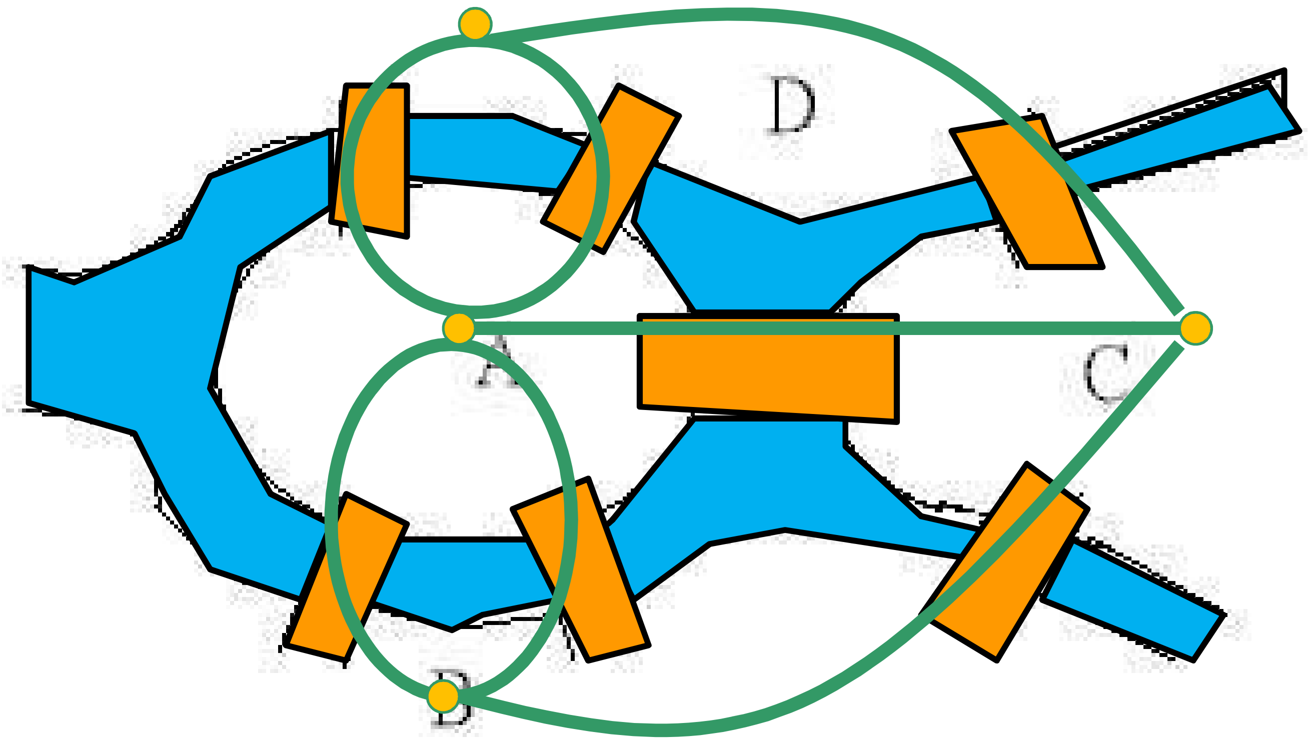
第八章 图与网络分析

- 图的基本概念
- 最短路径问题
- 最大流问题
- 最小费用流问题

图论起源

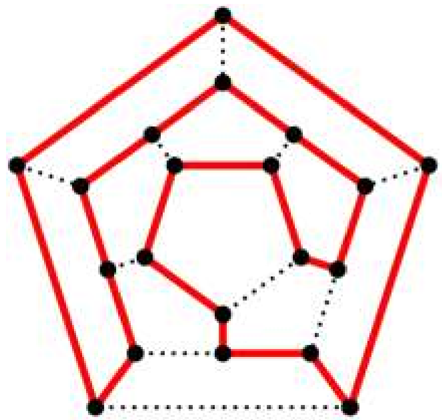
Euler **1736**年：柯尼斯堡七桥问题



图论发展

1859年，Hamilton"环球旅行 "问题

TSP (Travel Salesman Problem)旅行商问题



1936年，O Konig 有限图与无限图理论

1976， 地图着色问题"四色猜想"

现代分支众多，如代数图论、拓扑图论等

本课程主要讲"网络优化理论"

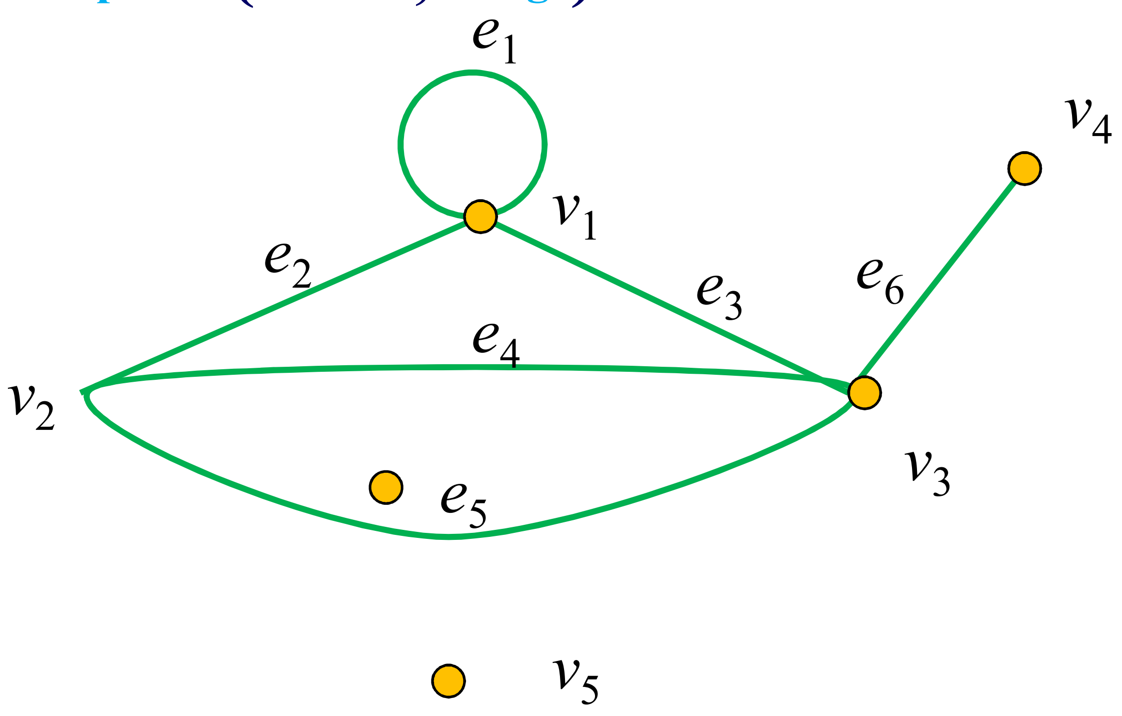
图的定义

■ 图 = 端点 + 边　　**G**raph = (**V**ertex, **E**dge)

➢ 端点 **V**={v_i}

➢ 边 **E**={e_i}

$e_i = (v_{i1}, v_{i2})$



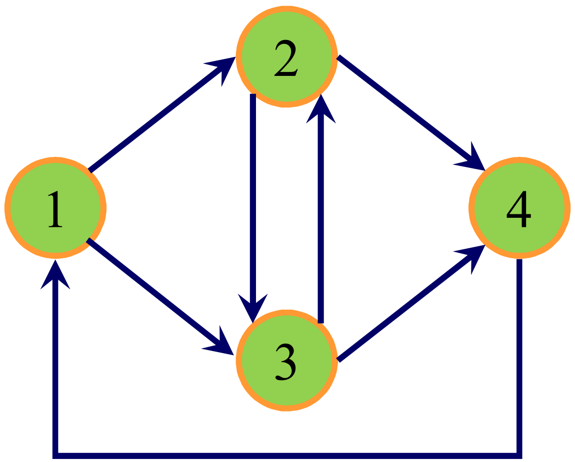
■ 无向图

■ 有向图，有向图的边也称为弧

邻接矩阵表示

- 邻接矩阵：

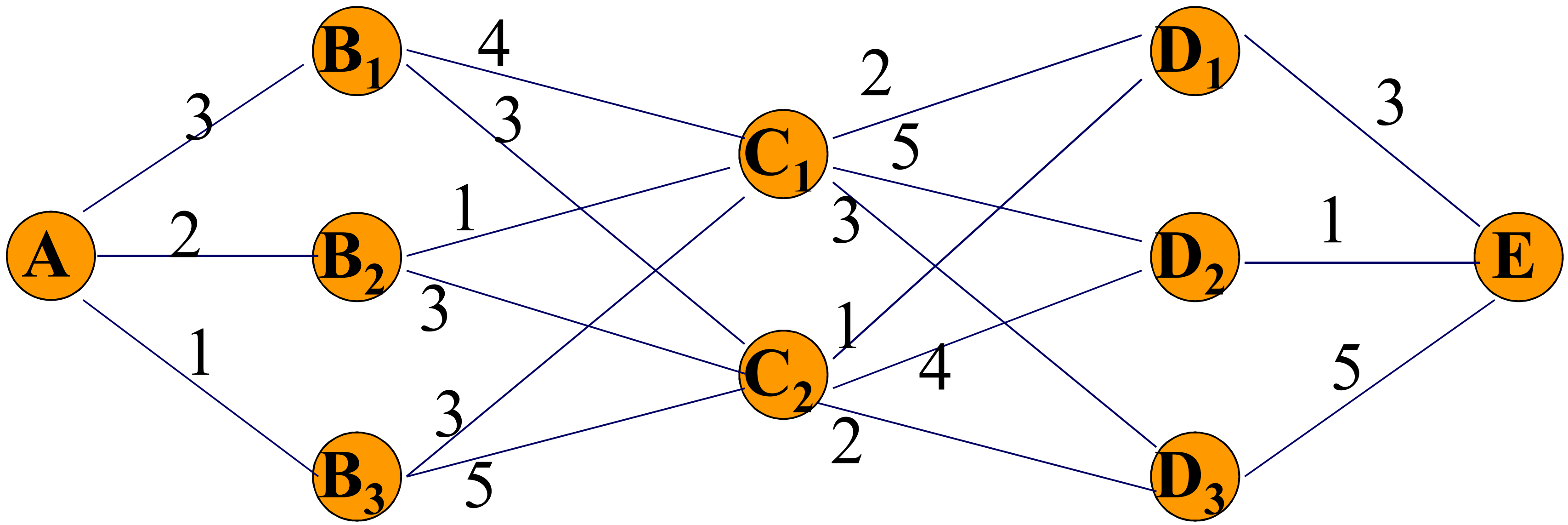
$$a_{ij}=\begin{cases}1 & (v_i,v_j)\in E\\ 0 & \text{其他}\end{cases}$$



$$\begin{bmatrix}0 & 1 & 1 & 0\\ 0 & 0 & 1 & 1\\ 0 & 1 & 0 & 1\\ 1 & 0 & 0 & 0\end{bmatrix}$$

网络

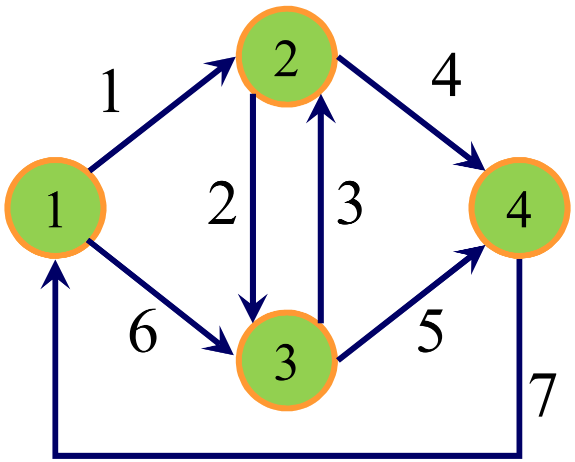
■网络：点或者边带权的图（赋权图）



权矩阵

- 权矩阵：

$$a_{ij}=\begin{cases} w_{ij} & (v_i,v_j)\in E \\ 0\text{或}\infty & \text{其他} \end{cases}$$

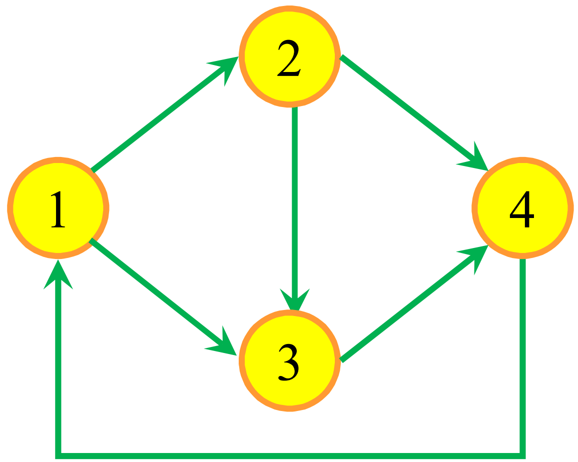


$$\begin{bmatrix} 0 & 1 & 6 & \infty \\ \infty & 0 & 2 & 4 \\ \infty & 3 & 0 & 5 \\ 7 & \infty & \infty & 0 \end{bmatrix}$$

链与道路

点、边交替序列，有$e_{it} = (v_{it-1}, v_{it})$ t=1,2,…,k

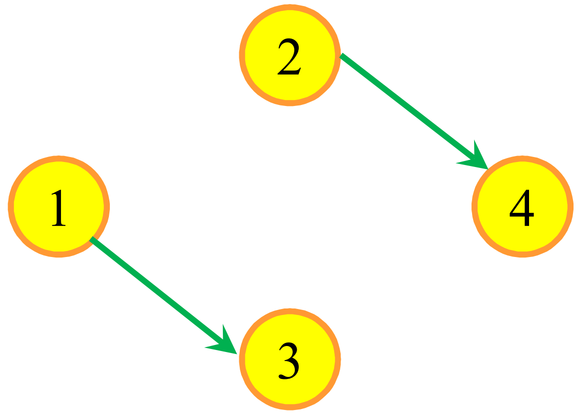
- 链：无方向要求 {(1,2) (2,3) (3,1) (1,2) (2,4)}
  - 圈：$v_{i0} = v_{ik}$ {(1,2) (2,3) (3,1)}
- 道路：方向一致{(1,2) (2,3) (3,4)}
  - 回路：$v_{i0} = v_{ik}$ {(1,2) (2,4) (4,1)}



连通图

■连通图：任意两点之间至少有**1**条链（道路）相连

■分图：不连通图中的连通子图

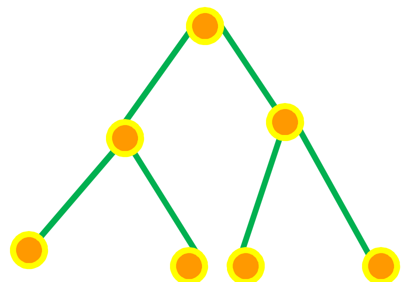


树的定义

■树： 不含圈的连通无向图 **T = (V, E)**

- 叶：次为**1**
- 分枝点：次大于**1**

■次 **deg(*v*)**、**d(*v*)**： 以*v*为端点的边数



树的性质

- 树T = (V, E)，$|V|=n$, $|E|=m$，则下列说法等价：

（1）T是一个树

（2）T无圈，且$m=n-1$

（3）T连通，且$m=n-1$

（4）T无圈，但每加一个新边，可得唯一的圈

（5）T连通，但每舍去一条边即不连通

（6）T中任意两点，有唯一的链相连

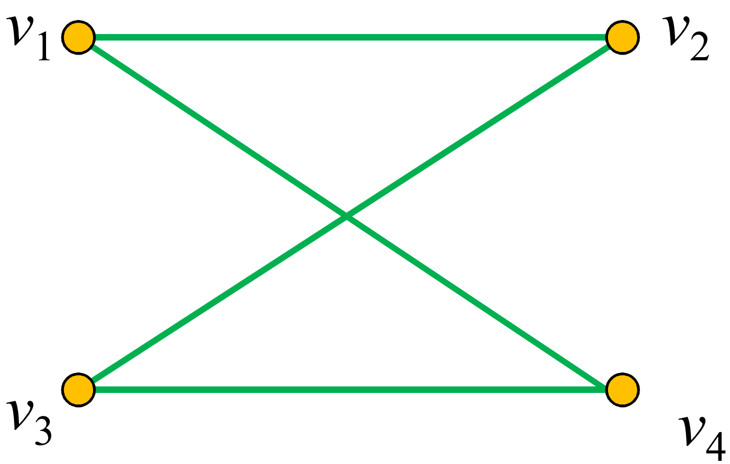
$$
\begin{array}{ccccc}
(1) & \Rightarrow & (2) & \Rightarrow & (3) \\
\Uparrow & & & & \Downarrow \\
(6) & \Leftarrow & (5) & \Leftarrow & (4)
\end{array}
$$

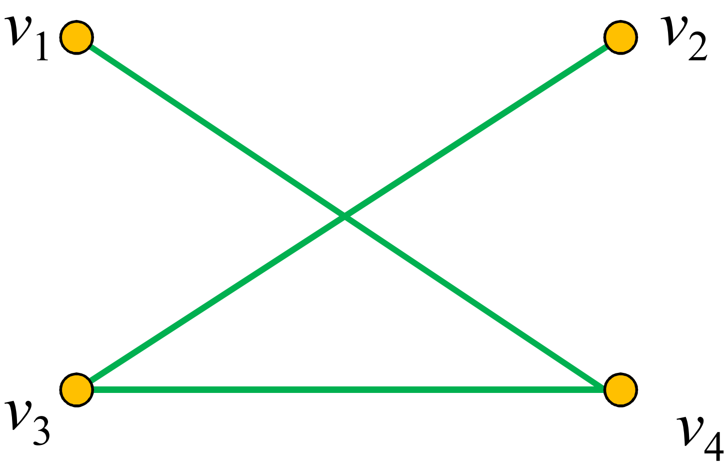
生成子图

■ 子图G′ = (V′, E′)

V′ ⊆ V　　E′ ⊆ E　　V′ 的边仅与E′ 的点相关联

■ 生成子图G′：V′ = V

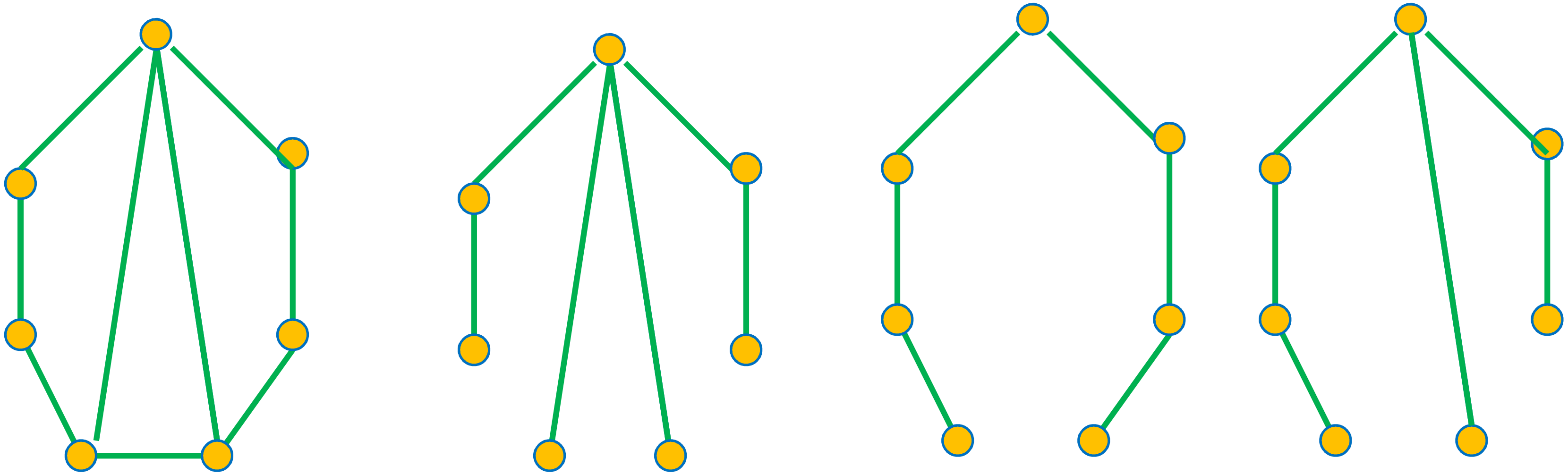




生成树

■生成树：连通图G的生成子图是1棵树（ V '=V）

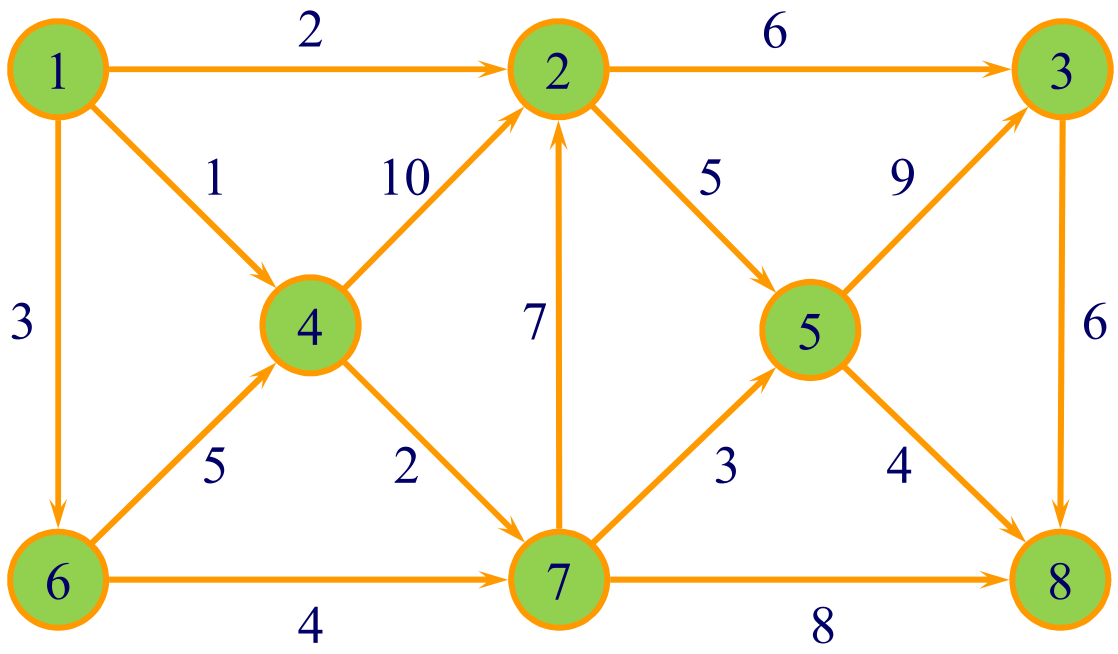
- 树枝：生成树中的边
- 弦：不在生成树中的边



第八章 图与网络分析

- 图的基本概念
- 最短路径问题
  - **Dijkstra**算法
  - 逐次逼近法
  - **Floyd**算法
- 最大流问题
- 最小费用流问题

最短路问题



求从**1**到**8**的最短路径

数学模型1

$$\min z = \sum_{(i,j)\in E} c_{ij} x_{ij}$$

s.t.

