Part VI

Graph Algorithms (III)

songyou@buaa.edu.cn

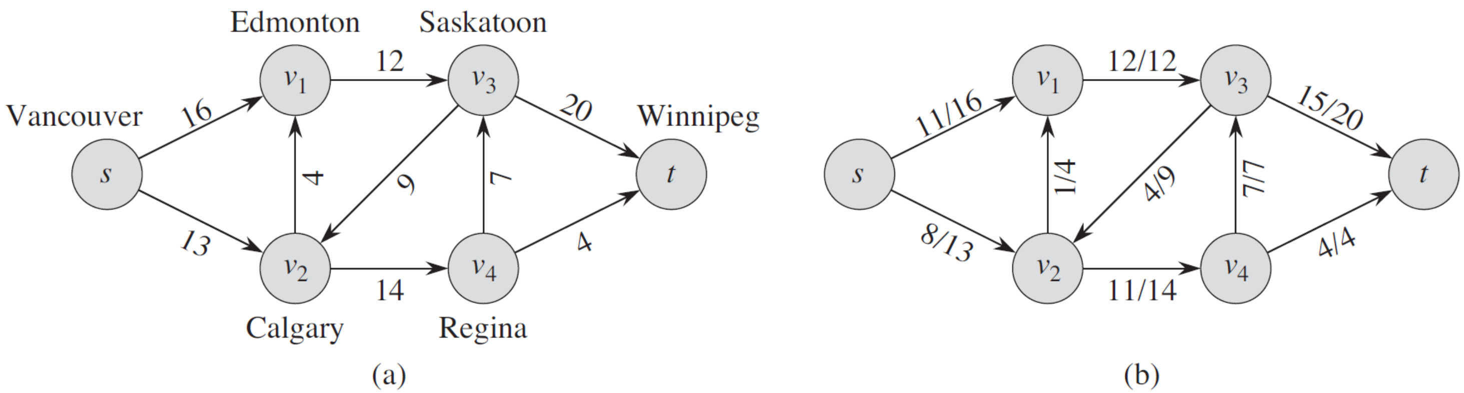
Graph Algorithms

- **Elementary Graph Algorithms**
  - **Representations of Graphs**
  - **BFS, DFS**
  - **Sort Topologically**
- **Single-Source Shortest Paths**
  - **Finding shortest paths from a given source vertex to all other vertices.**
- **All-Pairs Shortest Paths**
- **Maximum Flow**

26 Maximum Flow

26 Maximum Flow

Imagine a material coursing through a system from a source, where the material is produced, to a sink, where it is consumed. The source produces the material at some steady rate, and the sink consumes the material at the same rate.



(a) (b)

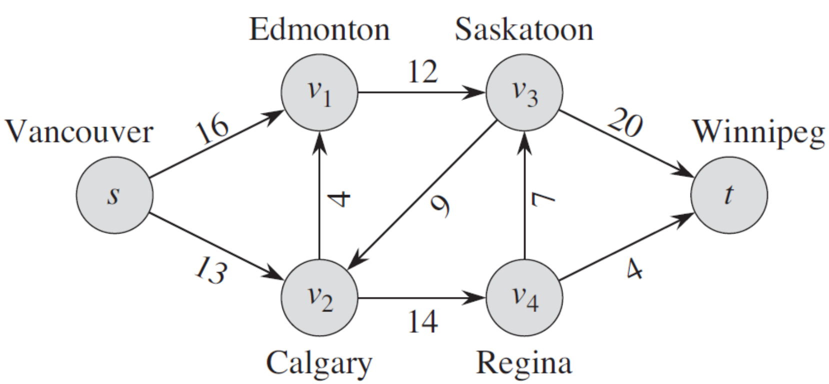
capacity network

最大流：又称为流网络的最大容量问题，或最小分割问题。

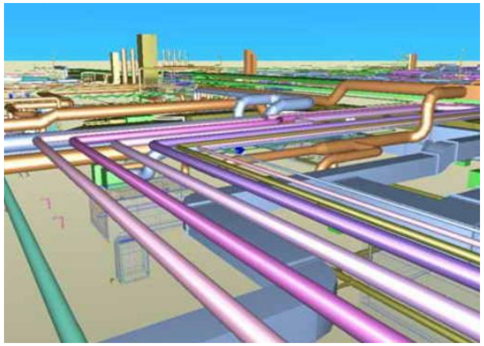
26 Maximum Flow

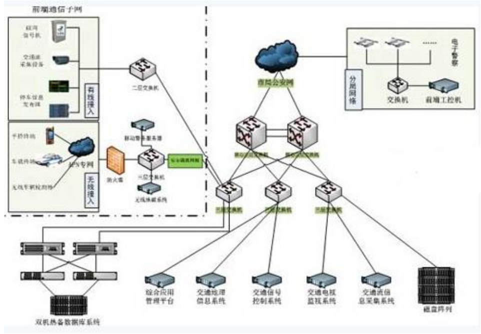
Flow networks

- Liquids flowing through pipes
- Parts through assembly lines
- Current through electrical networks
- Information through communication networks
- Cars through highway traffic networks



capacity network





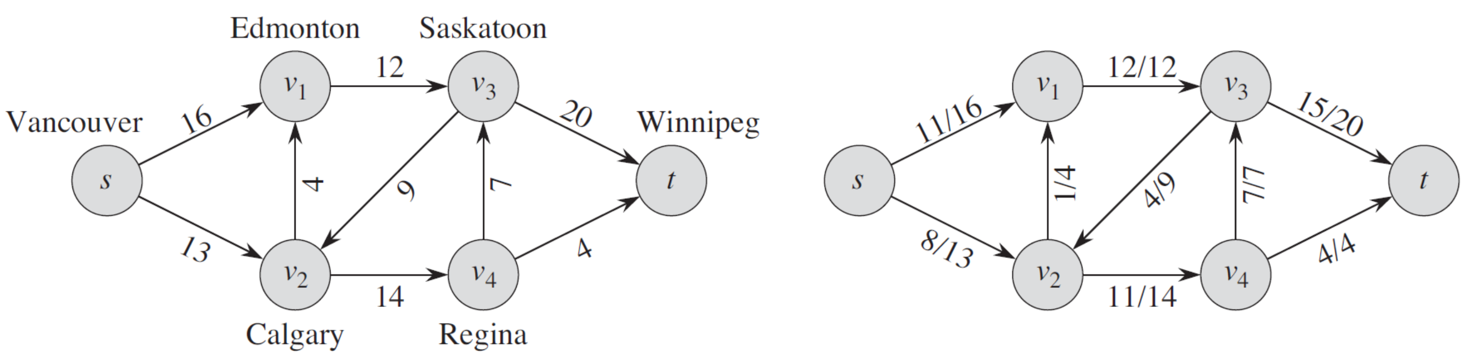
26 Maximum Flow

Basic definition

1. **Flow networks G**
2. **Flow f**
3. **Maximum-flow $f_{\max}$**
4. **Residual networks G_f** **(残留网络)**
5. **Residual capacity $c_f(u, v)$ of G_f** **(顶点间的残留容量)**
6. **Augmenting path p** **(增广路径)**
7. **Residual capacity $c_f(p)$ of p** **(路径上的残留容量)**
8. **Cut (S, T) and its net flow $f(S, T)$ and capacity $c(S, T)$**
9. **Max-flow min-cut** **(最大流最小割)**

26 Maximum Flow

- **Capacity**: a maximum rate at which the material can flow through the conduit.
- **Flow conservation**: the rate at which material enters a vertex must equal the rate at which it leaves the vertex.
- **Maximum-flow problem**: we wish to compute the greatest rate at which we can ship material from the source to the sink without violating any capacity constraints.



