运筹学

（图与网络分析）

王焕钢

清华大学自动化系

要点：图与网络分析的应用

求解如下线性规划问题：

$$\min C^T X$$

$$s.t.\ \ AX = B,\ 0 \le X \le D$$

$$A=\begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & -1 & 1 & 0 & 1 & 0 \\ 0 & -1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 & -1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & -1 & -1 \end{bmatrix} \qquad B=\begin{bmatrix} 10 \\ 0 \\ 0 \\ 0 \\ -10 \end{bmatrix}$$

$$C^T=\begin{bmatrix} 4 & 1 & 2 & 6 & 3 & 1 & 2 \end{bmatrix}$$

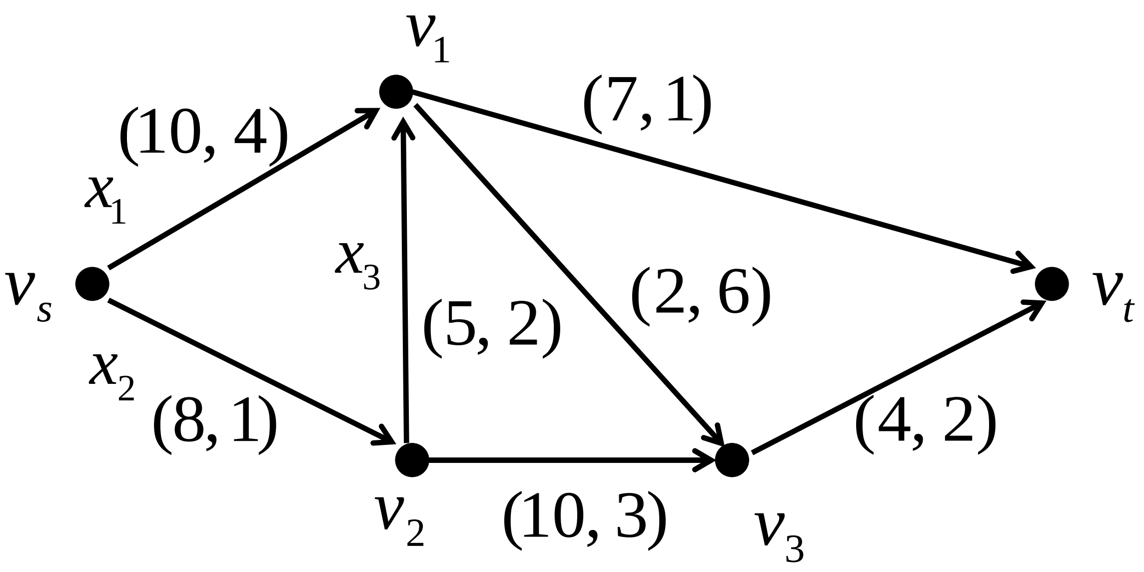
$$D^T=\begin{bmatrix} 10 & 8 & 5 & 2 & 10 & 7 & 4 \end{bmatrix}$$

求解如下线性规划问题：

$$\min C^T X$$

$$s.t.\ \ AX = B,\ 0 \le X \le D$$

$$A = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & -1 & 1 & 0 & 1 & 0 \\ 0 & -1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 & -1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & -1 & -1 \end{bmatrix}$$



$$B^T = \begin{bmatrix} 10 & 0 & 0 & 0 & -10 \end{bmatrix}$$

$$C^T = \begin{bmatrix} 4 & 1 & 2 & 6 & 3 & 1 & 2 \end{bmatrix}$$

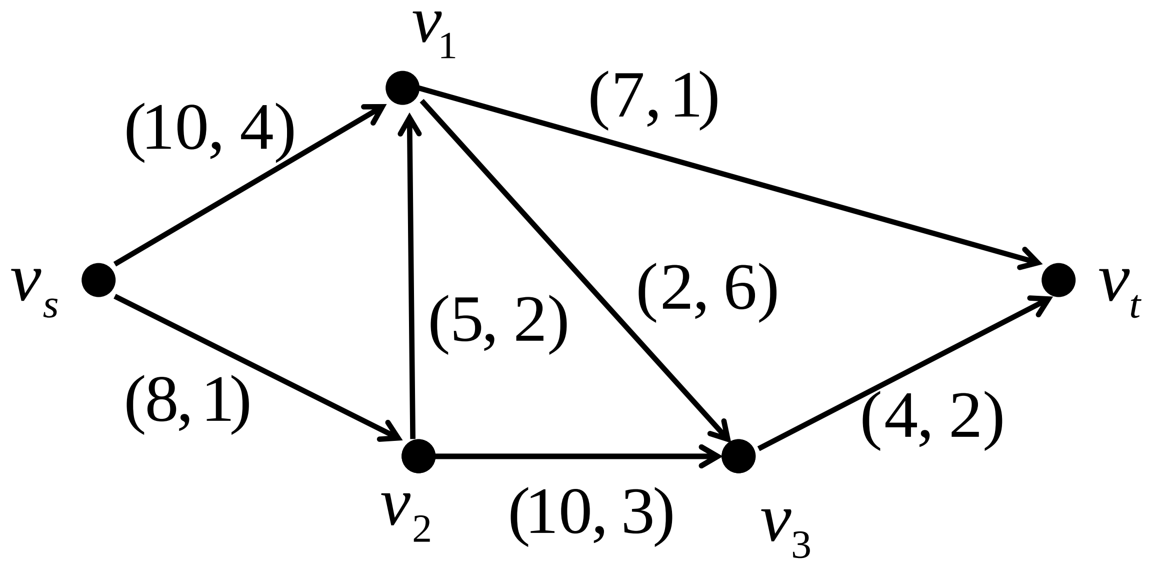
$$D^T = \begin{bmatrix} 10 & 8 & 5 & 2 & 10 & 7 & 4 \end{bmatrix}$$

v_s： $x_1 + x_2 = 10$

v_1： $-x_1 - x_3 + x_4 + x_6 = 0$

$\vdots$

等价于如下最小费用流问题



括号内第一个数字是容量，第二个是单位流量费用

目标：从发点到收点的总的流量费用最小

约束：1）容量约束，各边流量不大于容量

2）流量平衡约束，各点进出流量总和相等

3）从发点到收点的总流量为10

如下例

销地 / 产地	B_1	B_2	B_3	B_4	产量
A_1	x_{11} 8	x_{12} 6	x_{13} 10	x_{14} 9	35
A_2	x_{21} 9	x_{22} 12	x_{23} 13	x_{24} 7	50
A_3	x_{31} 14	x_{32} 9	x_{33} 16	x_{34} 5	40
销量	45	20	30	30	

数学规划模型为

$$\min \sum_{i=1}^{3}\sum_{j=1}^{4} c_{ij}x_{ij} = 8x_{11} + 6x_{12} + 10x_{13} + 9x_{14} + 9x_{21} + 12x_{22} + 13x_{23}$$

$$+ 7x_{24} + 14x_{31} + 9x_{32} + 16x_{33} + 5x_{34}$$

$$\text{s.t.}\quad x_{11} + x_{12} + x_{13} + x_{14} = 35$$

$$x_{21} + x_{22} + x_{23} + x_{24} = 50$$

$$x_{31} + x_{32} + x_{33} + x_{34} = 40$$

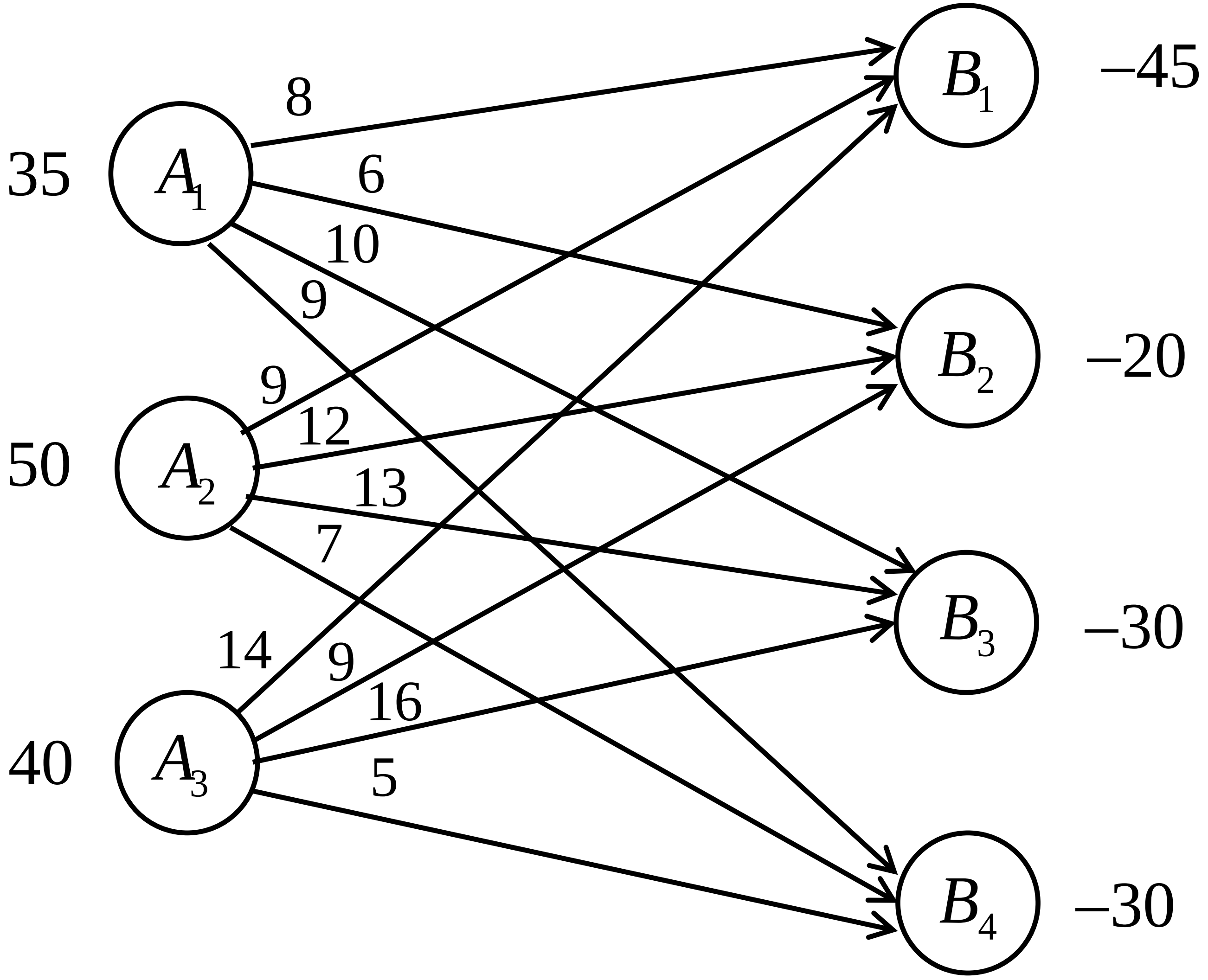
$$x_{11} + x_{21} + x_{31} = 45$$

$$x_{12} + x_{22} + x_{32} = 20$$

$$x_{13} + x_{23} + x_{33} = 30$$

$$x_{14} + x_{24} + x_{34} = 30 \qquad x_{ij} \ge 0,\ \forall 1 \le i \le 3, 1 \le j \le 4$$

怎样调运这些物品能使总费用最小？



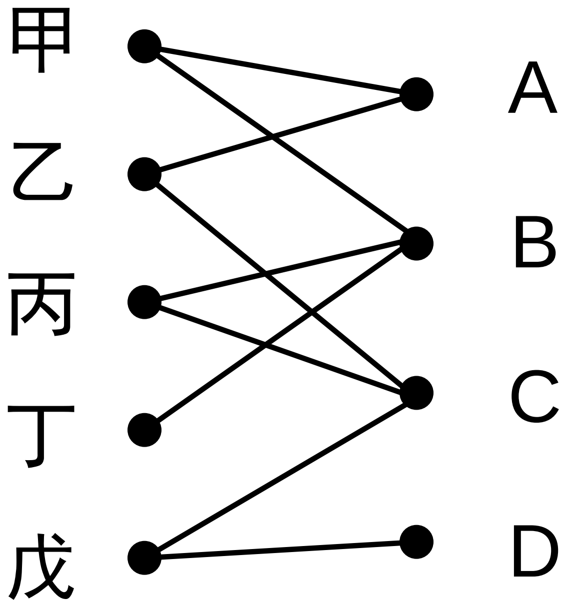
要点：图与网络的基本概念

图的定义

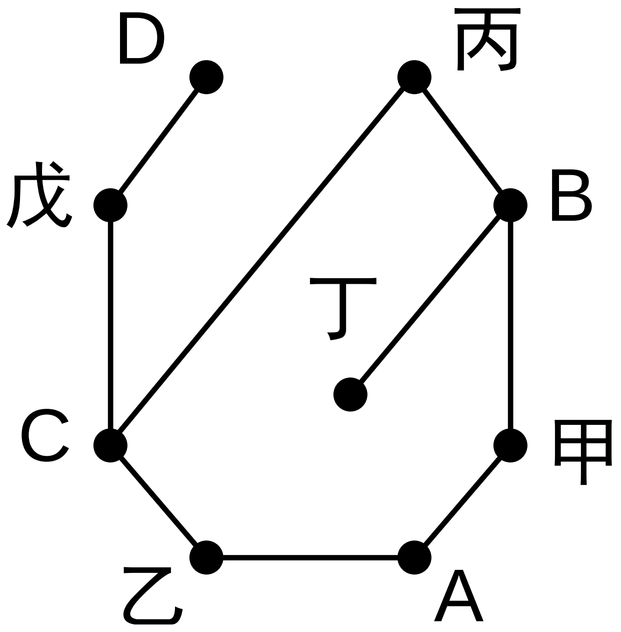
点集 V $\quad\rightarrow\quad$ 边集 E $\quad\rightarrow\quad$ 图 $G=(V,E)$

例 $V=\{$ 甲，乙，丙，丁，戊，A，B，C，D $\}$

$$E=\left\{\begin{array}{l}(\text{甲},\mathrm{A}),(\text{甲},\mathrm{B}),(\text{乙},\mathrm{A}),(\text{乙},\mathrm{C}),(\text{丙},\mathrm{B})\\(\text{丙},\mathrm{C}),(\text{丁},\mathrm{B}),(\text{戊},\mathrm{C}),(\text{戊},\mathrm{D})\end{array}\right\}$$



$\Leftrightarrow (V,E) \Leftrightarrow$



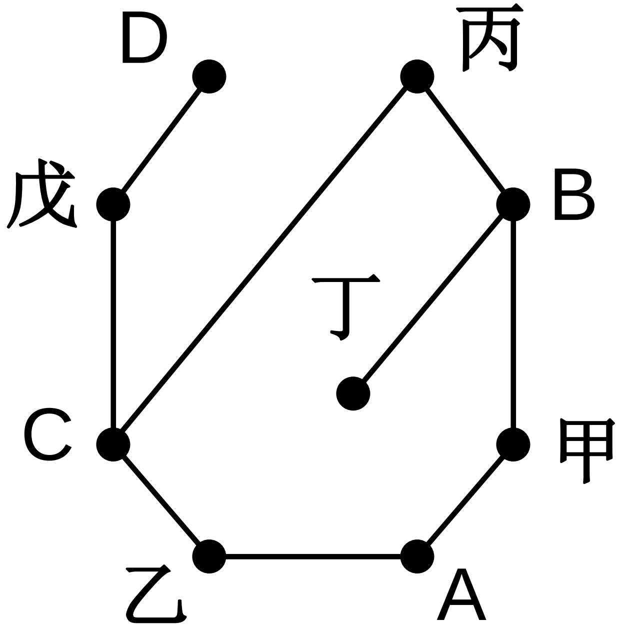
无向图

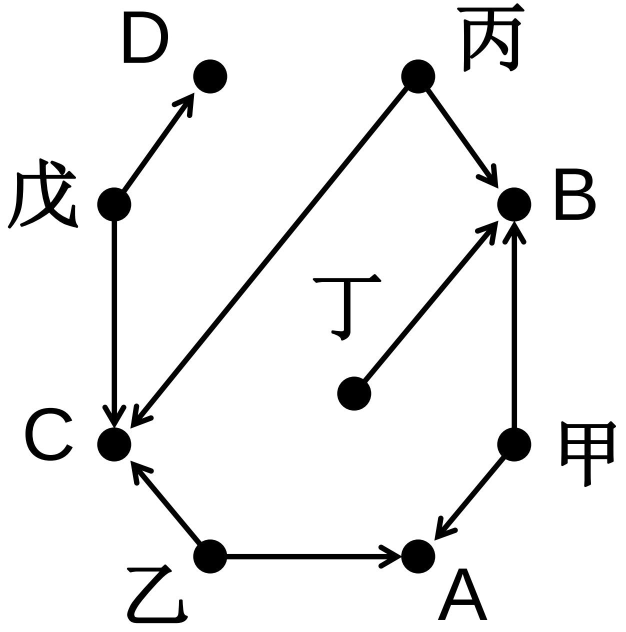
$$(\text{甲}, A) = (A, \text{甲})$$

有向图

$$(\text{甲}, A) \neq (A, \text{甲})$$

始点 甲 $\bullet\!\longrightarrow\!\bullet$ A $\bullet\!\longrightarrow\!\bullet$ 甲 终点