$$\sum_{(1,j)\in E} x_{1j} = 1 \qquad \sum_{(i,n)\in E} x_{in} = 1$$

自然满足

$$\sum_{(k,i)\in E} x_{ki} = \sum_{(i,j)\in E} x_{ij} \qquad i = 2, \cdots n-1$$

$$x_{ij} = 1或0$$

数学模型2

$$\min z=\sum_{i=1}^{n}\sum_{\substack{j=1\\ i\neq j}}^{n}c_{ij}x_{ij}$$

全连接网络

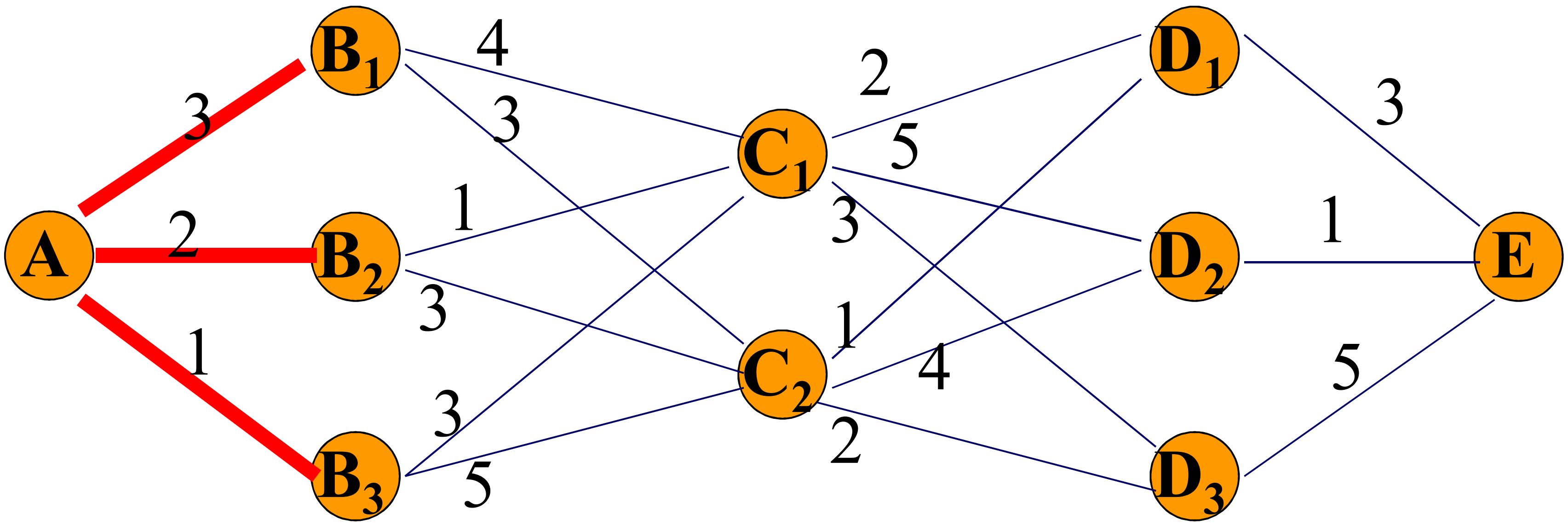
s.t.

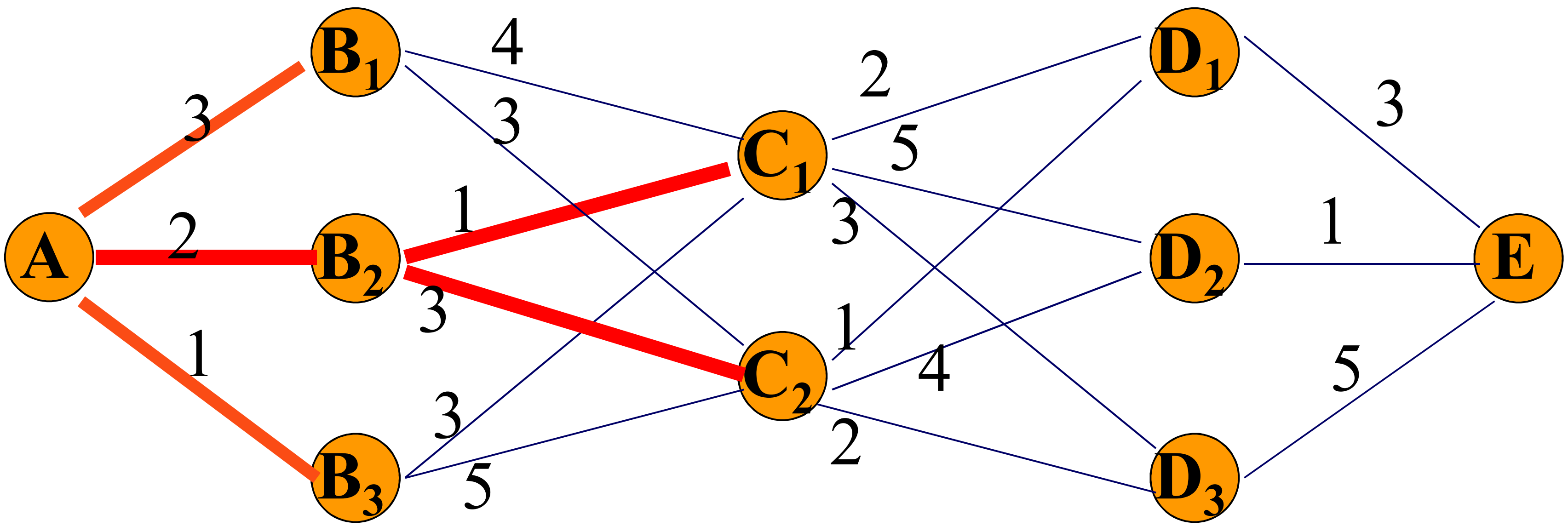
$$\sum_{j=2}^{n}x_{1j}=1$$

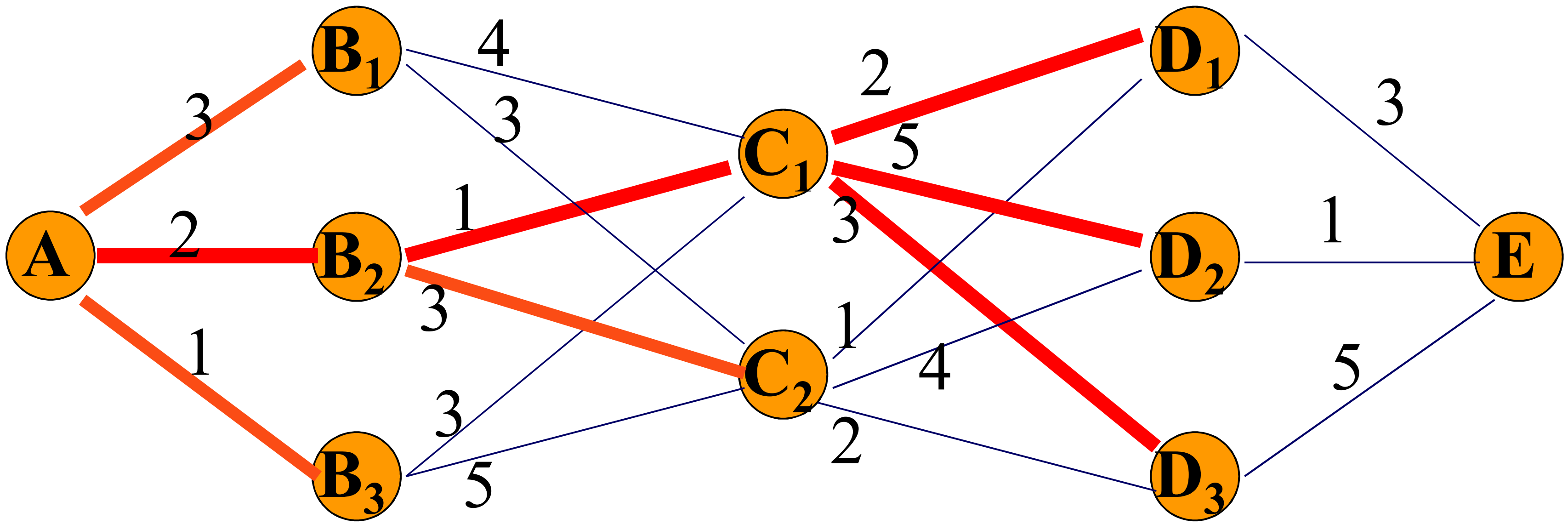
$$\sum_{\substack{k=1\\ k\neq i}}^{n}x_{ki}=\sum_{\substack{j=1\\ j\neq i}}^{n}x_{ij}\qquad i=2,\cdots n-1$$

$$x_{ij}=1\text{或}0$$

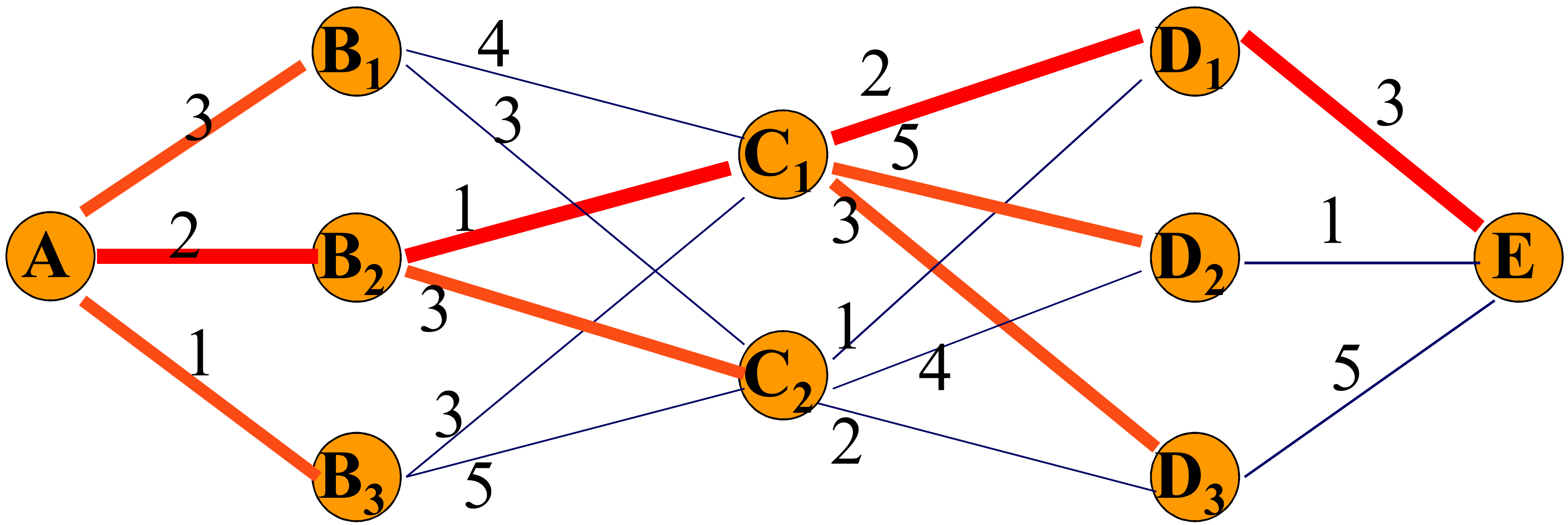
$$c_{ij}=\begin{cases}d_{ij} & (i,j)\in E\\ +\infty & (i,j)\notin E\end{cases}$$


A
B_1
B_2
B_3
C_1
C_2
D_1
D_2
D_3
E
3
2
1
4
3
1
3
3
5
2
5
3
1
4
2
3
1
5


A
B_1
B_2
B_3
C_1
C_2
D_1
D_2
D_3
E
3
2
1
4
3
1
3
3
5
2
5
3
1
4
2
3
1
5



A
B_1
B_2
B_3
C_1
C_2
D_1
D_2
D_3
E
3
2
1
4
3
1
3
3
5
2
5
3
1
4
2
3
1
5



A
B_1
B_2
B_3
C_1
C_2
D_1
D_2
D_3
E
3
2
1
4
3
1
3
3
5
2
5
3
1
4
2
3
1
5

动态规划的启示

1、新的最短路径节点必从已知的最短路径的节点展开（*最优性原理*）**;**

2、最短路径的搜索过程是一棵从起点展开的树。

Dijkstra算法的思路

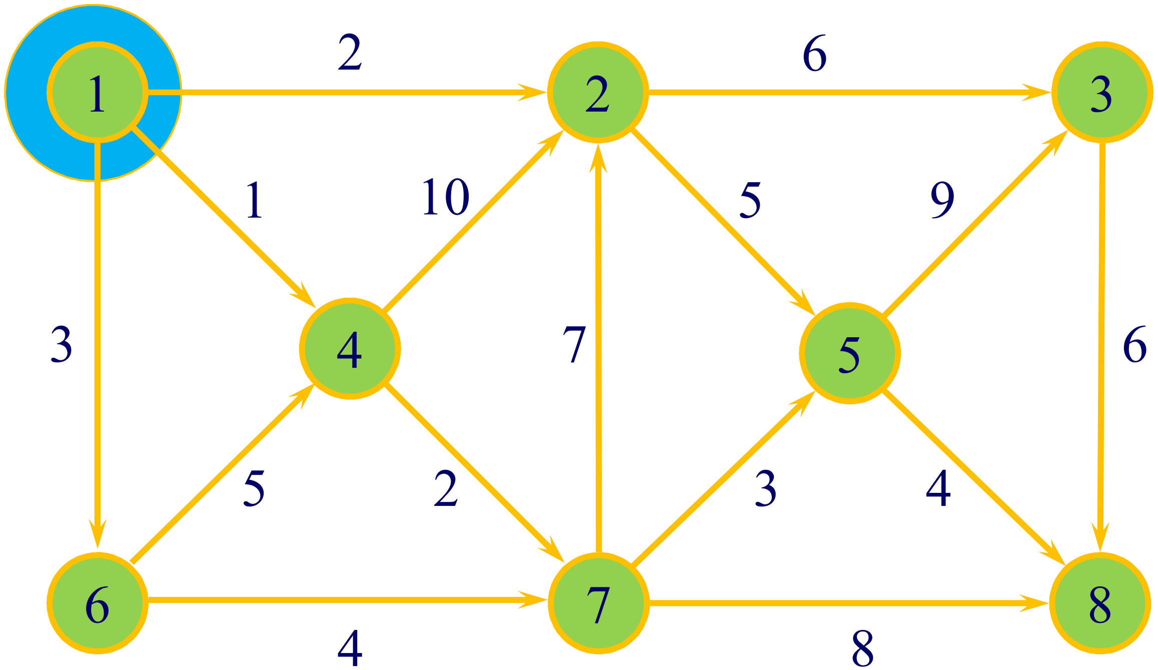
将所有的节点分为两类

1、找到了最短路径的节点，可作永久标号（***Permanent Label***），称为**P**点，**P**值为起点到该**P**点的最短路径。

2、没有找到最短路径的节点做暂定标号（**tentative label**），称为**T**点，**T**值为当前已搜索路径的最短长度（定界），用于过滤试探过程中不够短的路径。