(a)
(b)

capacity network

26.1 Flow networks

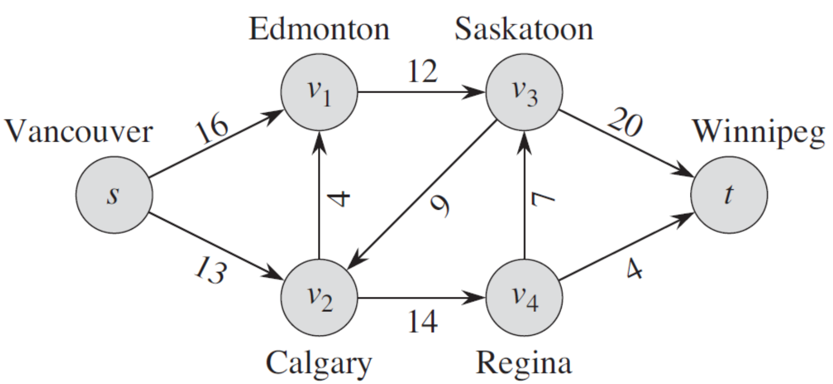
Flow networks and flows

- A flow network $G = (V, E)$ is a directed graph in which each edge $(u, v) \in E$ has a nonnegative capacity $c(u, v) \geq 0$. If $(u, v)\ !\in E$, $c(u, v) = 0$.
- Each vertex lies on some path from the source to the sink.
- ***source*** ***s***
- ***sink*** ***t***
- A ***flow*** in G is

  a real-valued function

  $f : V \times V \to \mathbf{R}$ that satisfies

  The following two properties:



capacity network

26.1 Flow networks

Flow networks and flows

A ***flow*** $f: V \times V \rightarrow \mathbf{R}$ that satisfies The following two properties:

(1) Capacity constraint: For all $u, v \in V$, we require

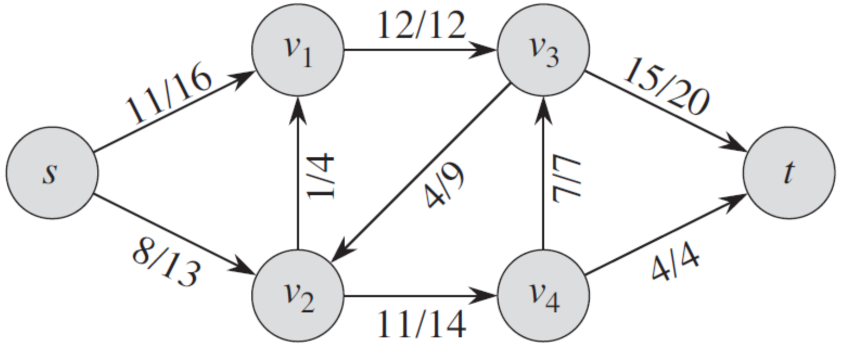
$$0 \le f(u, v) \le c(u, v) .$$

(2) Flow conservation: For all $u \in V - \{s, t\}$, we require

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

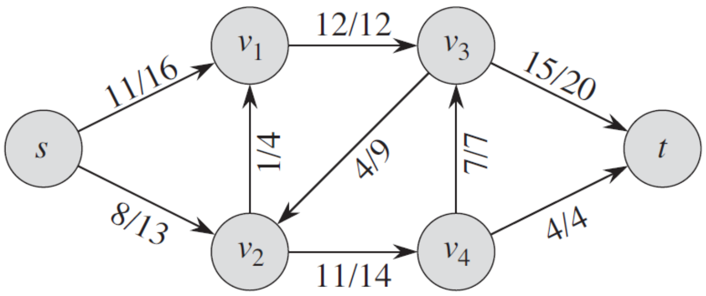
" flow in equals flow out. "

When $(u, v) \,!\!\in E$, there can be no flow from u to v, and $f(u, v) = 0$.



26.1 Flow networks

Flow networks and flows



- $f(u, v)$: the flow from vertex u to v.
- The value $|f|$ of a flow f is defined as

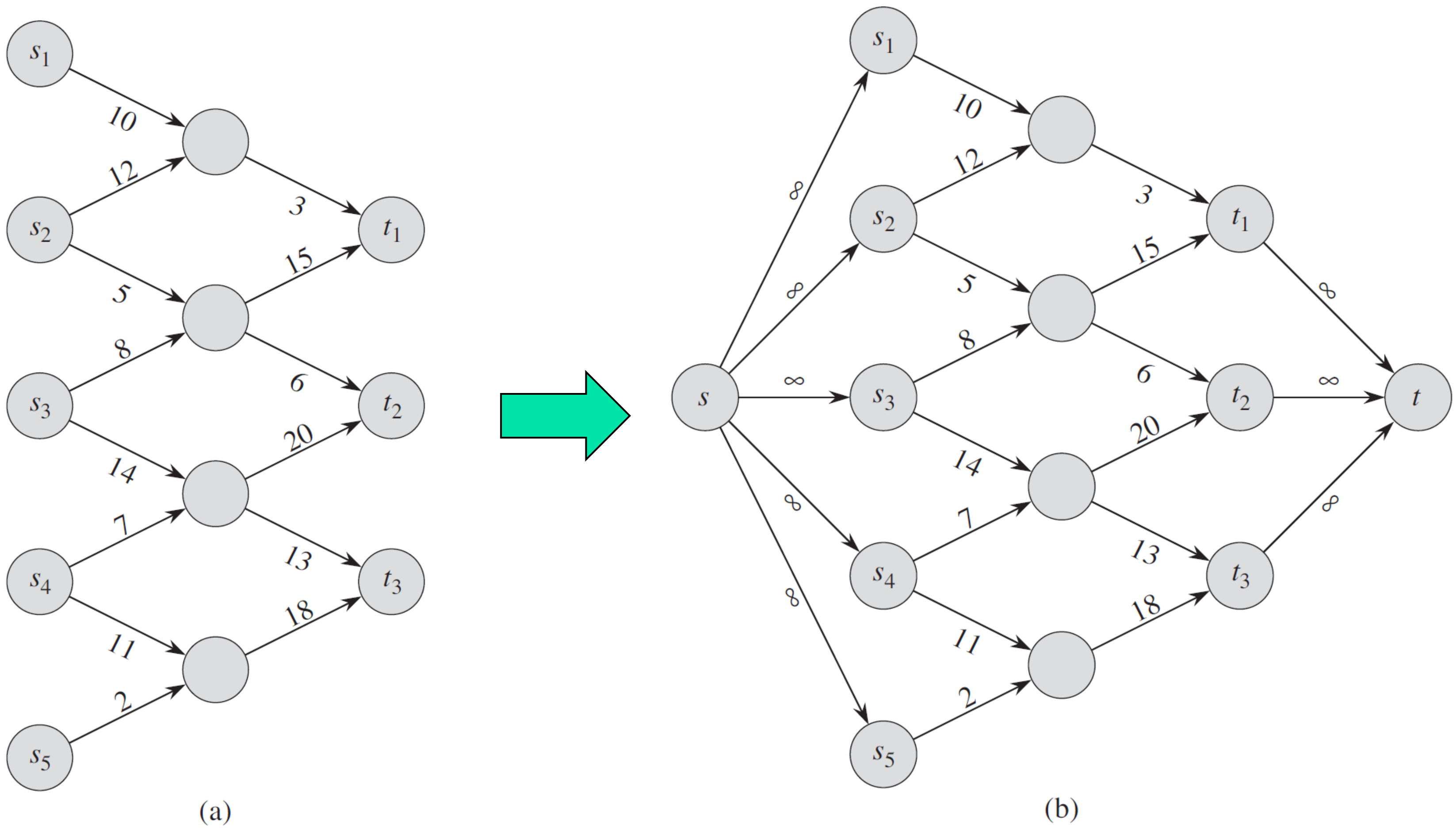
$$|f| = \sum_{v \in V} f(s, v) - \sum_{v \in V} f(v, s) ,$$

that is, the total flow out of the source minus the flow into the source. (Here, the $|\cdot|$ notation denotes flow value, not absolute value.)

- **Maximum-flow problem**: we are given a flow network G with source s and sink t , and we wish to find a flow of maximum value.

最大流：又称为流网络的最大容量问题，或最小分割问题。

Networks with multiple sources and sinks

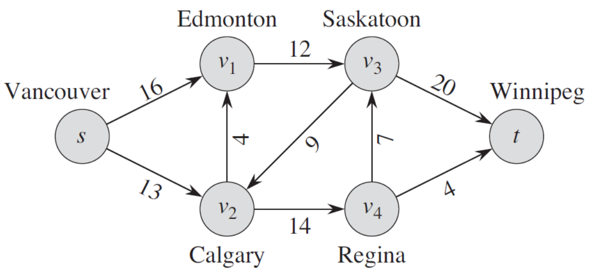


26.2 The Ford-Fulkerson method

A "method" rather than an "algorithm".

