $f$ along $p$
return $f$
```

Residual network

- Given a flow f and a network $G = (V, E)$
- the residual network consists of edges with capacities that represent how we can change the flow on the edges

Residual network

- Given a flow f and a network $G = (V, E)$
- the residual network consists of edges with capacities that represent how we can change the flow on the edges

Residual capacity:

$$c_f(u,v) = \begin{cases} c(u,v) - f(u,v) & \text{if } (u,v) \in E \\ f(v,u) & \text{if } (v,u) \in E \\ 0 & \text{otherwise} \end{cases}$$

Residual network

- Given a flow f and a network $G = (V, E)$
- the residual network consists of edges with capacities that represent how we can change the flow on the edges

Residual capacity:

$$c_f(u, v) = \begin{cases} c(u, v) - f(u, v) & \text{if } (u, v) \in E \\ f(v, u) & \text{if } (v, u) \in E \\ 0 & \text{otherwise} \end{cases}$$

Amount of capacity left

Amount of flow that can be reversed

Residual network

- Given a flow f and a network $G = (V, E)$
- the residual network consists of edges with capacities that represent how we can change the flow on the edges

Residual capacity:

$$c_f(u, v) = \begin{cases} c(u, v) - f(u, v) & \text{if } (u, v) \in E \longleftarrow \boxed{\text{Amount of capacity left}} \\ f(v, u) & \text{if } (v, u) \in E \longleftarrow \boxed{\text{Amount of flow that can be reversed}} \\ 0 & \text{otherwise} \end{cases}$$

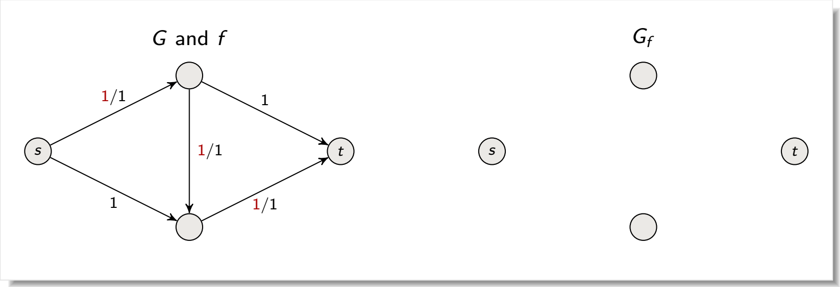
Residual network:

$G_f = (V, E_f)$ where $E_f = \{(u, v) \in V \times V : c_f(u, v) > 0\}$

Examples

Residual network: $G_f = (V, E_f)$ where $E_f = \{(u, v) \in V \times V : c_f(u, v) > 0\}$ and

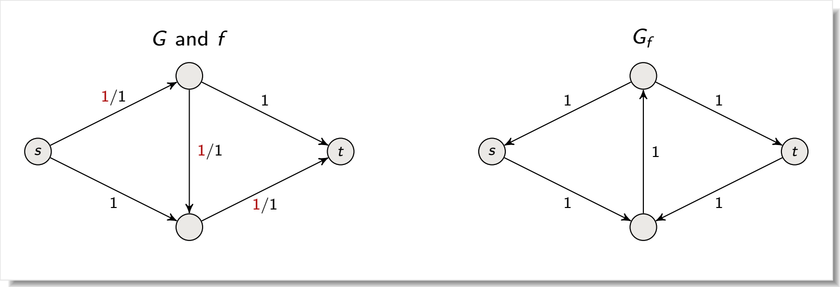
$$c_f(u, v) = \begin{cases} c(u, v) - f(u, v) & \text{if } (u, v) \in E \\ f(v, u) & \text{if } (v, u) \in E \\ 0 & \text{otherwise} \end{cases}$$



Examples

Residual network: $G_f = (V, E_f)$ where $E_f = \{(u,v) \in V \times V : c_f(u,v) > 0\}$ and

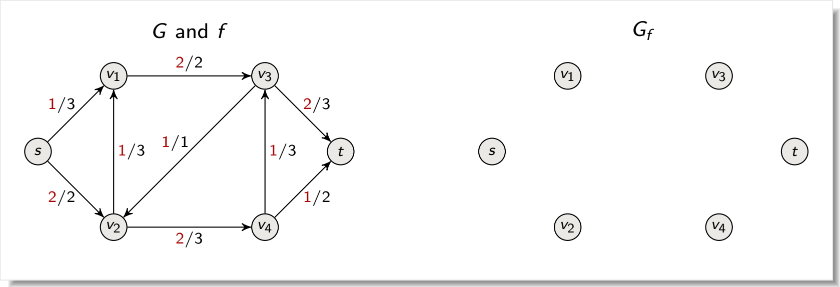
$$c_f(u,v) = \begin{cases} c(u,v) - f(u,v) & \text{if } (u,v) \in E \\ f(v,u) & \text{if } (v,u) \in E \\ 0 & \text{otherwise} \end{cases}$$



Examples

Residual network: $G_f = (V, E_f)$ where $E_f = \{(u,v) \in V \times V : c_f(u,v) > 0\}$ and

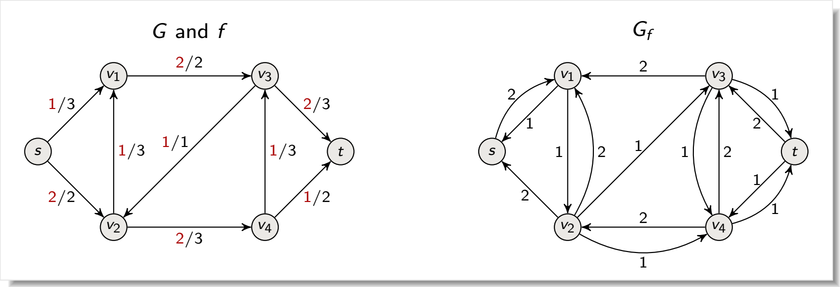
$$c_f(u,v) = \begin{cases} c(u,v) - f(u,v) & \text{if } (u,v) \in E \\ f(v,u) & \text{if } (v,u) \in E \\ 0 & \text{otherwise} \end{cases}$$



Examples

Residual network: $G_f = (V, E_f)$ where $E_f = \{(u,v) \in V \times V : c_f(u,v) > 0\}$ and

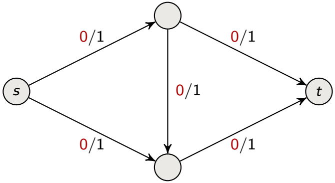
$$c_f(u,v) = \begin{cases} c(u,v) - f(u,v) & \text{if } (u,v) \in E \\ f(v,u) & \text{if } (v,u) \in E \\ 0 & \text{otherwise} \end{cases}$$



The Ford-Fulkerson Method'54

```
Ford-Fulkerson-Method($G, s, t$):

1. Initialize flow $f$ to 0
2.  while exists an augmenting path $p$ in the residual network $G_f$
3.         augment flow $f$ along $p$
4. return $f$
```

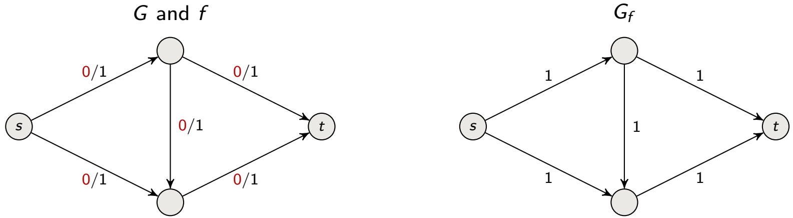
G and f



The Ford-Fulkerson Method'54

```
FORD-FULKERSON-METHOD($G, s, t$):

Initialize flow $f$ to 0
while exists an augmenting path $p$ in the residual network $G_f$
        augment flow $f$ along $p$
return $f$
```

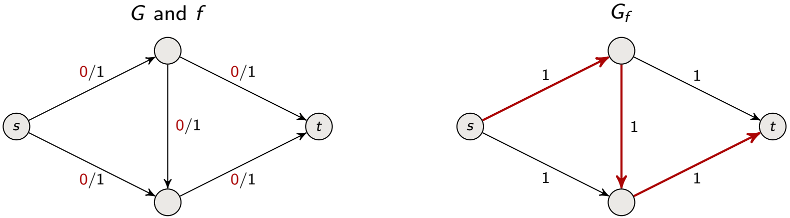
Augmenting path = simple path from s to t



The Ford-Fulkerson Method’54