问题：**T**点如何晋级为**P**点？

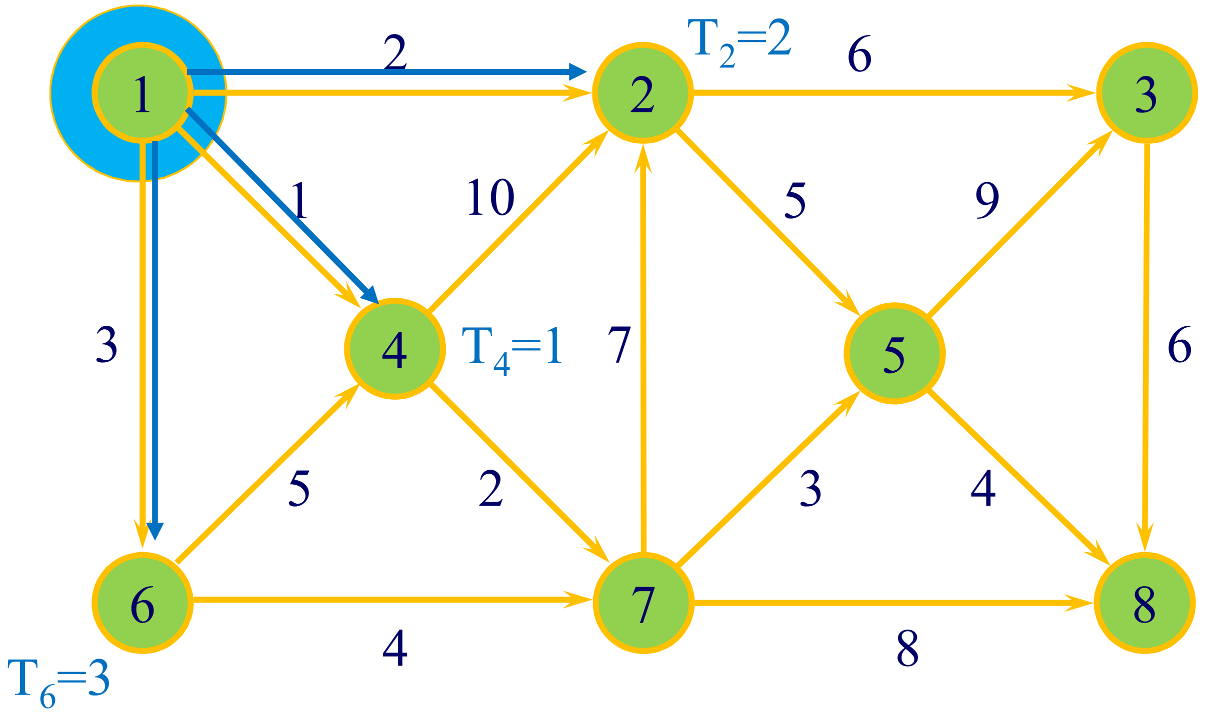
初始化



P点集合：X={1}，$\mathbf{P_1=0}$**；**

T点集合：Y={2,3,…,8}，$\mathbf{T_j=\infty,\ j\in Y}$

Step 1：T值更新



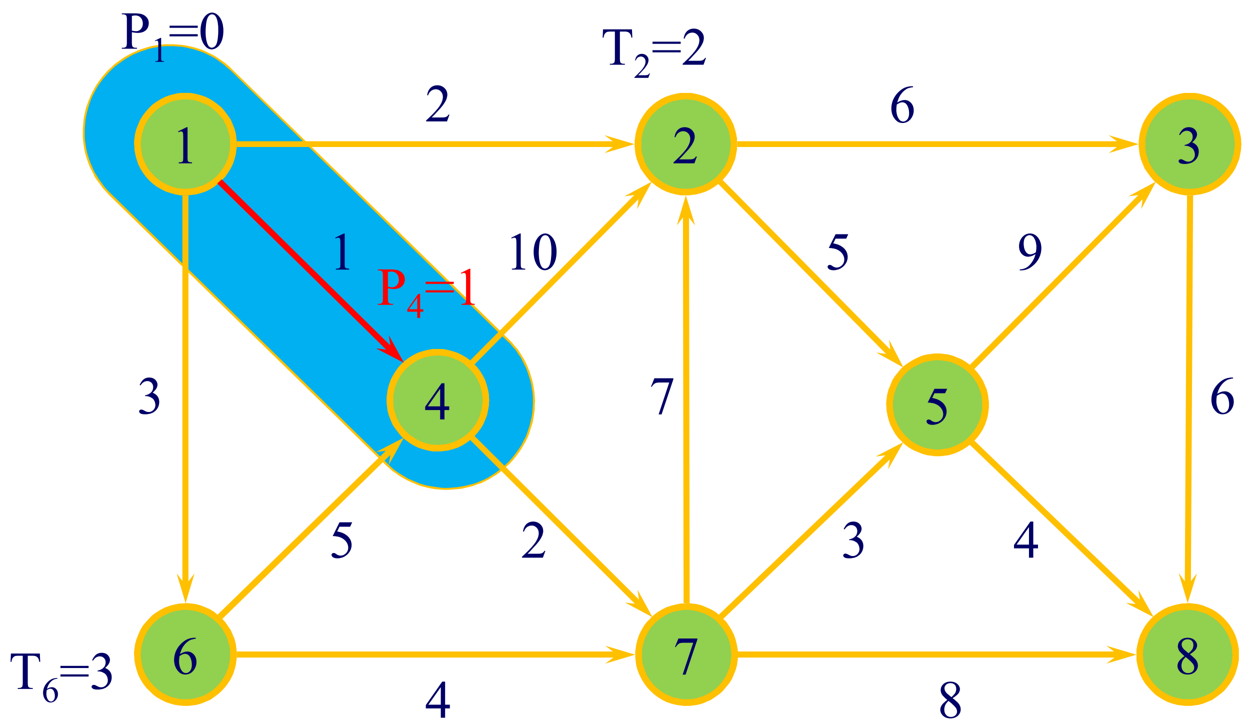
$T_2=\min\{P_1+l_{12}, T_2\}= 2$

$T_4=\min\{P_1+l_{14}, T_4\}= 1$

$T_6=\min\{P_1+l_{16}, T_6\}= 3$

问题：谁可作为新的P点？

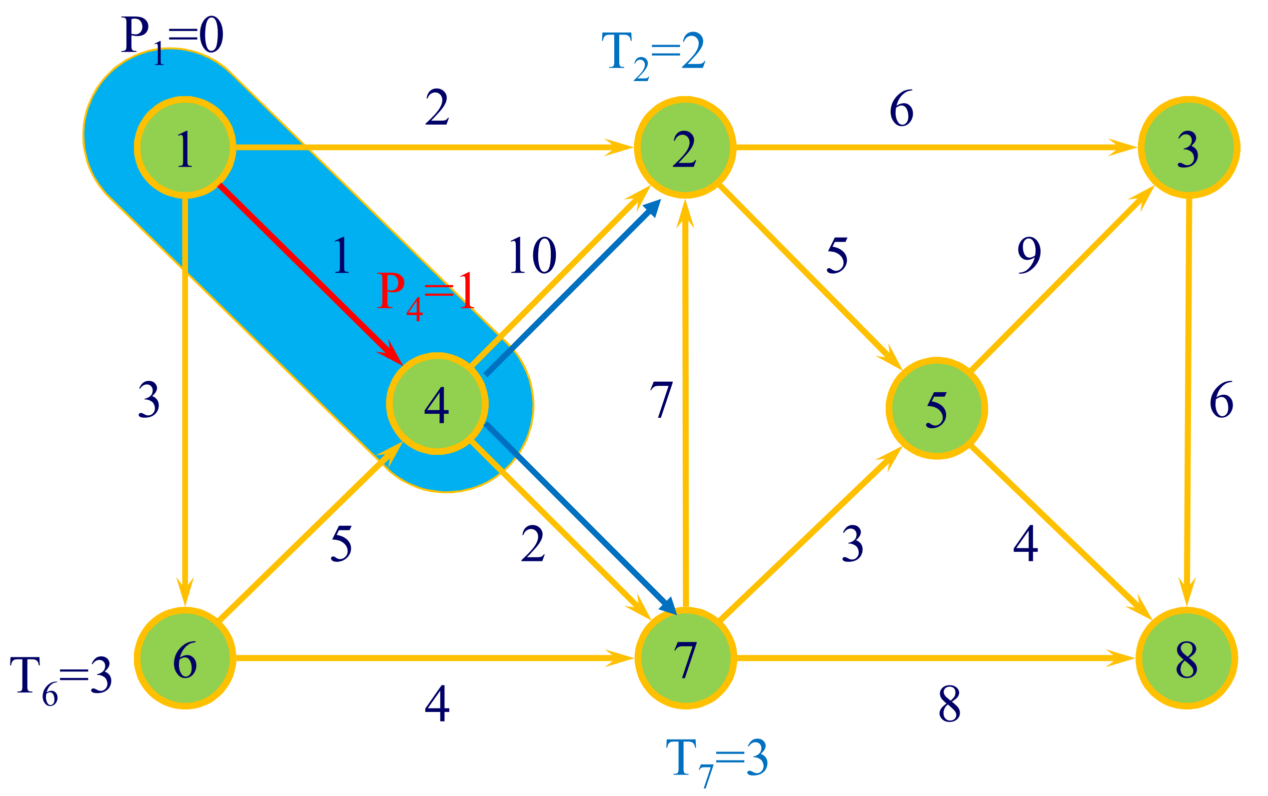
Step 1：P点搜索



Y={2,3,4,5,6,7,8}

min $\{T_i\}_{i\in Y}$ =min {2,∞,1, ∞, 3,∞,∞}=1； X={1,4}, P_4=1

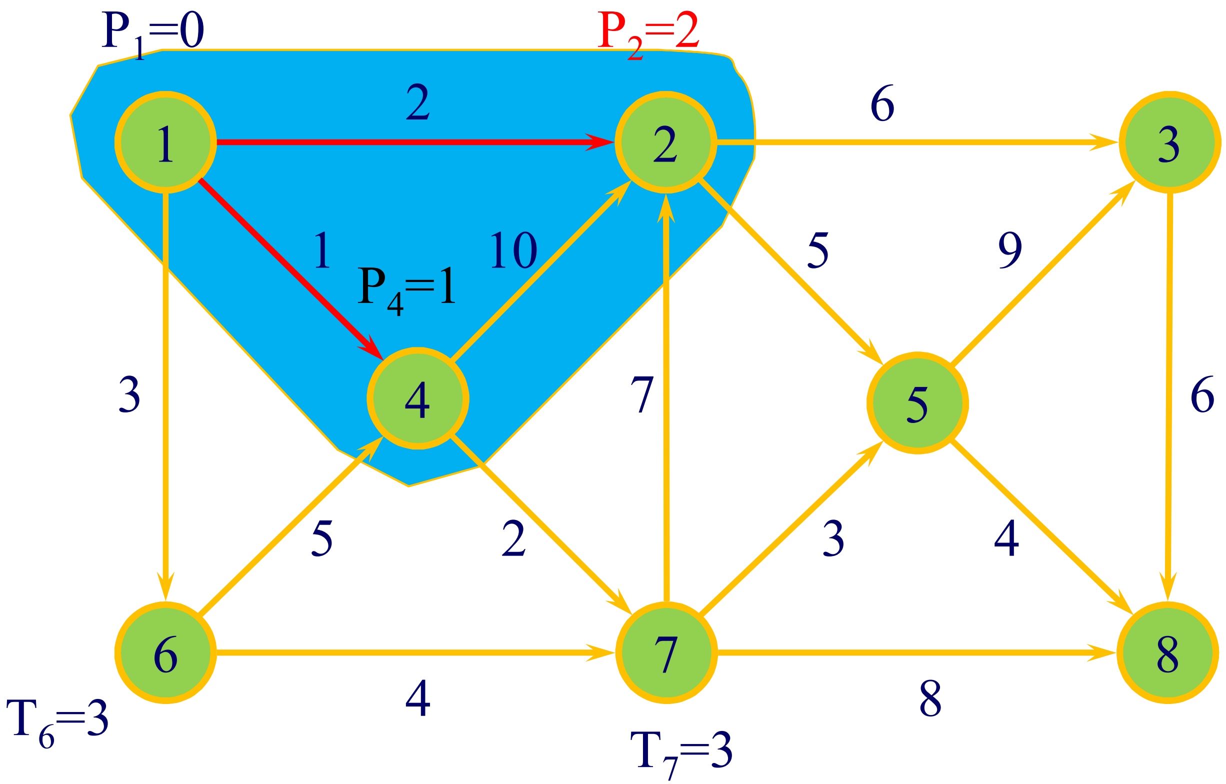
Step2：T值更新



T_2= min $\{P_4+l_{42}, T_2\}$=min $\{1+10, 2\}$=2

T_7=min $\{ P_4+l_{47}, T_7\}$=min $\{1+2, \infty\}$=3

Step2：P点搜索

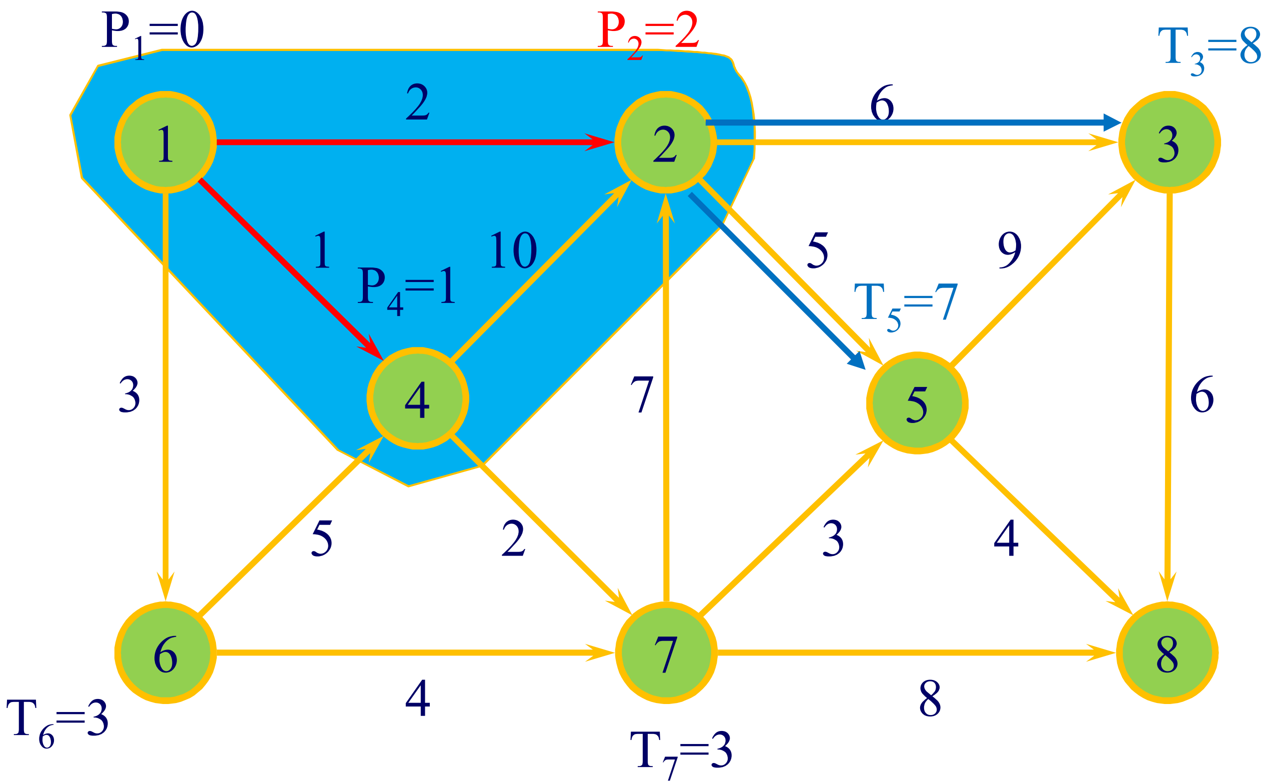


Y={2,3,5,6,7,8}

min $\{T_i\}_{i\in Y}$ =min {2,∞,∞, 3,3, ∞}=2

X={1,2,4,}, P_2=2

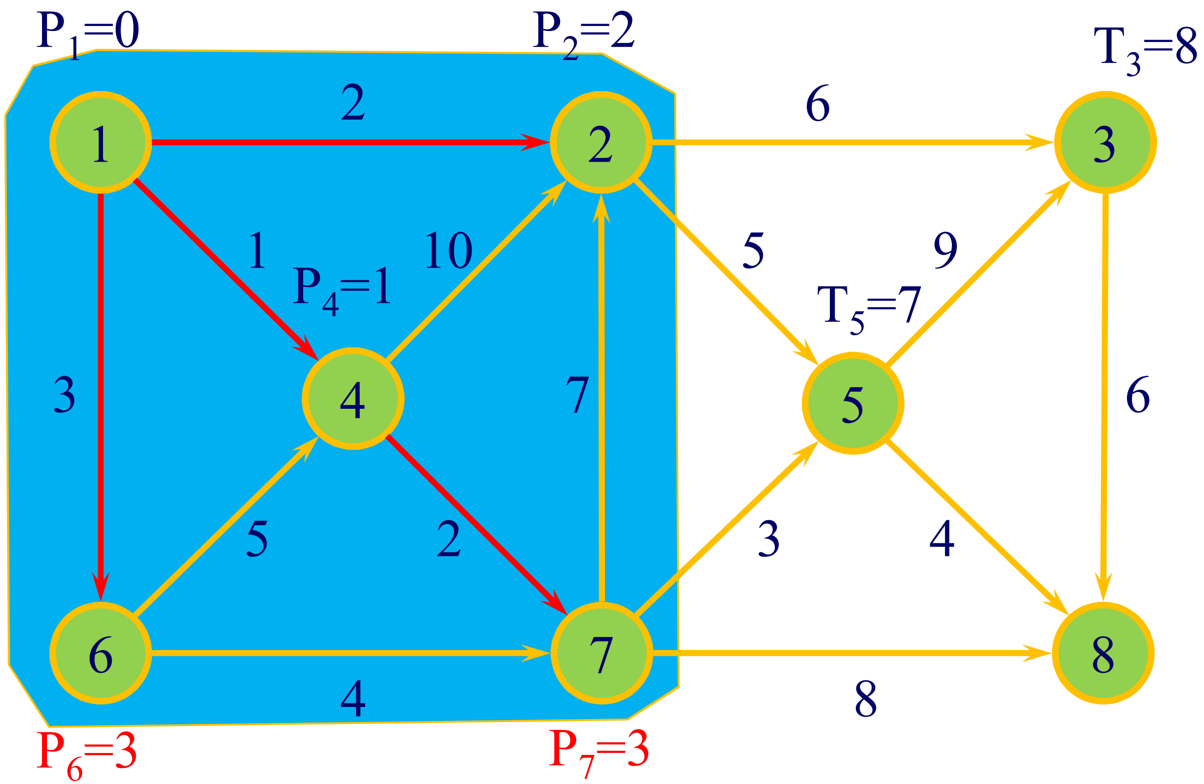
Step3：T值更新



T_3= min $\{P_2+l_{23}, T_3\}$=min $\{2+6, \infty\}$=8

T_5= min $\{P_2+l_{25}, T_5\}$=min $\{2+5, \infty\}$=7

Step3：P点搜索

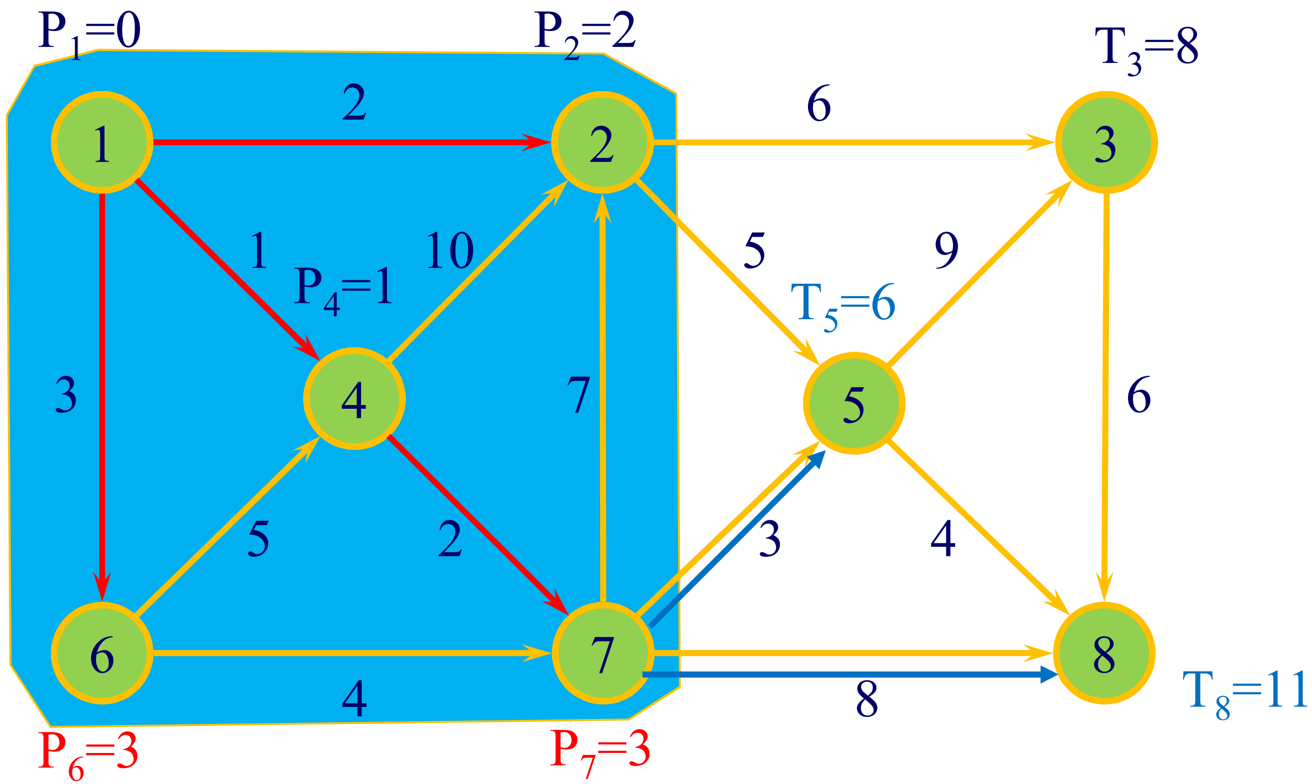


$Y=\{3,5,6,7,8\}$

$\min\ \{T_i\}_{i\in Y}=\min\ \{8,7,3,3,\infty\}=3$

$X=\{1,2,4,6,7\}$, $P_6=3$, $P_7=3$

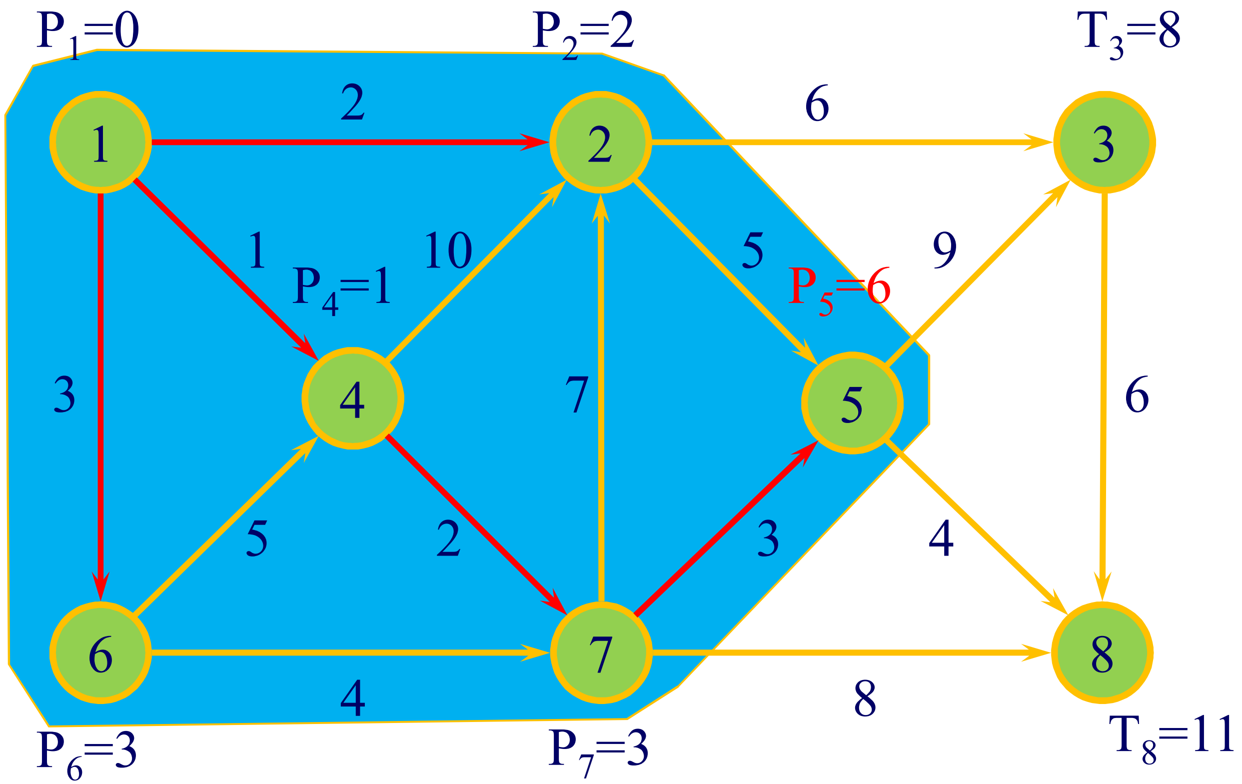
Step4：T值更新



T_5= min $\{P_7+l_{73}, T_5\}$=min $\{3+3, 7\}$=6

T_8= min $\{P_7+l_{73}, T_3\}$=min $\{3+8, \infty\}$=11

Step4：P点搜索

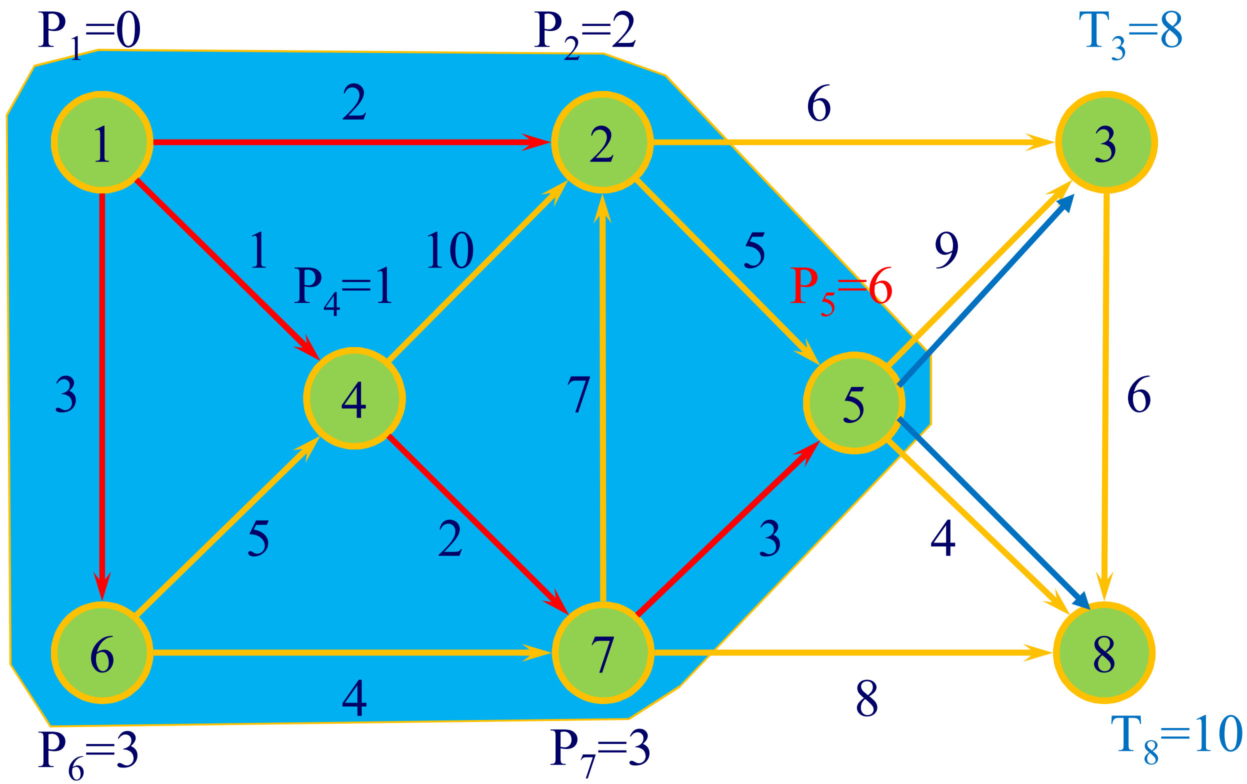


Y={3,5, 8}

min $\{T_i\}_{i\in Y}$ =min {8,6, 10}=6

X={1,2,4,5,6,7}, P_5=6

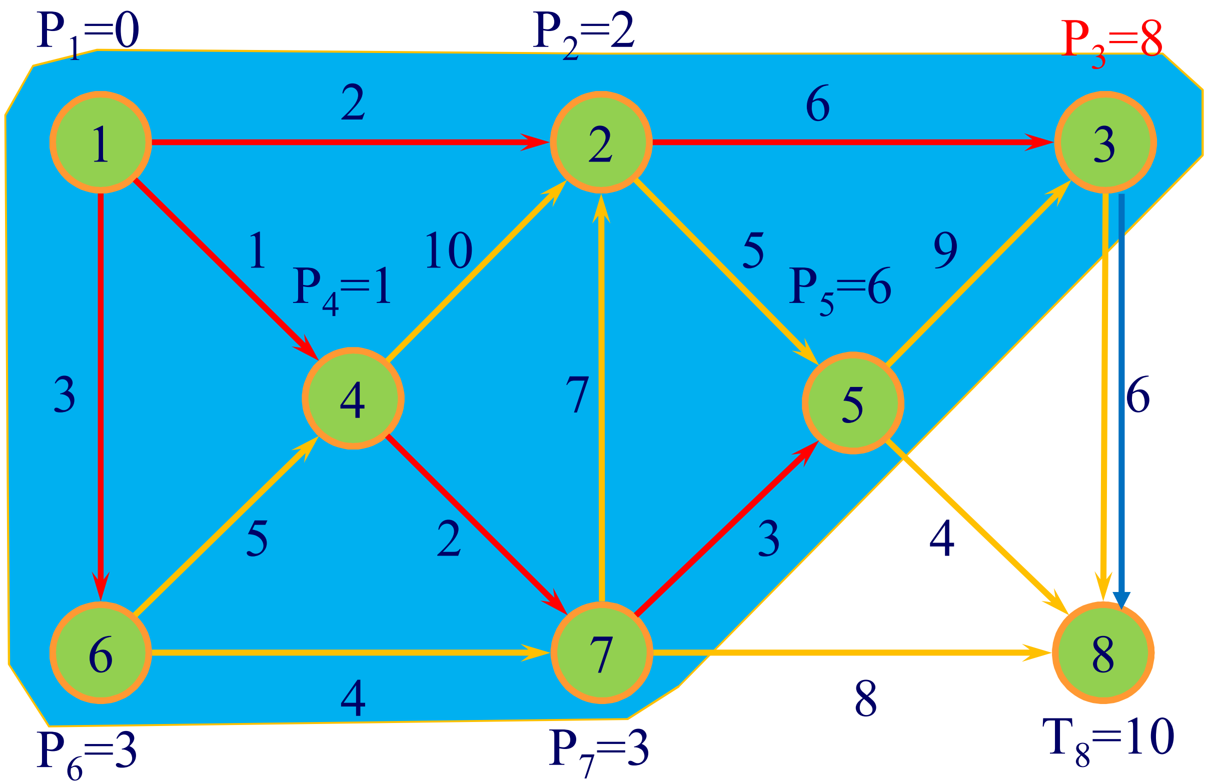
Step5：T值更新



T_3= min {P_5+l_{53}, T_5}=min {6+3, 8}=8

T_8= min {P_5+l_{53}, T_8}=min {6+4, 11}=10

Step5：P点搜索

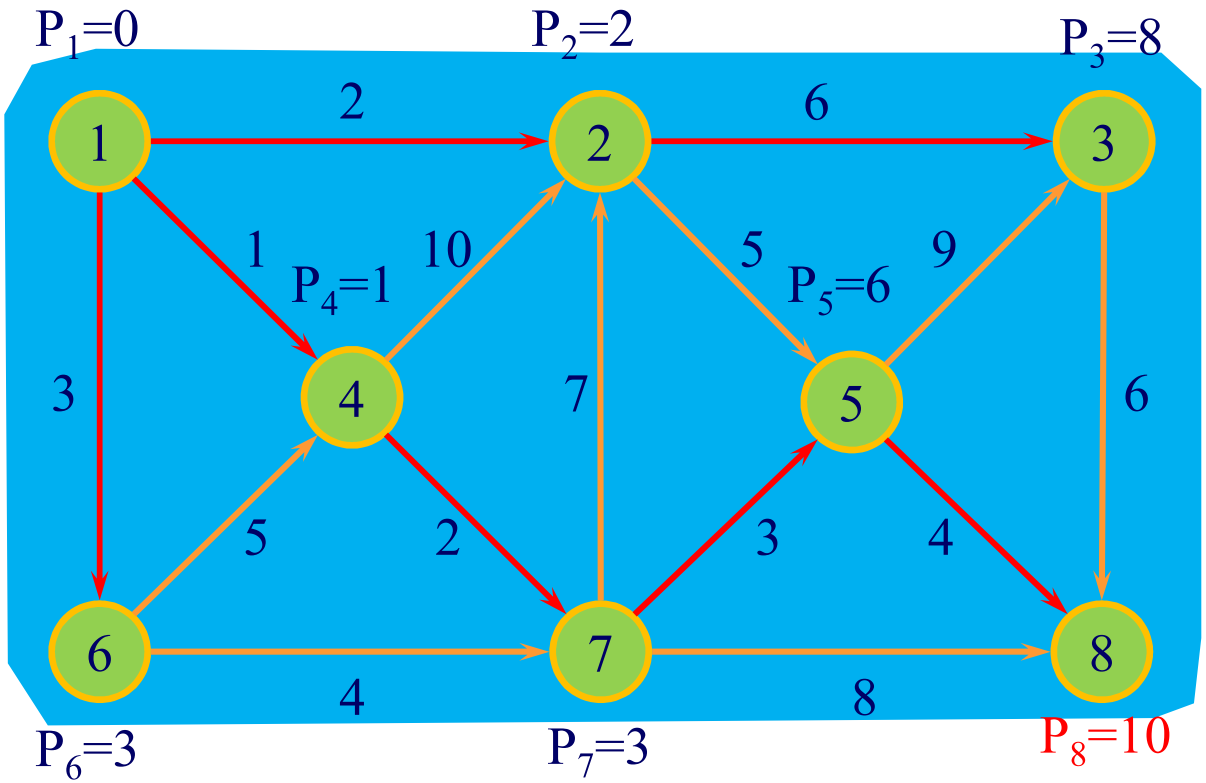


Y={3, 8}

min $\{T_i\}_{i\in Y}$ =min {8,10}=8

X={1,2,3,4,5,6,7}, $P_3=8$

Step6：T值更新+P点搜索



T_8= min $\{P_3+l_{38}, T_8\}$=min $\{8+6, 11\}$=10

Dijkstra算法

- $P(v_i)$： 到P点i的最短距离。
- $T^{(k)}(v_j)$： 搜索到第k步时，到T点j的最短距离。

1、设定初值

$T^{(0)}(v_1)=0$ $\quad$ $T^{(0)}(v_j)=\infty$ $\quad$ j=2,3,…,n

2、确定P点

$$P(v_i)=\min_j\{T^{(k)}(v_j)\}$$

3、更新T值

$$T^{(k+1)}(v_j)=\min_i\{T^{(k)}(v_j),P(v_i)+l_{ij}\}$$

实际考虑新的P点

$$l_{ij}=\begin{cases} d_{ij} & (i,j)\in E \\ +\infty & (i,j)\notin E \end{cases}$$

问题

1、Dijkstra算法最多需要迭代几步结束?

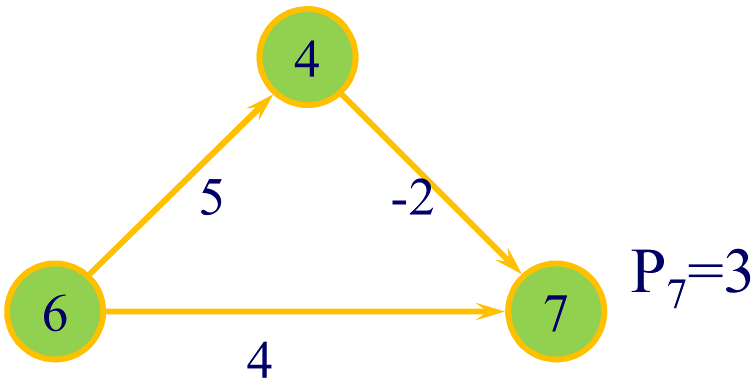
2、Dijkstra算法应用有何限制?

Dijkstra算法的限制条件

性质：

$$P(v_i)=\min_j\{T^{(k)}(v_j)\}$$