部分参考资料将有向图的边称为弧，以示区别

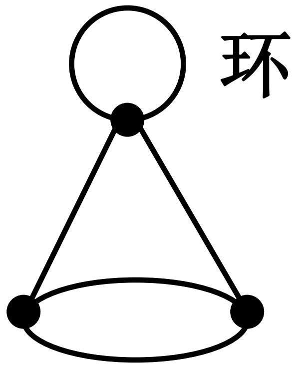
相邻　如果 $v_1, v_2 \in V$，$(v_1, v_2) \in E$，称 v_1, v_2 相邻，称 v_1, v_2 为 (v_1, v_2) 的端点

如果 $e_1, e_2 \in E$，并且有公共端点 $v \in V$，称 e_1, e_2 相邻，称 e_1, e_2 为 v 的关联边

对 $G=(V, E)$，$m(G)=|E|, n(G)=|V|$ 表示边数和点数

自回路 两端点相同的边，或称为环

多重边 两点之间多于一条不同的边



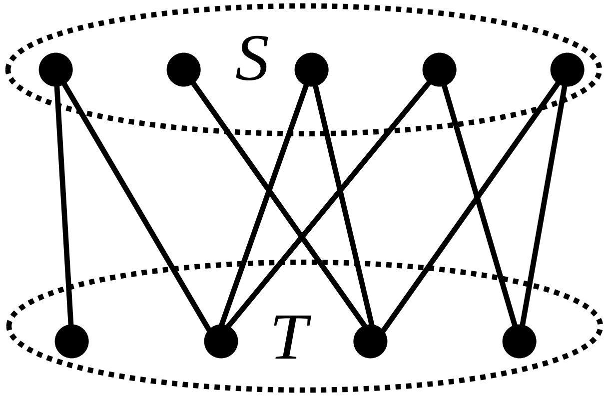
简单图与多重图

简单图 不含自回路和多重边的图

多重图 含有多重边的图

完全图 任意两个顶点间都有边相连的无向简单图称为完全图，有 n 个顶点的无向完全图记为 K_n ，任意两个顶点间都有且仅有一条边相连的有向简单图称为有向完全图

二分图 如果 V 可以划分为两个不相交的子集 $S,T,$ 使得 E 中每条边的两个端点必有一个属于 S ，一个属于 T ，称 $G=(V,E)$ 为二分图记为 $G=(S,T,E)$ 如右图



端点的次 以点 v 为端点的边数称为 v 的次，记为 $\deg(v)$ ，或简记为 $d(v)$

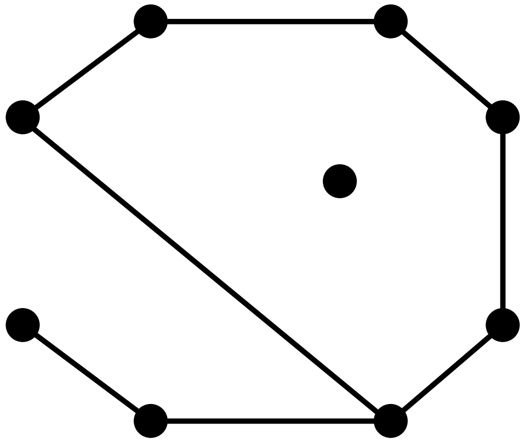
次为 0 的点称为孤立点，次为 1 的点称为悬挂点，连接悬挂点的边称为悬挂边，次为奇数的点称为奇点，次为偶数的点称为偶点

出次与入次

在有向图中，以 v 为始点的边数称为 v 的出次，用 $d^{+}(v)$ 表示，以 v 为终点的边数称为 v 的入次，用 $d^{-}(v)$表示

定理1 任何图中，顶点次数总和等于边数的 2 倍

定理2 任何图中，奇点的个数为偶数个



顶点次数总和等于 16

边数等于 8

奇点的个数为 2 个

定理1显然成立，下面证明定理2：根据定理1

奇点次数总和 ＋ 偶点次数总和 ＝ 偶数

（偶数）

若有奇数个奇点，其次数总和为奇数，上式不成立

子图

对于图 $G=(V,E)$，如果 E' 是 E 的子集，V' 是 V 的子集，并且 E' 中的边仅与 V' 中的顶点相关联，则称 $G'=(V',E')$ 是 G 的子图，特别是，若 $V'=V$，则称 G' 为 G 的支撑子图

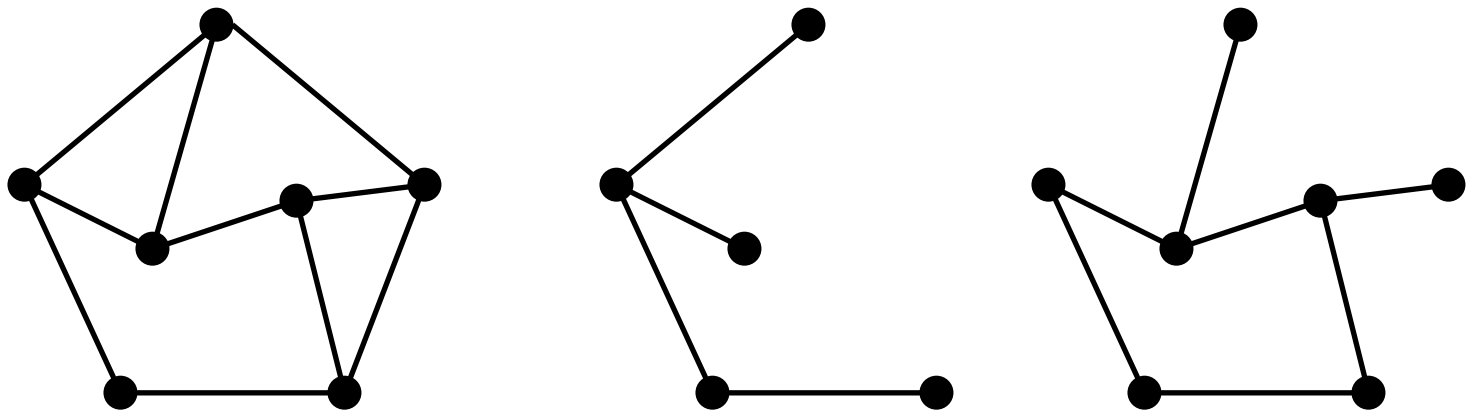
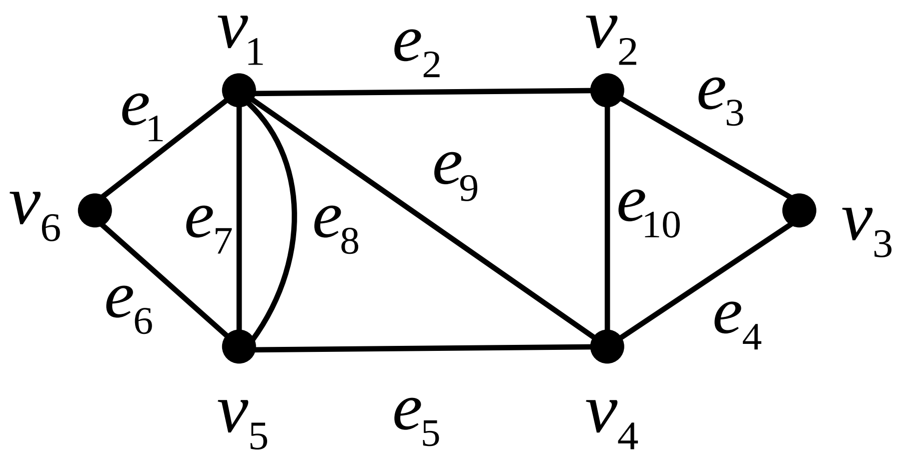


图　　子图　　支撑子图

路（链） 点、边交替（可重复）的序列，如

$$S=\{v_6,e_6,v_5,e_7,v_1,e_8,v_5,e_7,v_1,e_9,v_4,e_4,v_3\}$$



无重复边的路称为简单路，无重复点的路为初级路

$$S_1=\{v_6,e_6,v_5,e_5,v_4,e_4,v_3\}$$

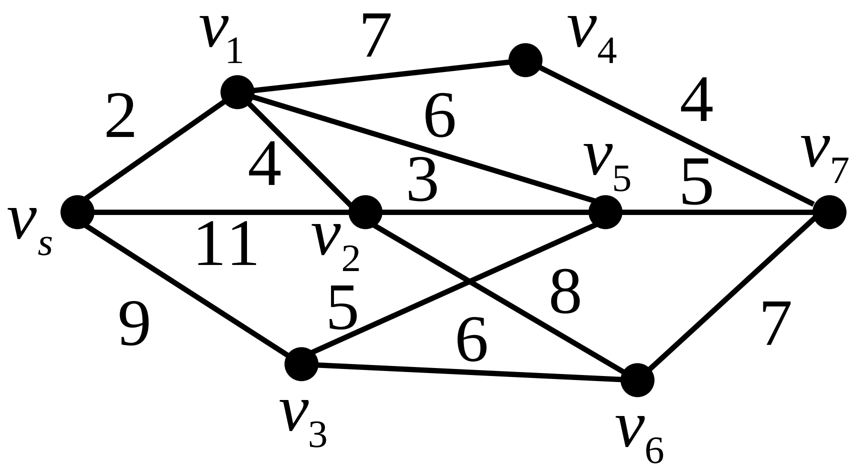
始点和终点为同一点的路称为回路

图的两点间若存在连接两点的一条路，称这两点连通

连通图 任意两点都连通的图

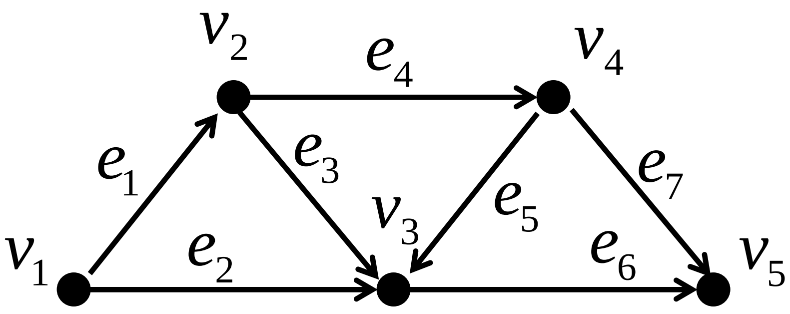
网络（赋权图）

每边赋有权（实数或实数向量）的图称为网络，记为 $G=\left(V,E,W\right)$，其中 $W=\left\{w(e),\ e\in E\right\}$ 是权的集合，无向图赋权构成无向网络，有向图赋权构成有向网络，下图为一个无向网络



要点：图与网络的矩阵描述

有向图的关联矩阵



$$
\begin{array}{c}
 \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5
\end{array}
\begin{array}{c}
\begin{array}{ccccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 & e_7 \end{array} \\
\begin{pmatrix}
1 & 1 & & & & & \\
-1 & & 1 & 1 & & & \\
 & -1 & -1 & & -1 & 1 & \\
 & & & -1 & 1 & & 1 \\
 & & & & & -1 & -1
\end{pmatrix}
\end{array}
$$

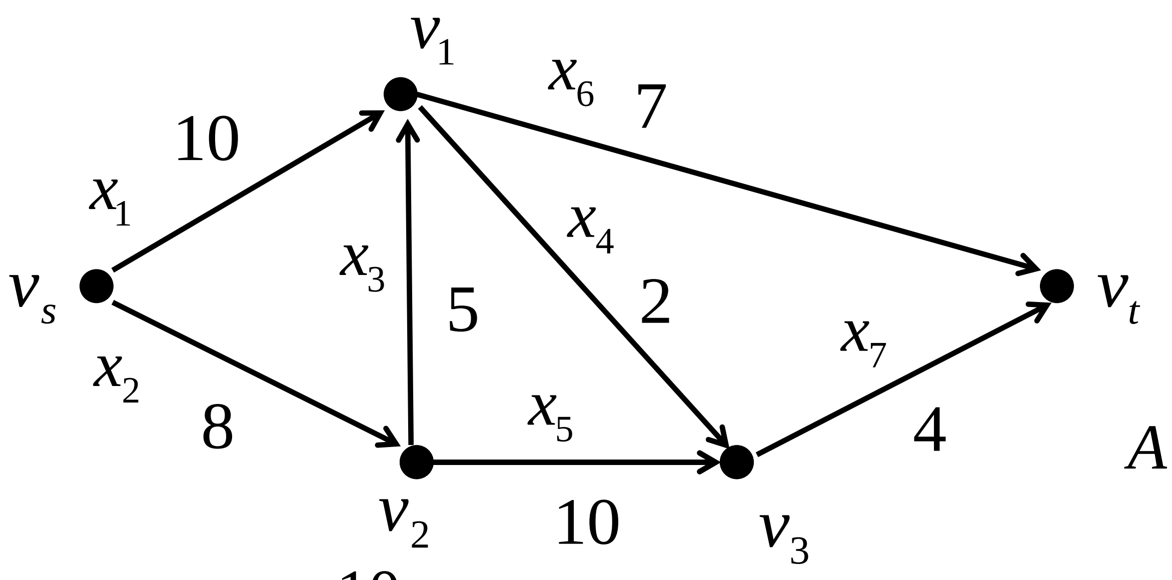
将有向图关联矩阵的-1换成1
就得到无向图关联矩阵

图的邻接矩阵

两点间有边为1，否则为0

$$
\begin{array}{c|ccccc}
 & v_1 & v_2 & v_3 & v_4 & v_5 \\
\hline
v_1 & 0 & 1 & 1 & 0 & 0 \\
v_2 & 1 & 0 & 1 & 1 & 0 \\
v_3 & 1 & 1 & 0 & 1 & 1 \\
v_4 & 0 & 1 & 1 & 0 & 1 \\
v_5 & 0 & 0 & 1 & 1 & 0
\end{array}
$$

网络的约束描述：



$$s.t.\ \ AX = B,\ 0 \le X \le D$$

$$A = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & -1 & 1 & 0 & 1 & 0 \\ 0 & -1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 & -1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & -1 & -1 \end{bmatrix}$$

v_s： $x_1 + x_2 = 10$

v_1： $-x_1 - x_3 + x_4 + x_6 = 0$

v_2： $-x_2 + x_3 + x_5 = 0$

v_3： $-x_4 - x_5 + x_7 = 0$

v_t： $-x_6 - x_7 = -10$

$$B^T = \begin{bmatrix} 10 & 0 & 0 & 0 & -10 \end{bmatrix}$$

$$D^T = \begin{bmatrix} 10 & 8 & 5 & 2 & 10 & 7 & 4 \end{bmatrix}$$

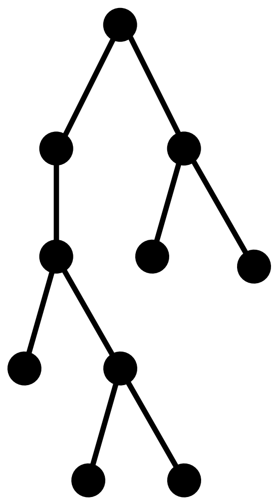
$x_1 \le 10, x_2 \le 8, x_3 \le 5, x_4 \le 2, x_5 \le 10, x_6 \le 7, x_7 \le 4$