The Ford-Fulkerson method depends on three important ideas:

- **residual networks** (剩余网络（残留网络），核心思想：存在一些边，在其上还能增加额外流，这些边就构成了残留网络的增广路径)
- **augmenting paths**
- **cuts**



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

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

26.2 The Ford-Fulkerson method

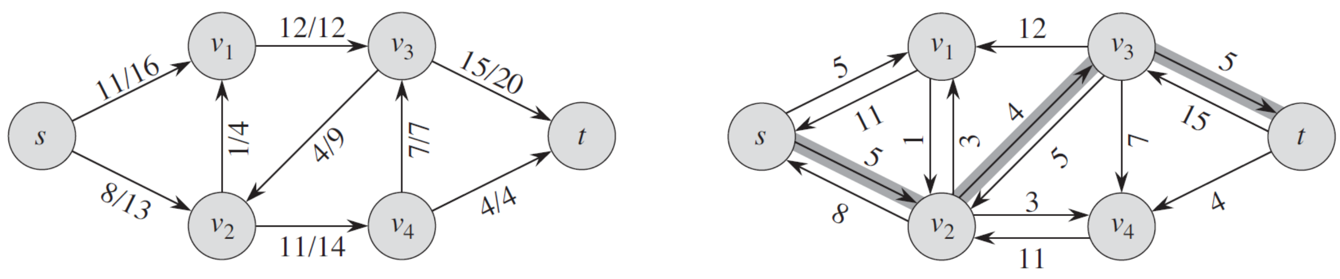
Residual networks

residual capacity

边上还能增加的额外流

反向流最多抵消正向流

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



(a) (b)

Example: let $u \leftarrow s$, $v \leftarrow v_1$, there are $c(u, v) = 16$ and $f(u, v) = 11$, then we can increase $f(u, v)$ by up to $c_f(u, v) = 5$ units before we exceed the capacity constraint on edge (u, v). We also wish to allow an algorithm to return up to 11 units of flow from v to u, and hence $c_f(v, u) = 11$.

26.2 The Ford-Fulkerson method

Residual networks

- **residual capacity**

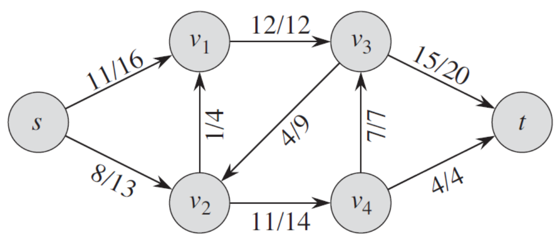
$$c_f(u,v) = \begin{cases} c(u,v) - f(u,v) & \text{if } (u,v) \in E\ , \\ f(v,u) & \text{if } (v,u) \in E\ , \\ 0 & \text{otherwise}\ . \end{cases}$$

- **residual network:** $G_f = (V, E_f)$, where

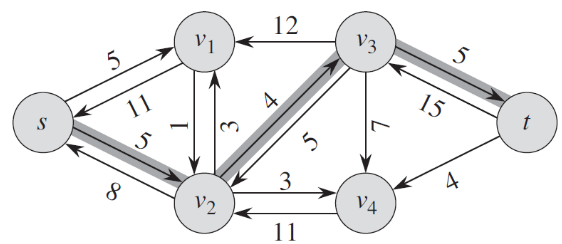
$$E_f = \{(u,v) \in V \times V : c_f(u,v) > 0\}$$

残留网络：顶点跟流网络一样，边（残留边）的权值为流网络的残留容量

流网络



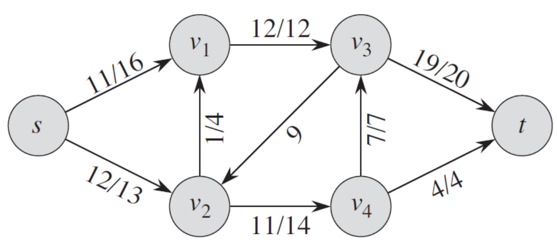
(a)



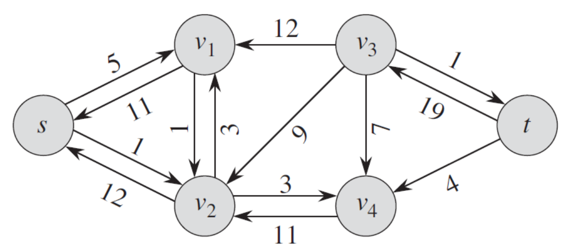
(b)

(a)的残留网络（粗线是增广路径）

在(a)图中，沿着其残留网络的增广路径上增加流以后新的流网络



(c)



(d)

(c)的残留网络

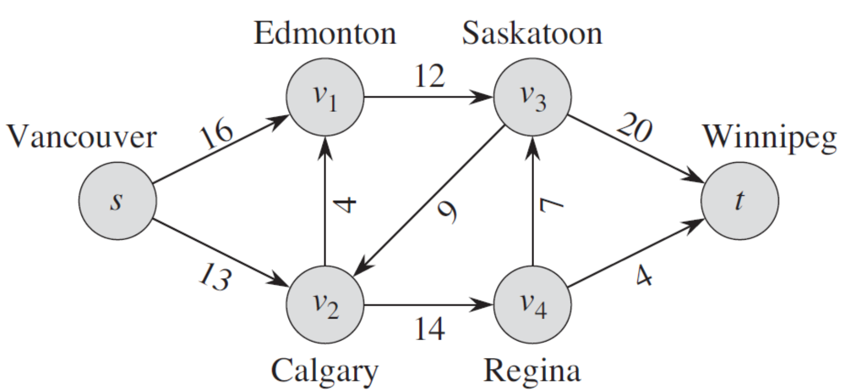
26.2 The Ford-Fulkerson method

Residual networks

- **residual capacity**

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

- 只有容量（流为零）的网络，其残留网络就是其自身。如下图既是流网络原图（流为零），也是残留网络。



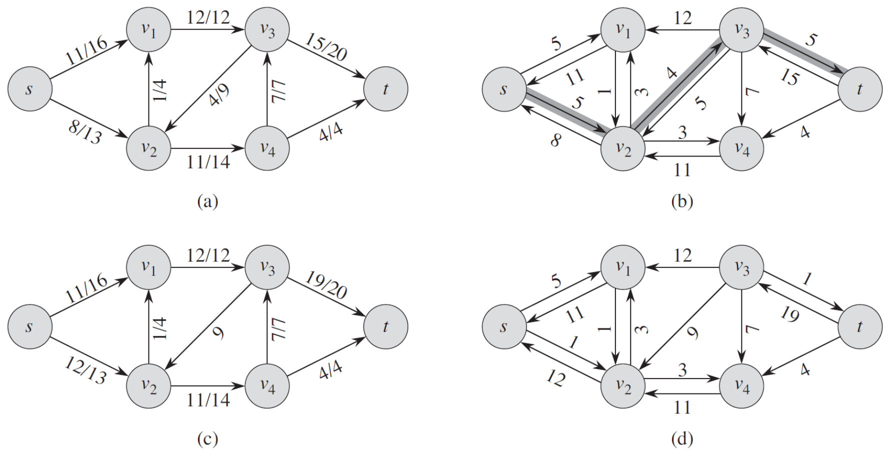
26.2 The Ford-Fulkerson method

Residual networks

Lemma 26.1 Let $G = (V, E)$ be a flow network, and let f be a flow in G. Let G_f be the residual network of G induced by f, and let **f' be a flow in G_f**. Then the flow sum $f + f'$ ($f \uparrow f'$) defined by equation (26.4) is a flow in G with value

$$|f + f'| = |f| + |f'|.$$

$(f_1 + f_2)(u, v) = f_1(u, v) + f_2(u, v)$ …… (26.4)



Proof Verify that the **capacity constraints**, **flow conservation** are obeyed.