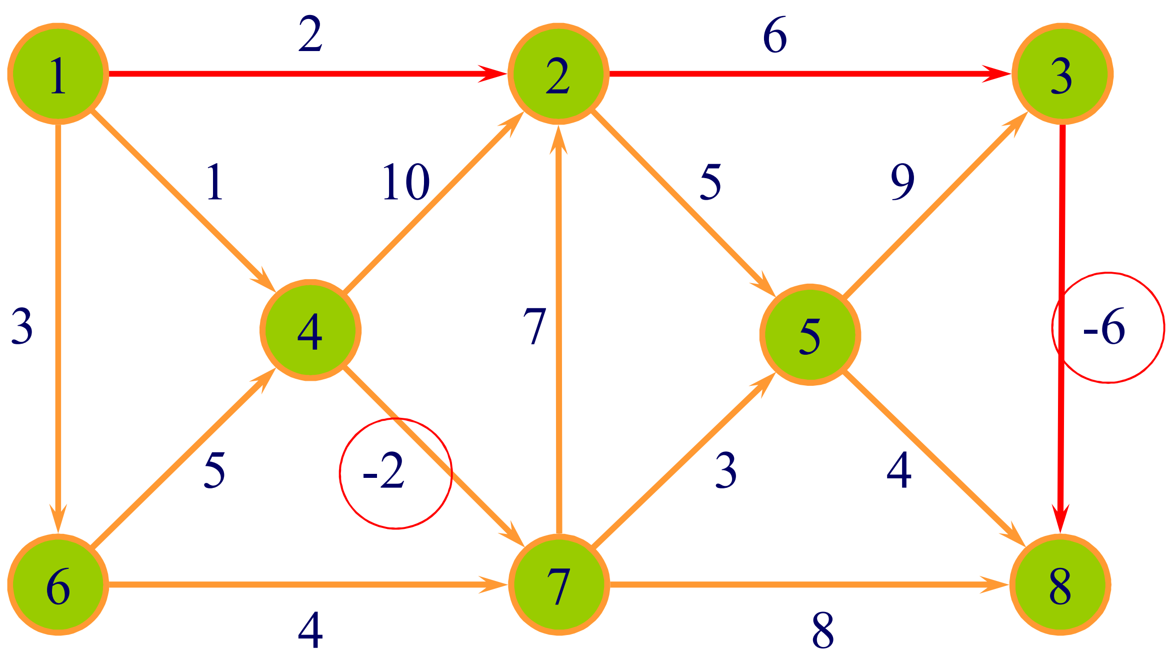
成立的条件是：所有边的权值非负



第八章 图与网络分析

- 图的基本概念
- 最短路径问题
  - **Dijkstra**算法
  - 逐次逼近法
  - **Floyd**算法
- 最大流问题
- 最小费用流问题

含负权值的最短路径



求从**1**到**8**的最短路径

迭代公式的搜索解释

$T^{(0)}(v_1)=0 \qquad T^{(0)}(v_j)=\infty \qquad j=2,3,\ldots,n$

$$T^{(k+1)}(v_j)=\min_i\{T^{(k)}(v_i)+l_{ij}\} \qquad l_{ij}=\begin{cases} d_{ij} & (i,j)\in E \\ +\infty & (i,j)\notin E \end{cases}$$

$T^{(1)}(v_j)$：起点直接到v_j的路径长度；

$T^{(1)}(v_i)+l_{ij}$：，沿当前最短路径经v_i中转到v_j的路径长度；

$T^{(2)}(v_j)$：最多经过1个中转点到v_j的最短路径长度；

…

$T^{(k)}(v_j)$：最多经过k-1个中间点到达v_j的最短路径长度。

问题：这样的搜索会找到最短路径吗？

逐次逼近法

1、设定初值

$$T^{(0)}(v_1)=0 \qquad T^{(0)}(v_j)=\infty \qquad j=2,3,\ldots,n$$

2、更新**T**值

$$T^{(k+1)}(v_j)=\min_i\{T^{(k)}(v_i)+l_{ij}\} \qquad l_{ij}=\begin{cases} d_{ij} & (i,j)\in E \\ +\infty & (i,j)\notin E \end{cases}$$

3、收敛条件

$$T^{(k+1)}(v_j)=T^{(k)}(v_j) \qquad j=1,2,\ldots,n$$

空格为∞

$$T^{(k+1)}(v_j)=\min_i\{T^{(k)}(v_i)+l_{ij}\}$$

节点	l_{ij}								$T^{(0)}_j$	$T^{(1)}_j$	$T^{(2)}_j$	$T^{(3)}_j$	$T^{(4)}_j$
	1	2	3	4	5	6	7	8					
1	0	2		1		3			0	0	0	0	0
2		0	6		5					2	2	2	2
3			0					-6			8	8	8
4		10		0			-2			1	1	1	1
5			9		0			4			7	2	2
6				5		0	4			3	3	3	3
7		7			3		0	8			-1	-1	-1
8								0				2	2

$8\leftarrow3\leftarrow2\leftarrow1$

问题

1、逐次逼近法是否一定收敛？

2、逐次逼近法最多需迭代几次？

3、逐次逼近法是否是动态规划算法？

$$T^{(k+1)}(v_j) = \min_i \{T^{(k)}(v_i) + l_{ij}\}$$

第八章 图与网络分析

- 图的基本概念
- 最短路径问题
  - **Dijkstra**算法
  - 逐次逼近法
  - **Floyd**算法
- 最大流问题
- 最小费用流问题

任意两点最短距离问题

■目标：求取网络中所有的两点间最短距离

- 初始权矩阵：$\mathbf{D}^{(0)}= (d_{ij}^{(0)})_{n\times n} =(l_{ij})_{n\times n}$ $\quad l_{ij} = \begin{cases} d_{ij} & (i,j)\in E \\ +\infty & (i,j)\notin E \end{cases}$

- 迭代权矩阵：$\mathbf{D}^{(k)}= (d_{ij}^{(k)})_{n\times n}$

➢ 思路1：两点间用逐次逼近法求解

$$d_{ij}^{(k)} = \min_{l}\{d_{il}^{(k-1)} + l_{lj}\} \qquad l=1,2,\ldots,n \qquad 1\le k \le n$$

➢ 思路2：l_{ij}用上一步迭代结果替代

$$d_{ij}^{(k)} = \min_{l}\{d_{il}^{(k-1)} + d_{lj}^{(k-1)}\} \qquad l=1,2,\ldots,n \qquad 1\le k \le n$$

Floyd算法

权矩阵：$D^{(k)}=(d_{ij}^{(k)})_{n\times n}$

初值：$D^{(0)}=(d_{ij}^{(0)})_{n\times n}=(l_{ij})_{n\times n}$

$$l_{ij}=\begin{cases} d_{ij} & (i,j)\in E \\ +\infty & (i,j)\notin E \end{cases}$$

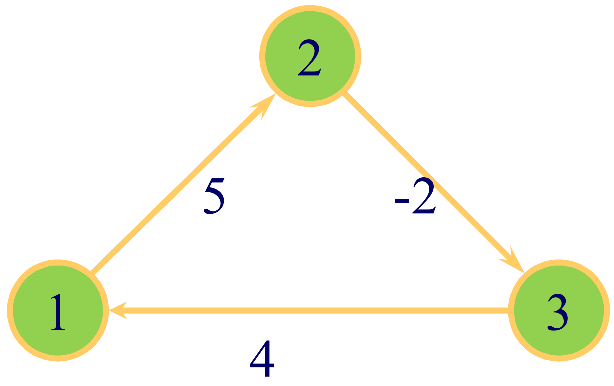
$$d_{ij}^{(k)}=\min\{d_{ij}^{(k-1)},d_{ik}^{(k-1)}+d_{kj}^{(k-1)}\}$$