```
FORD-FULKERSON-METHOD($G, s, t$):

1. Initialize flow $f$ to 0
2. while exists an augmenting path $p$ in the residual network $G_f$
3.         augment flow $f$ along $p$
4. return $f$
```

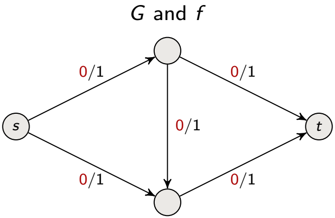
Exists augmenting path **p**

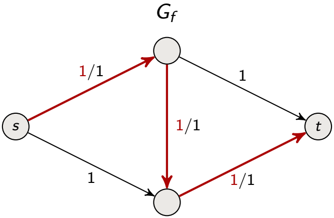


The Ford-Fulkerson Method'54

```
FORD-FULKERSON-METHOD($G, s, t$):

1. Initialize flow $f$ to 0
2. while exists an augmenting path $p$ in the residual network $G_f$
3.        augment flow $f$ along $p$
4. return $f$
```

Exists augmenting path **p** with flow $\mathbf{f_p}$ of value = min capacity on **p**

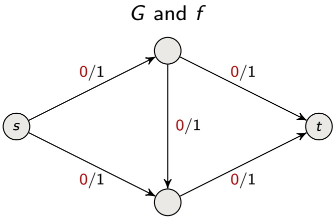


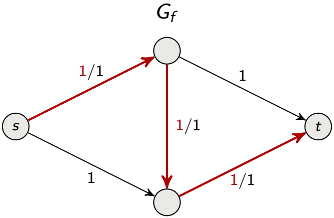


The Ford-Fulkerson Method'54

```
FORD-FULKERSON-METHOD($G, s, t$):

1. Initialize flow $f$ to 0
2.  while exists an augmenting path $p$ in the residual network $G_f$
3.      augment flow $f$ along $p$
4. return $f$
```

$\mathbf{f}$ is updated by changing the flow on an edge $(\mathbf{u}, \mathbf{v})$ by $\mathbf{f_p(u, v) - f_p(v, u)}$

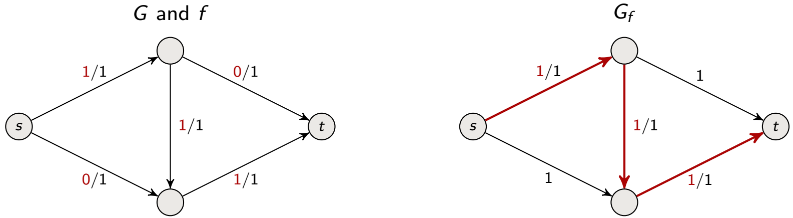




The Ford-Fulkerson Method'54

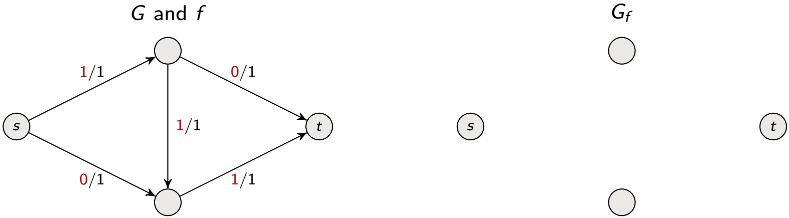
```
FORD-FULKERSON-METHOD($G, s, t$):

1. Initialize flow $f$ to 0
2.  while exists an augmenting path $p$ in the residual network $G_f$
3.      augment flow $f$ along $p$
4. return $f$
```

$\mathbf{f}$ is updated by changing the flow on an edge $(\mathbf{u},\mathbf{v})$ by $\mathbf{f_p(u,v) - f_p(v,u)}$



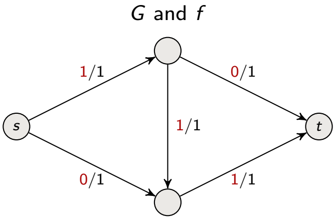
The Ford-Fulkerson Method'54

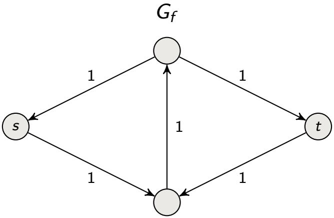
```
FORD-FULKERSON-METHOD($G, s, t$):

1. Initialize flow $f$ to 0
2. while exists an augmenting path $p$ in the residual network $G_f$
3.         augment flow $f$ along $p$
4. return $f$
```



The Ford-Fulkerson Method'54

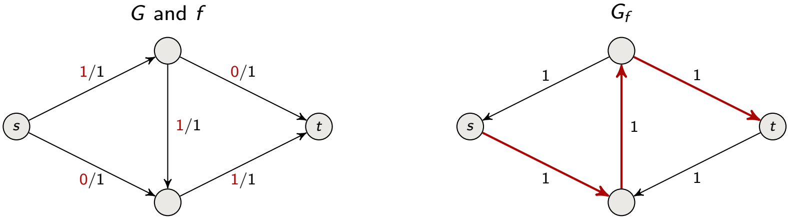
```
FORD-FULKERSON-METHOD($G, s, t$):

1. Initialize flow $f$ to 0
2. while exists an augmenting path $p$ in the residual network $G_f$
3.         augment flow $f$ along $p$
4. return $f$
```





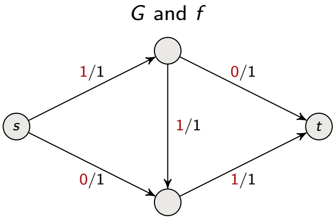
The Ford-Fulkerson Method'54

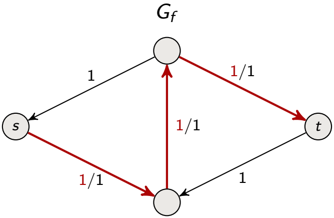
```
FORD-FULKERSON-METHOD($G, s, t$):

Initialize flow $f$ to 0
while exists an augmenting path $p$ in the residual network $G_f$
        augment flow $f$ along $p$
return $f$
```



The Ford-Fulkerson Method'54

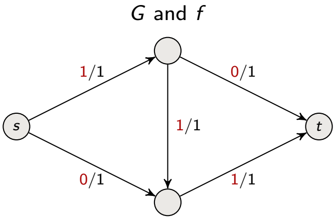
```
FORD-FULKERSON-METHOD($G, s, t$):

1. Initialize flow $f$ to 0
2. while exists an augmenting path $p$ in the residual network $G_f$
3.         augment flow $f$ along $p$
4. return $f$
```

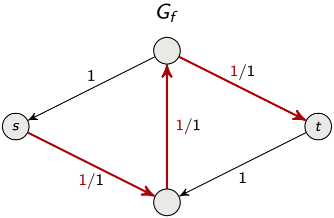




The Ford-Fulkerson Method'54

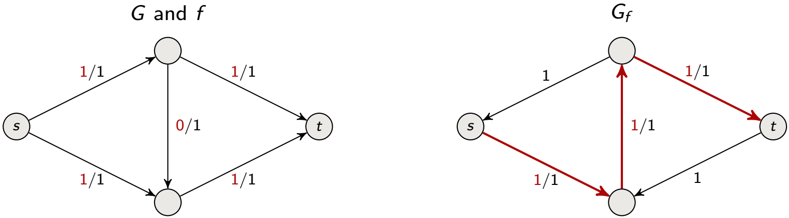
```
FORD-FULKERSON-METHOD($G, s, t$):

1. Initialize flow $f$ to 0
2.  while exists an augmenting path $p$ in the residual network $G_f$
3.      augment flow $f$ along $p$
4. return $f$
```





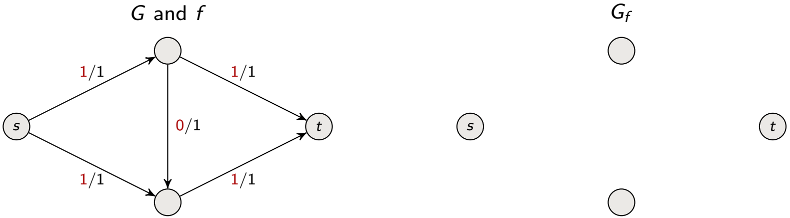
The Ford-Fulkerson Method'54

```
FORD-FULKERSON-METHOD($G, s, t$):