… …

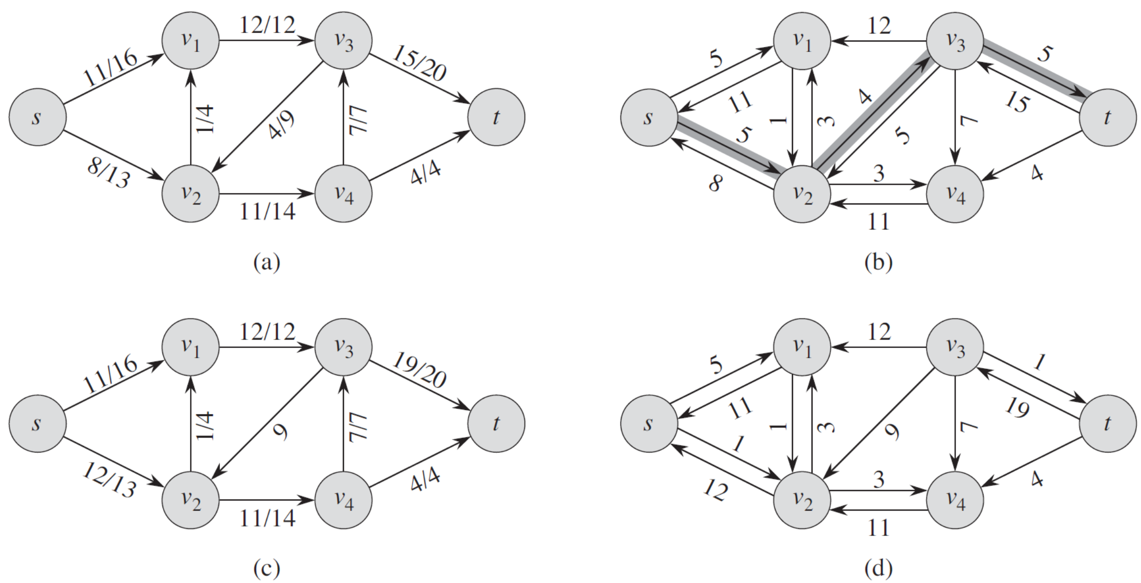
Capacity constraint: For all u, v, we have $f(u, v) \le c(u, v)$

Flow conservation: For all $u \in V - \{s, t\}$, flow in equals flow out

26.2 The Ford-Fulkerson method

Residual networks

How to find a flow *f'* in G_f?



26.2 The Ford-Fulkerson method

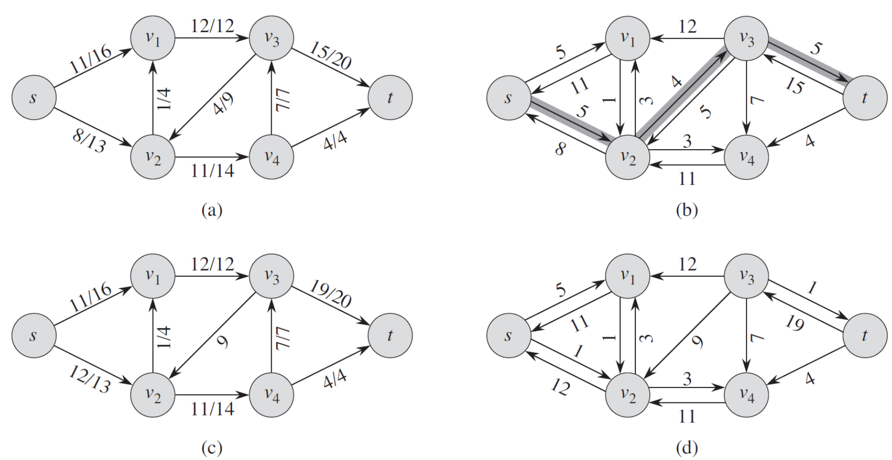
Augmenting paths（增广路径）

- An **augmenting path *p*** is a simple path from s to t in the residual network G_f .
- ***residual capacity*** of p : the maximum amount by which we can increase the flow on each edge in the augmenting path p .

  $c_f\,(p) = \min\,\{c_f(u, v) : (u, v) \text{ is on } p\}$. （容量最小的那条边 (u, v)， 也称为关键边）

- **Lemma 26.2** Let $G = (V, E)$ be a flow network, let f be a flow in G, and let p be an augmenting path in G_f. Define a function $f_p : V \times V \rightarrow \mathbf{R}$ by

$$f_p(u, v) = \begin{cases} c_f(p) & \text{if } (u, v) \text{ is on } p\,, \\ 0 & \text{otherwise}\,. \end{cases}$$



Then, f_p is a flow in G_f

with value $|f_p| = c_f\,(p) > 0$.

残留网络上的流

Proof : verify two properties of flow…

Augmenting paths

$$f_p(u,v) = \begin{cases} c_f(p) & \text{if } (u,v) \text{ is on } p \,, \\ 0 & \text{otherwise} \,. \end{cases}$$

Corollary 26.3 Let $G = (V, E)$ be a flow network, let f be a flow in G, and let p be an augmenting path in G_f. Then the function $(f \uparrow f_p)$: $V \times V \rightarrow \mathbf{R}$, is a flow in G with value $|f| + |f_p| > |f|$.

Proof **:**

Immediately,

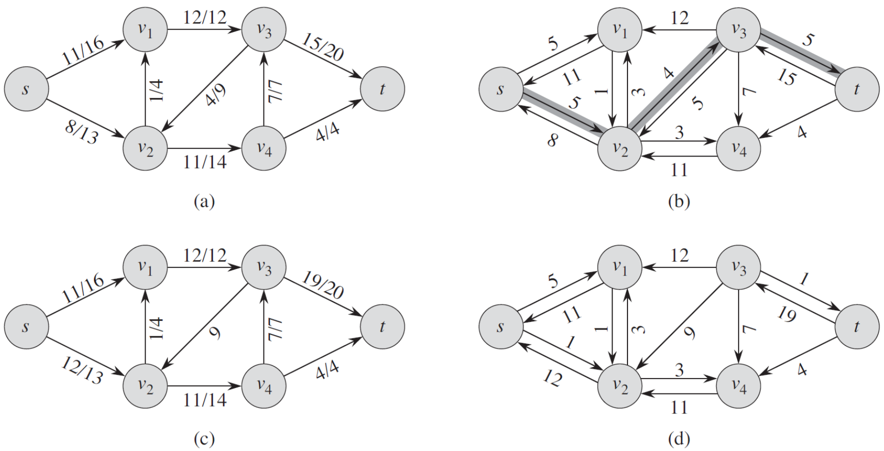
from Lemmas 26.2 and 26.1,

{

Lemmas 26.2: f_p is a flow in G_f.

Lemmas 26.1: $f + f_p$ is a flow in G.

}

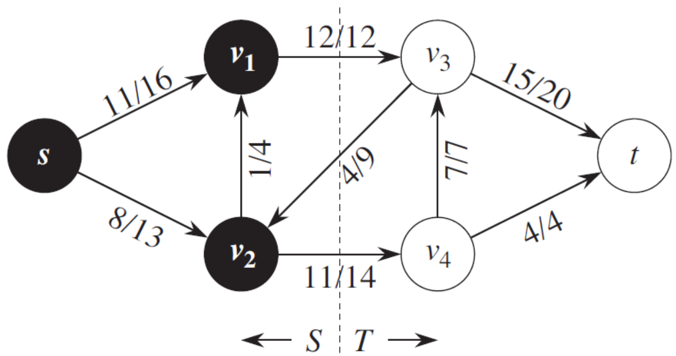


26.2 The Ford-Fulkerson method

Cuts of flow networks

- A ***cut*** (S, T) of flow network $G = (V, E)$ is a partition of V into S and $T = V - S$ such that $s \in S$ and $t \in T$. (source is in S, sink is in T.)
- If f is a flow, then the ***net flow*** across the cut (S, T) is defined to be $f(S, T)$.

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



- The ***capacity*** of the cut (S, T) is $c(S, T)$.

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

注意：不含
反向容量

- A ***minimum cut*** of a network is a cut whose capacity is minimum over all cuts of the network.（一个网络的最小割是网络中具有最小容量的割）

$f(S, T) = 19$

$c(S, T) = 26$

26.2 The Ford-Fulkerson method

Cuts of flow networks

Lemma 26.4 Let f be a flow in a flow network G, and let (S, T) be any cut of G. Then the net flow across (S, T) is $f(S, T) = |f|$.