$k=1,2,\ldots,n$

最多经过前k个节点的最短路径

问题：Floyd算法需几步收敛？

$$d_{ij}^{(k)} = \min\{d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)}\}$$



$$\begin{array}{cc} & \begin{array}{ccc} d_{i1}^{(0)} & & \end{array} \\ \begin{array}{c} d_{1j}^{(0)} \\ \\ \end{array} & \begin{bmatrix} 0 & 5 & \infty \\ \infty & 0 & -2 \\ 4 & \infty & 0 \end{bmatrix} \end{array} \Rightarrow \begin{array}{cc} & \begin{array}{ccc} & d_{i2}^{(1)} & \end{array} \\ \begin{array}{c} \\ d_{2j}^{(1)} \\ \end{array} & \begin{bmatrix} 0 & 5_{12} & \infty \\ \infty & 0 & -2_{23} \\ 4_{31} & 9_{312} & 0 \end{bmatrix} \end{array} \Rightarrow$$

$$\begin{array}{cc} & \begin{array}{ccc} & & d_{i3}^{(2)} \end{array} \\ \begin{array}{c} \\ \\ d_{3j}^{(2)} \end{array} & \begin{bmatrix} 0 & 5_{12} & 3_{123} \\ \infty & 0 & -2_{23} \\ 4_{31} & 9_{312} & 0 \end{bmatrix} \end{array} \Rightarrow \begin{bmatrix} 0 & 5_{12} & 3_{123} \\ 2_{231} & 0 & -2_{23} \\ 4_{31} & 9_{312} & 0 \end{bmatrix}$$

最短路问题的应用

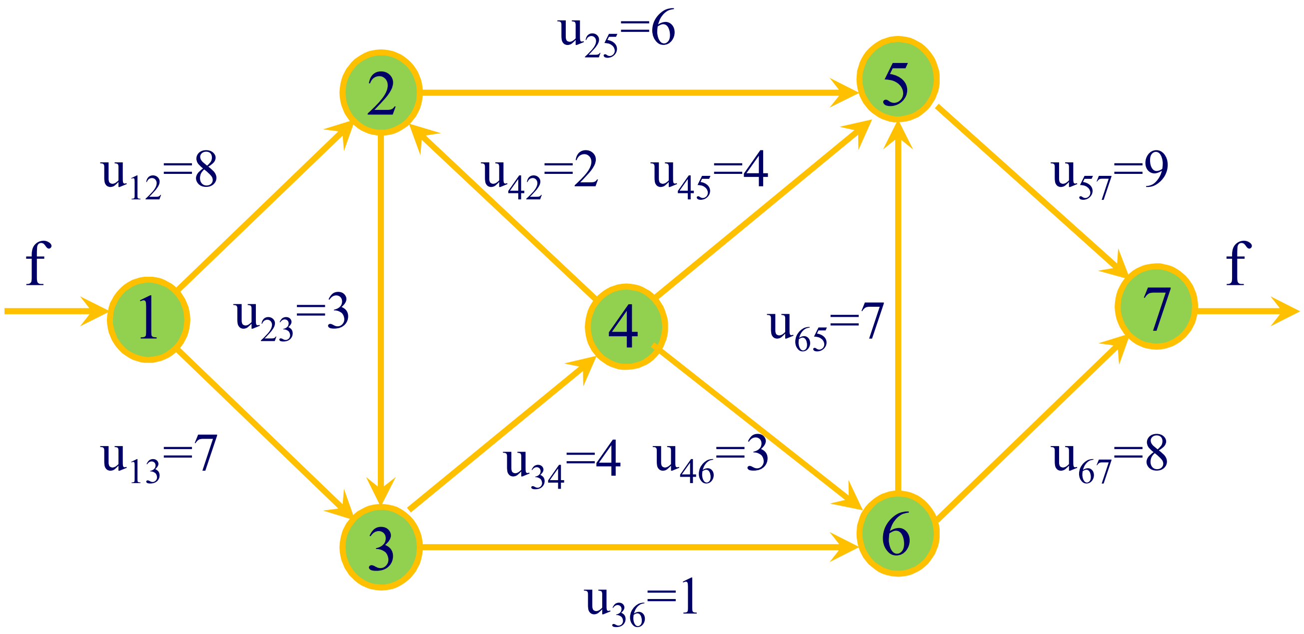
选址问题

设备更新问题：抽象路径长度，如成本

第八章 图与网络分析

- 图的基本概念
- 最短路径问题
- 最大流问题
  - **Ford-Fulkerson算法**
  - 最大流-最小割定理
- 最小费用流问题

最大流问题



u25=6
u12=8
u42=2
u45=4
u57=9
f
u23=3
u65=7
f
u13=7
u34=4
u46=3
u67=8
u36=1
1
2
3
4
5
6
7

最大流问题模型

■ 最大流问题：求网络的最大可行流

$$\max f = \sum_k x_{kt} = \sum_j x_{sj}$$

s起点/源source

t终点/汇sink

s.t. $$\sum_k x_{ki} = \sum_j x_{ij} \quad i \neq s,\ i \neq t \quad i = 1, 2, \ldots, n$$

流量平衡条件

$$0 \leq x_{ij} \leq u_{ij} \quad i, j = 1, 2, \ldots, n$$

容量限制条件

u_{ij}为对应边的容量上限

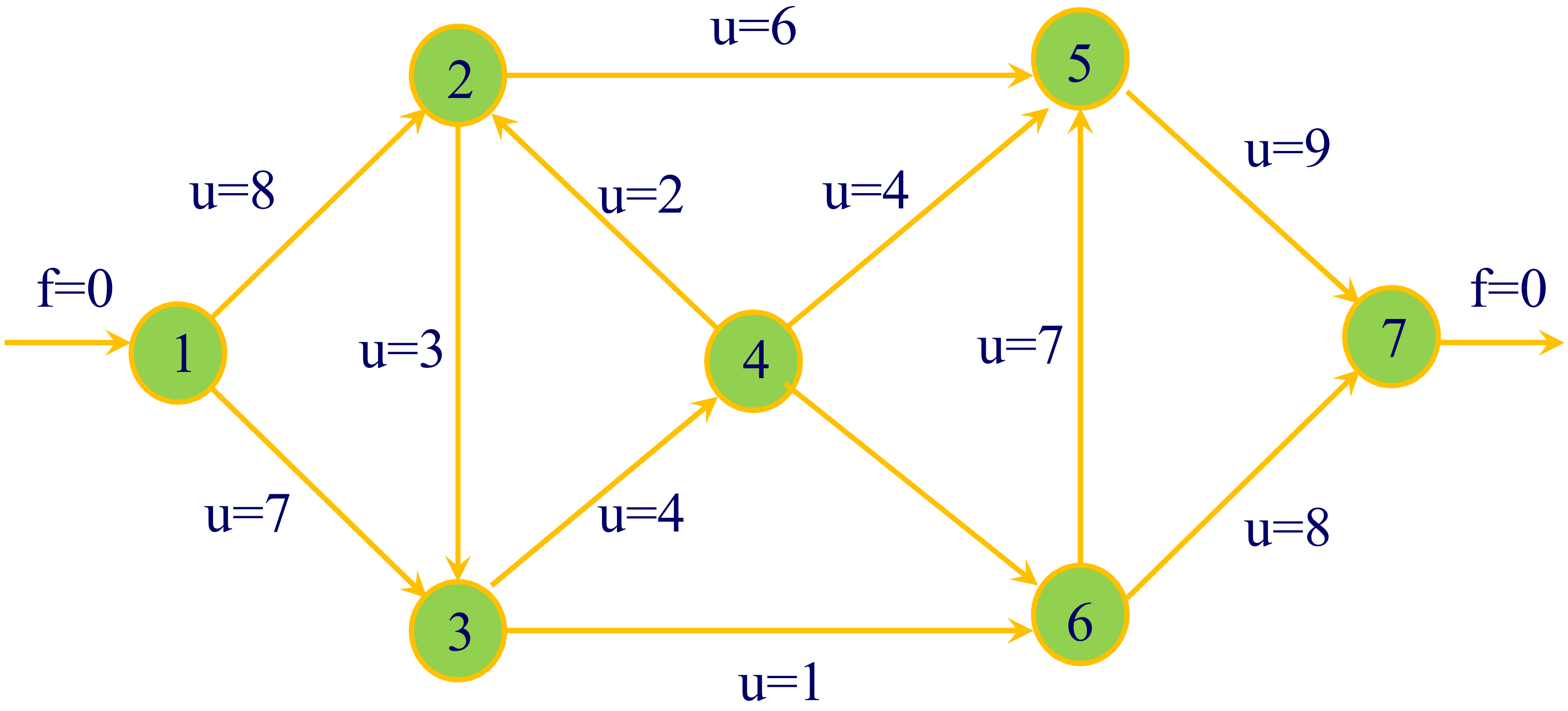
Ford-Fulkerson算法

- 思想：不断的增加流量，直至网络饱和
- 算法：从当前可行流开始，沿可增加流量的路径增大流量。

问题1：如何增加网络流量？

问题2：流量不可增大时是否是最优解？

初始可行流



给出一个初始的可行流x_{ij}=0

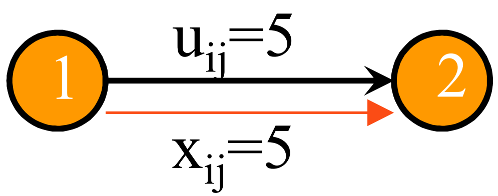
不饱和边与可增广链

- 不饱和边：流量没有达到容量限制的边
- 可增广链：从起点到终点方向一致的不饱和边构成的链。

正向饱和/不饱和边

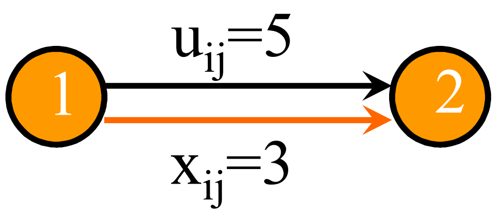
- 正向边：与参考链（可增广链）方向一致的边。

1、如果$x_{ij}=u_{ij}$，边从i到j的方向是饱和的；



（1，2）是饱和的

2、如果$x_{ij}<u_{ij}$，边从i到j的方向是不饱和的；

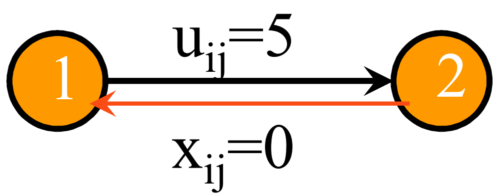


（1，2）是不饱和的
间隙为$\Delta_{12}=u_{12}-x_{12}=5-3=2$

反向饱和/不饱和边

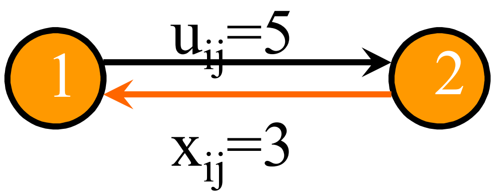
- 反向边：与参考链的方向相反的边。

3、如果x_{ij}=0，边从j到i的方向是饱和的；



（2，1）是饱和的

4、如果x_{ij}>0，边从j到i的方向是不饱和的；

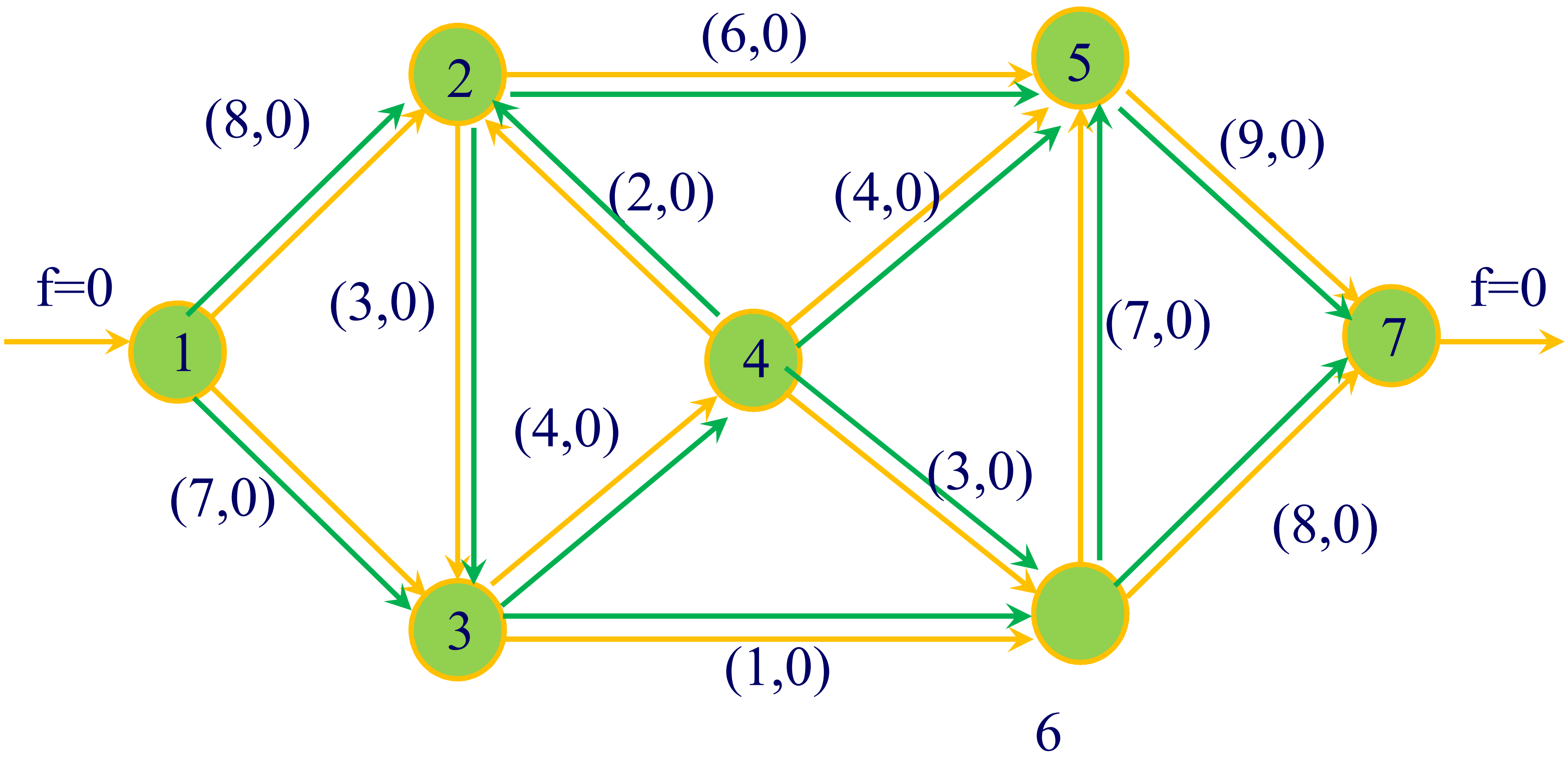


（2，1）是不饱和的
间隙为$\Delta_{12}=x_{12}=3$

最大流性质

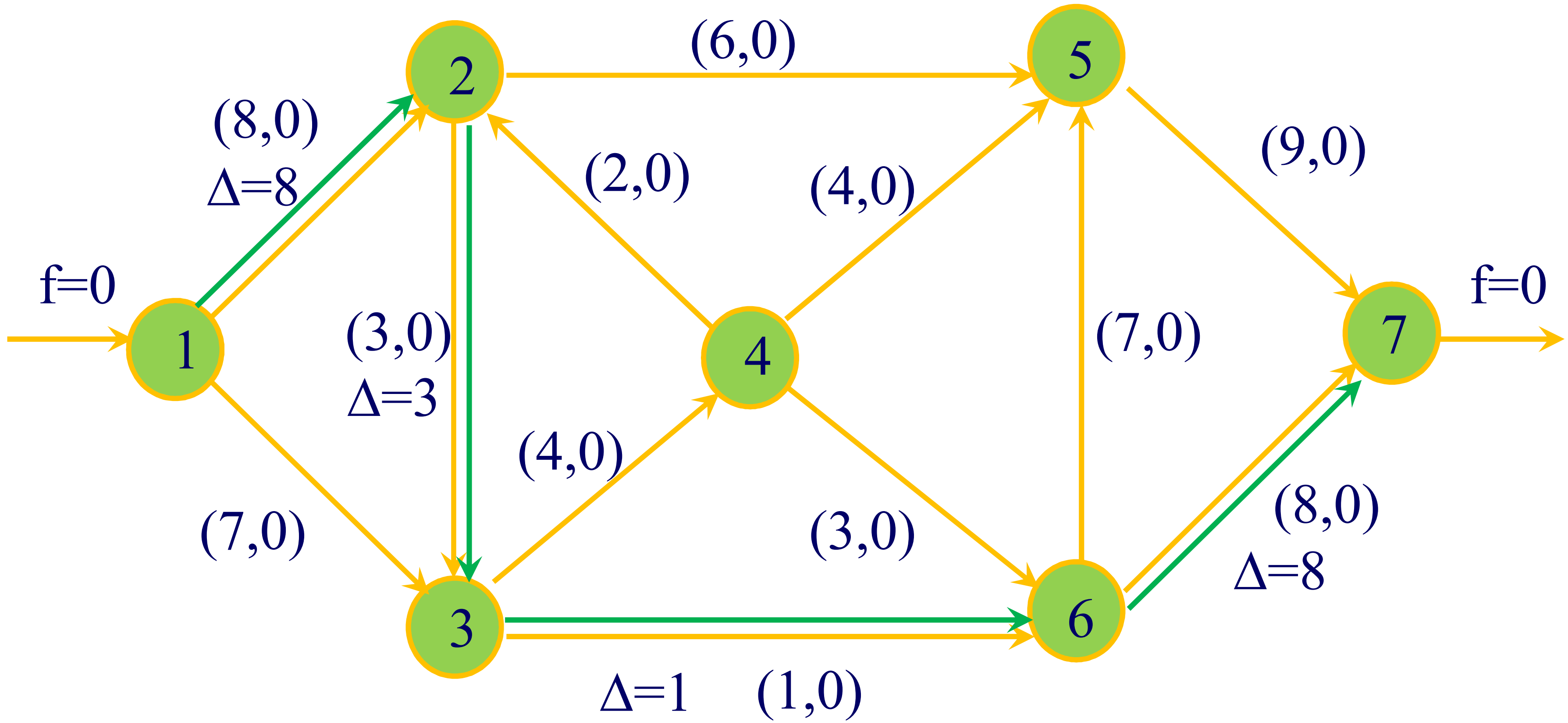
- 现象：若存在连接起点和终点的链的每一条边都是不饱和边，即存在可增广链，则可使总流量增加。