Initialize flow $f$ to 0
while exists an augmenting path $p$ in the residual network $G_f$
    augment flow $f$ along $p$
return $f$
```



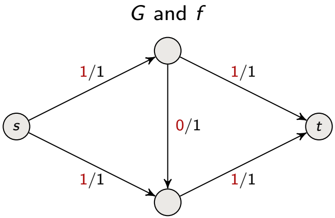
The Ford-Fulkerson Method'54

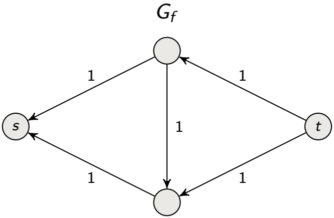
```
FORD-FULKERSON-METHOD($G, s, t$):

1. Initialize flow $f$ to 0
2. while exists an augmenting path $p$ in the residual network $G_f$
3.         augment flow $f$ along $p$
4. return $f$
```



The Ford-Fulkerson Method'54

```
Ford-Fulkerson-Method($G, s, t$):

1. Initialize flow $f$ to 0
2. while exists an augmenting path $p$ in the residual network $G_f$
3.         augment flow $f$ along $p$
4. return $f$
```

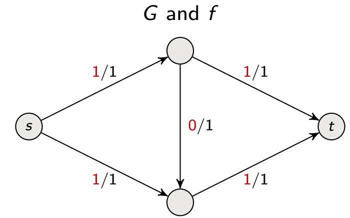


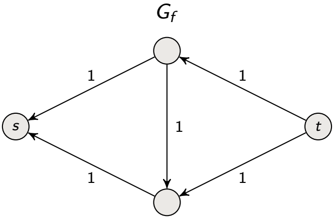


The Ford-Fulkerson Method'54

```
FORD-FULKERSON-METHOD($G, s, t$):

1. Initialize flow $f$ to 0
2. while exists an augmenting path $p$ in the residual network $G_f$
3.         augment flow $f$ along $p$
4. return $f$
```

No augmenting path and flow of value 2 is optimal

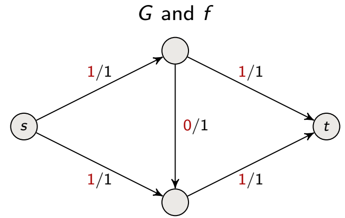


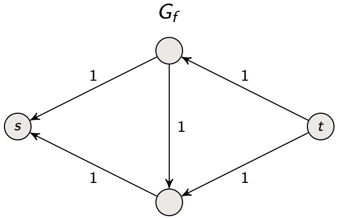


The Ford-Fulkerson Method'54

```
FORD-FULKERSON-METHOD($G, s, t$):

1. Initialize flow $f$ to 0
2.  while exists an augmenting path $p$ in the residual network $G_f$
3.      augment flow $f$ along $p$
4. return $f$
```

No augmenting path and flow of value 2 is optimal

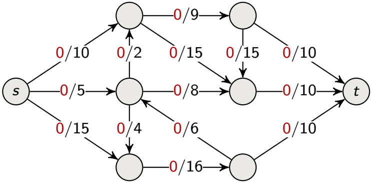




The Ford-Fulkerson Method'54

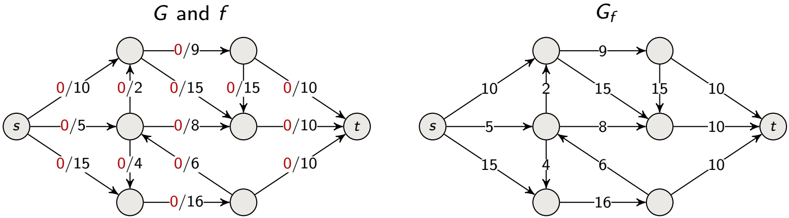
```
FORD-FULKERSON-METHOD($G, s, t$):

1. Initialize flow $f$ to 0
2.  while exists an augmenting path $p$ in the residual network $G_f$
3.          augment flow $f$ along $p$
4. return $f$
```

G and f



The Ford-Fulkerson Method'54

```
FORD-FULKERSON-METHOD($G, s, t$):

Initialize flow $f$ to 0
while exists an augmenting path $p$ in the residual network $G_f$
        augment flow $f$ along $p$
return $f$
```

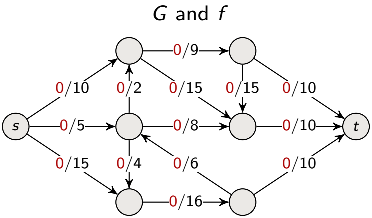


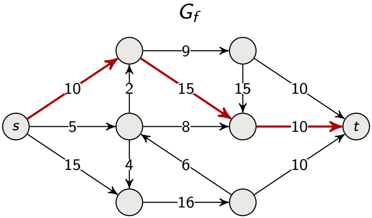
G and f

G_f

The Ford-Fulkerson Method'54

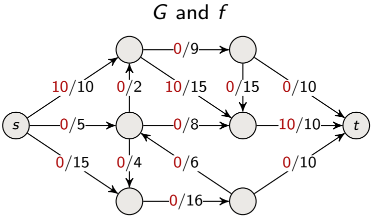
```
FORD-FULKERSON-METHOD($G, s, t$):

1. Initialize flow $f$ to 0
2. **while exists an augmenting path $p$ in the residual network $G_f$**
3.         augment flow $f$ along $p$
4. **return** $f$
```

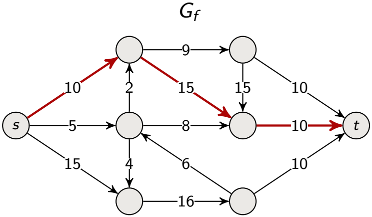




The Ford-Fulkerson Method’54

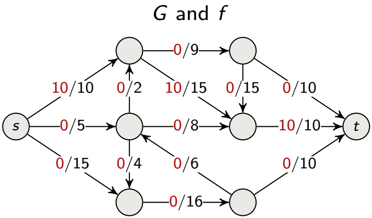
```
FORD-FULKERSON-METHOD($G, s, t$):

1. Initialize flow $f$ to 0
2.  while exists an augmenting path $p$ in the residual network $G_f$
3.       augment flow $f$ along $p$
4. return $f$
```

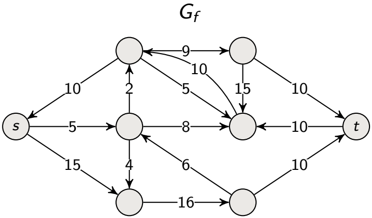




The Ford-Fulkerson Method'54

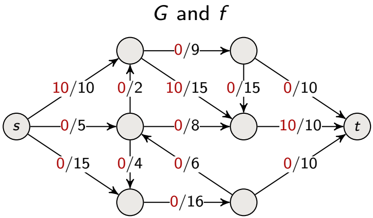
```
FORD-FULKERSON-METHOD(G, s, t):

1. Initialize flow f to 0
2. while exists an augmenting path p in the residual network G_f
3.         augment flow f along p
4. return f
```

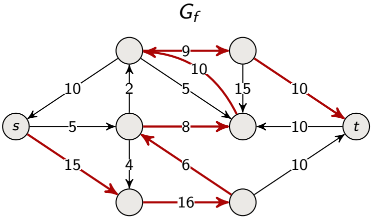




The Ford-Fulkerson Method’54

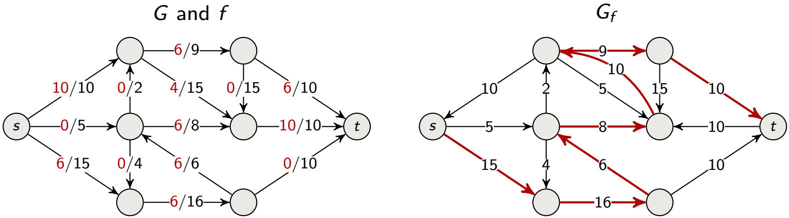
```
FORD-FULKERSON-METHOD($G, s, t$):

1. Initialize flow $f$ to 0
2. while exists an augmenting path $p$ in the residual network $G_f$
3.         augment flow $f$ along $p$
4. return $f$
```





The Ford-Fulkerson Method'54

```
FORD-FULKERSON-METHOD($G, s, t$):