(任意割的容量都相等)

Proof ...略（根据流的定义与流守恒性质来证明，证明略）

流的定义

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

$\Leftrightarrow$

切割的净流定义

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

流守恒性质

$$\sum_{v \in V} f(v, u) = \sum_{v \in V} f(u, v) \quad \left(\sum_{v \in V} f(u, v) - \sum_{v \in V} f(v, u) = 0 \right)$$

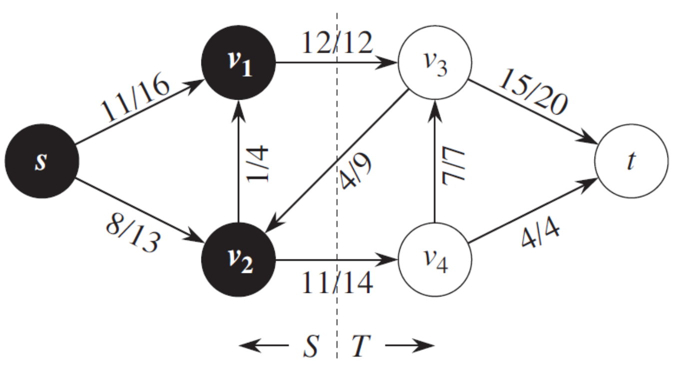
Cuts of flow networks

- **Lemma 26.4** Let f be a flow in a flow network G, and let (S, T) be any cut of G. Then the net flow across (S, T) is $f(S, T) = |f|$.
- **Corollary 26.5** The value of any flow f in a flow network G is bounded from above by the capacity of any cut of G.

  (任意割的容量都是流的上界)

  Proof 很显然。根据切割的净流与容量的定义来证明。

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



26.2 The Ford-Fulkerson method

Cuts of flow networks

Theorem 26.6: (Max-flow min-cut theorem)

If f is a flow, then the following conditions are equivalent:

1. f is a maximum flow in G.
2. The residual network G_f contains no augmenting paths.
3. $|f| = c(S, T)$ for some cut (S, T) of G.

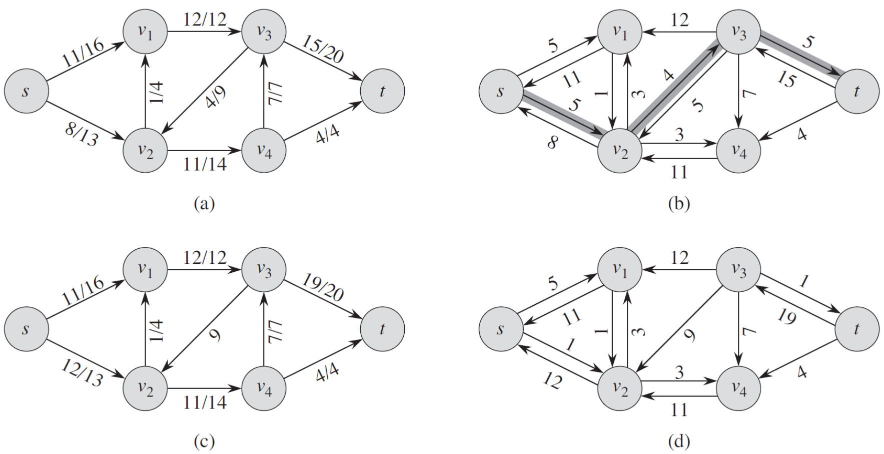
Proof …

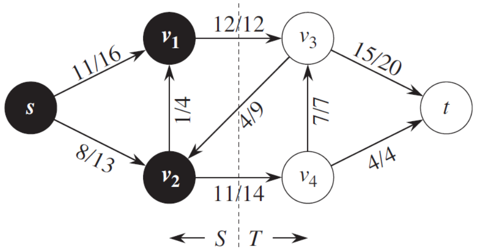
证明思路:

1=>2

2=>3

3=>1





26.2 The Ford-Fulkerson method

Cuts of flow networks

Theorem 26.6: (Max-flow min-cut theorem)

If f is a flow, then the following conditions are equivalent:

1. f is a maximum flow in G.
2. The residual network G_f contains no augmenting paths.
3. $|f| = c(S, T)$ for some cut (S, T) of G.

最大流求解算法:

① 流网络比较小时：穷举出所有切割(cut)，求出最小cut。

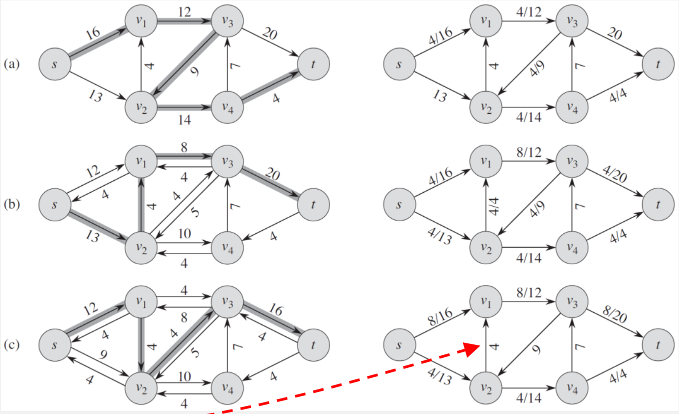
② 流网络比较大时：求残留网络，找增广路径（求路径上的残留容量），在流网络中沿增广路径压入残留（剩余）容量。

26.2 The Ford-Fulkerson method

The basic Ford-Fulkerson algorithm

FORD-FULKERSON(G, s, t)

for each edge $(u, v) \in E$
    $f[u, v] \leftarrow 0$
while there exists a path p from s to t in the residual network G_f
    $c_f(p) \leftarrow \min \{c_f(u, v) : (u, v)$ is in $p\}$
    for each edge (u, v) in p
        if $(u, v) \in E$
            $f[u, v] \leftarrow f[u, v] + c_f(p)$
        else $f[v, u] \leftarrow f[v, u] - c_f(p)$