- 性质：可行流为最大流的充要条件是：
  网络不存在可增广链



找到所有的不饱和边，以及各边可以调整流量的方向

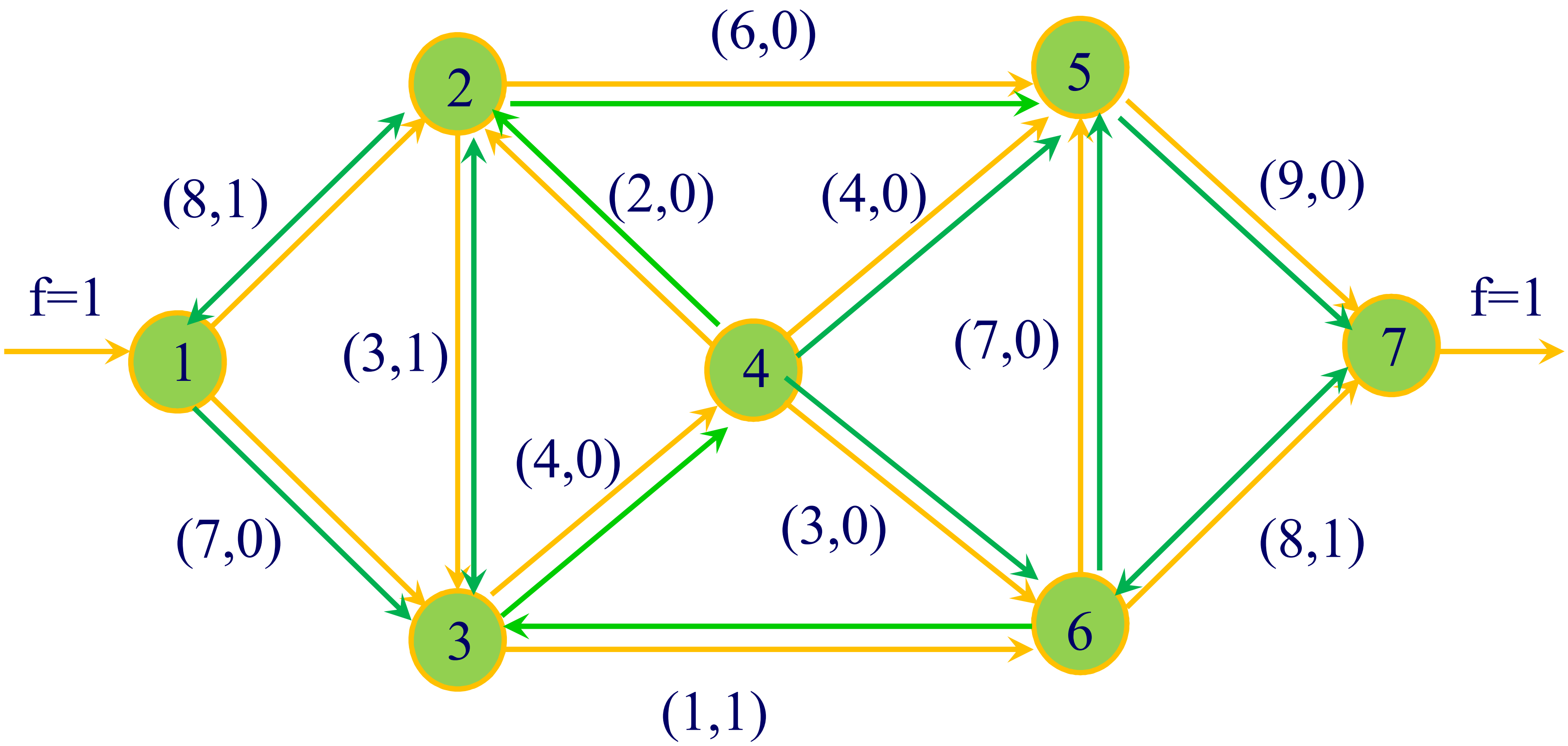
找到一条从1到7的可增广链



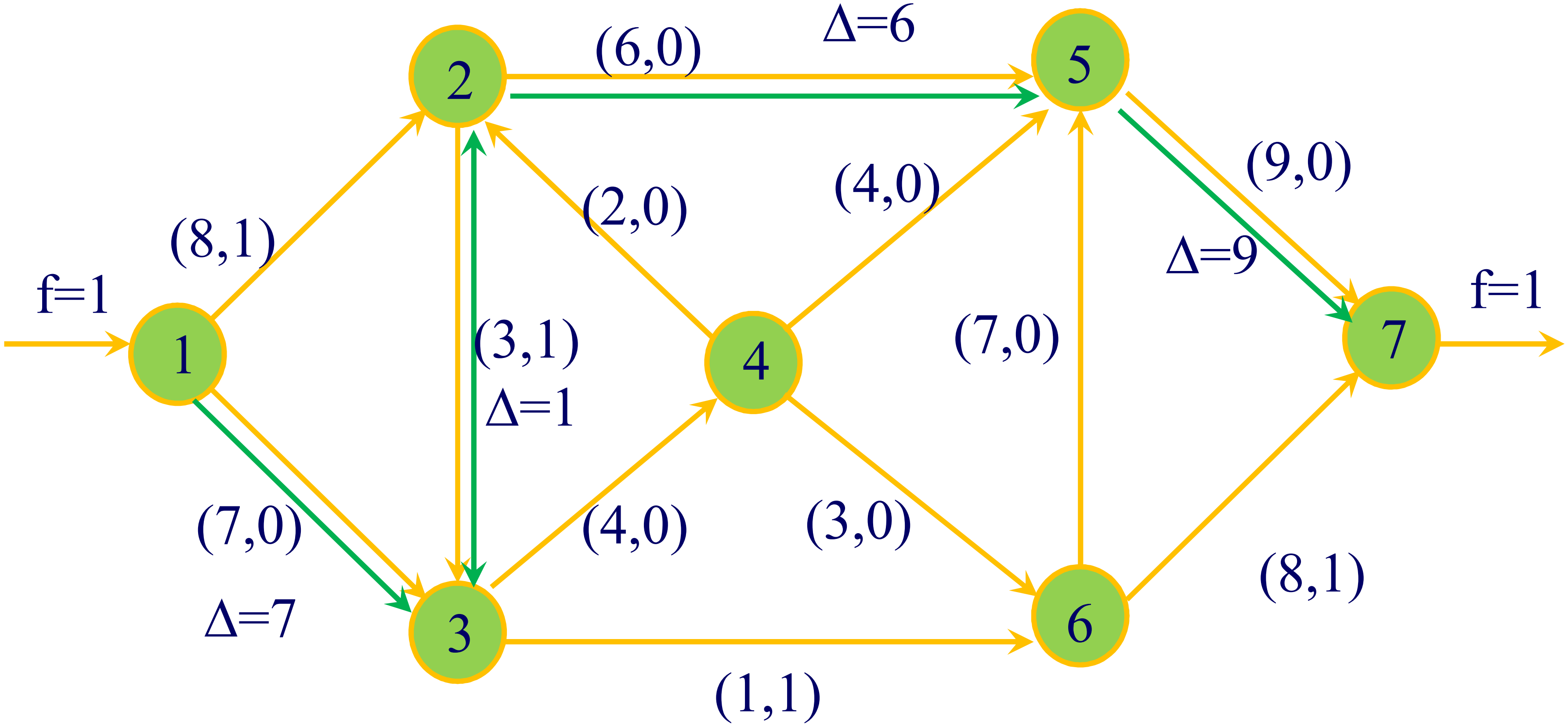
链的间隙为：$\Delta = \min\{8,3,1,8\}=1$

调整链的流量：$x_{ij}'=x_{ij}+\Delta$

调整流量，**f=1**。继续求出网络的不饱和边



求出一条从1到7的可增广链

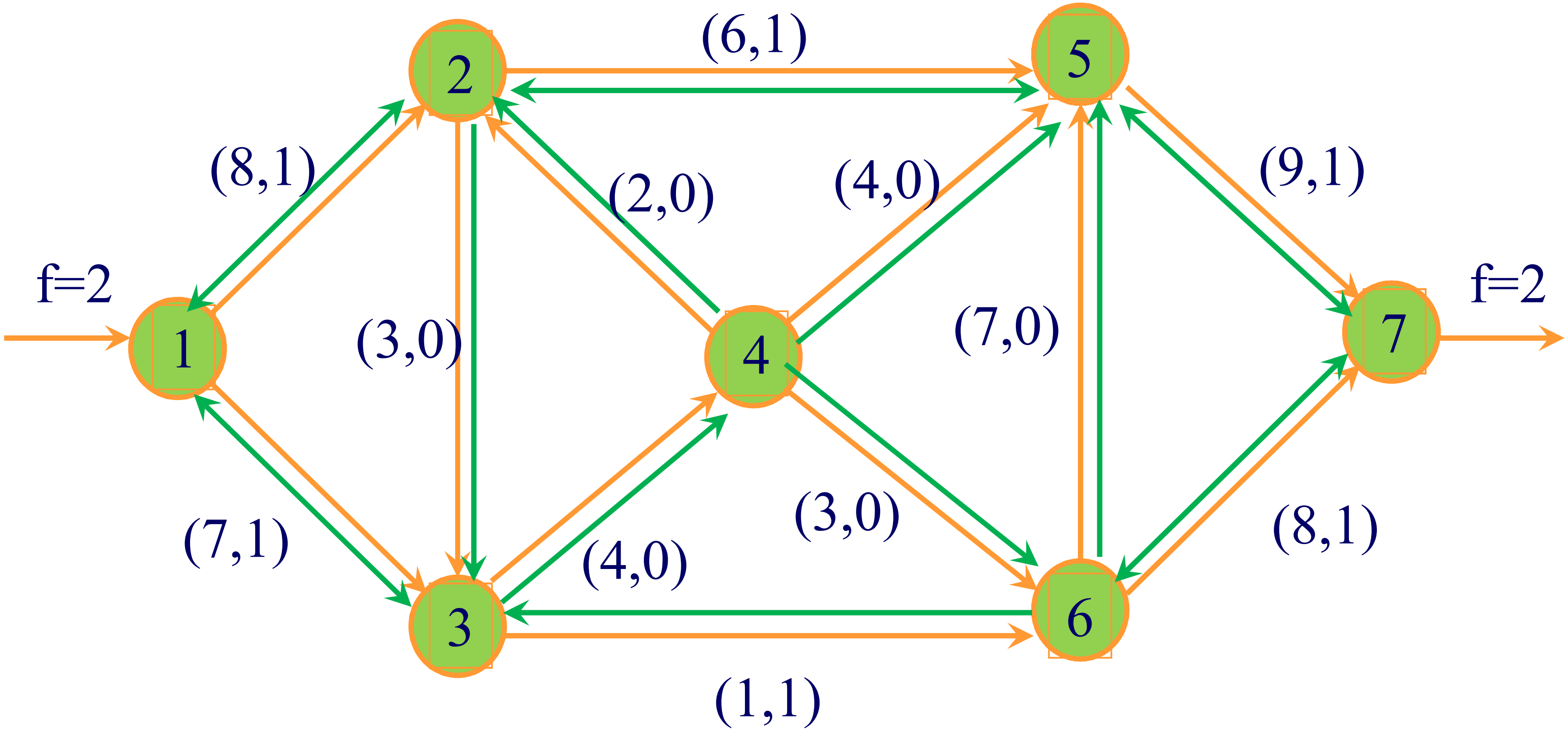


Δ=min {7,1,6,9}=1,

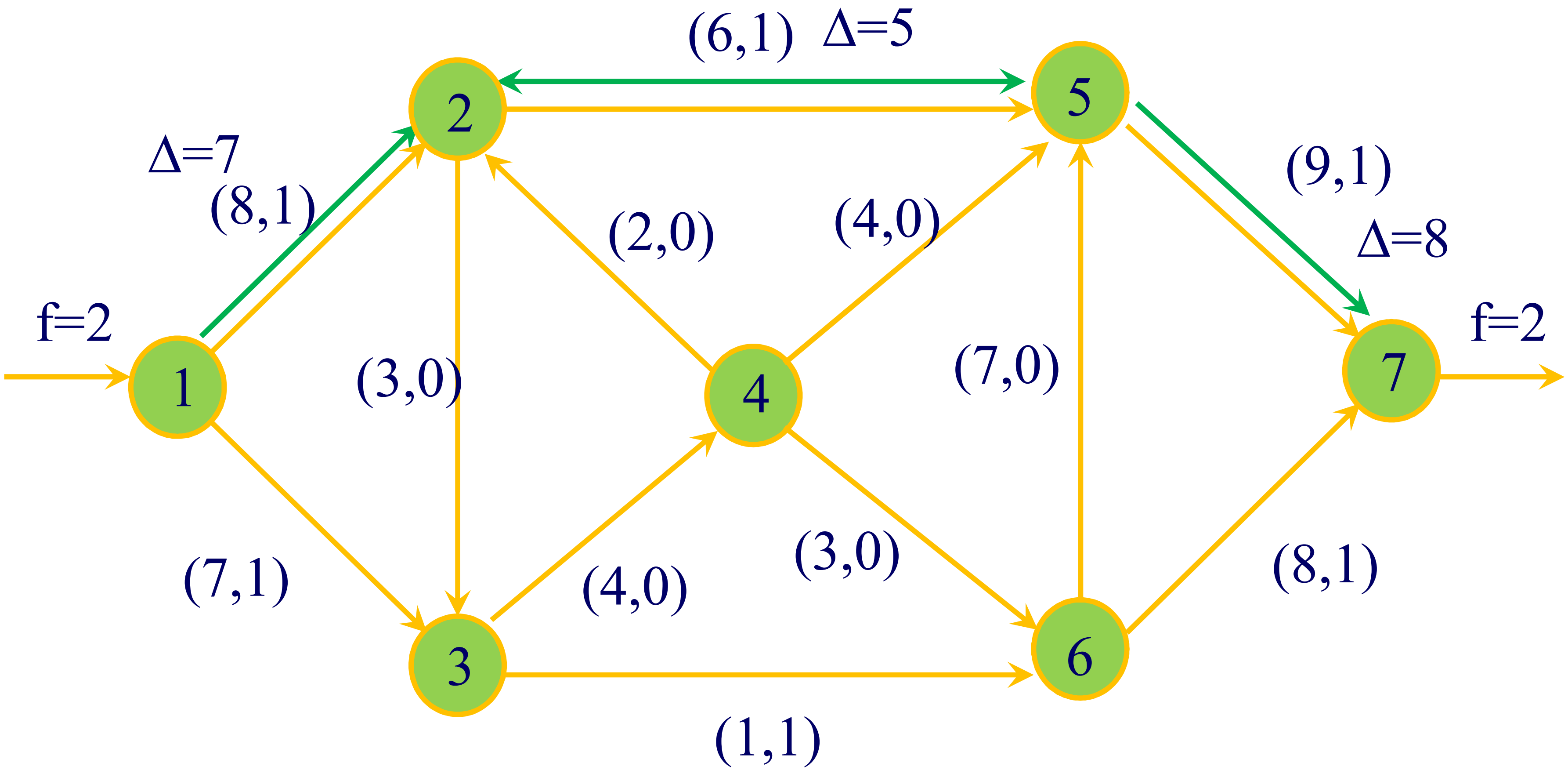
调整流量： 正向边：$x_{ij}'=x_{ij}+\Delta$, 反向边： $x_{ij}'=x_{ij}-\Delta$