要点：树的概念与特性

树　连通且不含圈的无向图（森林：无圈的图）

定理 1

设 $T=(V,E)$，$n=|V|\geq 3$，$m=|E|$，则以下等价：



$m=9$

$n=10$

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，但每加一新边（不加点）即得唯一一个圈

5）T 连通，但舍去任一边就不连通

6）T 中任意两点，有唯一路相连

定理 2 每个树**至少有**两个次为 1 的点。

若 $T=(V,E)$ 恰好有**两个**次为 1 的点，则其它点的次必为 2，因此 $T=(V,E)$ 是**一条链**（路）

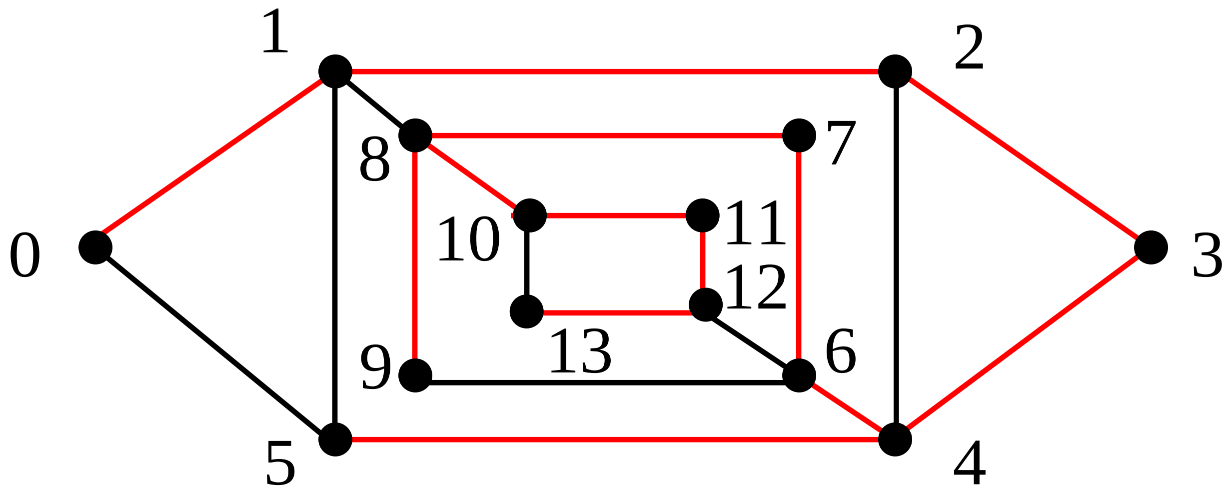
要点：支撑树与最小支撑树

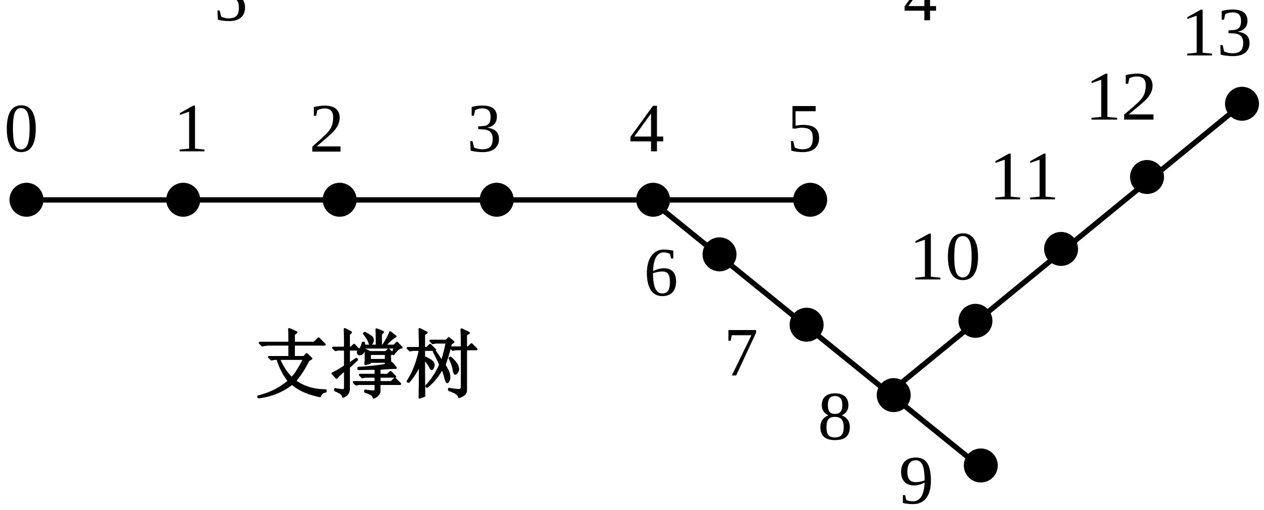
图的支撑树 如果 $G=(V,E)$ 的支撑子图 $G'=(V,E')$ 是树，称其为 G 的支撑树， G 中属于支撑树的边称为**树枝**，不属于支撑树的边称为**弦**

定理 3 图 $G=(V,E)$ 有支撑树的充要条件是 G 是**连通图**

1）深探法

从任意点开始，边标号边前进，只至标完所有点

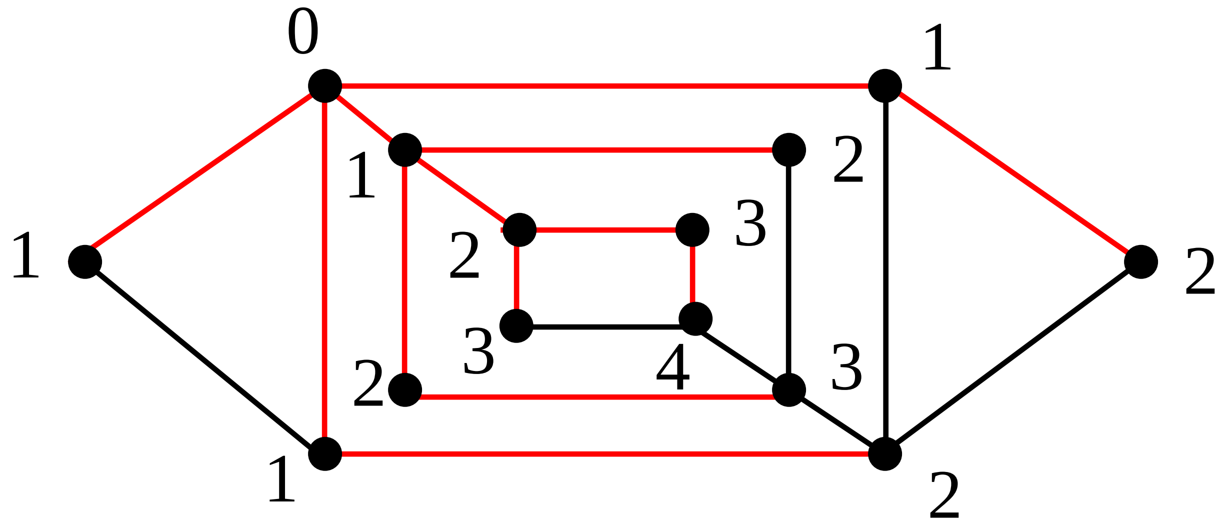


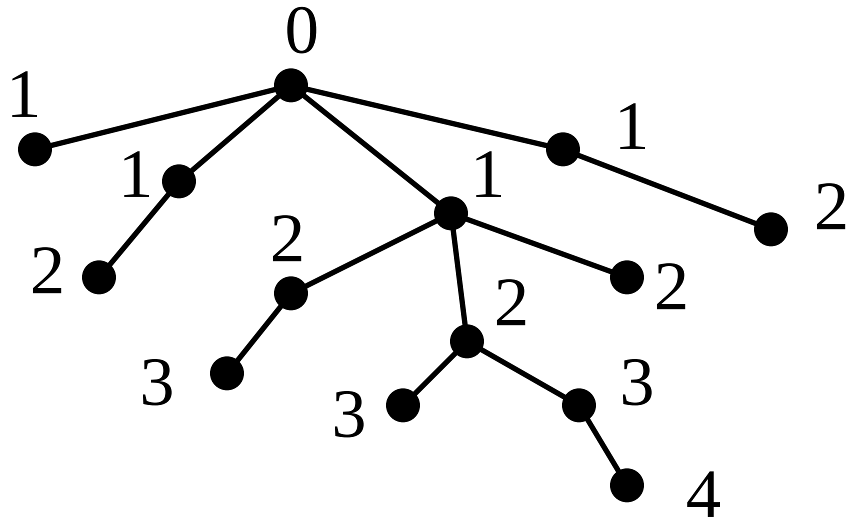


支撑树

2）广探法

从任意点开始，把当前标号点附近标号完再前进





支撑树

最小支撑树（在交通网、电力网等设计中广泛应用）

连通图 $G=(V,E)$ 的每条边上有非负权 $l(e)$ ，一个

支撑树 $T=(V,E')$ 的所有树枝上的权的总和：

$$L(T)=\sum_{e\in E'} l(e)$$

称为这个支撑树的权（长） $\Rightarrow$ $G=(V,E,W)$

具有最小权的支撑树称为最小支撑树（最小树）

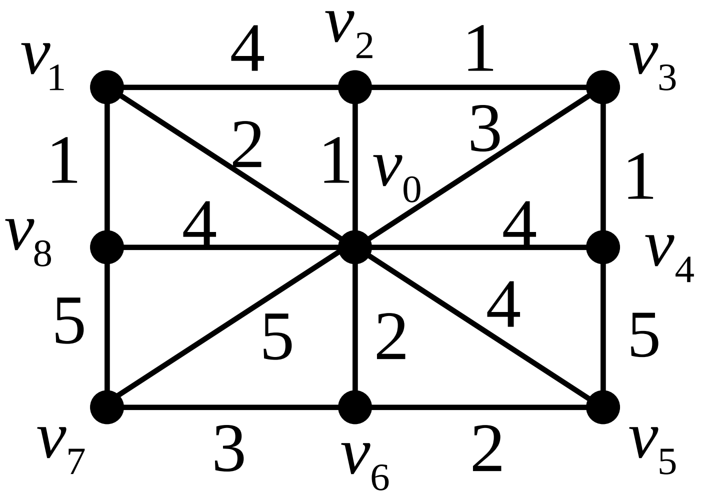
要点：求最小支撑树的避圈算法

求最小支撑树的 Kruskal 算法（避圈法）

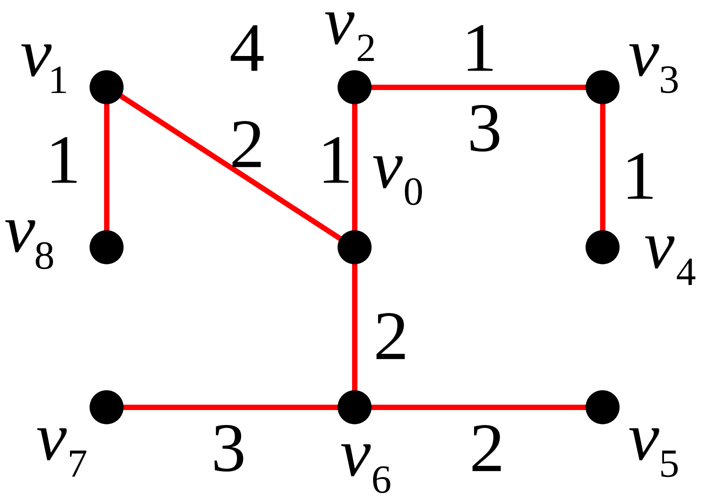
Kruskal 于1956年提出，是一种“贪心算法”

将所有边按权值从小到大排序，从权值最小的边开始选树枝，如果可能形成圈则跳过，直到选够顶点数减 1 的树枝

例



$\Rightarrow$



所有边从小到大排列

$$(v_0,v_2)=1,(v_2,v_3)=1,(v_3,v_4)=1,(v_1,v_8)=1,(v_0,v_1)=2$$

$$(v_0,v_6)=2,(v_5,v_6)=2,(v_0,v_3)=3,(v_6,v_7)=3,(v_0,v_4)=4$$

$$(v_0,v_5)=4,(v_0,v_8)=4,(v_1,v_2)=4$$

从小到大顺序选择不构成圈的边形成右上支撑树

性质：加入任何弦形成的圈中，弦的权值最大

定理　T 是最小支撑树的充要条件是：加入任何弦形成的圈中，弦的权值最大

$\Rightarrow$　Kruskal 算法产生的是最小支撑树

证明必要性：

如果加入某个弦形成的圈中有比该弦的权值更大的树枝，则用该弦代替最大数值形成的支撑树的总权值会变小，和最小支撑树定义矛盾

证明充分性：

设 T_1 是满足条件的支撑树，T_2 是所有最小支撑树中和 T_1 不同的树枝数最少的树（一定存在），记

$$T_1 = \left\{e_1, e_2, \cdots, e_m, \hat{T}\right\}_2, \quad T_2 = \left\{\bar{e}_1, \bar{e}_2, \cdots, \bar{e}_m, \hat{T}\right\}$$

其中 $w(\bar{e}_1) \le w(\bar{e}_i), \ \forall 2 \le i \le m$

将 $\bar{e}_1$ 加入 T_1 会形成回路，一定有 $e_k \in T_1 \setminus \hat{T} \Rightarrow w(e_k) \le w(\bar{e}_1)$

将 e_k 加入 T_2 会形成回路，一定有 $\bar{e}_j \in T_2 \setminus \hat{T}$

$w(e_k) \le w(\bar{e}_1) \le w(\bar{e}_j) \quad \Rightarrow \quad T_2(e_k \setminus \bar{e}_j)$ 仍是最小支撑树