算法：求残留网络G_f，找增广路径p，求路径上的残留容量$c_f(p)$，在流网络中沿增广路径在每条边上压入残留（剩余）容量。

Ford-Fulkerson Algorithm

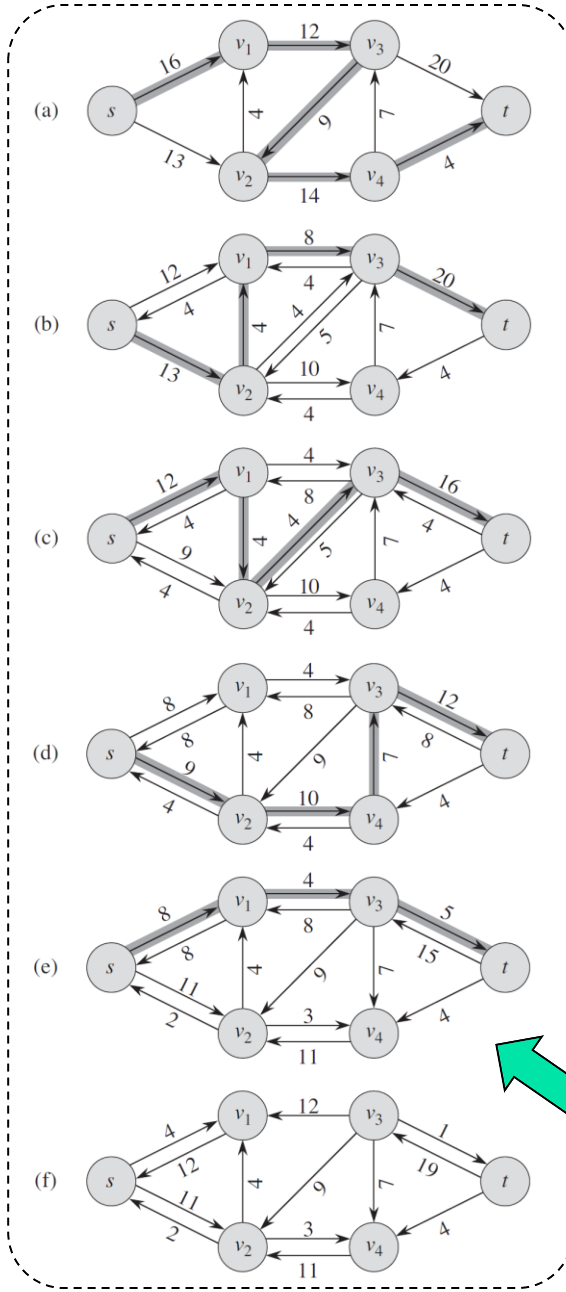
虚线框里的是residual network

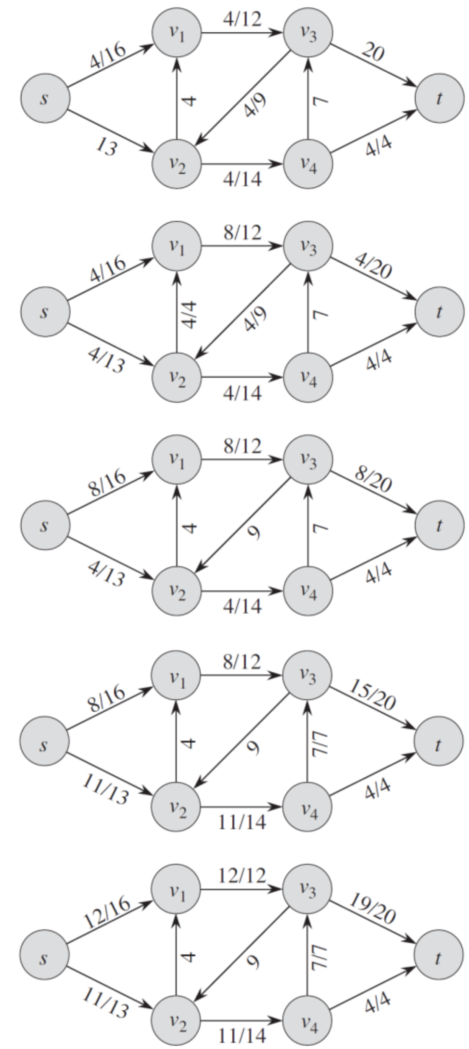
1. 初始，图a-L，流f为0，增广路径的残留容量为4；
2. 图a-R，沿增广路径可压入流4，图中的流为4；
3. 图b-L是图a-R的残留网络，图b-L的一个增广路径的残留容量是4；
4. 在流网络图a-R的基础上，沿图b-L的增广路径可压入流4，得到图b-R；

……

图f中不存在增广路径（不能再增加流，因此图e-2中的流就是最大流。

增广路径选取方法不同，计算效率不同。

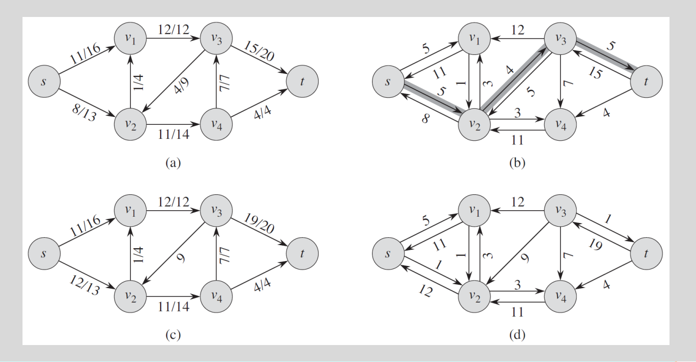




The residual network G_f.
A shaded: augmenting path p.

26.2 The Ford-Fulkerson method

Analysis of Ford-Fulkerson



FORD-FULKERSON(G, s, t) **$O(E\cdot f^*)$**

```
for each edge (u, v) ∈ E
    f[u, v] ← 0
while there exists a path p from s to t in the residual network G_f
    c_f(p) ← min {c_f(u, v) : (u, v) is in p}
    for each edge (u, v) in p
        if (u, v) ∈ E
            f[u, v] ← f[u, v] + c_f(p)
        else f[v, u] ← f[v, u] - c_f(p)
```

设最大流为 f^*：

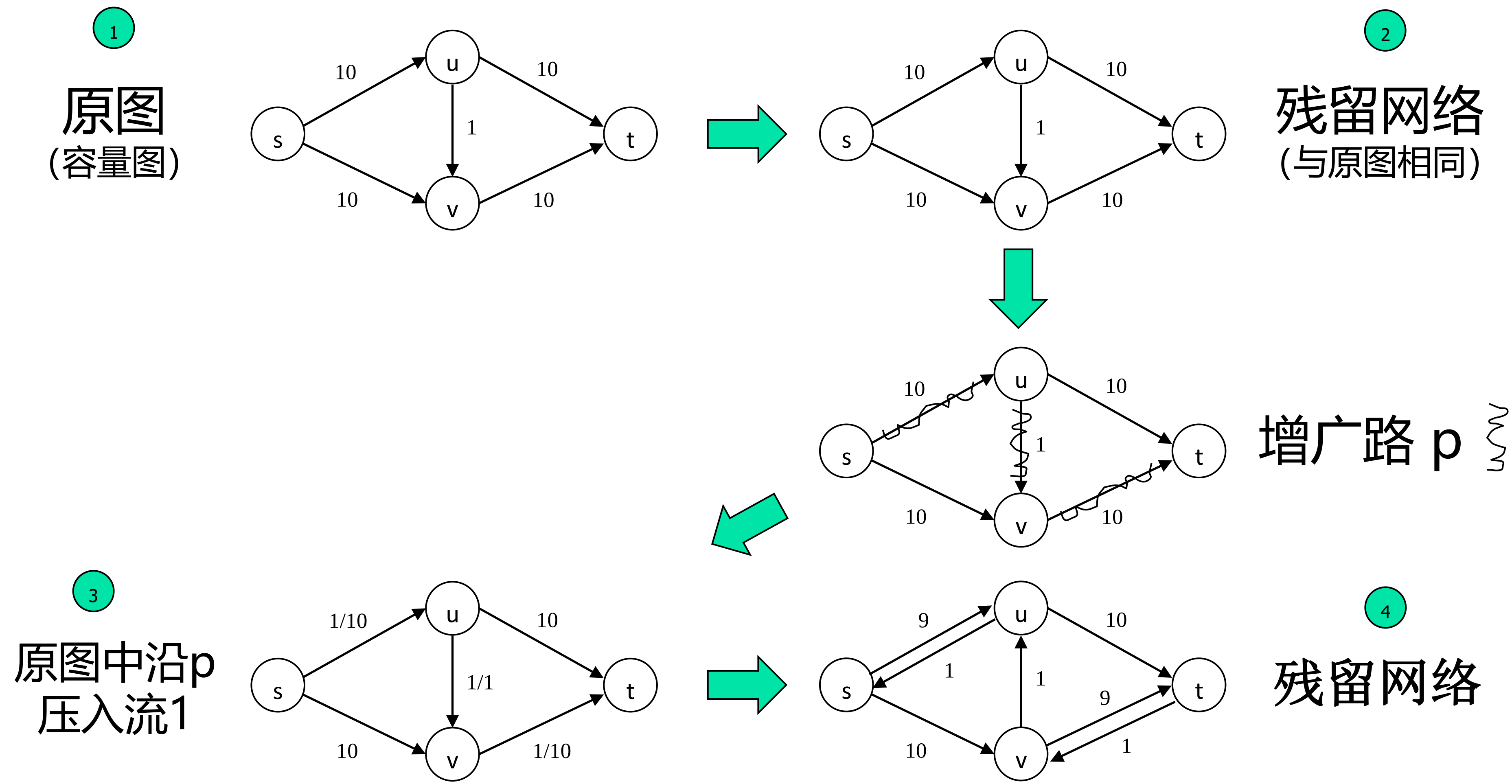
每次找到增广路径，流至少增加1，流从0增加到f^*，时间为O(f^*)；

每次找增广路径和给边增加流的操作，时间为O(E)；

总的时间， $O(E\cdot f^*)$

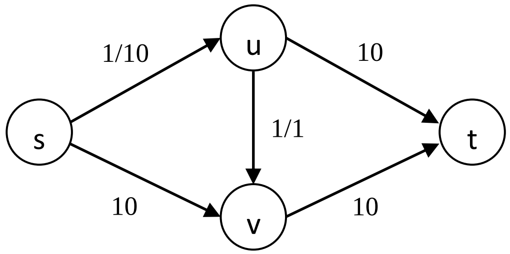
When the capacities are integral and the optimal flow value f^* is small, the running time of the F-F algorithm is good.