Initialize flow $f$ to 0
while exists an augmenting path $p$ in the residual network $G_f$
    augment flow $f$ along $p$
return $f$
```

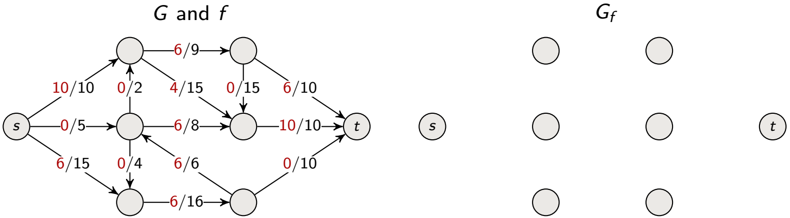


G and f

G_f

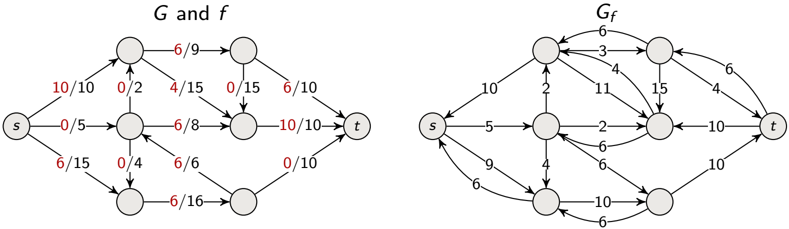
The Ford-Fulkerson Method'54

```
FORD-FULKERSON-METHOD($G$, $s$, $t$):

Initialize flow $f$ to 0
while exists an augmenting path $p$ in the residual network $G_f$
    augment flow $f$ along $p$
return $f$
```



The Ford-Fulkerson Method'54

```
FORD-FULKERSON-METHOD($G, s, t$):

Initialize flow $f$ to 0
while exists an augmenting path $p$ in the residual network $G_f$
        augment flow $f$ along $p$
return $f$
```

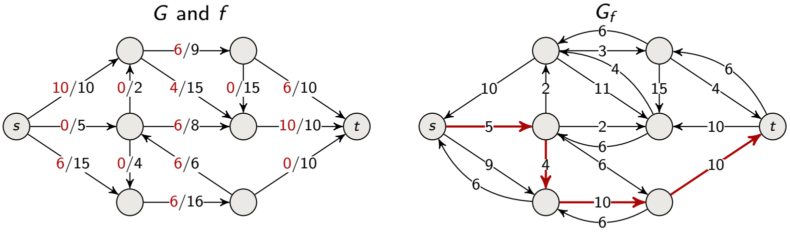


G and f

G_f

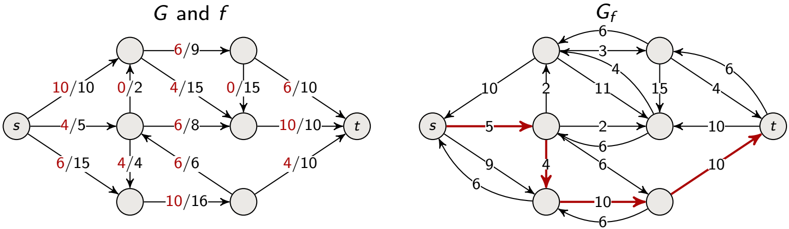
The Ford-Fulkerson Method'54

```
FORD-FULKERSON-METHOD($G, s, t$):

1. Initialize flow $f$ to 0
2. while exists an augmenting path $p$ in the residual network $G_f$
3.         augment flow $f$ along $p$
4. return $f$
```



The Ford-Fulkerson Method’54

```
FORD-FULKERSON-METHOD($G, s, t$):

Initialize flow $f$ to 0
while exists an augmenting path $p$ in the residual network $G_f$
    augment flow $f$ along $p$
return $f$
```

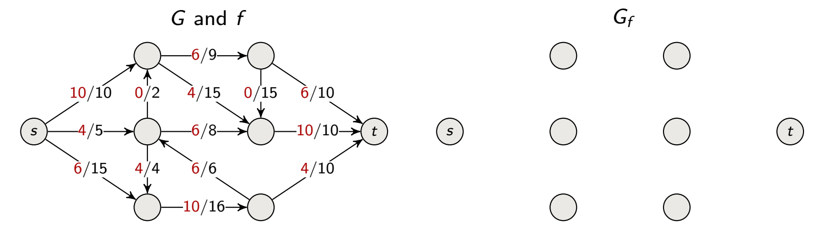


G and f

G_f

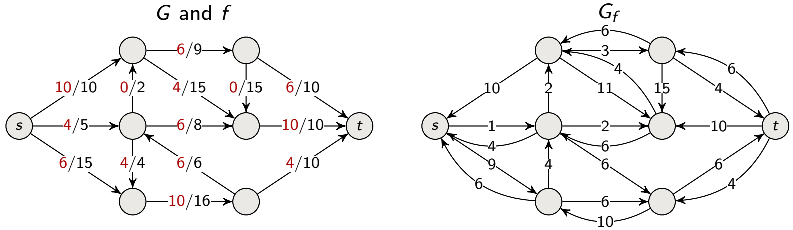
The Ford-Fulkerson Method'54

```
FORD-FULKERSON-METHOD($G, s, t$):

Initialize flow $f$ to 0
while exists an augmenting path $p$ in the residual network $G_f$
        augment flow $f$ along $p$
return $f$
```



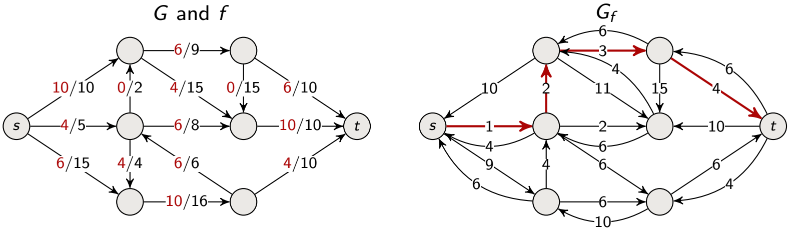
The Ford-Fulkerson Method’54

```
FORD-FULKERSON-METHOD($G, s, t$):

Initialize flow $f$ to 0
while exists an augmenting path $p$ in the residual network $G_f$
    augment flow $f$ along $p$
return $f$
```



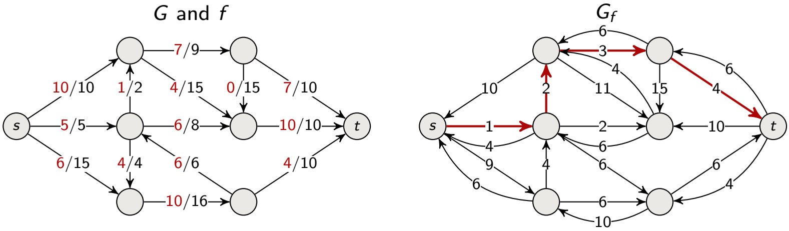
The Ford-Fulkerson Method'54

```
FORD-FULKERSON-METHOD($G, s, t$):

Initialize flow $f$ to 0
while exists an augmenting path $p$ in the residual network $G_f$
        augment flow $f$ along $p$
return $f$
```



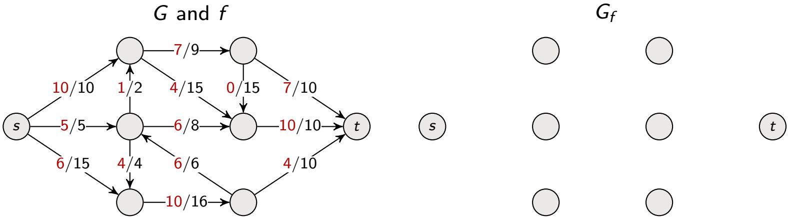
The Ford-Fulkerson Method'54

```
Ford-Fulkerson-Method($G, s, t$):

Initialize flow $f$ to 0
while exists an augmenting path $p$ in the residual network $G_f$
    augment flow $f$ along $p$
return $f$
```



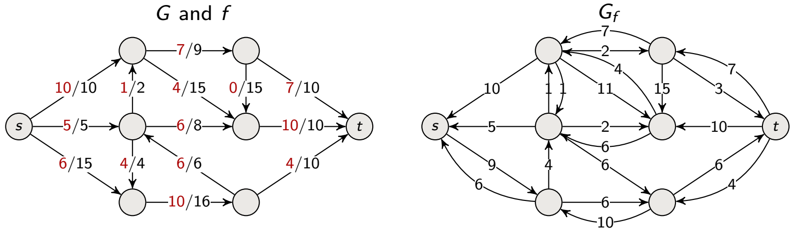
The Ford-Fulkerson Method'54

```
FORD-FULKERSON-METHOD($G, s, t$):