$T_2(e_k \setminus \bar{e}_j)$ 和 T_1 的不同树枝数为 $m-1$，矛盾！ $\Rightarrow \quad m = 0$

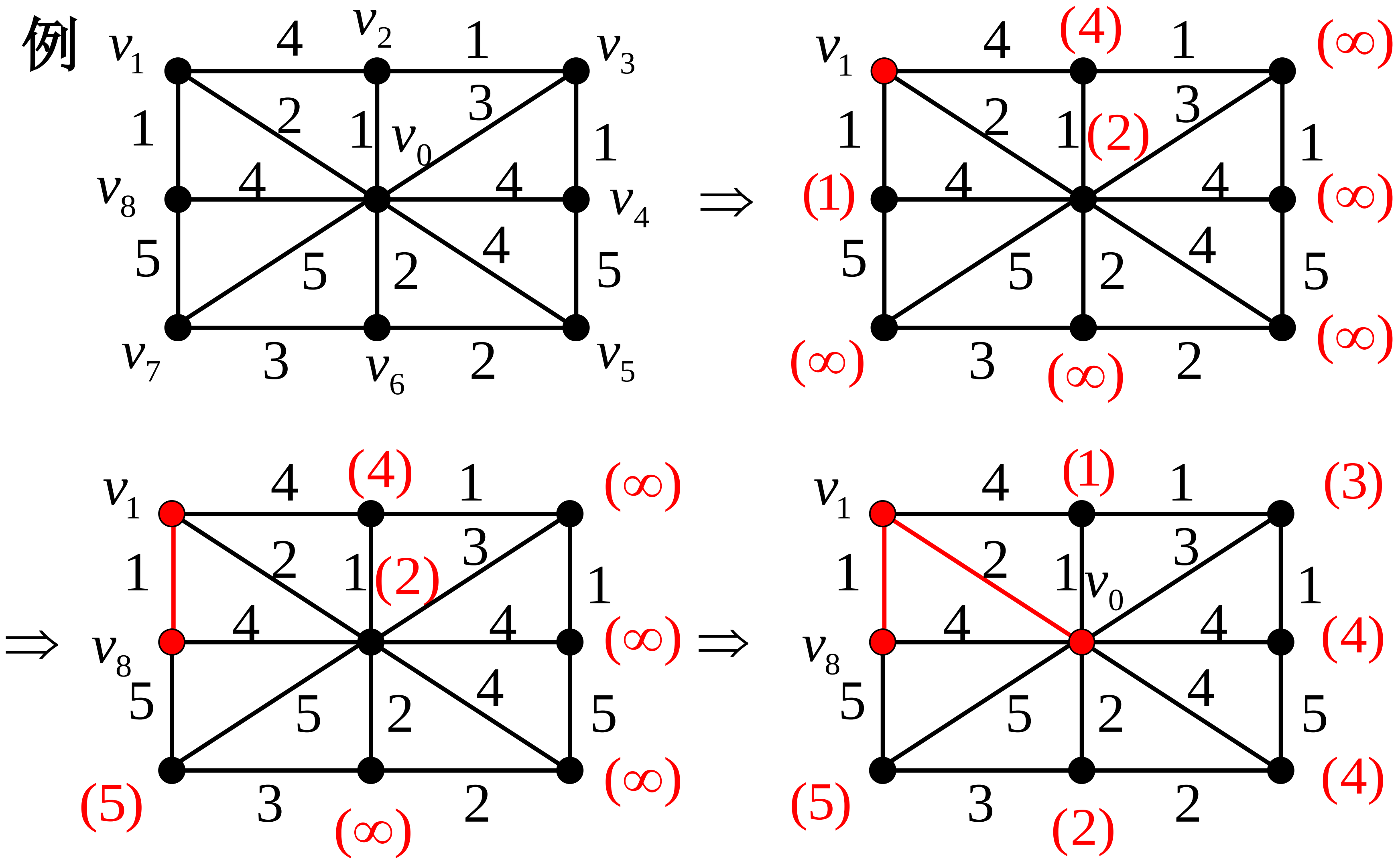
要点：求最小支撑树的 Dijkstra 算法

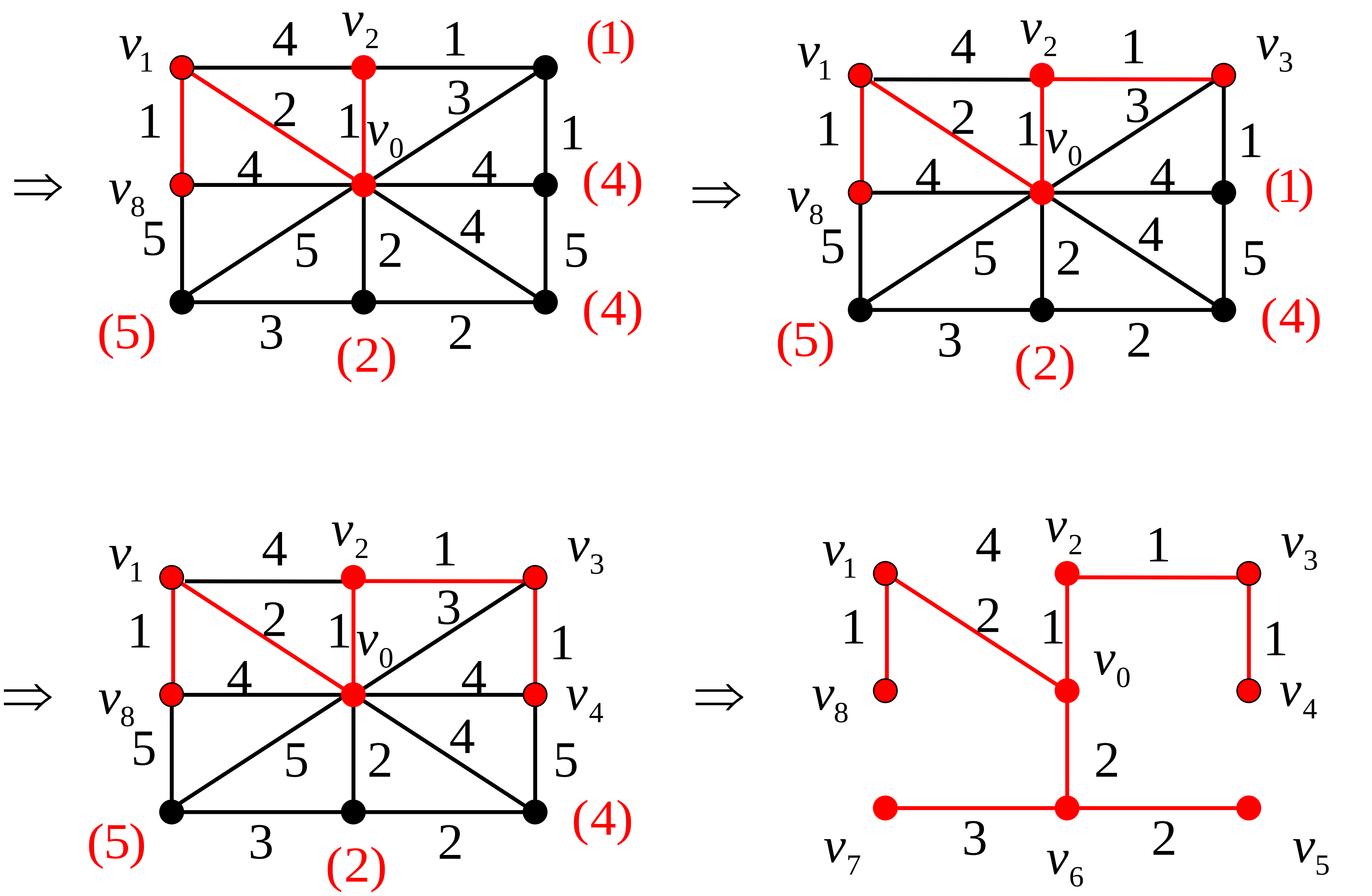
求最小支撑树的 Dijkstra 算法

从任意点开始逐渐增加某个点集，记为 S，每次从不在 S 的点集里选择距 S 一步距离最小的点加入 S，将相应边取为树枝，直至 S 包含所有的点

求最小支撑树的 Dijkstra 算法（1959年提出，$O(n^2)$）

例

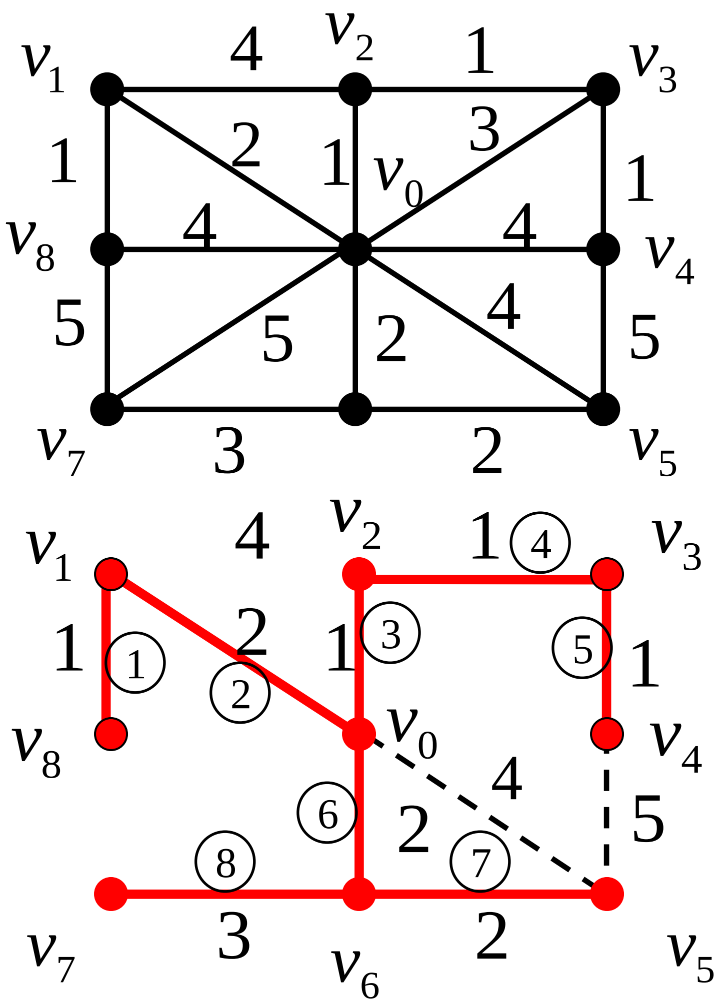



⇒
v1 v2 v0 v8
4 1 1 2 1 3 1 4 4 5 5 2 4 5 3 2
(1) (4) (4) (5) (2)
⇒
v1 v2 v3 v0 v8
4 1 1 2 1 3 1 4 4 5 5 2 4 5 3 2
(1) (4) (5) (2)
⇒
v1 v2 v3 v0 v8 v4
4 1 1 2 1 3 1 4 4 5 5 2 4 5 3 2
(4) (5) (2)
⇒
v1 v2 v3 v0 v8 v4 v7 v6 v5
4 1 1 2 1 1 2 3 2

Dijkstra算法生成的支撑树的性质

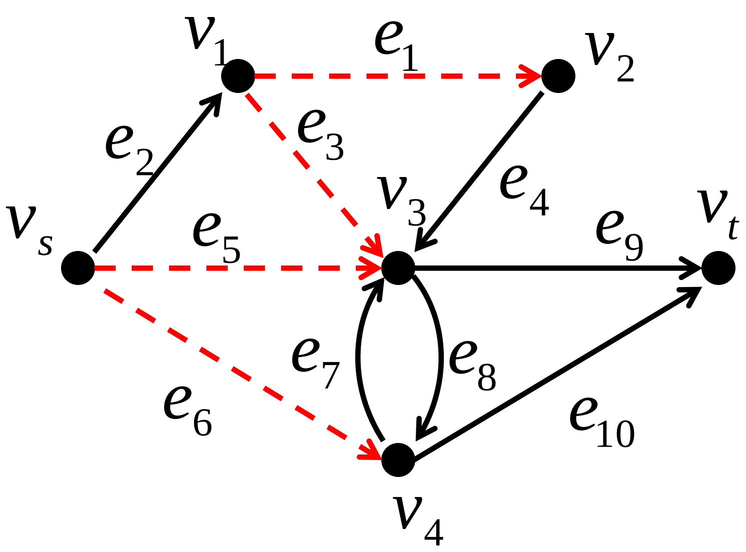
加入任何弦形成的回路中，弦的权值最大

理由：以右图回路为例，其中圆中数字为进入树枝的顺序。首先容易看出，第 7 次进入回路的树枝权一定不大于弦的权，第 6 次进入的树枝同样，然后可看出，第 5 次进入的一定不大于第 6 次进入的，然后以此类推可知上述性质成立

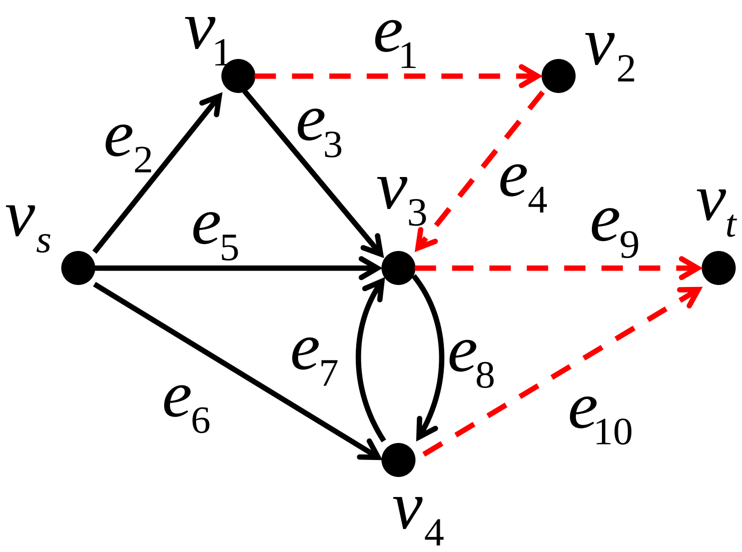


边割 对于图 $G=(V,E)$，任取 $S\subset V$，其补集为 $\bar{S}$，若 S 和 $\bar{S}$ 都不是空集，称两个端点分属 S 和 $\bar{S}$ 的**边的集合**为 G 的一个**边割**，记为 $\{S,\bar{S}\}$