$f'=f+\Delta=2$

调整流量，继续求出网络的不饱和边

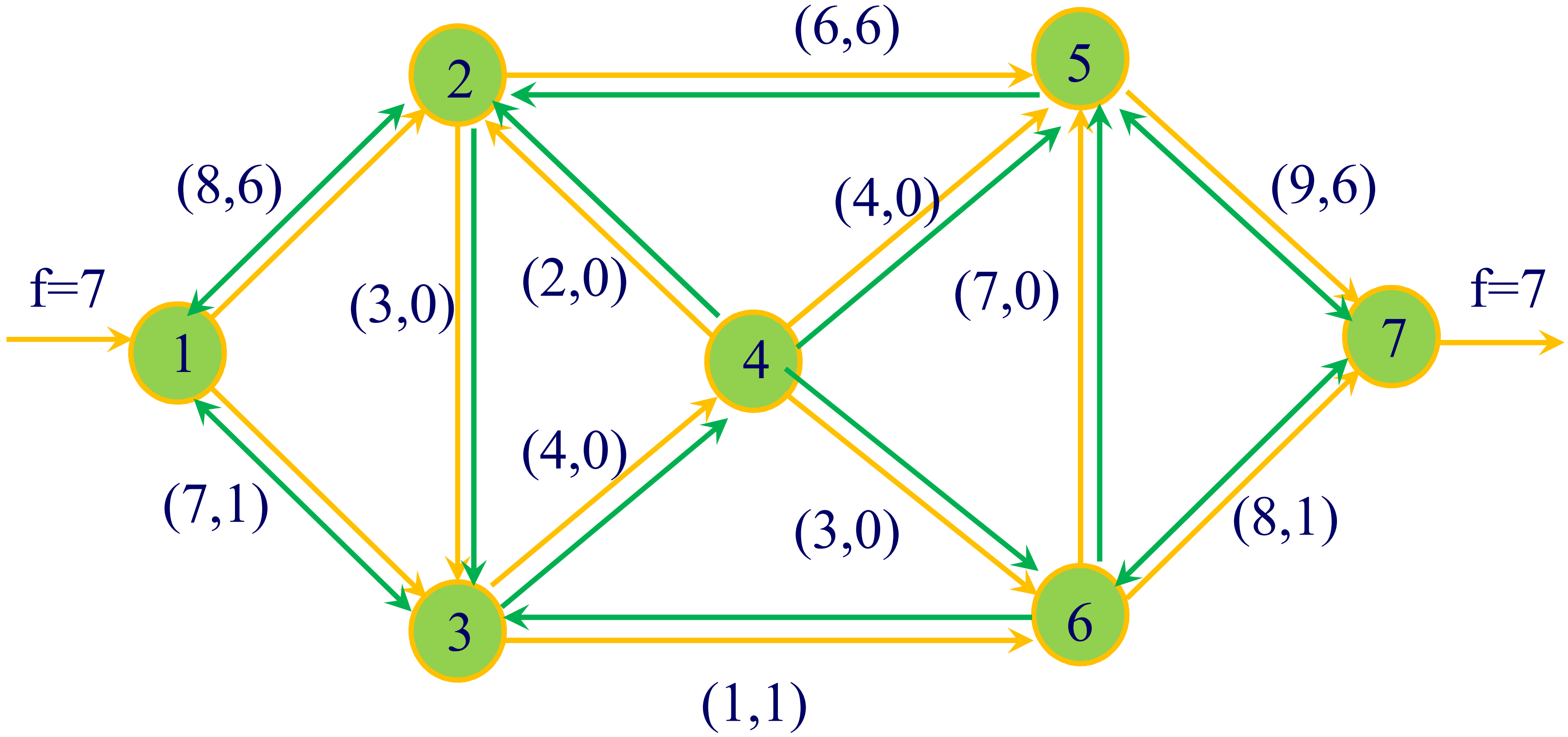


求出一条从1到7的可增广链

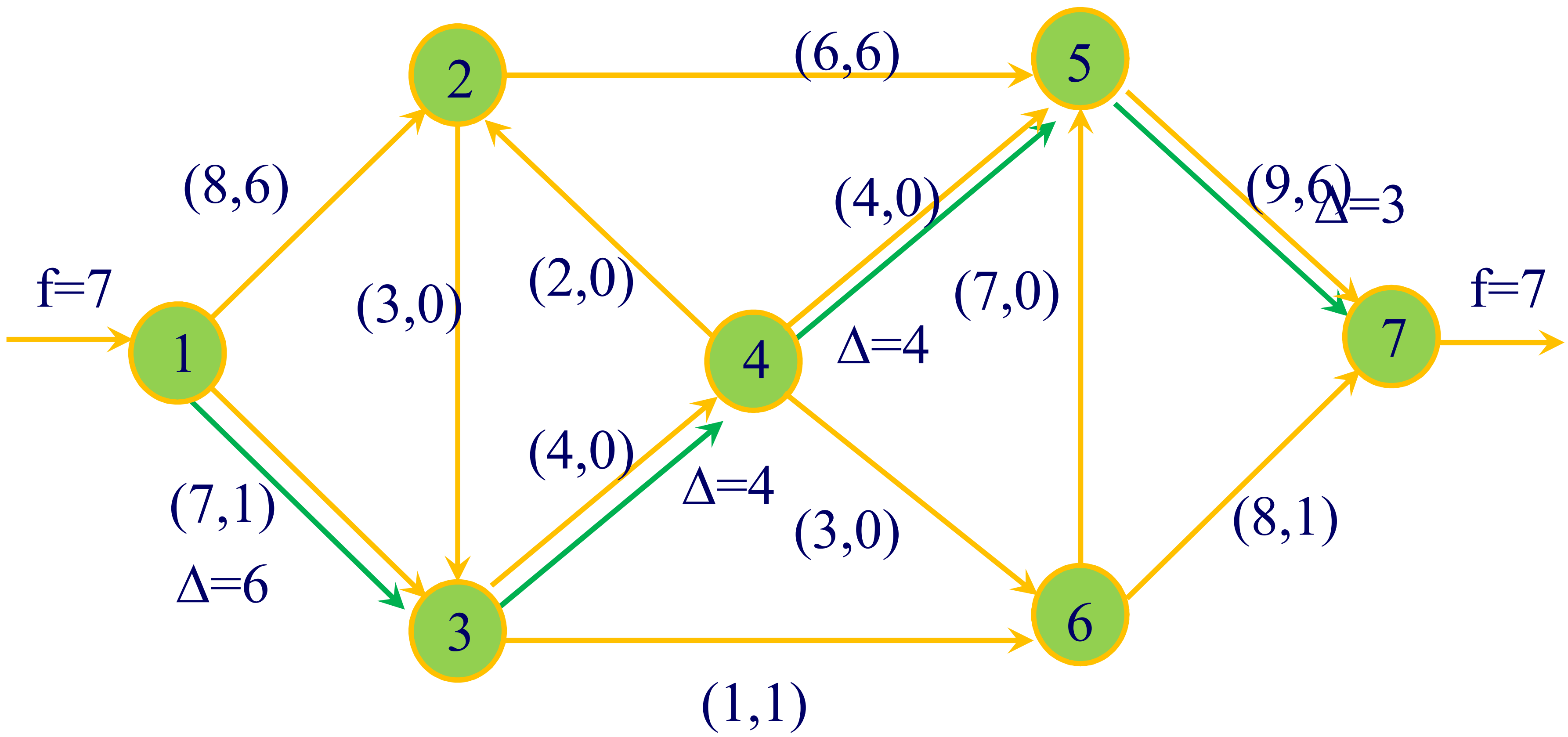


$\Delta=\min\{7,5,8\}=5$, 调整流量 $x_{ij}'=x_{ij}+5$, $f'=f+5=2+5=7$

调整流量，继续求出网络的不饱和边

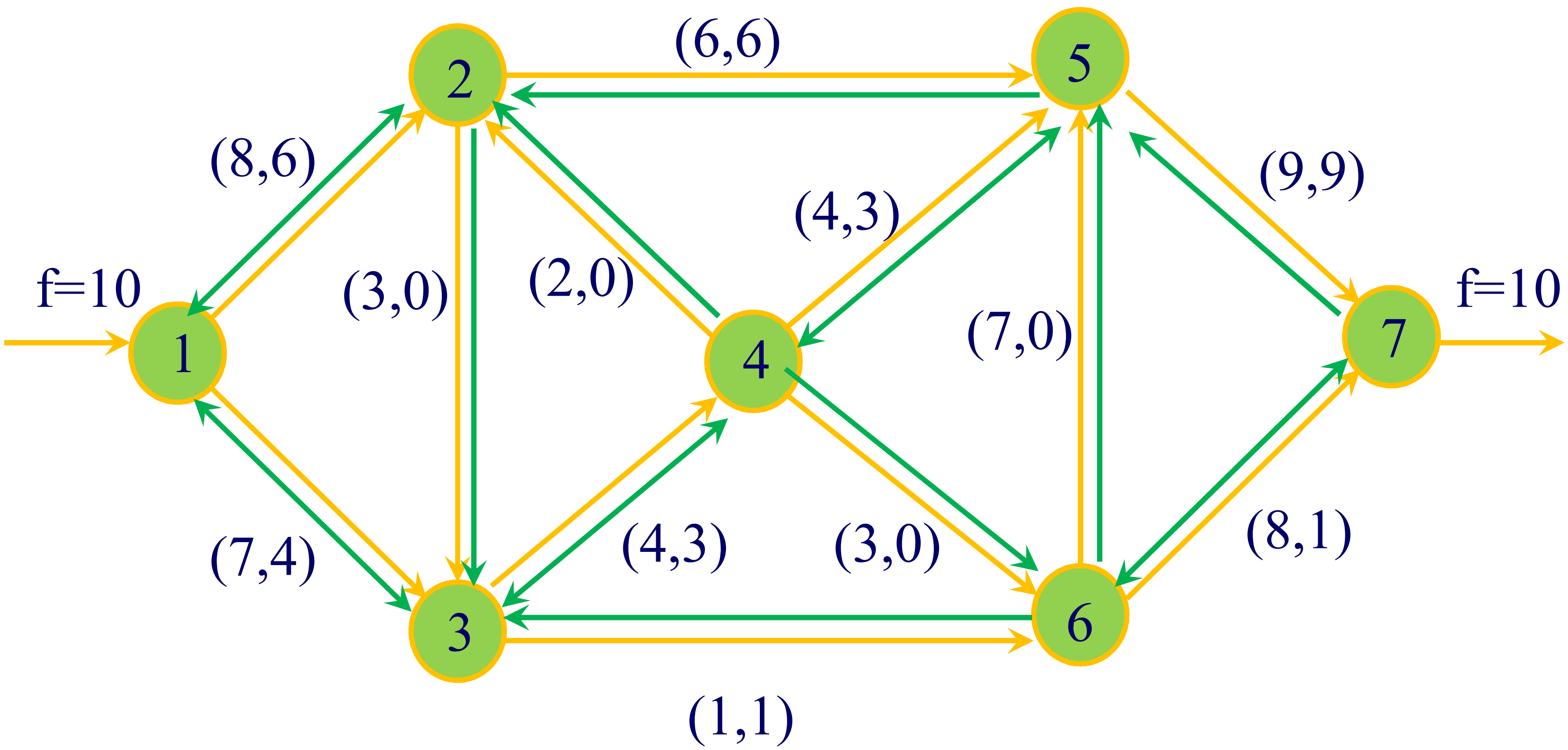


求出一条从1到7的可增广链

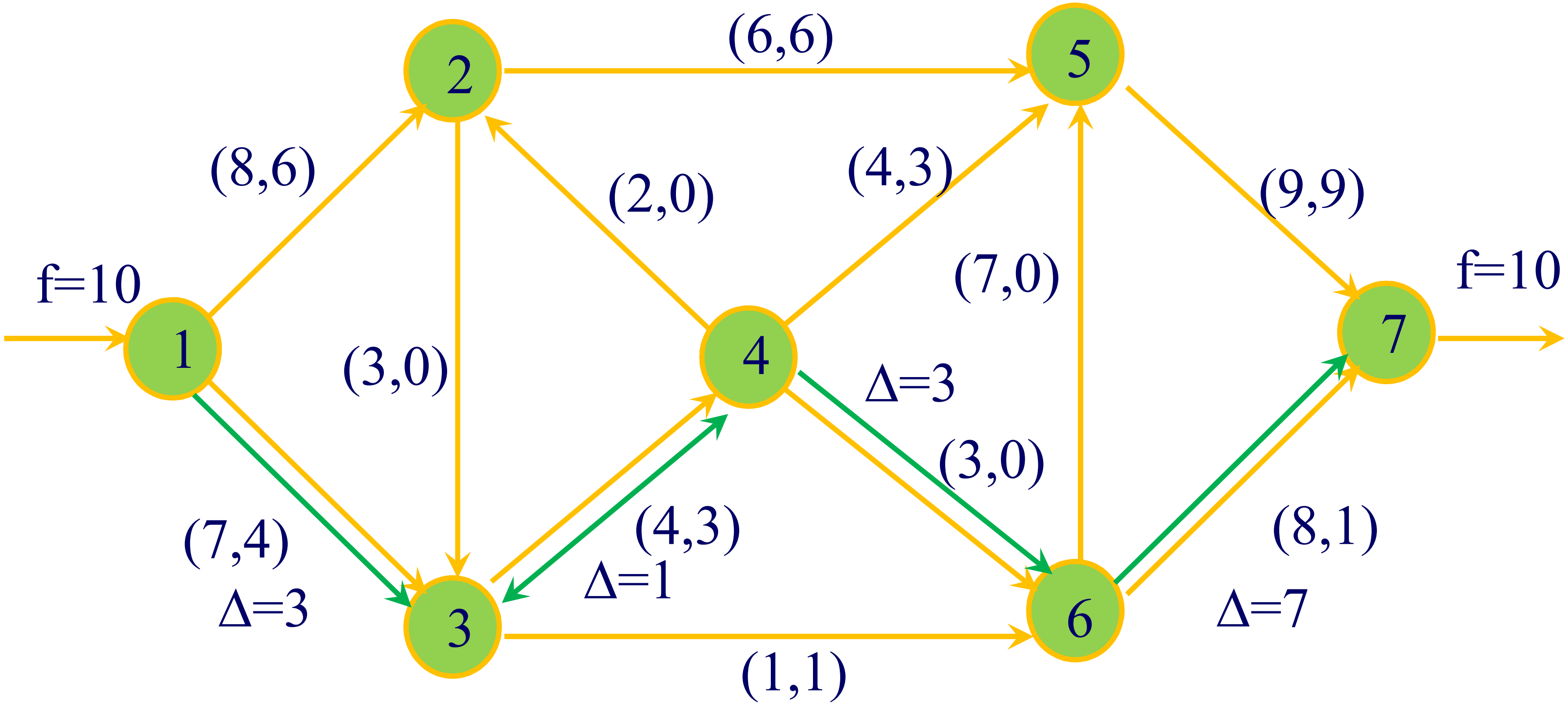


$\Delta=\min\{6,7,4,3\}=3$, 调整流量 $x_{ij}'=x_{ij}+3$, $f'=f+3=7+3=10$

调整流量，继续求出网络的不饱和边

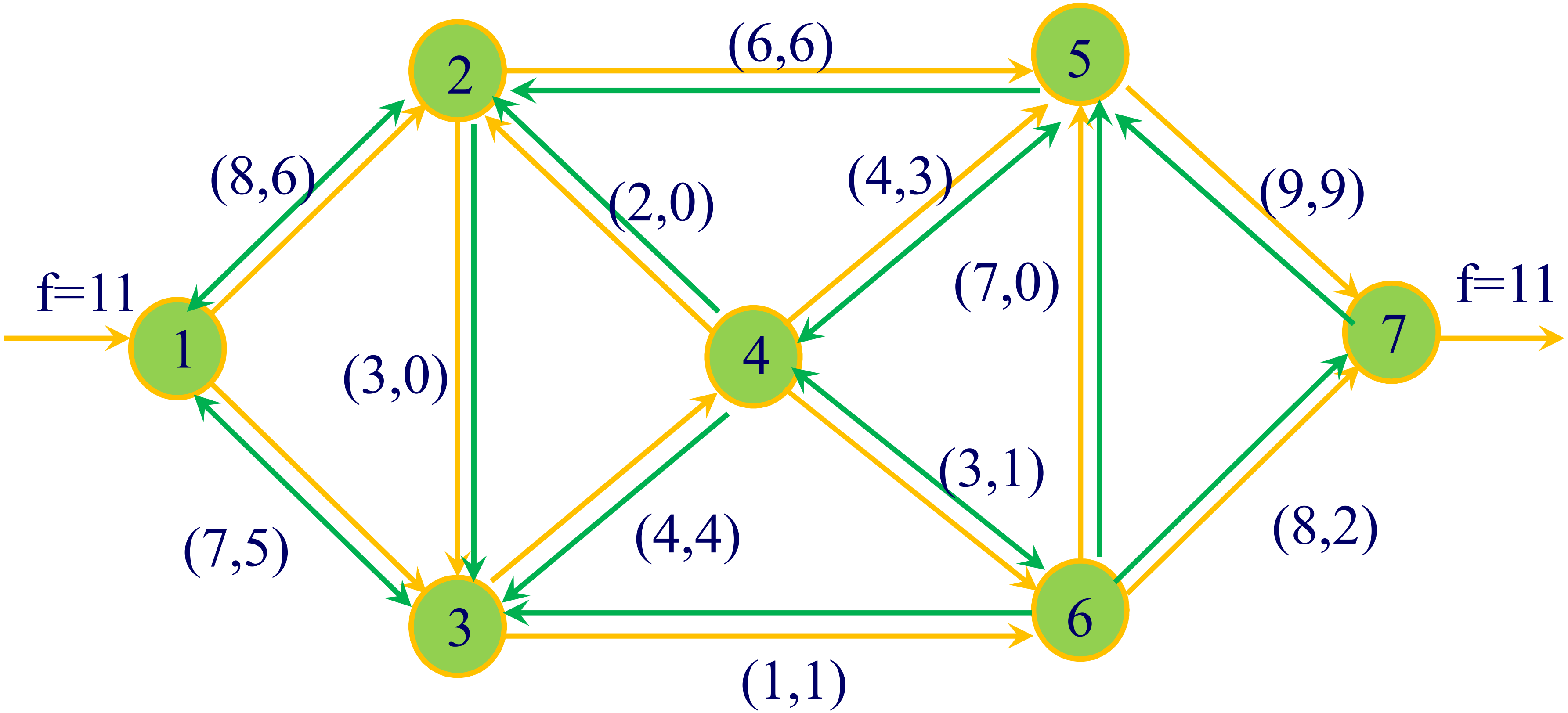


求出一条从1到7的可增广链



$\Delta=\min\{3,1,3,7\}=1$, 调整流量 $x_{ij}'=x_{ij}+1$, $f'=f+1=10+1=11$

调整流量，继续求出网络的不饱和边

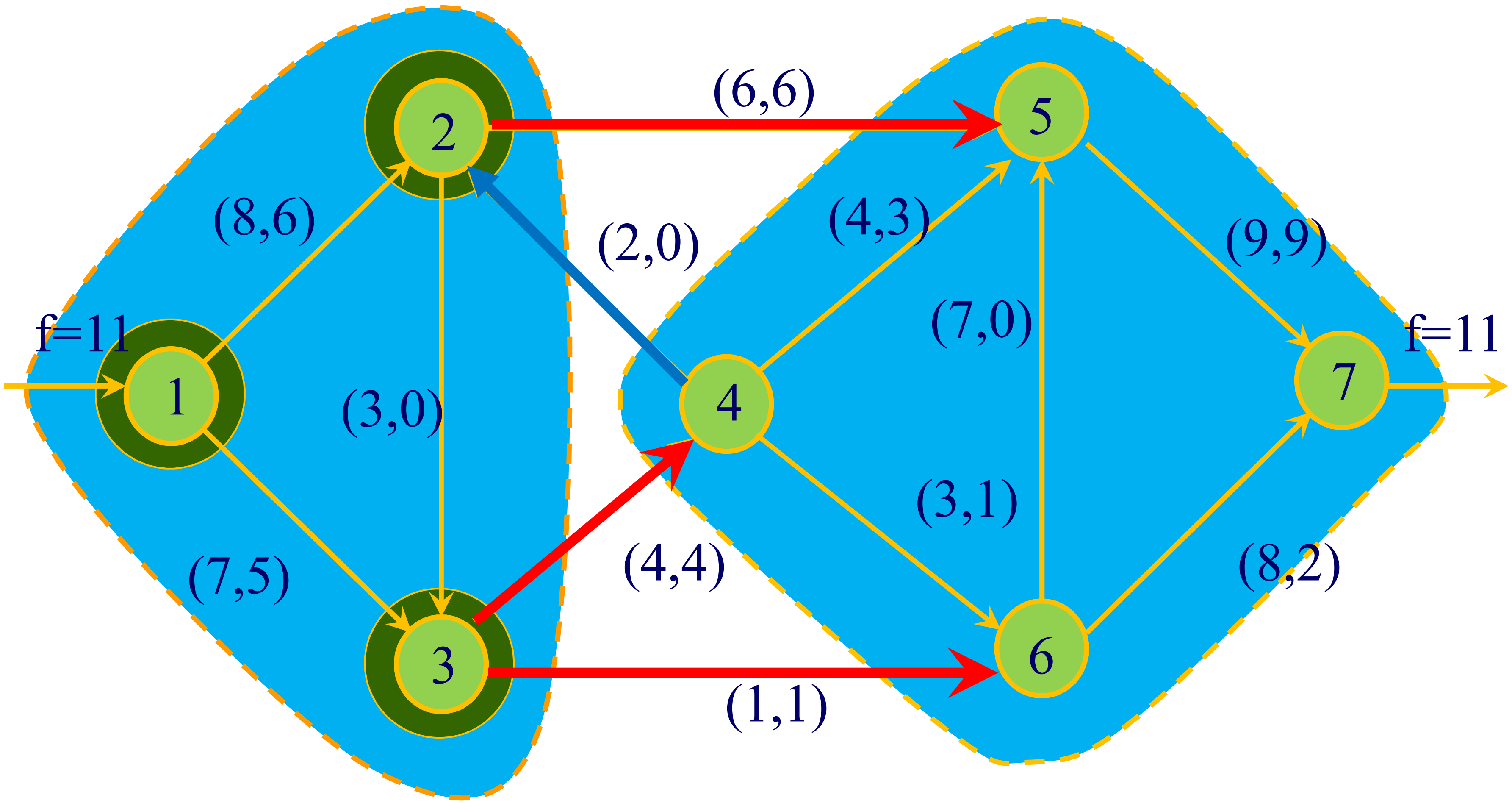


已找不到一条从**1**到**7**的可增广链，从**1**开始可以到达的节点为**1**，**2**，**3**

第八章 图与网络分析

- 图的基本概念
- 最短路径问题
- 最大流问题
  - **Ford-Fulkerson算法**
  - 最大流-最小割定理
- 最小费用流问题

最大流分析



割集/截集（**cut set)**

割集容量

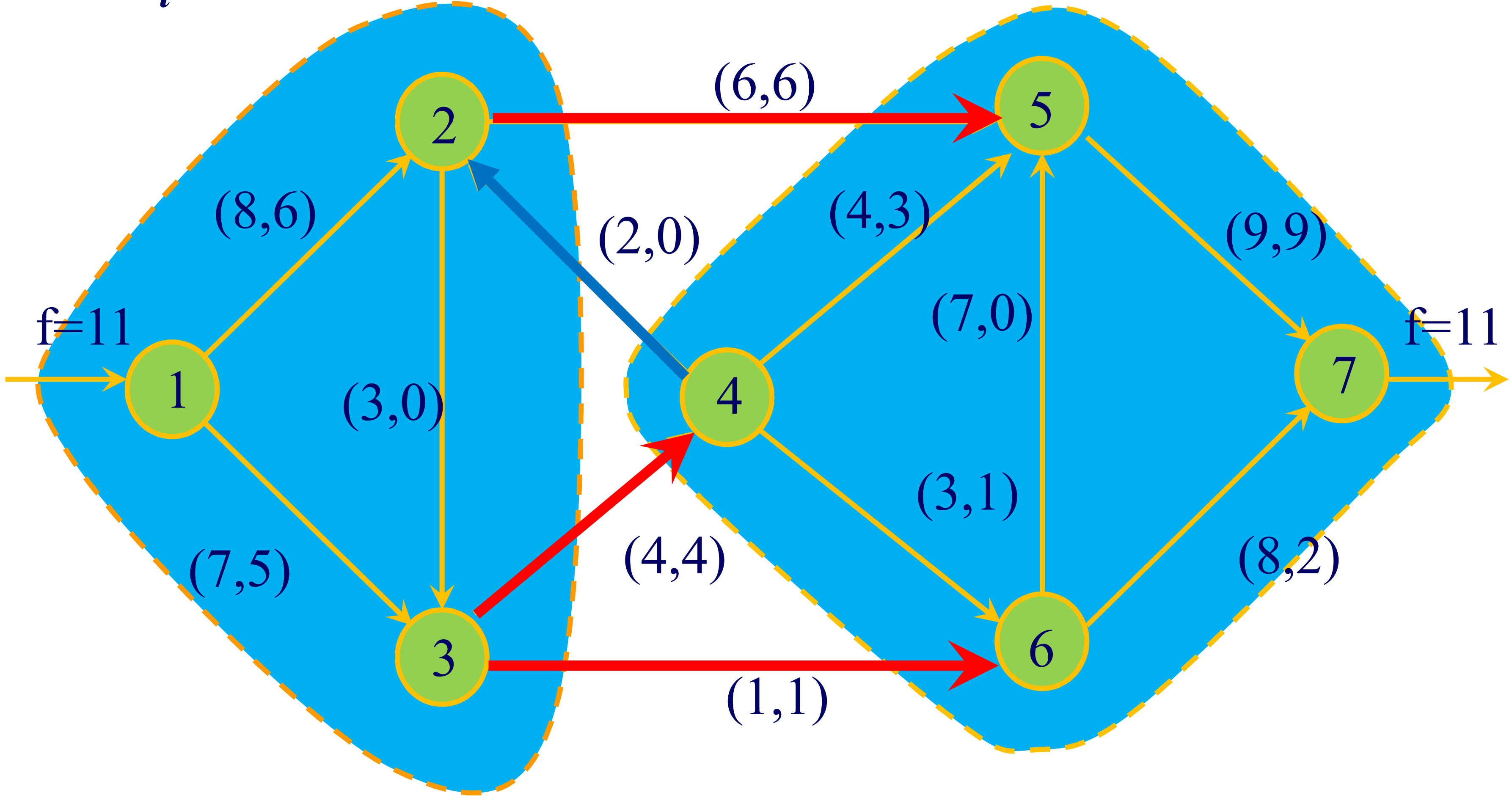
割集容量：对网络G=(V,E,U)，E'为将起点v_s和终点v_t分离在S、T两部分的割集，则割集容量 $U(E')$为所有起点在S、终点在T的边的容量之和。