1. Initialize flow $f$ to 0
2. while exists an augmenting path $p$ in the residual network $G_f$
3.         augment flow $f$ along $p$
4. return $f$
```



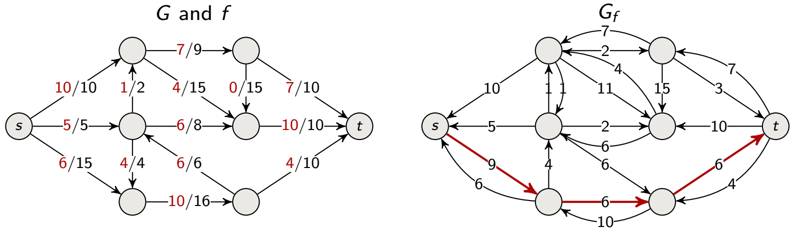
The Ford-Fulkerson Method'54

```
FORD-FULKERSON-METHOD($G, s, t$):

Initialize flow $f$ to 0
while exists an augmenting path $p$ in the residual network $G_f$
        augment flow $f$ along $p$
return $f$
```



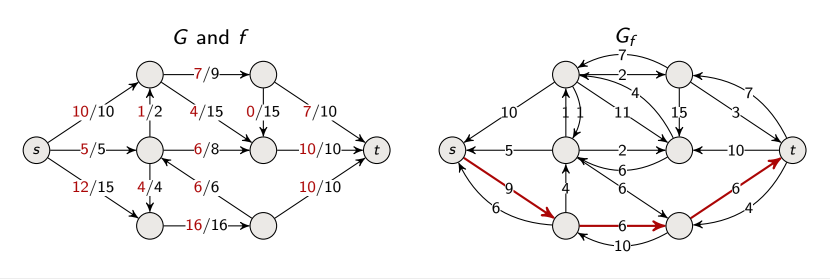
The Ford-Fulkerson Method’54

```
FORD-FULKERSON-METHOD($G, s, t$):

Initialize flow $f$ to 0
while exists an augmenting path $p$ in the residual network $G_f$
    augment flow $f$ along $p$
return $f$
```



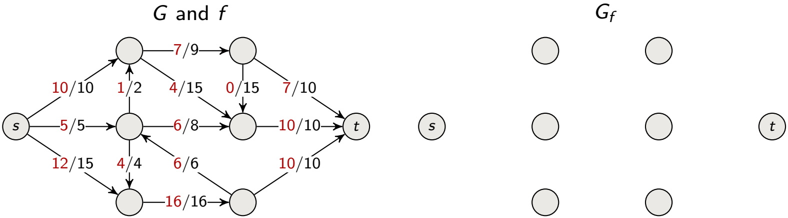
The Ford-Fulkerson Method'54

```
FORD-FULKERSON-METHOD($G, s, t$):

1. Initialize flow $f$ to 0
2.  while exists an augmenting path $p$ in the residual network $G_f$
3.       augment flow $f$ along $p$
4. return $f$
```



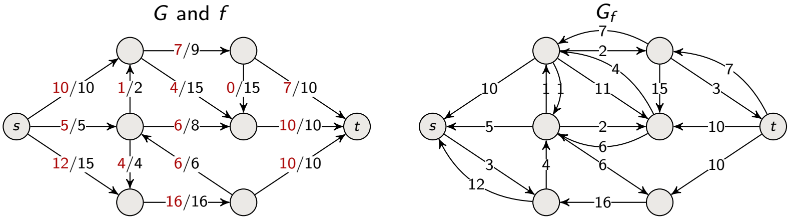
The Ford-Fulkerson Method'54

```
FORD-FULKERSON-METHOD($G, s, t$):

1. Initialize flow $f$ to 0
2. while exists an augmenting path $p$ in the residual network $G_f$
3.         augment flow $f$ along $p$
4. return $f$
```



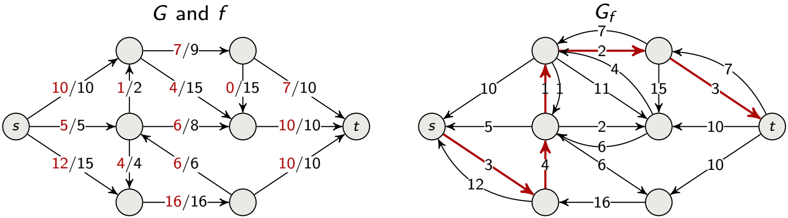
The Ford-Fulkerson Method'54

```
FORD-FULKERSON-METHOD($G, s, t$):

Initialize flow $f$ to 0
while exists an augmenting path $p$ in the residual network $G_f$
        augment flow $f$ along $p$
return $f$
```



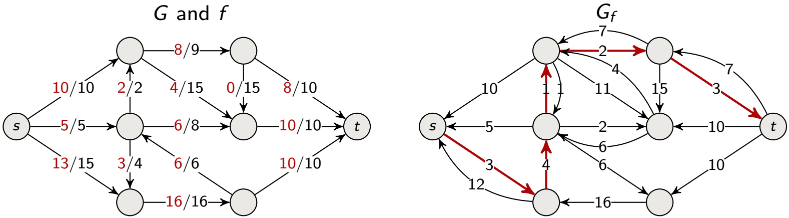
The Ford-Fulkerson Method'54

```
FORD-FULKERSON-METHOD($G, s, t$):

1. Initialize flow $f$ to 0
2. while exists an augmenting path $p$ in the residual network $G_f$
3.         augment flow $f$ along $p$
4. return $f$
```



The Ford-Fulkerson Method'54

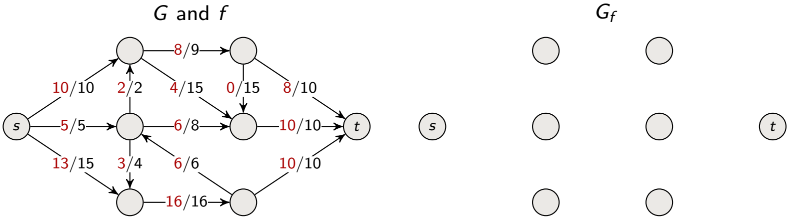
```
FORD-FULKERSON-METHOD($G, s, t$):

1. Initialize flow $f$ to 0
2.  while exists an augmenting path $p$ in the residual network $G_f$
3.       augment flow $f$ along $p$
4. return $f$
```


G and f

G_f

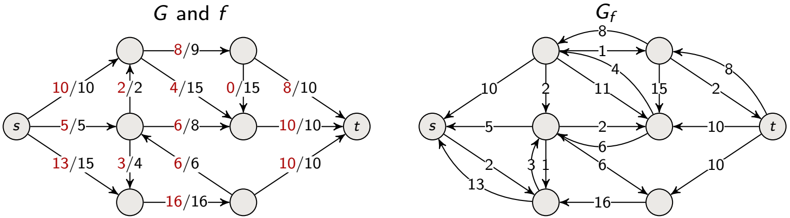
The Ford-Fulkerson Method'54

```
FORD-FULKERSON-METHOD($G, s, t$):

1. Initialize flow $f$ to 0
2. while exists an augmenting path $p$ in the residual network $G_f$
3.         augment flow $f$ along $p$
4. return $f$
```



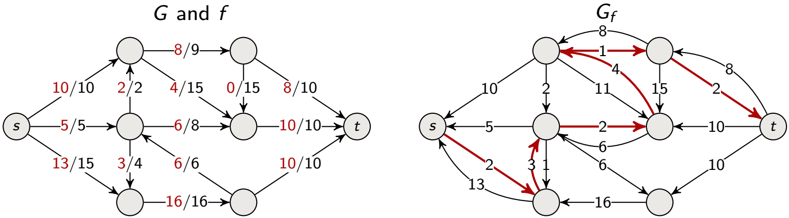
The Ford-Fulkerson Method'54

```
FORD-FULKERSON-METHOD($G, s, t$):

Initialize flow $f$ to 0
while exists an augmenting path $p$ in the residual network $G_f$
        augment flow $f$ along $p$
return $f$
```



The Ford-Fulkerson Method'54

```
FORD-FULKERSON-METHOD($G, s, t$):