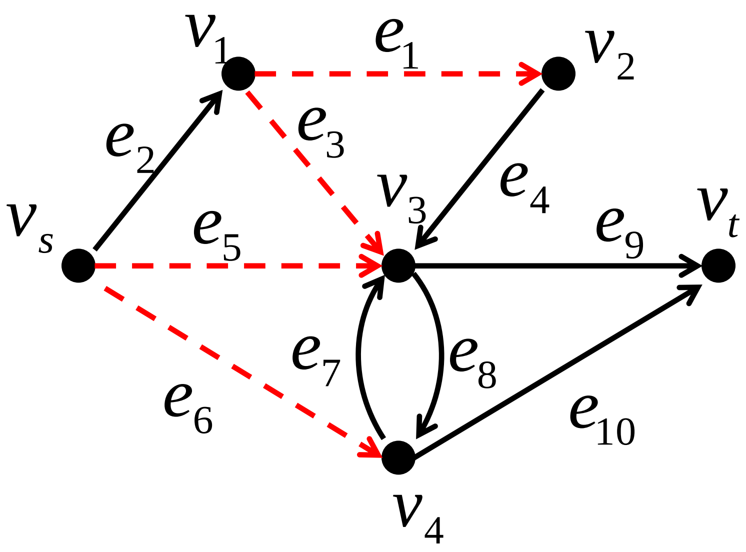
$\{e_1,e_3,e_5,e_6\}$

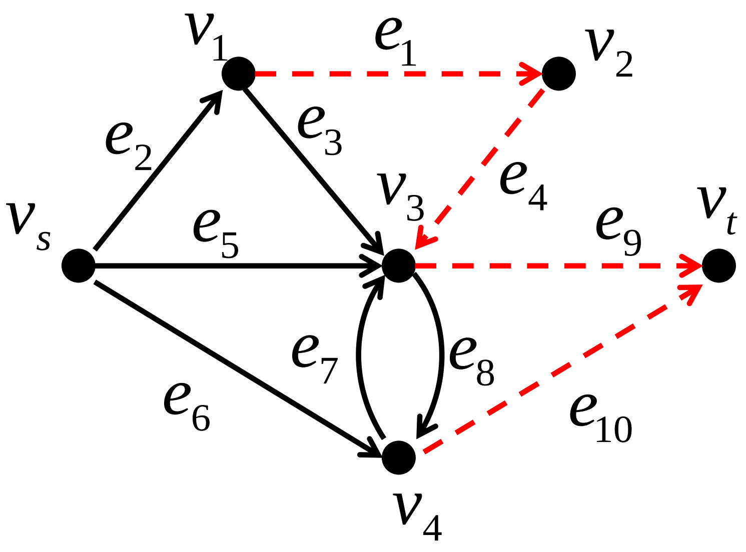


$\{e_1,e_4,e_9,e_{10}\}$



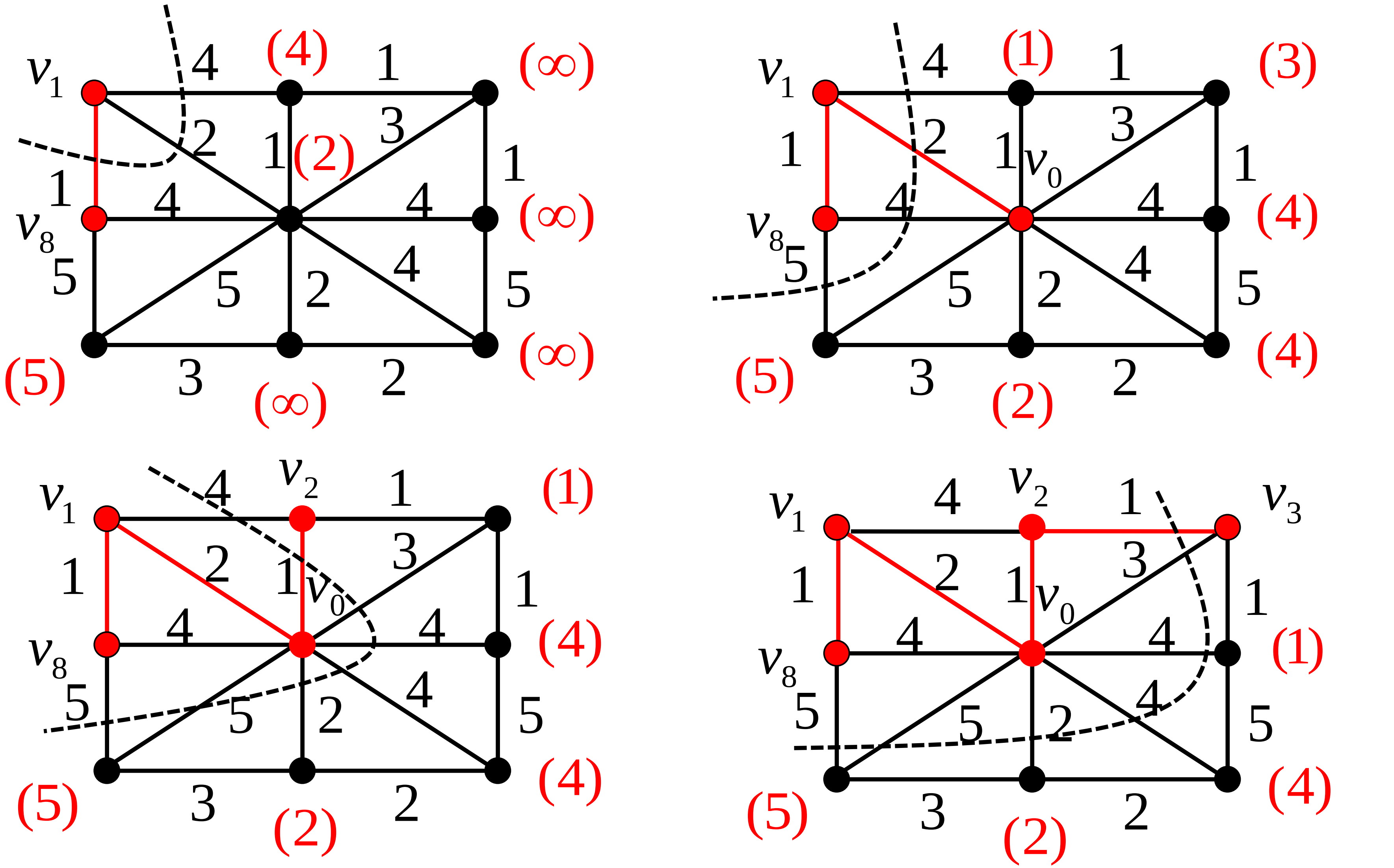
割集 当除去边割后，连通图变为不连通，而除去边割的真子集后，连通图仍然连通





$\{e_1,e_3,e_5,e_6\}$是割集，$\{e_1,e_4,e_9,e_{10}\}$是边割，不是割集

求最小支撑树的 Dijkstra 算法的性质

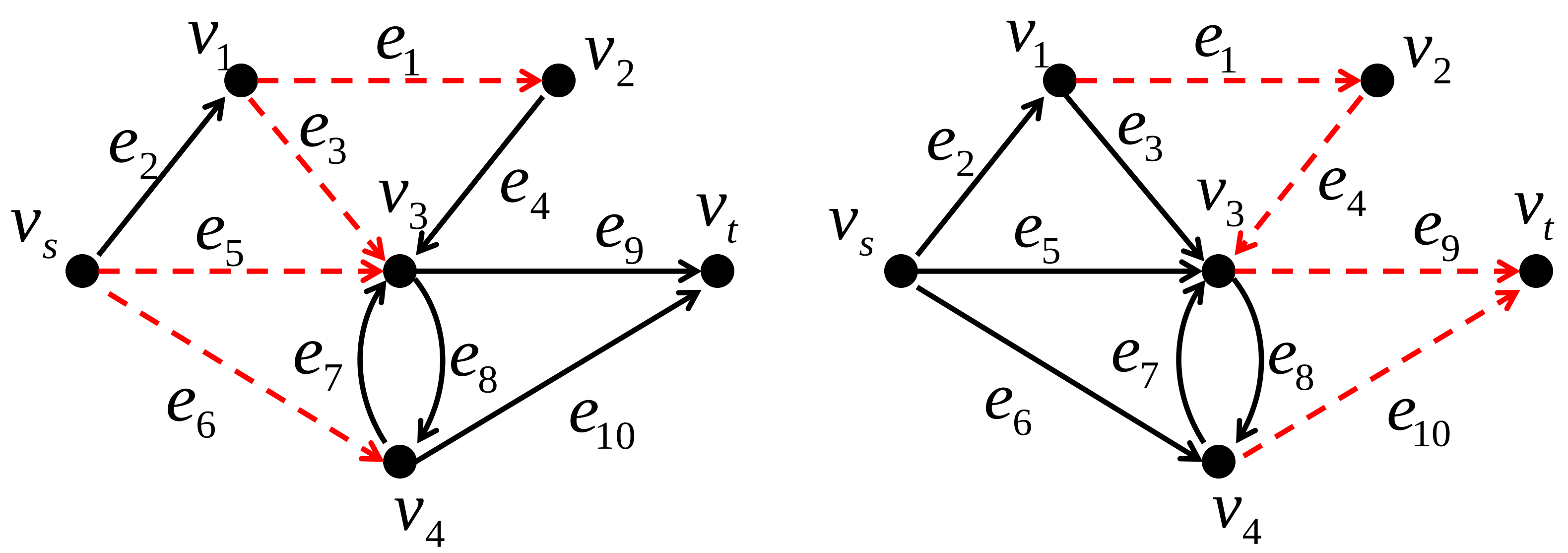


新增边是虚线所指出的割集中的最短的边

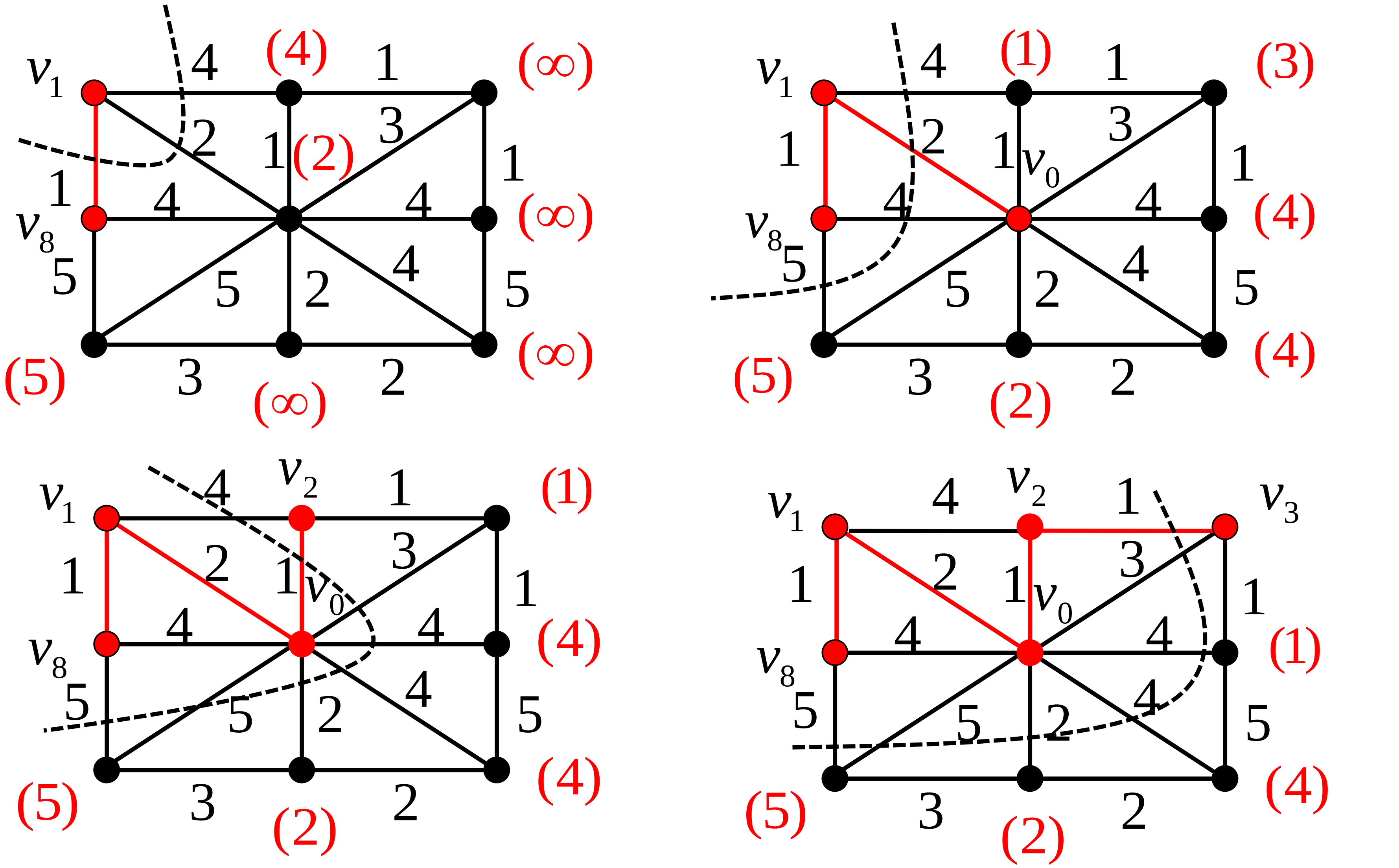
要点：最小支撑树的Dijkstra算法性质

边割 对于图$G=(V,E)$，任取$S\subset V$，其补集为$\bar{S}$，若S和$\bar{S}$都不是空集，称两个端点分属S和$\bar{S}$的边的集合为G的一个边割，记为$\{S,\bar{S}\}$

割集 当除去边割后，连通图变为不连通，而除去边割的真子集后，连通图仍然连通

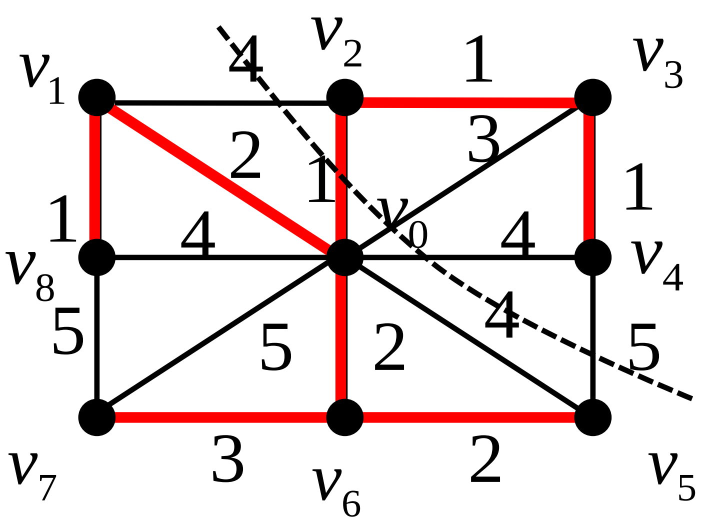


求最小支撑树的 Dijkstra 算法的性质

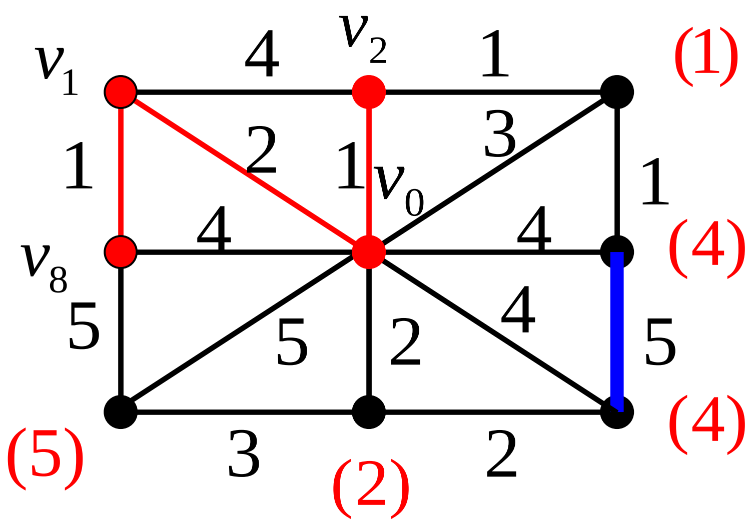


新增边是虚线所指出的割集中的最短的边

如何比较右图 (v_0, v_2) 和 (v_4, v_5) ？



将 v_2 加入 ***S*** 时两者没有比较



不能直接得到“树枝最短”的性质！

利用“弦的权值最大的性质”可以得到上述性质

定理：T 是最小支撑树的充要条件是：任何树枝都是所在的唯一割集中权值最小的边

必要性：如果不是，用权值最小的边代替相应树枝可得总权值更小的支撑树

充分性：加入任何弦形成的回路中，弦和回路上任何树枝都在某个唯一的割集上，所以弦的权值最大

$\Rightarrow$ **Dijkstra算法产生的是最小支撑树**

算法复杂性： **Kruskal** $n^2 \log_2 n \quad > \quad n^2$ **Dijkstra**

要点：不定期最短路问题的Dijkstra算法

不定期最短路问题的 Dijkstra 算法

一般性问题：

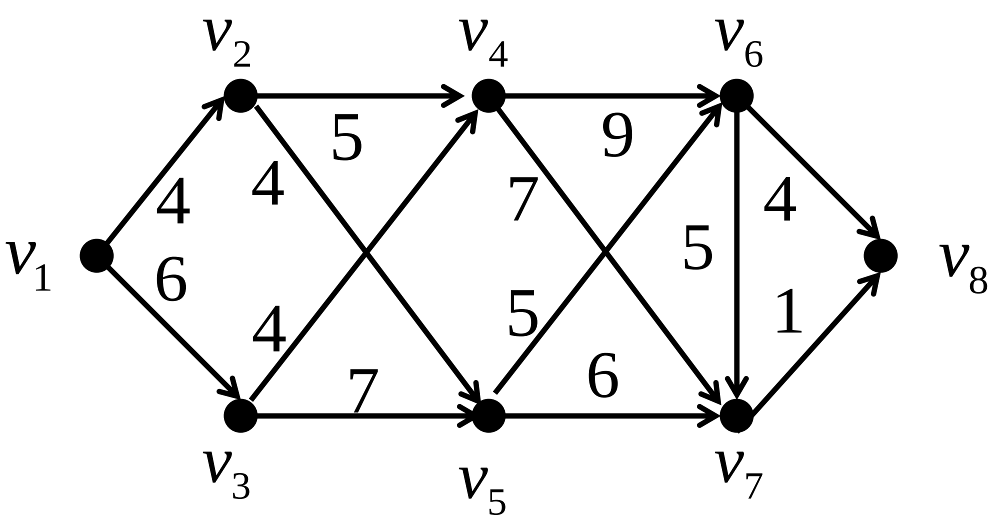
连通图 $G=(V,E)$ 各边 (v_i,v_j) 有权 l_{ij}（$l_{ij}=\infty$ 表示两点间无边），任意给定两点 v_s,v_t，求一条道路 μ，使它是从 v_s 到 v_t 的所有道路中总权最小的道路，即