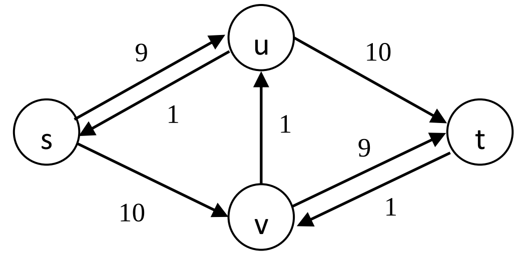
If *f** is large? an example...



3

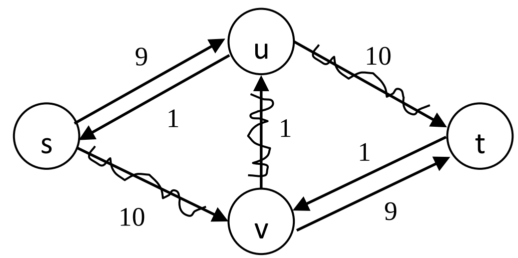
原图中沿p
压入流1





4

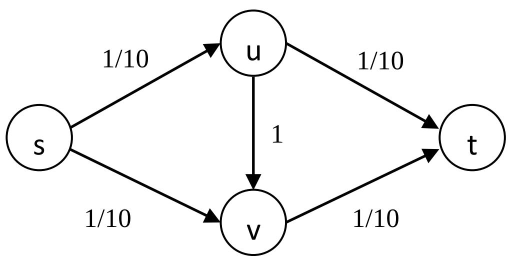
残留网络

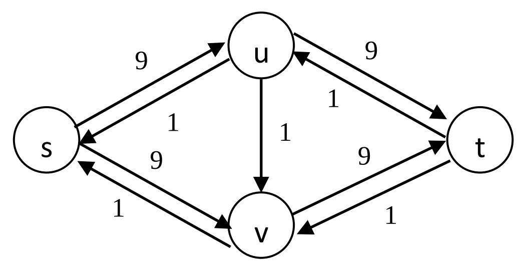


增广路 p

5

上一流图中
沿p压入流1



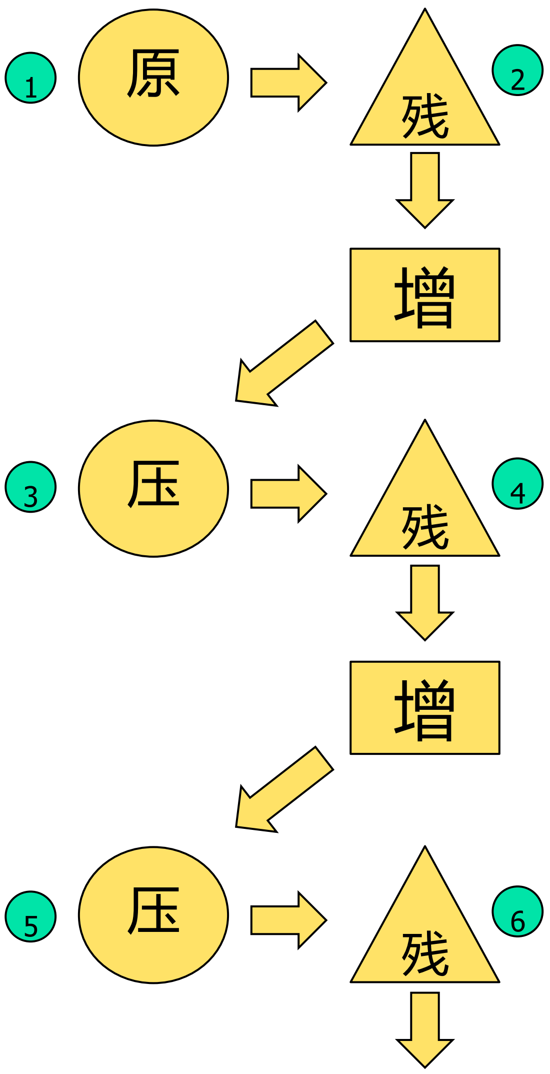


6

残留网络

If *f is large?**
an example...

If the optimal flow value *f is large, the F-F algorithm is not good.**



26.2 The Ford-Fulkerson method

FORD-FULKERSON(G, s, t)

1 **for** each edge $(u, v) \in E$
2 $\quad f[u, v] \leftarrow 0$
3 **while** there exists a path p from s to t in the residual network G_f
4 $\quad c_f(p) \leftarrow \min \{c_f(u, v) : (u, v) \text{ is in } p\}$
5 $\quad$ **for** each edge (u, v) in p
6 $\qquad$ **if** $(u, v) \in E$
7 $\qquad\quad f[u, v] \leftarrow f[u, v] + c_f(p)$
8 $\qquad$ **else** $f[v, u] \leftarrow f[v, u] - c_f(p)$

$O(E \cdot f^*)$

The Edmonds-Karp algorithm

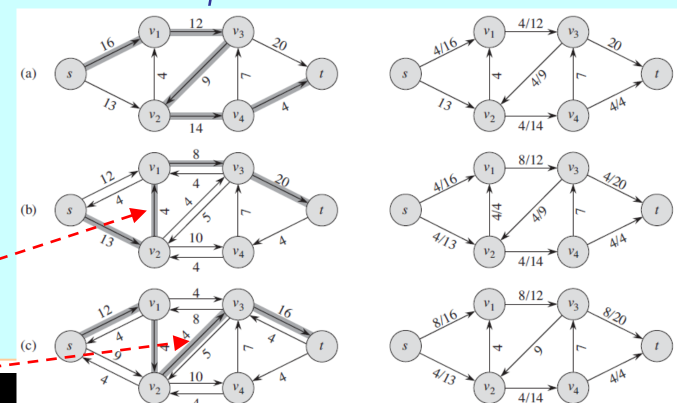
We can improve the bound on F-F by finding the augmenting path p in line 3 with **a breadth-first search**. That is, we choose p as a shortest path from s to t in the residual network, where each edge has unit distance (weight). We call the F-F method so implemented the **Edmonds-Karp algorithm**. The E-K algorithm runs in $O(VE^2)$ time. *Proof* …?

26.2 The Ford-Fulkerson method

EDMONDS-KARP(G, s, t)

$O(V \cdot E^2)$

for each edge $(u, v) \in E$
  $f[u, v] \leftarrow 0$
while there exists a path p from s to t in the residual network G_f **(using BFS)**
  $c_f(p) \leftarrow \min\{c_f(u, v) : (u, v) \text{ is in } p\}$
  for each edge (u, v) in p
    if $(u, v) \in E$
      $f[u, v] \leftarrow f[u, v] + c_f(p)$
    else $f[v, u] \leftarrow f[v, u] - c_f(p)$



证明思想：关键边（增广路径p上的最小容量边）。

沿着p增加流一次，关键边消失；边(u, v)最多O(V)次作为关键边；共E条边；E-K算法执行中的关键边数量$O(V \cdot E)$ **(关键边全部消失，不再有增广路径，最大流找到)**。每次找增广路径和给边增加流的操作，时间为O(E)。总的时间， $O(V \cdot E^2)$

26.2 The Ford-Fulkerson method

Idea:

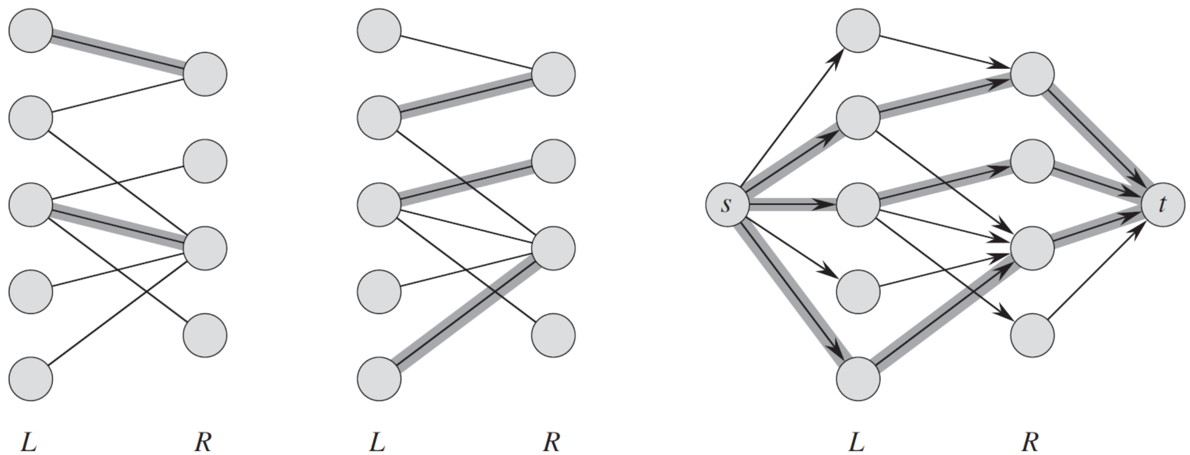
- **residual networks**
- **augmenting paths**
- **Cuts**

Method: The Ford-Fulkerson method

Algorithm: EK

Code:

26.3 Maximum bipartite matching



(a) (b) (c)

Practical applications

- L : machines ; R : tasks
- L : students ; R : scholarships
- L : students ; R : mentors
- L : gentlemen ; R : ladies
- …

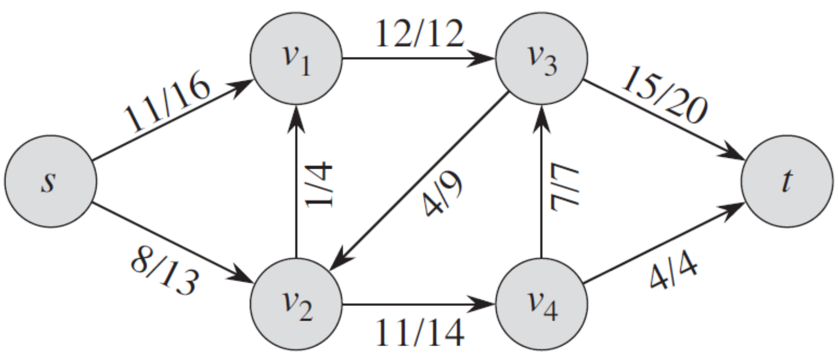
26.4 Push-relabel algorithms *

26.5 The relabel-to-front algorithm *

chapter 29 Linear Programming *

Exercise

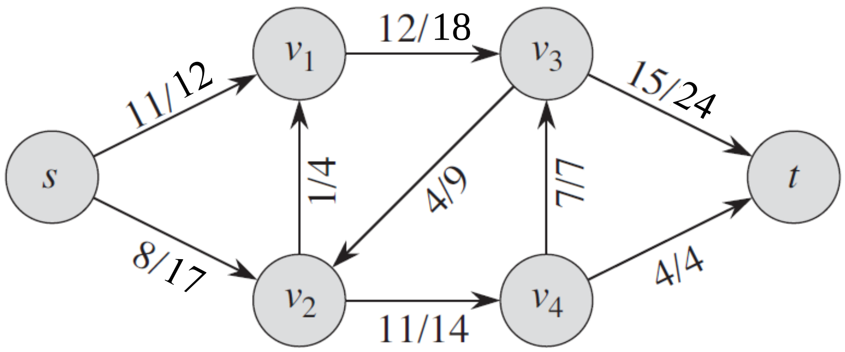
- **In Figure 26.1(b), what is the flow across the cut (S, T) = ({s , v_2 , v_4}, {v_1 , v_3 , t})? What is the capacity of this cut?**
- **What is the minimum cut to the figure? What is the maximum flow?**

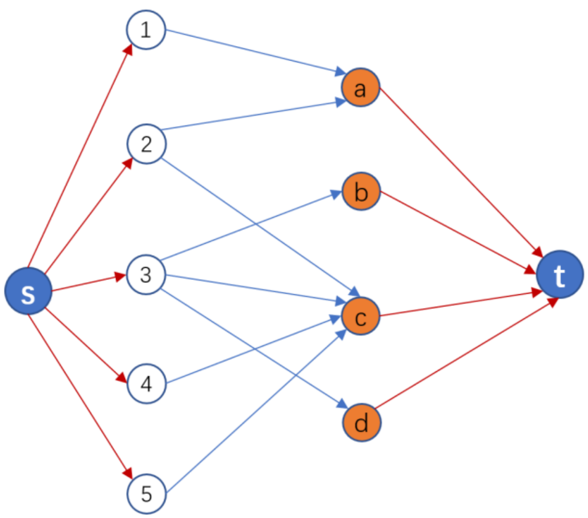


Exercise

In the followed Figures, for each,

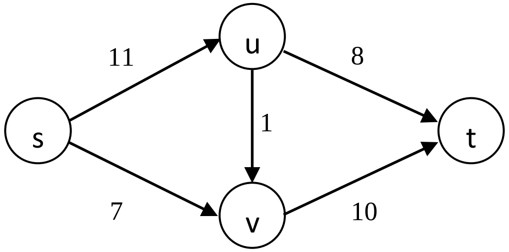
- **What is the <u>minimum cut</u>?**
- **What is the maximum flow?**

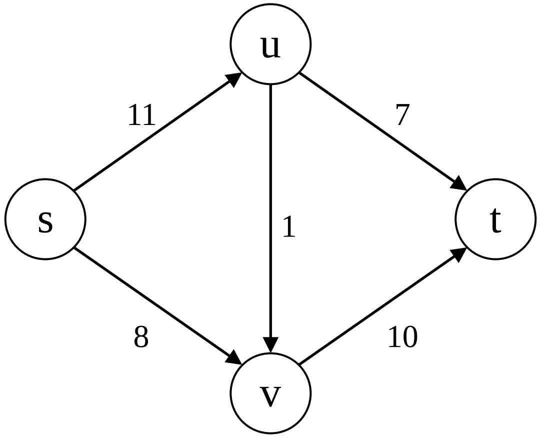


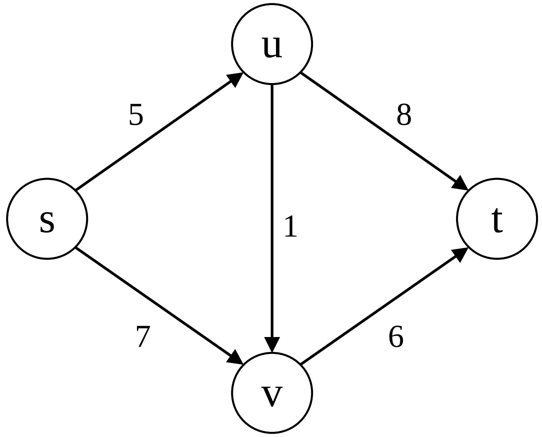


Exercise

What is the maximum flow?

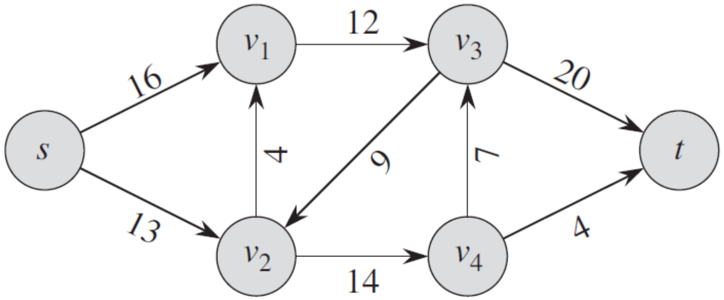






Exercise

采用E-K算法，画出左图的最大流求解过程。



EDMONDS-KARP(G, s, t)

for each edge $(u, v) \in E$

  $f[u, v] \leftarrow 0$

while there exists a path p from s to t in the residual network G_f **(using BFS)**

  $c_f(p) \leftarrow \min \{c_f(u, v) : (u, v) \text{ is in } p\}$

  for each edge (u, v) in p

    if $(u, v) \in E$

      $f[u, v] \leftarrow f[u, v] + c_f(p)$

    else $f[u, v] \leftarrow f[u, v] - c_f(p)$