Initialize flow $f$ to 0
while exists an augmenting path $p$ in the residual network $G_f$
        augment flow $f$ along $p$
return $f$
```

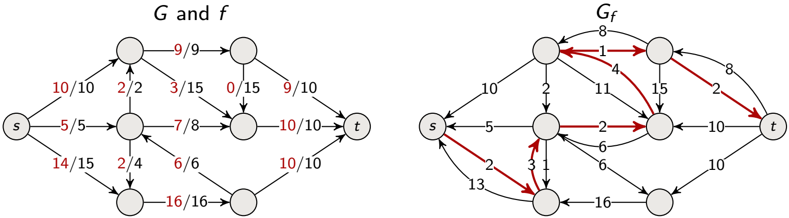


G and f

G_f

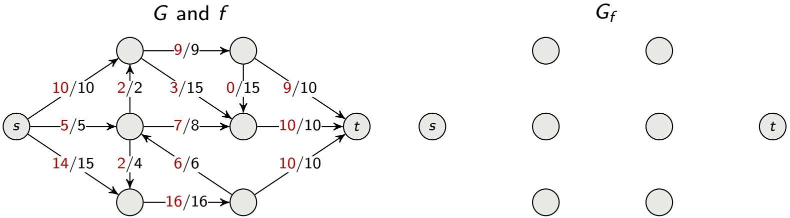
The Ford-Fulkerson Method'54

```
FORD-FULKERSON-METHOD($G, s, t$):

1. Initialize flow $f$ to 0
2.   while exists an augmenting path $p$ in the residual network $G_f$
3.        augment flow $f$ along $p$
4. return $f$
```



The Ford-Fulkerson Method'54

```
FORD-FULKERSON-METHOD($G, s, t$):

Initialize flow $f$ to 0
while exists an augmenting path $p$ in the residual network $G_f$
    augment flow $f$ along $p$
return $f$
```

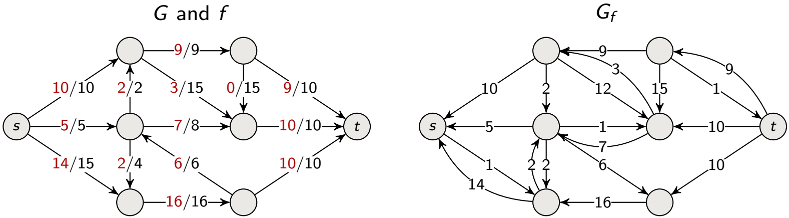


G and f

G_f

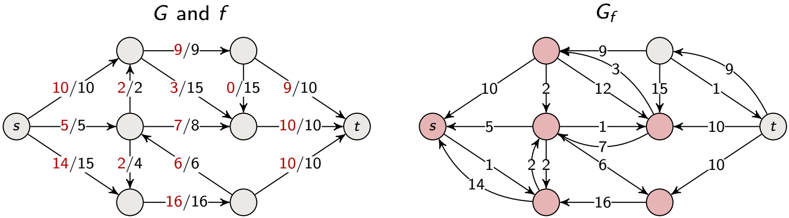
The Ford-Fulkerson Method'54

```
FORD-FULKERSON-METHOD($G, s, t$):

Initialize flow $f$ to 0
while exists an augmenting path $p$ in the residual network $G_f$
    augment flow $f$ along $p$
return $f$
```



The Ford-Fulkerson Method'54

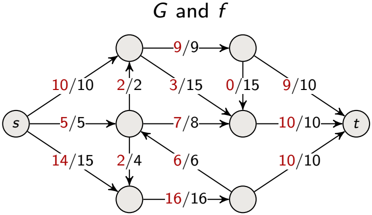
```
FORD-FULKERSON-METHOD($G, s, t$):

Initialize flow $f$ to 0
while exists an augmenting path $p$ in the residual network $G_f$
        augment flow $f$ along $p$
return $f$
```

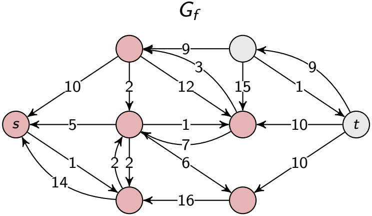


The Ford-Fulkerson Method'54

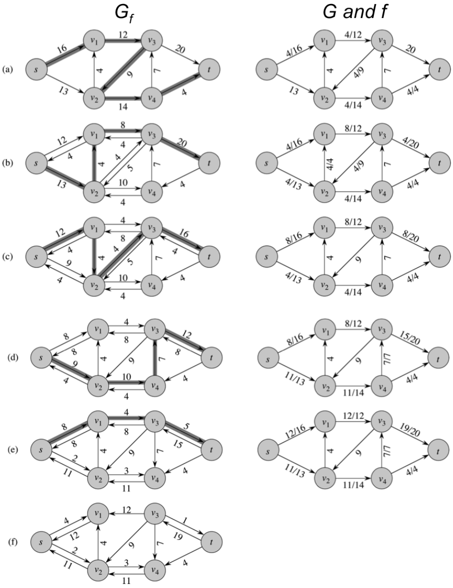
```
FORD-FULKERSON-METHOD($G, s, t$):