$$L(\mu_*)=\min_{\mu\in\Omega_{st}} L(\mu) = \sum_{(v_i,v_j)\in\mu} l_{ij}$$

其中 Ω_{st} 表示从 v_s 到 v_t 的所有道路的集合

适用 Dijkstra 算法的问题：所有权值非负

求 v_1 到 v_8 最短路



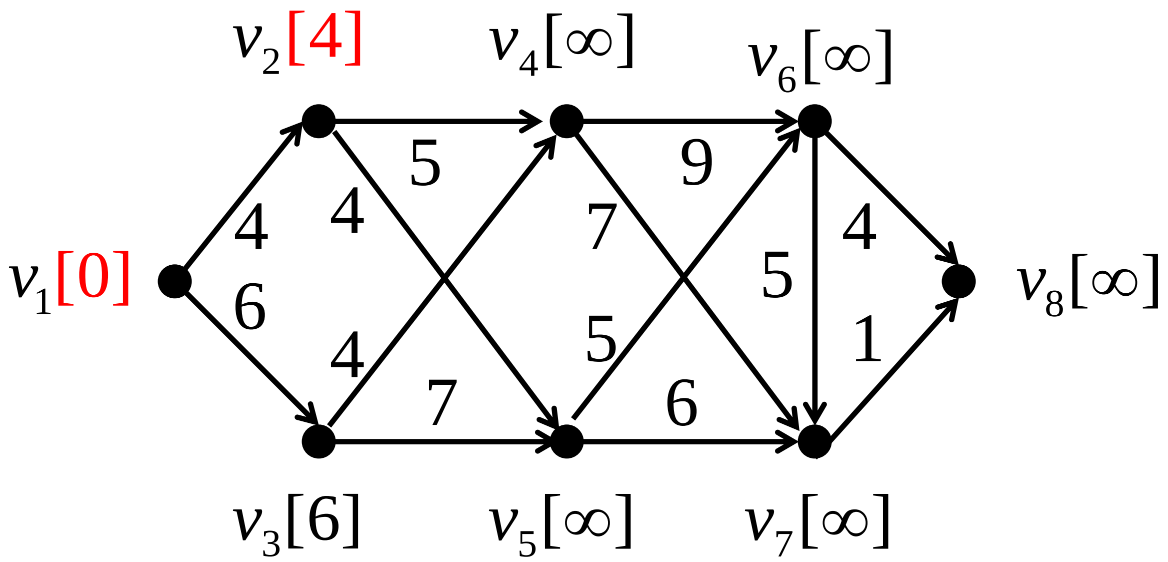
值迭代公式（顺推）

$$f_1\left(v_j\right)=l_{1j}, \ \forall j$$

$$f_{k+1}\left(v_j\right)=\min_i\left\{f_k\left(v_i\right)+l_{ij}\right\}, \ \forall j$$

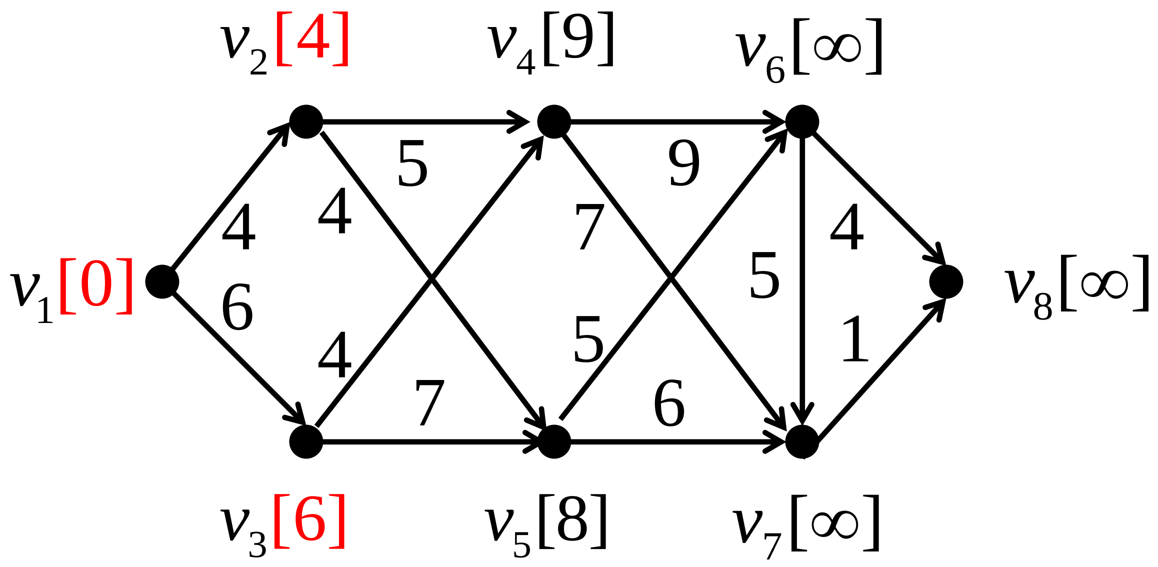
其中 $f_k\left(v_j\right)$ 表示经过 $k-1$ 个中间点到达 v_j 的最短路

$k = 1$



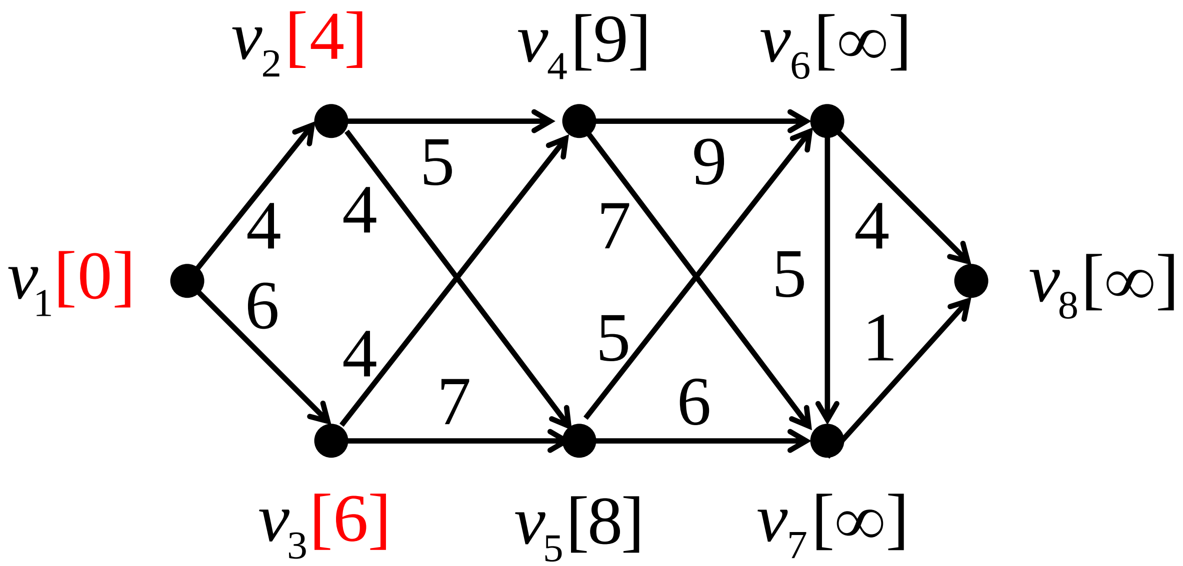
$v_1[0]$
$v_2[4]$
$v_3[6]$
$v_4[\infty]$
$v_5[\infty]$
$v_6[\infty]$
$v_7[\infty]$
$v_8[\infty]$
4 6 5 4 4 7 9 7 5 6 5 4 1

$k = 2$



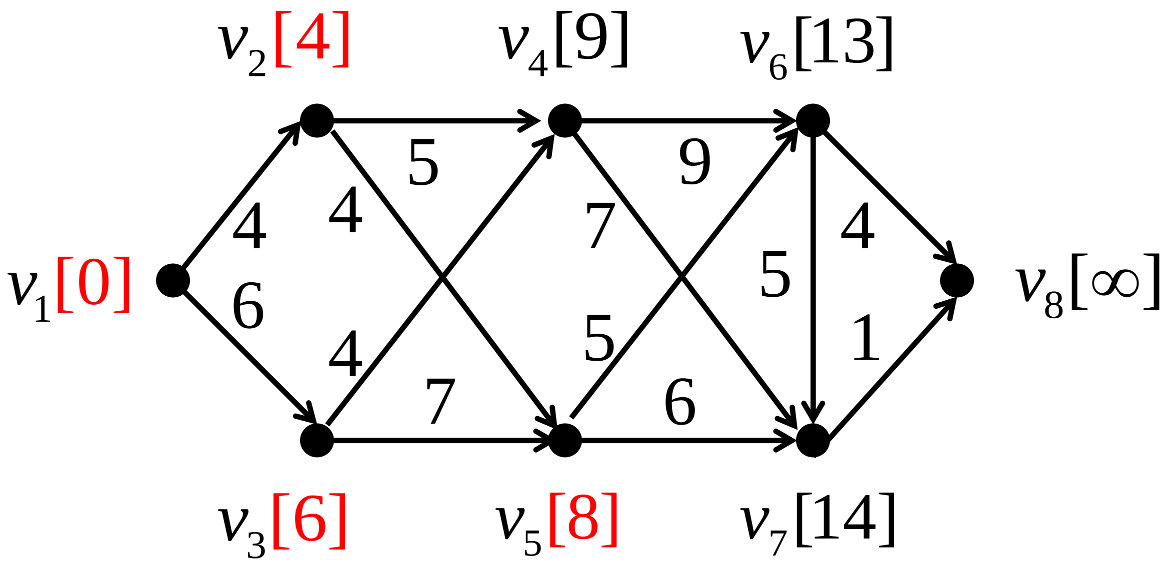
$v_1[0]$
$v_2[4]$
$v_3[6]$
$v_4[9]$
$v_5[8]$
$v_6[\infty]$
$v_7[\infty]$
$v_8[\infty]$
4 6 5 4 4 7 9 7 5 6 5 4 1

$k=2$



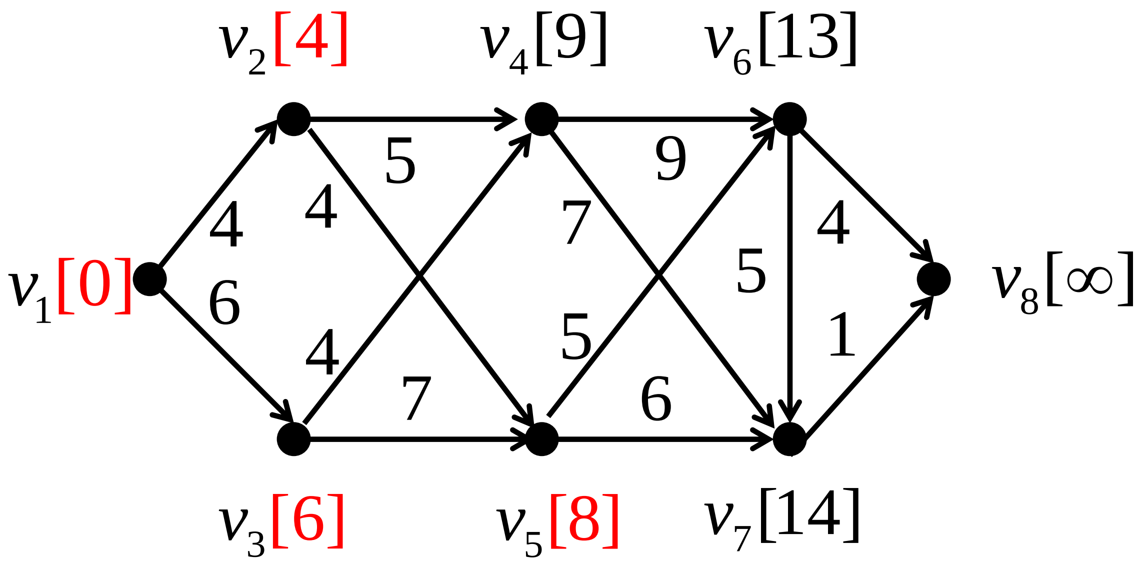
$v_2[4]$
$v_4[9]$
$v_6[\infty]$
$v_1[0]$
$v_8[\infty]$
$v_3[6]$
$v_5[8]$
$v_7[\infty]$
5
9
4
4
7
5
4
6
1
4
5
7
6

$k=3$



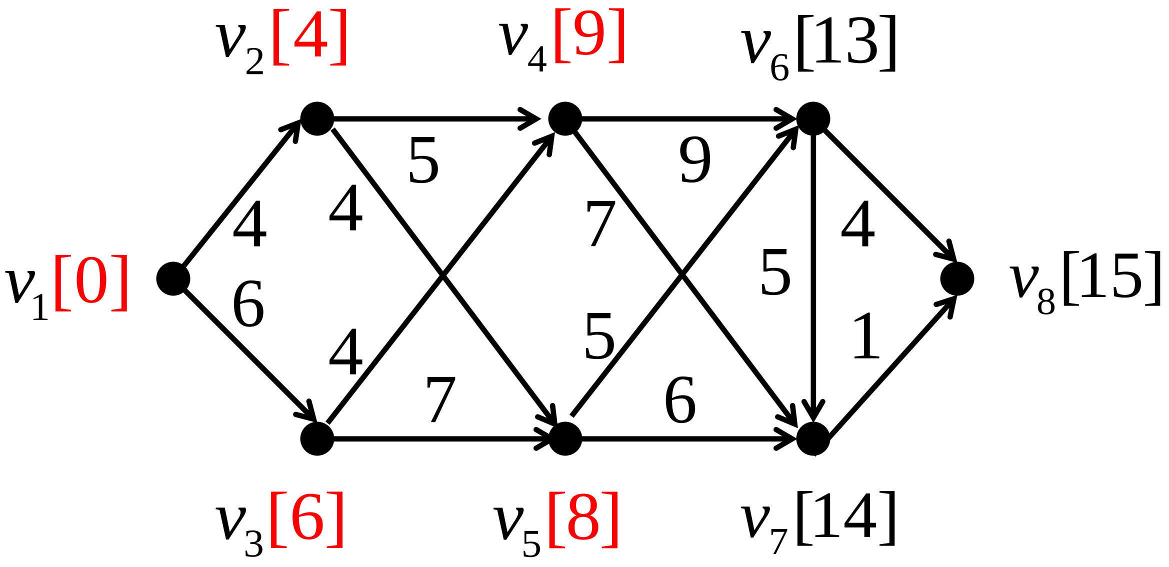
$v_2[4]$
$v_4[9]$
$v_6[13]$
$v_1[0]$
$v_8[\infty]$
$v_3[6]$
$v_5[8]$
$v_7[14]$
5
9
4
4
7
5
4
6
1
4
5
7
6

$k=3$



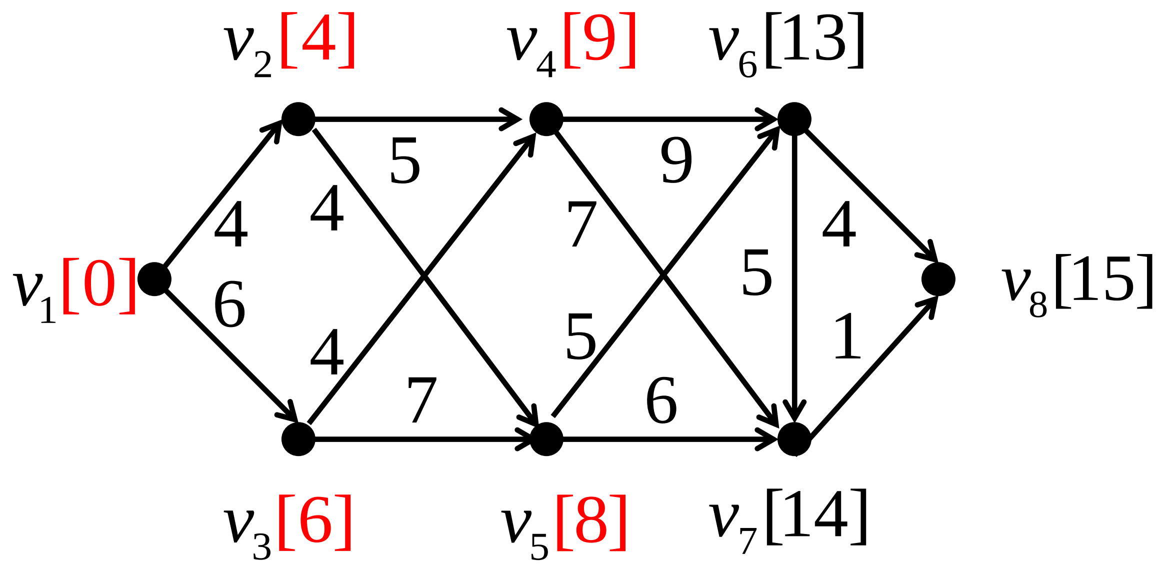
v2[4]
v4[9]
v6[13]
v1[0]
v8[∞]
v3[6]
v5[8]
v7[14]
4
4
6
5
4
7
7
5
9
6
5
4
1