- 定理：设f为网络G=(V,E,U)的任一可行流，E'为将起点v_s和终点v_t分离的割集，则有$f \le U(E')$

证明：略。

最大流-最小割定理

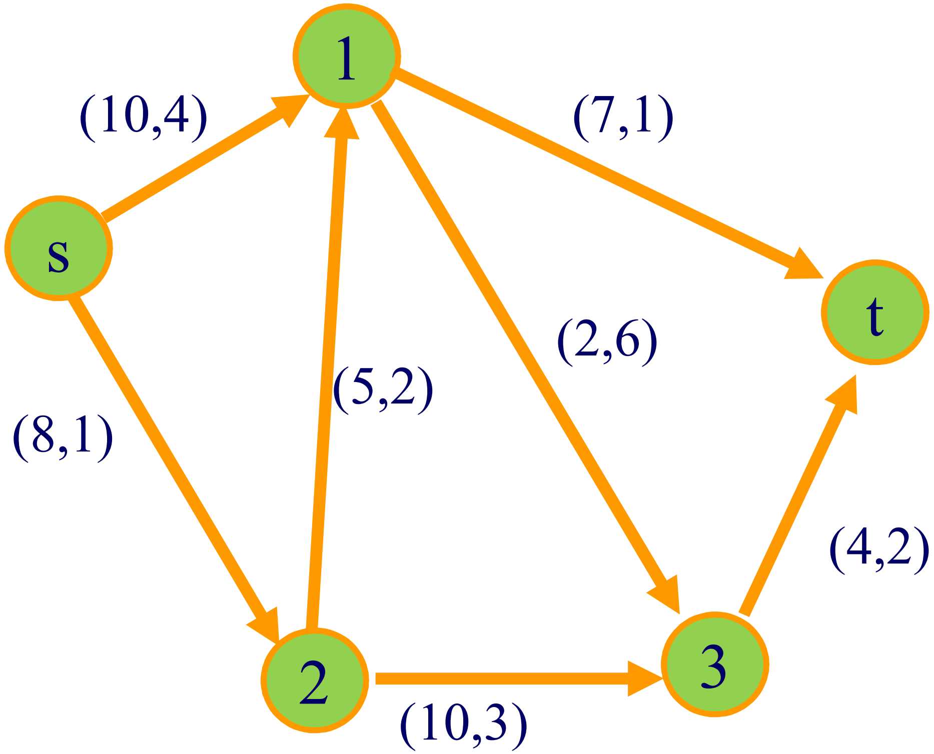
- 定理： 网络G中，从v_s到v_t的最大流等于分离v_s和v_t的最小割的容量



第八章 图与网络分析

- 图的基本概念
- 最短路径问题
- 最大流问题
- 最小费用流问题

例



已知$(\mathbf{u}_{ij},\mathbf{c}_{ij})$，求$f_g=\mathbf{10}$的最小费用流

最小费用流问题

- 最小费用流问题：在容量网络**G=(V,E,U,C)**中，设边(v_i,v_j)的容量为u_{ij}，单位流量的费用为c_{ij}，求一个可行流f，使$f=f_g$时，总费用最小。

$$\min \sum_{(v_i,v_j)\in E} c_{ij}x_{ij}$$

$$\text{s.t.} \quad \sum_k x_{ki} = \sum_j x_{ij} \qquad v_i \in V \setminus \{v_s, v_t\}$$

$$\sum_k x_{kt} = \sum_j x_{sj} = f_g$$

$$0 \le x_{ij} \le u_{ij} \qquad i,j=1,2,\dots,n$$

求解思路

思路：**1**。先寻找一个初始最小费用可行流$f_0 < f_g$

2。沿着可增广链增加$f = f_0 + \Delta$，使$f \to f_g$

问题：如何保证$f = f_0 + \Delta$是最小费用流？

最小费用可增广链

■可增广链的费用：设μ为起点v_s到终点v_t的一条可增广链，其费用为**z(μ)**

$$z(\mu)=\sum_{\mu^+}c_{ij}-\sum_{\mu^-}c_{ij}$$

■最小费用可增广链：链的费用最小

◆ 定理：设f为流量f_0的最小费用流， μ为起点v_s到终点v_t的一条最小费用可增广链，设f'为经过μ调整流量θ后得到的新的可行流，则f'一定是流量为f_0+ θ的最小费用流

对偶算法

对偶算法的思路：

1。以$\boldsymbol{f_0}$ **=0**为初始可行流

2。找到最小费用可增广链

3。作最大可行调整，使$\boldsymbol{f_0} + \theta = \boldsymbol{f_g}$，若不能达到，重复第**2**步

$$\theta = \min\left\{\min_{\mu^+}(u_{ij} - x_{ij}), \min_{\mu-}(x_{ij}), f_g - f_0\right\}$$

问题：如何保证$\boldsymbol{f} = \boldsymbol{f_0} + \Delta$是最小费用流？

最小费用可增广链的寻找

■虚拟（单位费用）长度网络**L(*f*)**

1。 $(v_i,v_j)\in$ E，

$$l_{ij}=\begin{cases} c_{ij} & x_{ij}<u_{ij} \\ +\infty & x_{ij}=u_{ij} \end{cases}$$

饱和边不可增加流量

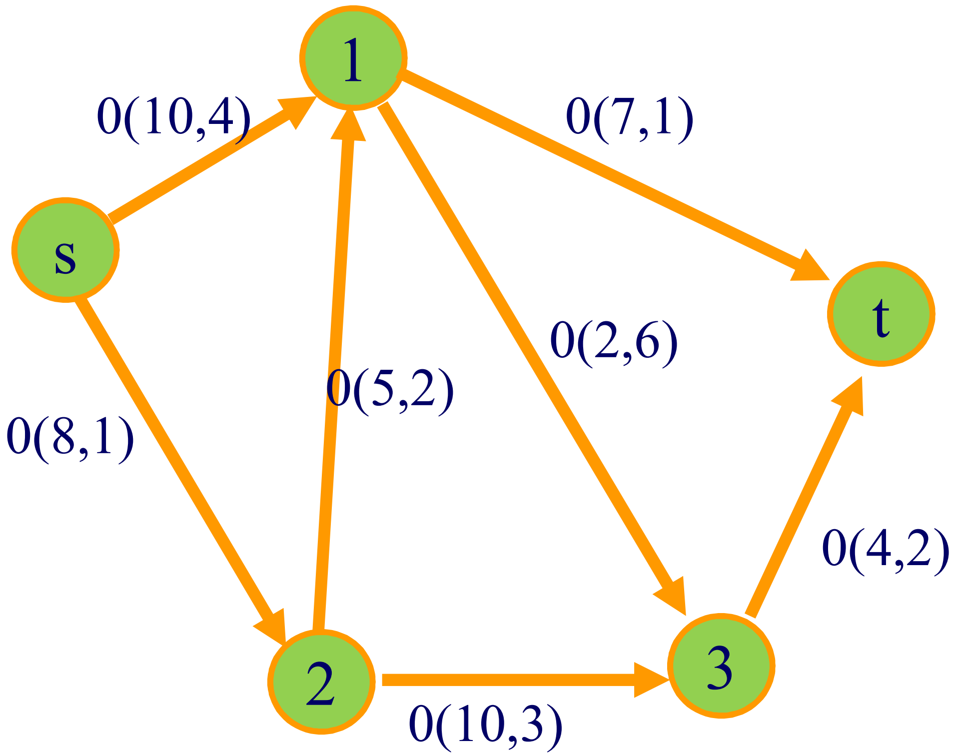
2。 $(v_i,v_j)\in$ E，考虑反向边

$$l_{ji}=\begin{cases} -c_{ij} & x_{ij}>0 \\ +\infty & x_{ij}=0 \end{cases}$$

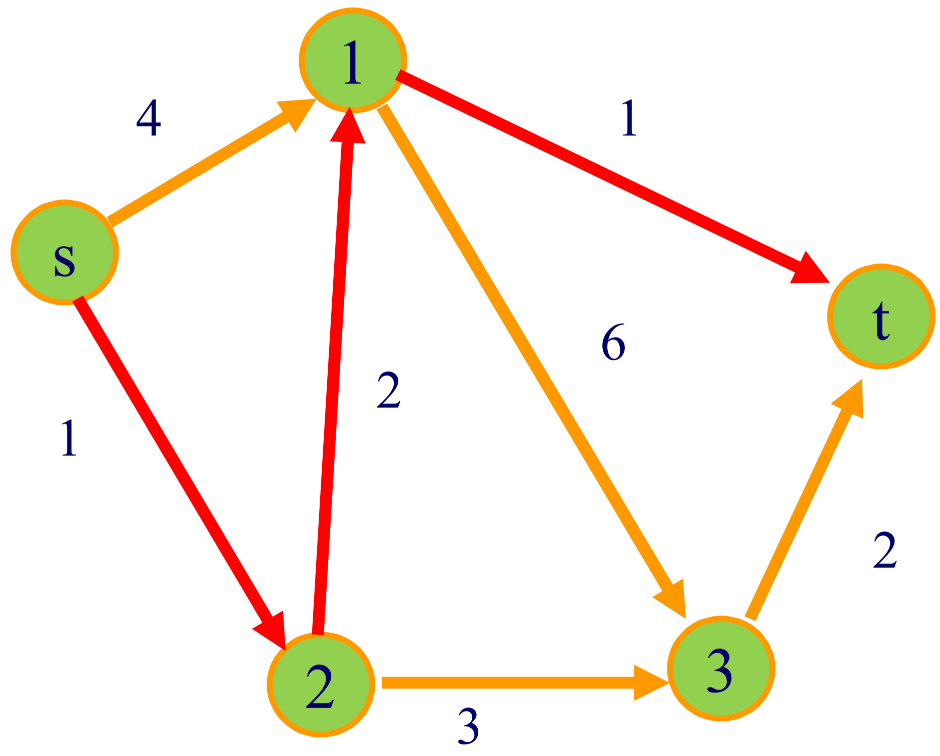
饱和边不可增加流量

求*f*的最小费用可增广链⇔求**L(*f*)**中起点与终点间的最短路

$f_0=0$



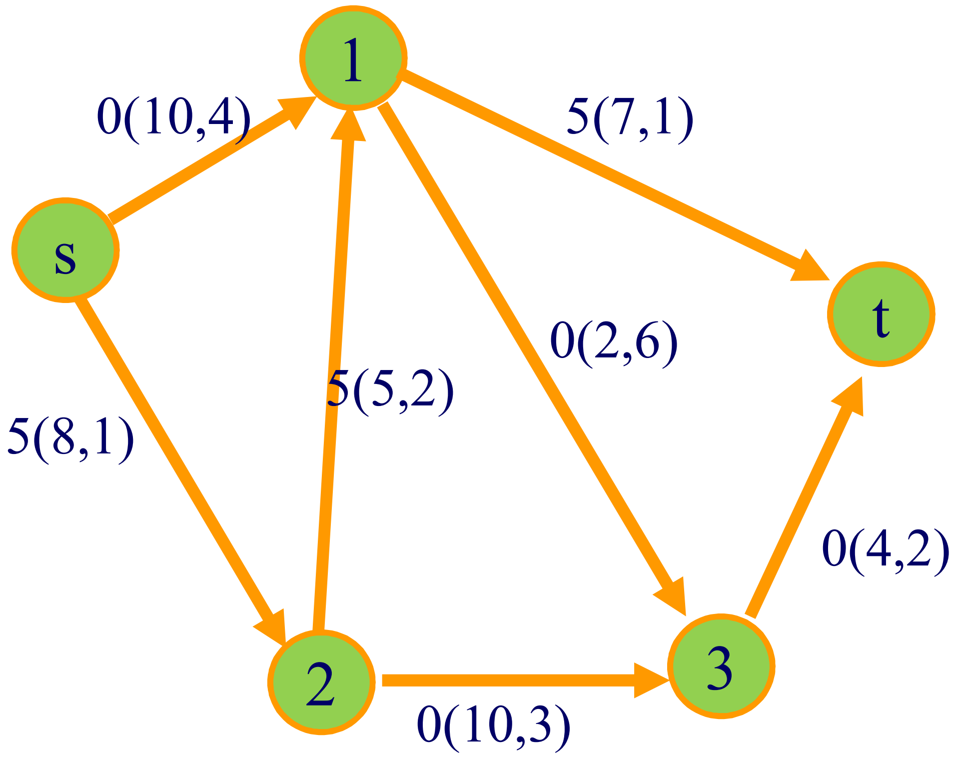
f_0



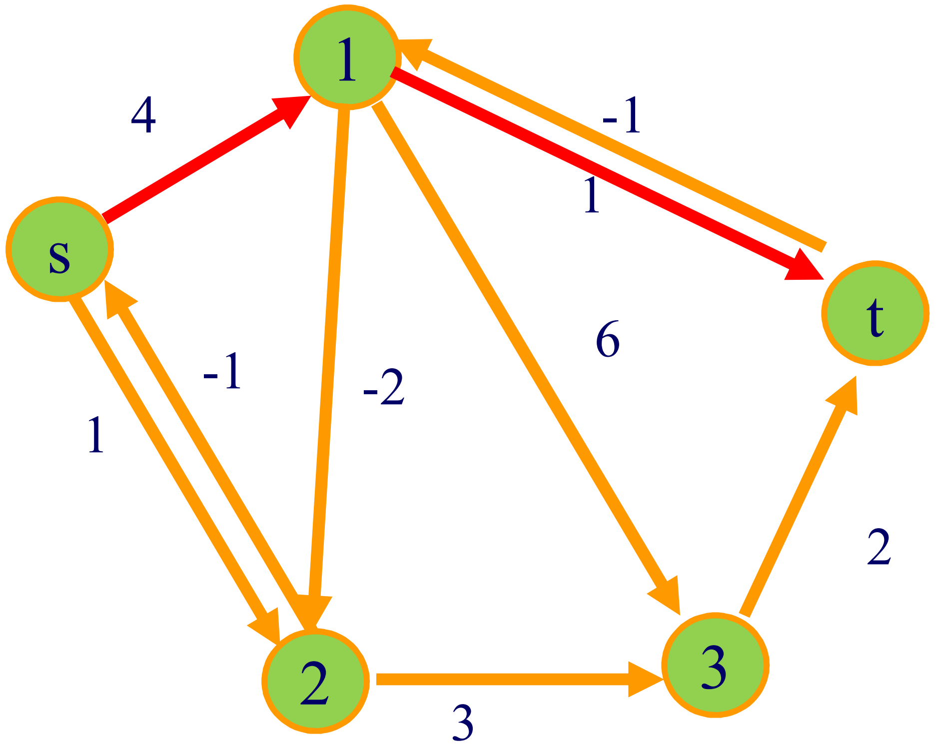
$L(f_0)$

μ=s→2 →1→t θ=min(8,5,7)=5 $f_1= f_0+5=5$

f_1=5



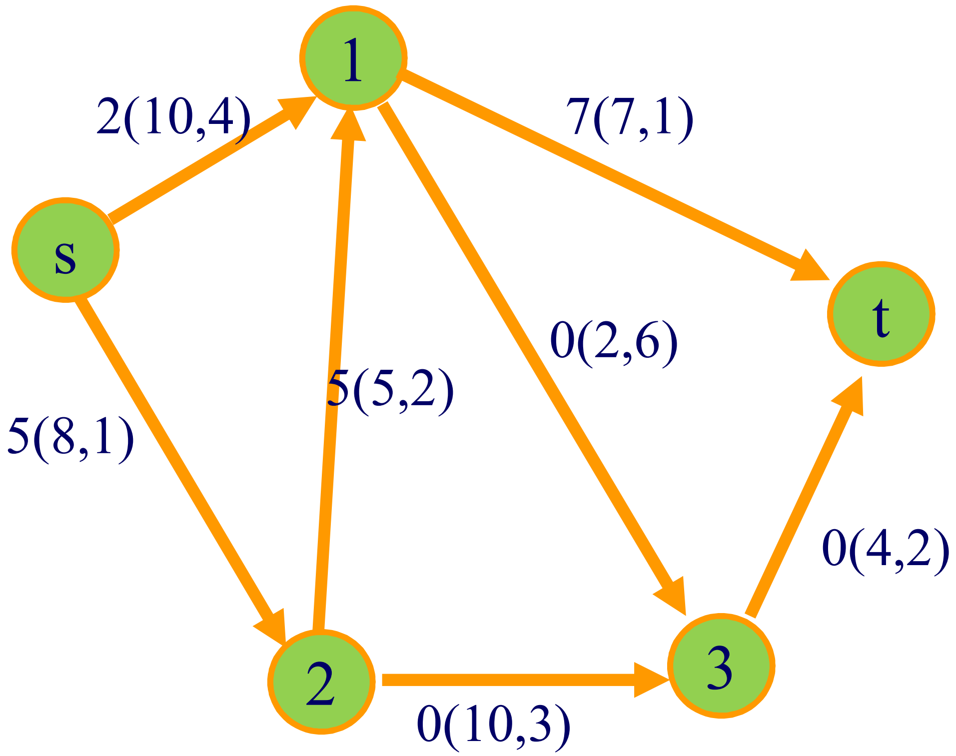
f_1



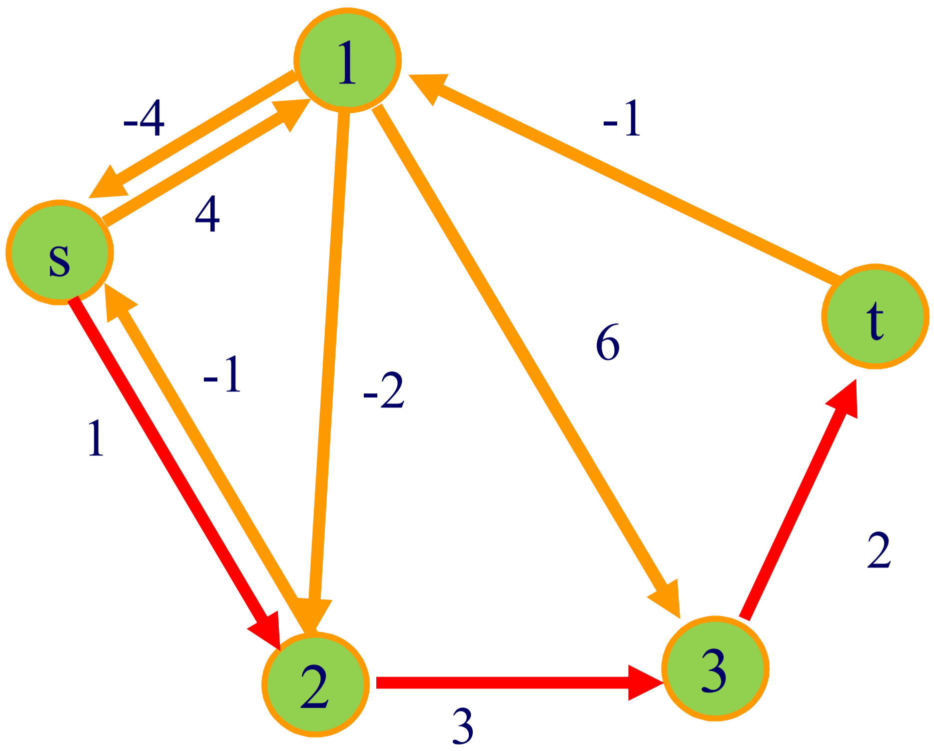
L(f_1)

μ=s→1→t θ=min(10,2)=2 f_2= f_1+2=7

f_2=7



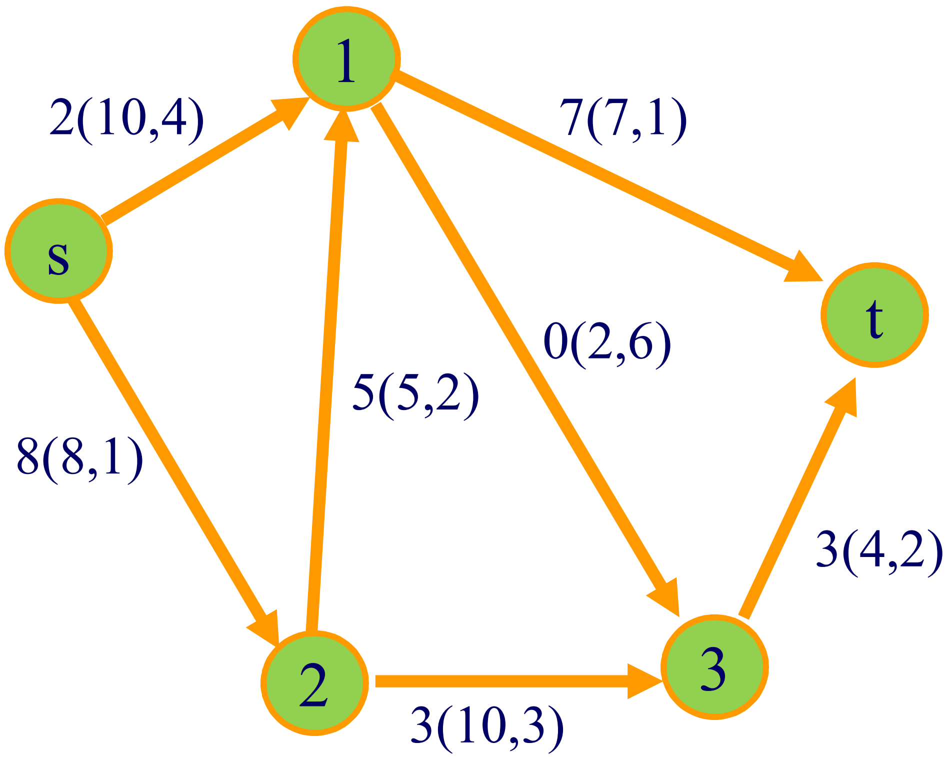
f_2



L(f_2)

μ=s →2 → 3→t   θ=min(3,10,4)=3 f_3= f_2+3=10

$$f_3 = 10 = f_g$$



f_3