1. Initialize flow $f$ to 0
2.  while exists an augmenting path $p$ in the residual network $G_f$
3.        augment flow $f$ along $p$
4. return $f$
```

No augmenting path and flow of value 29 is optimal





Study and understand Example!


G_f

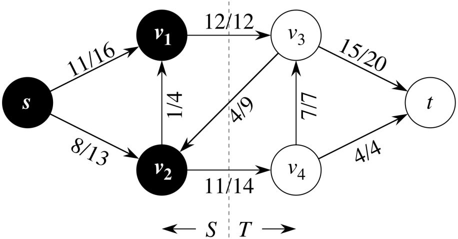
G and f

WHY IS RETURNED FLOW OPTIMAL? (MIN-CUTS)

Cuts in flow networks

A cut of flow network $G(V, E)$ is

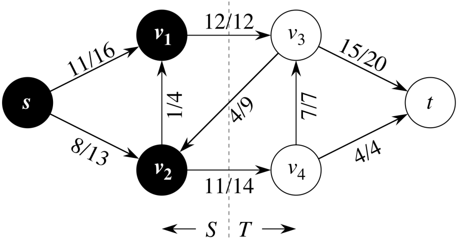
- a partition of V into S and $T = V \setminus S$
- such that $s \in S$ and $t \in T$



Net flow across a cut

The net flow across cut (S, T) is

$$f(S,T) = \underbrace{\sum_{u \in S, v \in T} f(u,v)}_{\text{flow leaving } S} - \underbrace{\sum_{u \in S, v \in T} f(v,u)}_{\text{flow entering } S}$$

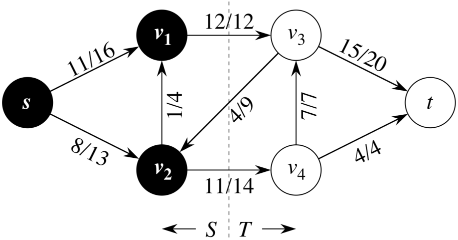


Net flow across a cut

The net flow across cut (S, T) is

$$f(S,T) = \underbrace{\sum_{u \in S, v \in T} f(u,v)}_{\text{flow leaving } S} - \underbrace{\sum_{u \in S, v \in T} f(v,u)}_{\text{flow entering } S}$$

What is the net flow of this cut?

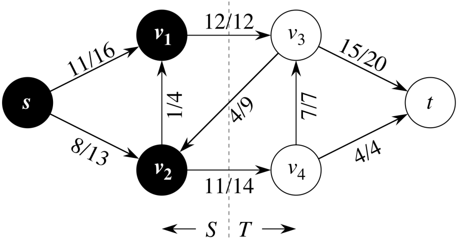


Net flow across a cut

The net flow across cut (S, T) is

$$f(S,T) = \underbrace{\sum_{u \in S, v \in T} f(u,v)}_{\text{flow leaving } S} - \underbrace{\sum_{u \in S, v \in T} f(v,u)}_{\text{flow entering } S}$$

What is the net flow of this cut? $12 + 11 - 4 = 19$

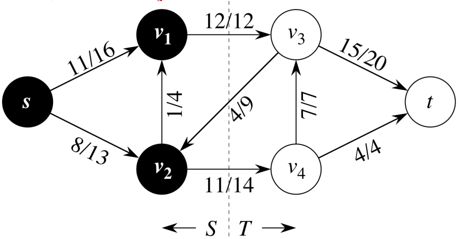


Net flow across a cut

The net flow across cut (S, T) is

$$f(S, T) = \underbrace{\sum_{u \in S, v \in T} f(u, v)}_{\text{flow leaving } S} - \underbrace{\sum_{u \in S, v \in T} f(v, u)}_{\text{flow entering } S}$$

What is the net flow of this cut? $12 + 11 - 4 = 19$ Note that this equals the value of the flow; it's always the case!



Net flow equals flow value for any cut

Theorem

For any cut (S, T), $|f| = f(S, T)$.

Net flow equals flow value for any cut

Theorem

For any cut (S, T), $|f| = f(S, T)$.

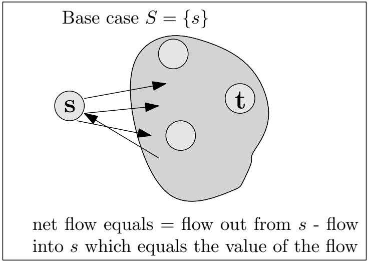
Proof by induction on the size of S.

Net flow equals flow value for any cut

Theorem

For any cut (S, T), $|f| = f(S, T)$.

Proof by induction on the size of S.


Base case $S = \{s\}$
s
t
net flow equals = flow out from s - flow into s which equals the value of the flow

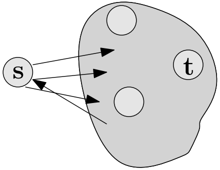
Net flow equals flow value for any cut

Theorem

For any cut (S, T)*,* $|f| = f(S, T)$*.*

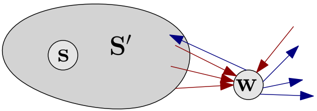
Proof by induction on the size of S.

Base case $S = \{s\}$



net flow equals = flow out from s - flow into s which equals the value of the flow

Inductive Step $S = S' \cup \{w\}$

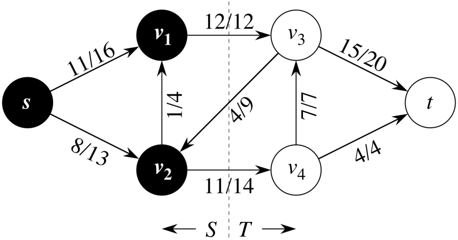


New net flow = Old net flow + $\underbrace{\text{flow on blue edges - flow on red edges}}_{\text{0 by flow conservation}}$

Capacity a cut

The capacity of a cut (S, T) is

$$c(S, T) = \sum_{u \in S, v \in T} c(u, v)$$

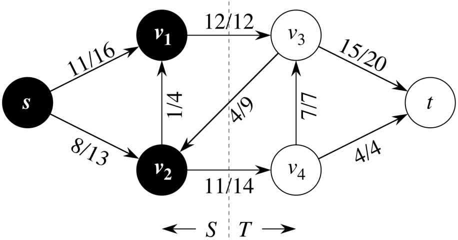


Capacity a cut

The capacity of a cut (S, T) is

$$c(S,T) = \sum_{u \in S, v \in T} c(u,v)$$

What is the capacity of this cut?

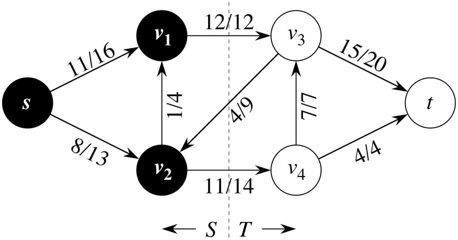


Capacity a cut

The capacity of a cut (S, T) is

$$c(S,T) = \sum_{u \in S, v \in T} c(u,v)$$

What is the capacity of this cut? $12 + 14 = 26$



Flow is at most capacity of a cut

For any flow f and any cut (S, T):

$$|f| = f(S, T)$$

Flow is at most capacity of a cut

For any flow f and any cut (S, T):

$$\begin{aligned} |f| &= f(S, T) \\ &= \sum_{u \in S, v \in T} f(u, v) - \sum_{u \in S, v \in T} f(v, u) \end{aligned}$$

Flow is at most capacity of a cut

For any flow f and any cut (S, T):

$$\begin{aligned} |f| &= f(S, T) \\ &= \sum_{u \in S, v \in T} f(u, v) - \sum_{u \in S, v \in T} f(v, u) \\ &\leq \sum_{u \in S, v \in T} f(u, v) \end{aligned}$$

Flow is at most capacity of a cut

For any flow f and any cut (S, T):

$$\begin{aligned}
|f| &= f(S, T) \\
&= \sum_{u \in S, v \in T} f(u, v) - \sum_{u \in S, v \in T} f(v, u) \\
&\leq \sum_{u \in S, v \in T} f(u, v) \\
&\leq \sum_{u \in S, v \in T} c(u, v)
\end{aligned}$$

Flow is at most capacity of a cut

For any flow f and any cut (S, T):

$$\begin{aligned}
|f| &= f(S, T) \\
&= \sum_{u \in S, v \in T} f(u, v) - \sum_{u \in S, v \in T} f(v, u) \\
&\leq \sum_{u \in S, v \in T} f(u, v) \\
&\leq \sum_{u \in S, v \in T} c(u, v) \\
&= c(S, T)
\end{aligned}$$

Flow is at most capacity of a cut

For any flow f and any cut (S, T):

$$\begin{aligned}
|f| &= f(S, T) \\
&= \sum_{u \in S, v \in T} f(u, v) - \sum_{u \in S, v \in T} f(v, u) \\
&\leq \sum_{u \in S, v \in T} f(u, v) \\
&\leq \sum_{u \in S, v \in T} c(u, v) \\
&= c(S, T)
\end{aligned}$$

Max-flow is at most capacity of a cut

Max-flow is at most capacity of a cut

Therefore: **max-flow** $\leq$ **min-cut**

Max-flow is at most capacity of a cut

Therefore: **max-flow** $\leq$ **min-cut**

We shall prove

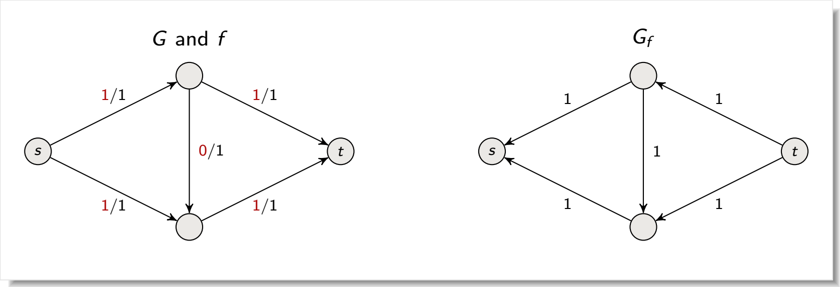
Theorem (max-flow min-cut theorem)

max-flow = min-cut

Examples

Consider f obtained by running Ford-Fulkerson and let

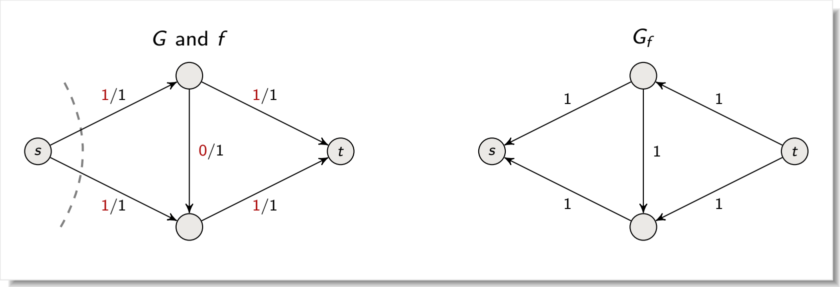
$S = \{v \in V : \text{there is a path from } s \text{ to } v \text{ in } G_f\}$ and $T = V \setminus S$



Examples

Consider f obtained by running Ford-Fulkerson and let

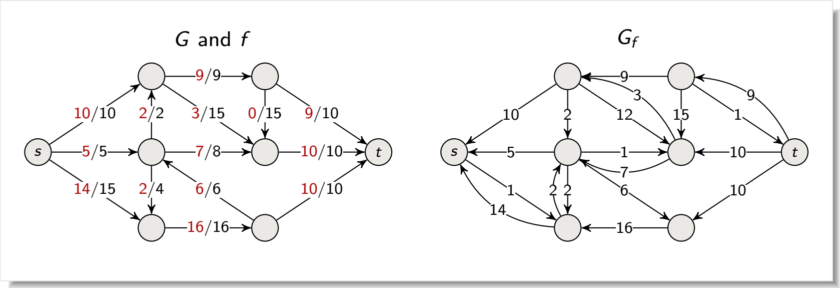
$$S = \{v \in V : \text{there is a path from } s \text{ to } v \text{ in } G_f\} \qquad \text{and} \qquad T = V \setminus S$$



Examples

Consider f obtained by running Ford-Fulkerson and let

$$S = \{v \in V : \text{there is a path from } s \text{ to } v \text{ in } G_f\} \qquad \text{and} \qquad T = V \setminus S$$



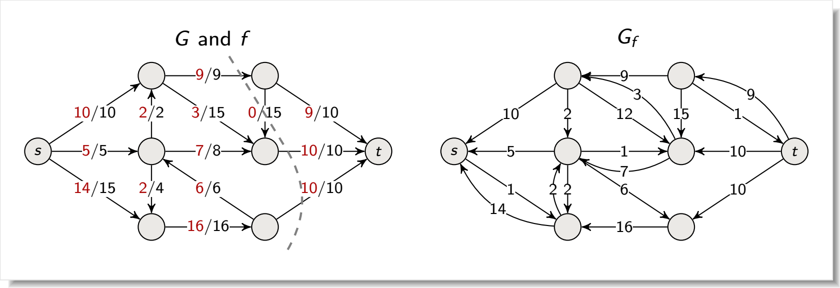
G and f

G_f

Examples

Consider f obtained by running Ford-Fulkerson and let

$S = \{v \in V : \text{there is a path from } s \text{ to } v \text{ in } G_f\}$ and $T = V \setminus S$