$k=4$



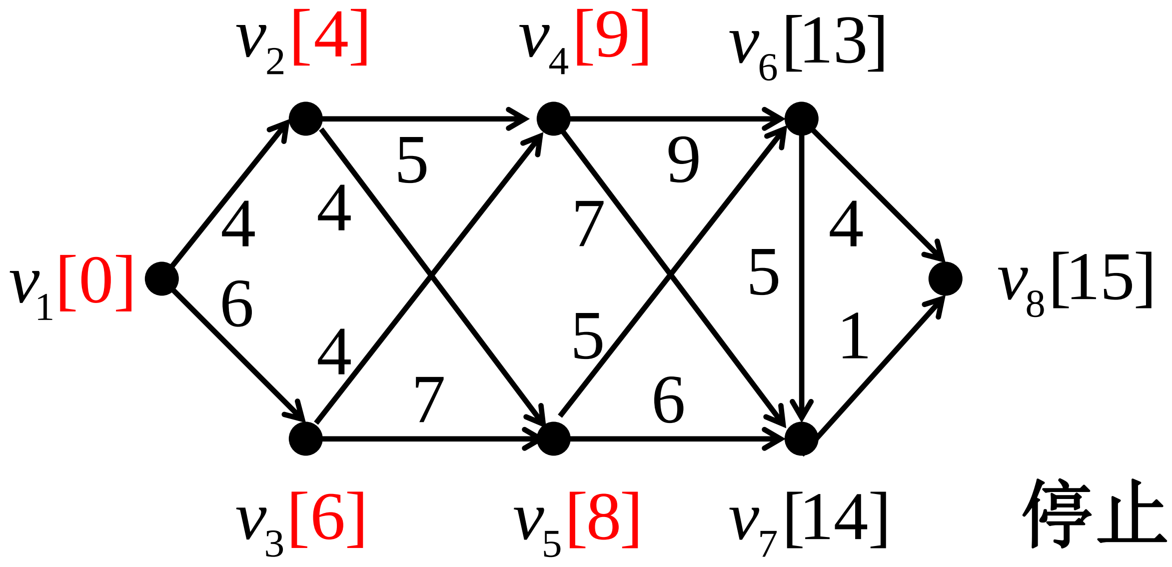
v2[4]
v4[9]
v6[13]
v1[0]
v8[15]
v3[6]
v5[8]
v7[14]
4
4
6
5
4
7
7
5
9
6
5
4
1

$k=4$



$v_2[4]$
$v_4[9]$
$v_6[13]$
$v_1[0]$
$v_8[15]$
$v_3[6]$
$v_5[8]$
$v_7[14]$
5
9
4
4
7
4
5
6
4
5
1
7
6

$k=5$



$v_2[4]$
$v_4[9]$
$v_6[13]$
$v_1[0]$
$v_8[15]$
$v_3[6]$
$v_5[8]$
$v_7[14]$
5
9
4
4
7
4
5
6
4
5
1
7
6

停止

用值迭代法求非负权值不定期最短路问题的特点：

每步迭代后有一个新的最优函数值不再发生变化，这个新最优函数值是未固定的函数值中数值最小的

理由： $f_{k+1}\left(v_j\right)=\min_i\left\{f_k\left(v_i\right)+l_{ij}\right\}$

$$\Rightarrow \quad \min_j f_k\left(v_j\right)\le \min_j f_{k+1}\left(v_j\right)$$

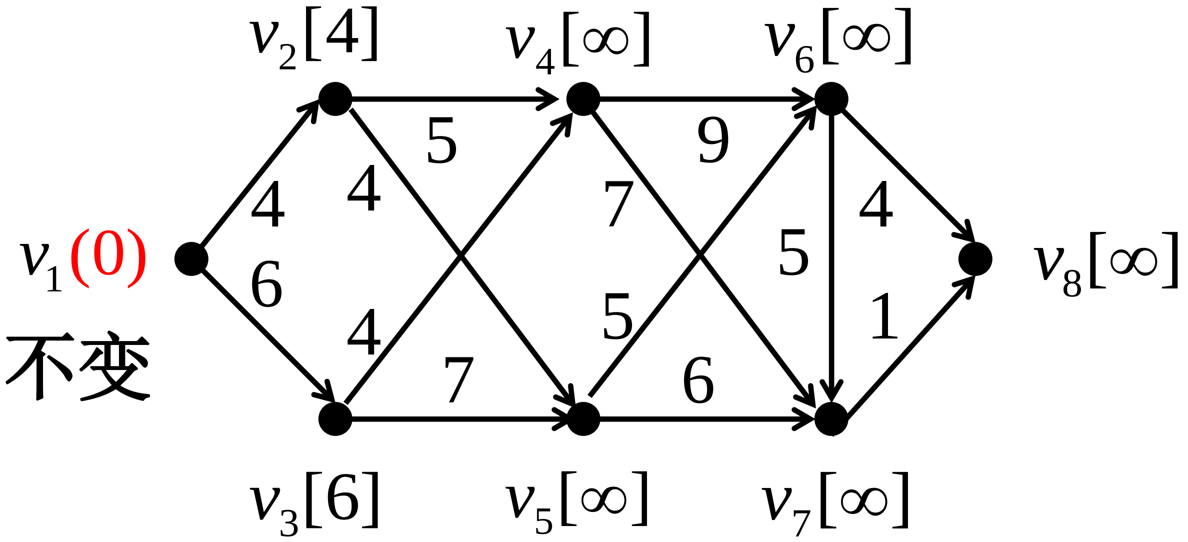
$\Rightarrow$ 达到最小值的函数值不会改变

利用上述特点的做法（ Dijkstra 算法）：

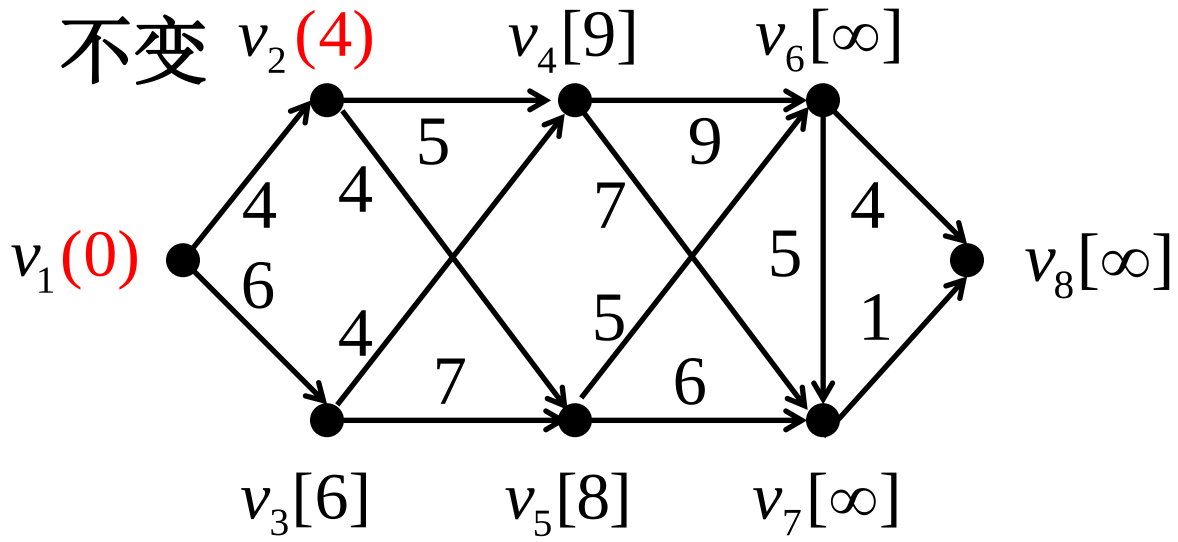
每次确定一个不变的函数值，同时仅修改经过新确定的不变节点到其他节点的路程

用 Dijkstra 算法求下面的最短路问题

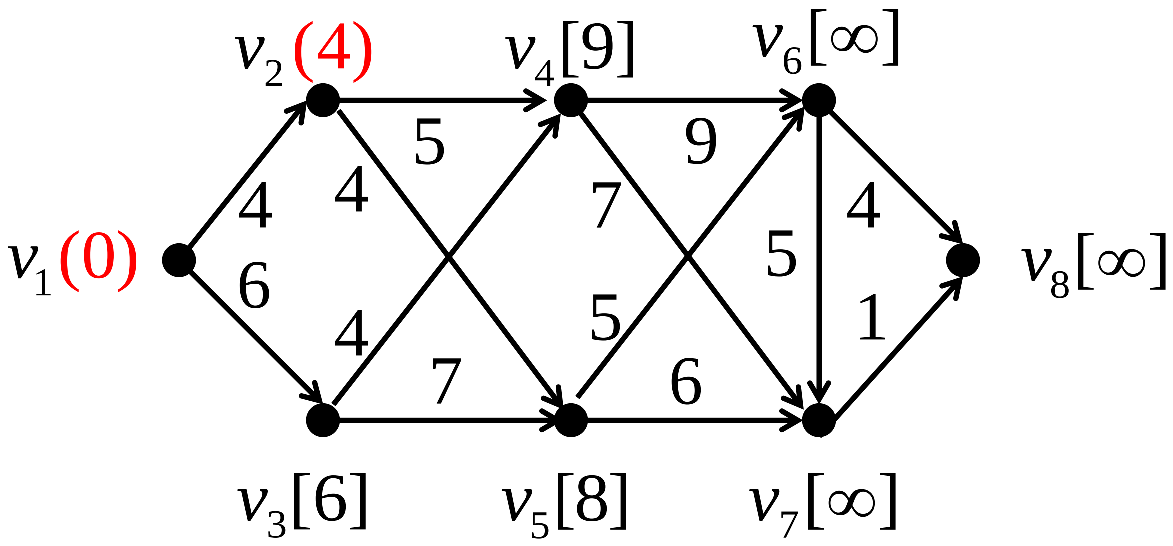
$k=0$



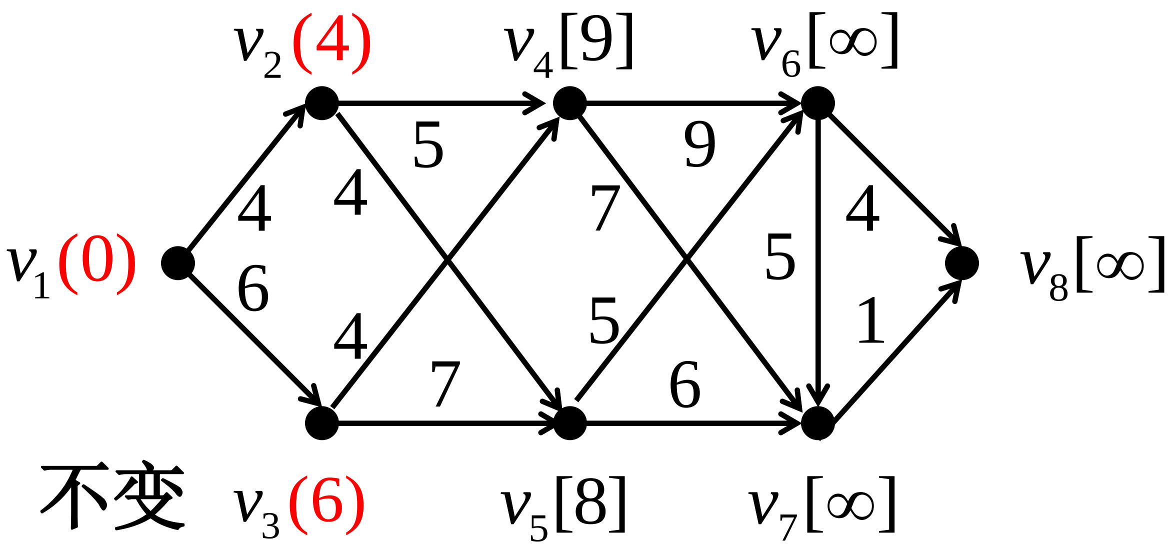
$k=1$



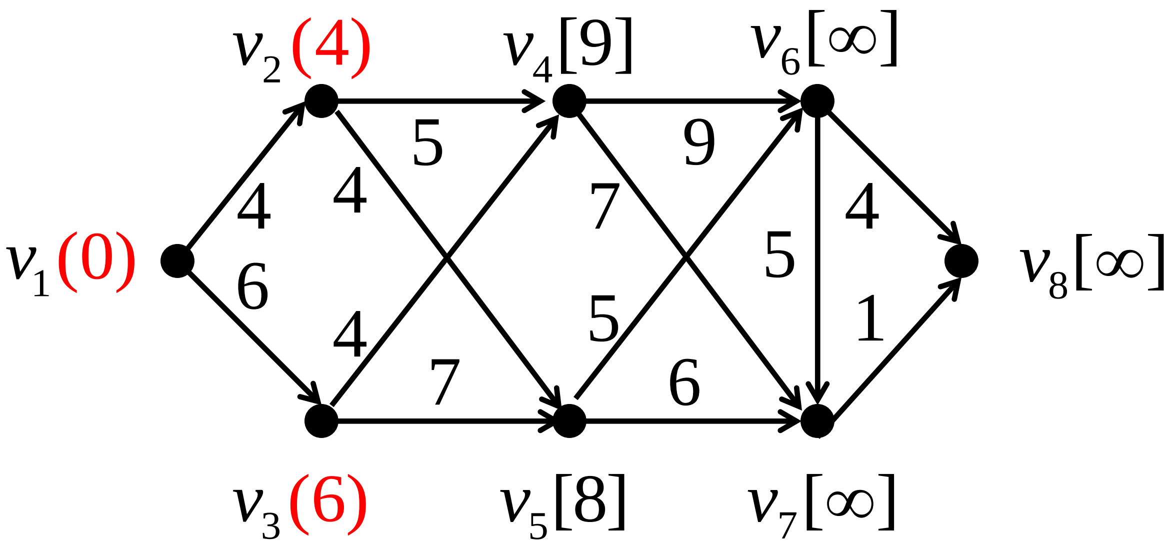
$k=1$


$v_1(0)$
$v_2(4)$
$v_3[6]$
$v_4[9]$
$v_5[8]$
$v_6[\infty]$
$v_7[\infty]$
$v_8[\infty]$