Max-flow min-cut theorem

Let $G = (V, E)$ be a flow network with source s and sink t and capacities c and a flow f.

The following are equivalent:

1. f is a maximum flow
2. G_f has no augmenting path
3. $|f| = c(S, T)$ for a minimum cut (S, T)

Max-flow min-cut theorem

Let $G = (V, E)$ be a flow network with source s and sink t and capacities c and a flow f.

The following are equivalent:

1. f is a maximum flow
2. G_f has no augmenting path
3. $|f| = c(S, T)$ for a minimum cut (S, T)

Proof.

Max-flow min-cut theorem

Let $G = (V, E)$ be a flow network with source s and sink t and capacities c and a flow f.

The following are equivalent:

1. f is a maximum flow
2. G_f has no augmenting path
3. $|f| = c(S, T)$ for a minimum cut (S, T)

Proof. (1) $\Rightarrow$ (2): Suppose toward contradiction that G_f has an augmenting path p.

However, then Ford-Fulkerson method would augment f by p to obtain a flow if increased value which contradicts that f is a maximum flow

Max-flow min-cut theorem

Let $G = (V, E)$ be a flow network with source s and sink t and capacities c and a flow f.

The following are equivalent:

1. f is a maximum flow
2. G_f has no augmenting path
3. $|f| = c(S, T)$ for a minimum cut (S, T)

Proof. (2) ⇒ (3): $S =$ set of nodes reachable from s in residual network, $T = V \setminus S$

Max-flow min-cut theorem

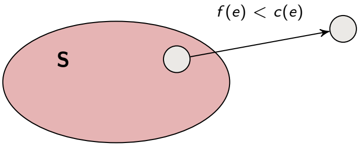
Let $G = (V, E)$ be a flow network with source s and sink t and capacities c and a flow f.

The following are equivalent:

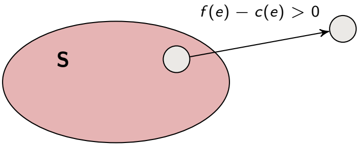
1. f is a maximum flow
2. G_f has no augmenting path
3. $|f| = c(S, T)$ for a minimum cut (S, T)

Proof. (2) $\Rightarrow$ (3): $S =$ set of nodes reachable from s in residual network, $T = V \setminus S$

Every edge flowing out of S in G must be at capacity, otherwise we can reach a node outside S in the residual network.



Original graph



Residual network

Max-flow min-cut theorem

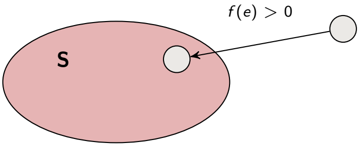
Let $G = (V, E)$ be a flow network with source s and sink t and capacities c and a flow f.

The following are equivalent:

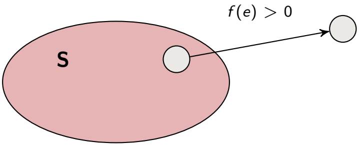
1. f is a maximum flow
2. G_f has no augmenting path
3. $|f| = c(S, T)$ for a minimum cut (S, T)

Proof. (2) $\Rightarrow$ (3): $S =$ set of nodes reachable from s in residual network, $T = V \setminus S$

Every edge flowing into S in G must have flow 0, otherwise we can reach a node outside S in the residual network.



Original graph



Residual network

Max-flow min-cut theorem

Let $G = (V, E)$ be a flow network with source s and sink t and capacities c and a flow f.

The following are equivalent:

1. f is a maximum flow
2. G_f has no augmenting path
3. $|f| = c(S, T)$ for a minimum cut (S, T)

Proof. (2) $\Rightarrow$ (3): $S =$ set of nodes reachable from s in residual network, $T = V \setminus S$

Therefore

$$|f| = f(S, T)$$

Max-flow min-cut theorem

Let $G = (V, E)$ be a flow network with source s and sink t and capacities c and a flow f.

The following are equivalent:

1. f is a maximum flow
2. G_f has no augmenting path
3. $|f| = c(S, T)$ for a minimum cut (S, T)

Proof. (2) $\Rightarrow$ (3): $S =$ set of nodes reachable from s in residual network, $T = V \setminus S$

Therefore

$$\begin{aligned} |f| &= f(S, T) \\ &= \sum_{u \in S, v \in T} f(u, v) - \sum_{u \in S, v \in T} f(v, u) \end{aligned}$$

Max-flow min-cut theorem

Let $G = (V, E)$ be a flow network with source s and sink t and capacities c and a flow f.

The following are equivalent:

1. f is a maximum flow
2. G_f has no augmenting path
3. $|f| = c(S, T)$ for a minimum cut (S, T)

Proof. (2) $\Rightarrow$ (3): $S =$ set of nodes reachable from s in residual network, $T = V \setminus S$

Therefore

$$\begin{aligned} |f| &= f(S, T) \\ &= \sum_{u \in S, v \in T} f(u, v) - \sum_{u \in S, v \in T} f(v, u) \\ &= \sum_{u \in S, v \in T} c(u, v) \end{aligned}$$

Max-flow min-cut theorem

Let $G = (V, E)$ be a flow network with source s and sink t and capacities c and a flow f.

The following are equivalent:

1. f is a maximum flow
2. G_f has no augmenting path
3. $|f| = c(S, T)$ for a minimum cut (S, T)

Proof. (2) $\Rightarrow$ (3): $S =$ set of nodes reachable from s in residual network, $T = V \setminus S$

Therefore

$$\begin{aligned} |f| &= f(S, T) \\ &= \sum_{u \in S, v \in T} f(u, v) - \sum_{u \in S, v \in T} f(v, u) \\ &= \sum_{u \in S, v \in T} c(u, v) \\ &= c(S, T) \end{aligned}$$

Max-flow min-cut theorem

Let $G = (V, E)$ be a flow network with source s and sink t and capacities c and a flow f.

The following are equivalent:

1. f is a maximum flow
2. G_f has no augmenting path
3. $|f| = c(S, T)$ for a minimum cut (S, T)

Proof. (3) $\Rightarrow$ (1): Recall that $|f| \leq c(S, T)$ for all cuts (S, T).

Max-flow min-cut theorem

Let $G = (V, E)$ be a flow network with source s and sink t and capacities c and a flow f.

The following are equivalent:

1. f is a maximum flow
2. G_f has no augmenting path
3. $|f| = c(S, T)$ for a minimum cut (S, T)

Proof. (3) $\Rightarrow$ (1): Recall that $|f| \leq c(S, T)$ for all cuts (S, T).

Therefore, if the value of flow is equal to the capacity of some cut, then it cannot be further improved.

Max-flow min-cut theorem

Let $G = (V, E)$ be a flow network with source s and sink t and capacities c and a flow f.

The following are equivalent:

1. f is a maximum flow
2. G_f has no augmenting path
3. $|f| = c(S, T)$ for a minimum cut (S, T)

Proof. (3) ⇒ (1): Recall that $|f| \leq c(S, T)$ for all cuts (S, T).

Therefore, if the value of flow is equal to the capacity of some cut, then it cannot be further improved.

So f is a maximum flow

Summary

- Flow Networks
- Ford-Fulkerson Method
- Cuts
- Max-flow $=$ min cut theorem