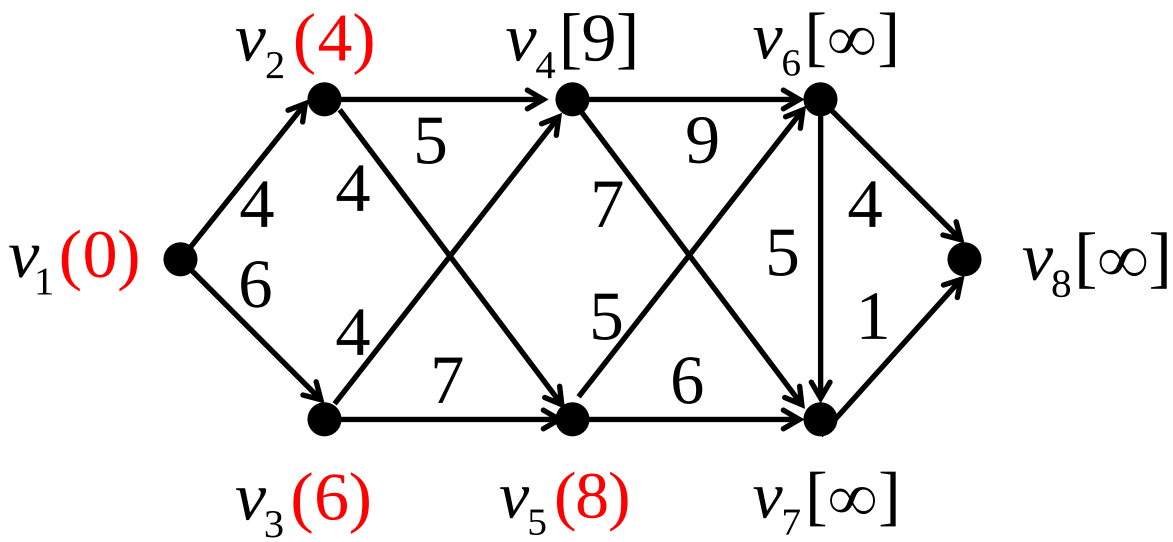
$k=2$


$v_1(0)$
$v_2(4)$
$v_3(6)$
不变
$v_4[9]$
$v_5[8]$
$v_6[\infty]$
$v_7[\infty]$
$v_8[\infty]$

$k=2$

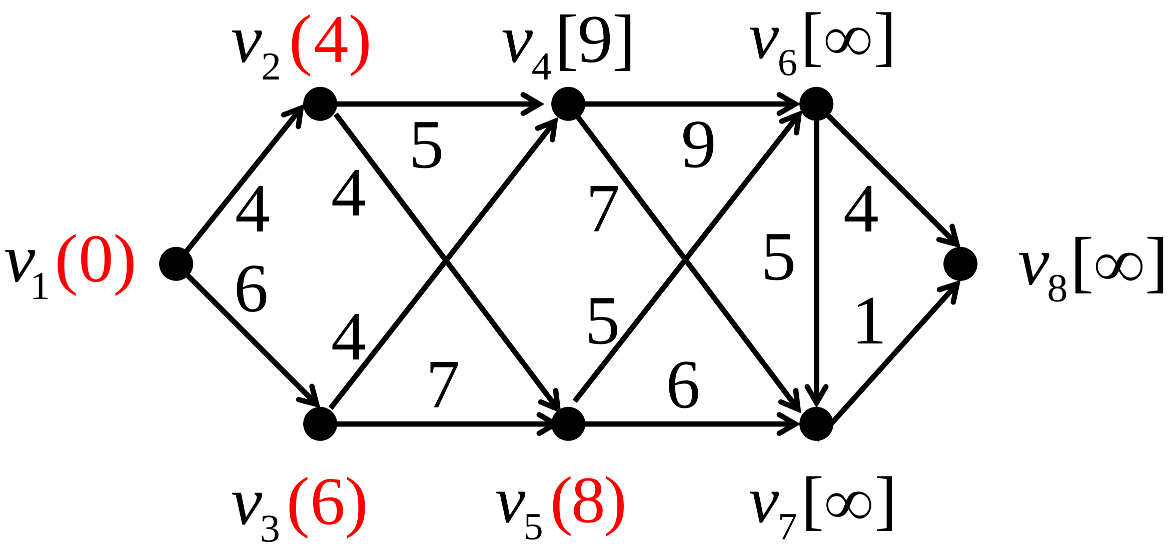

$v_1(0)$
$v_2(4)$
$v_3(6)$
$v_4[9]$
$v_5[8]$
$v_6[\infty]$
$v_7[\infty]$
$v_8[\infty]$
5
9
4
4
7
5
4
6
4
5
1
7
6

$k=3$


$v_1(0)$
$v_2(4)$
$v_3(6)$
$v_4[9]$
$v_5(8)$
$v_6[\infty]$
$v_7[\infty]$
$v_8[\infty]$
5
9
4
4
7
5
4
6
4
5
1
7
6

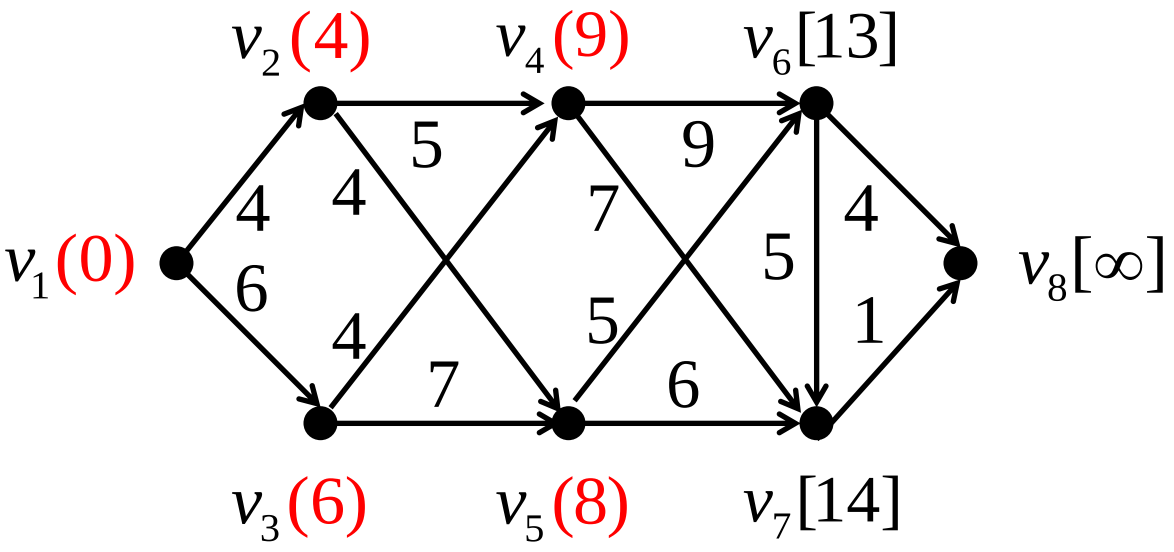
不变

$k=3$

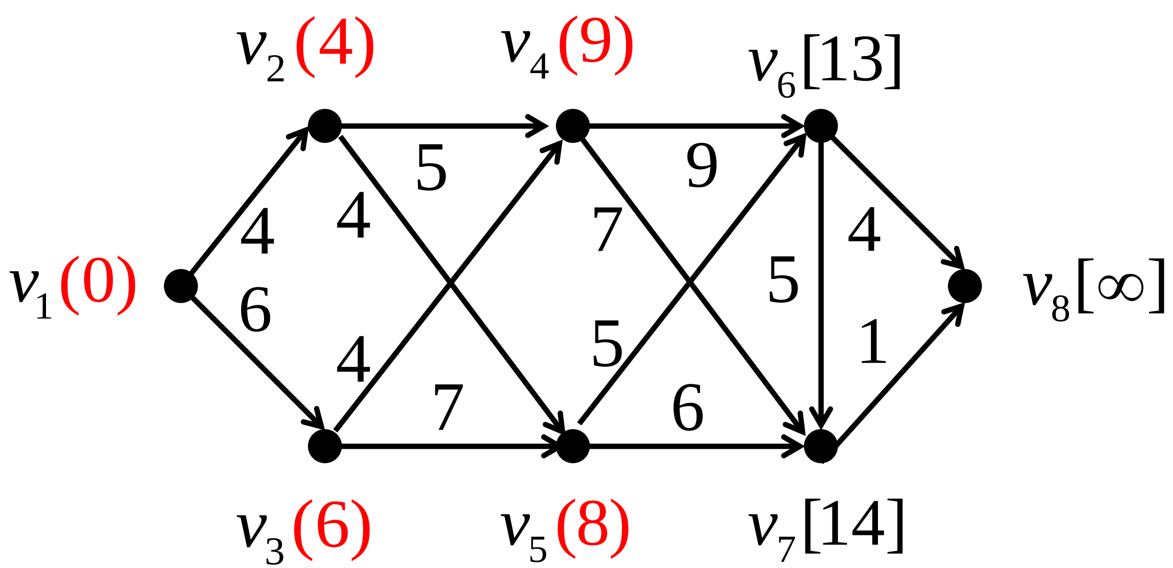

$v_2(4)$
$v_4[9]$
$v_6[\infty]$
$v_1(0)$
$v_8[\infty]$
$v_3(6)$
$v_5(8)$
$v_7[\infty]$
5
9
4
4
7
4
5
6
4
5
1
7
6

不变

$k=4$

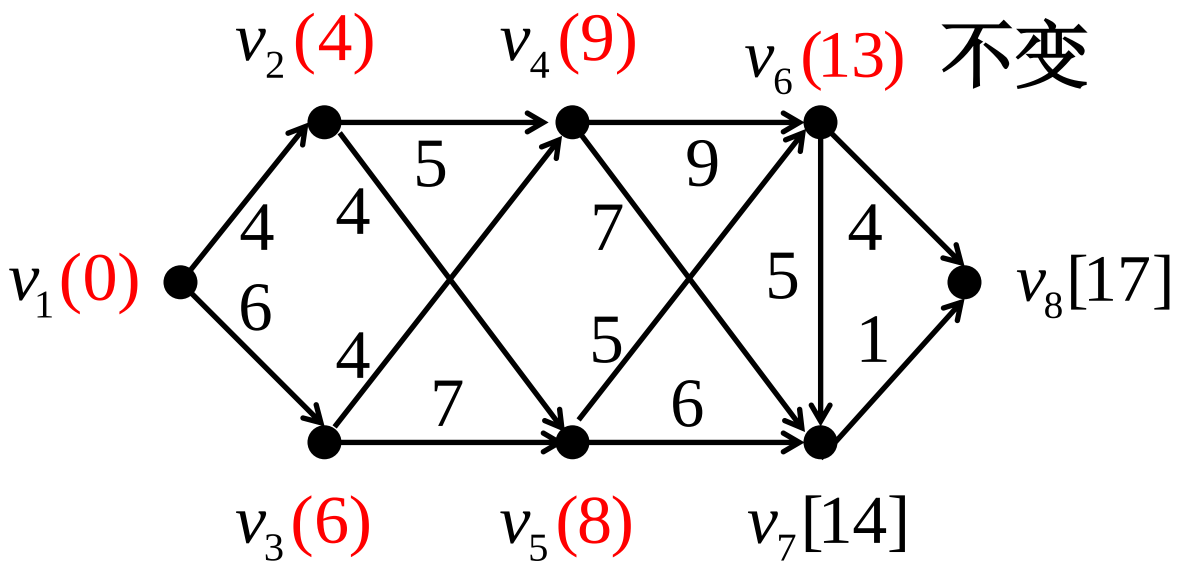

$v_2(4)$
$v_4(9)$
$v_6[13]$
$v_1(0)$
$v_8[\infty]$
$v_3(6)$
$v_5(8)$
$v_7[14]$
5
9
4
4
7
4
5
6
4
5
1
7
6

$k=4$



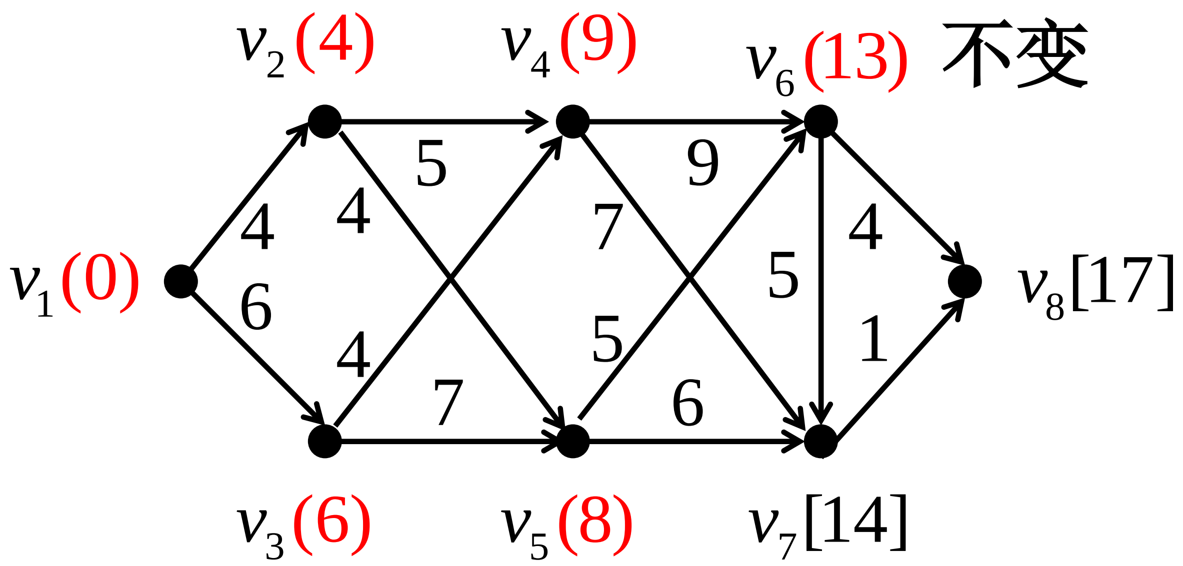
$v_1(0)$
$v_2(4)$
$v_3(6)$
$v_4(9)$
$v_5(8)$
$v_6[13]$
$v_7[14]$
$v_8[\infty]$
4
6
5
4
4
7
9
7
5
6
5
4
1

$k=5$

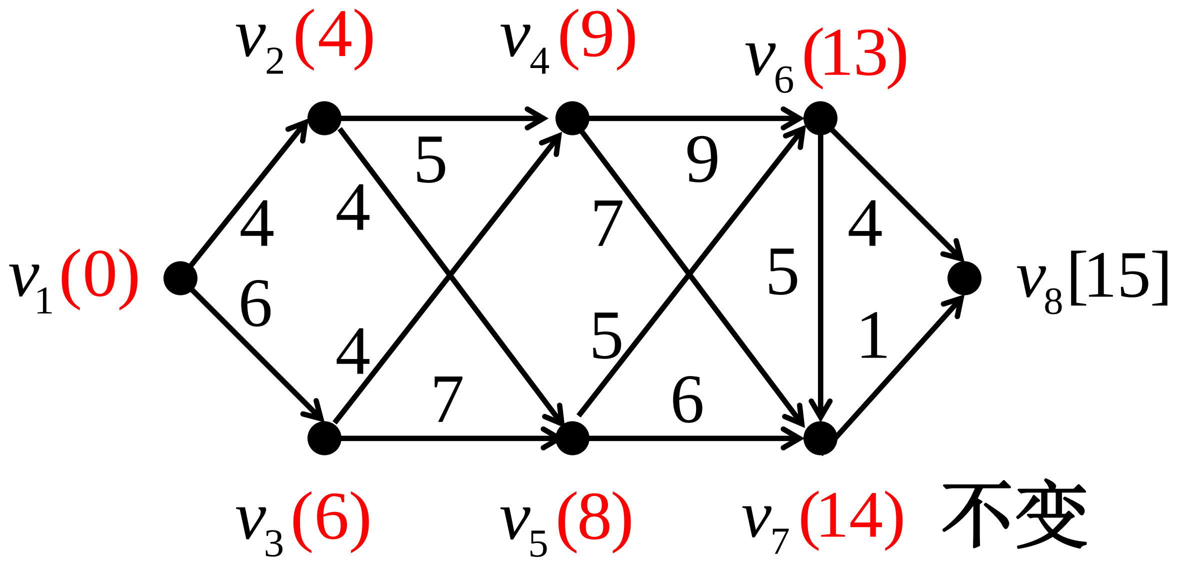


$v_1(0)$
$v_2(4)$
$v_3(6)$
$v_4(9)$
$v_5(8)$
$v_6(13)$
不变
$v_7[14]$
$v_8[17]$
4
6
5
4
4
7
9
7
5
6
5
4
1

$k=5$

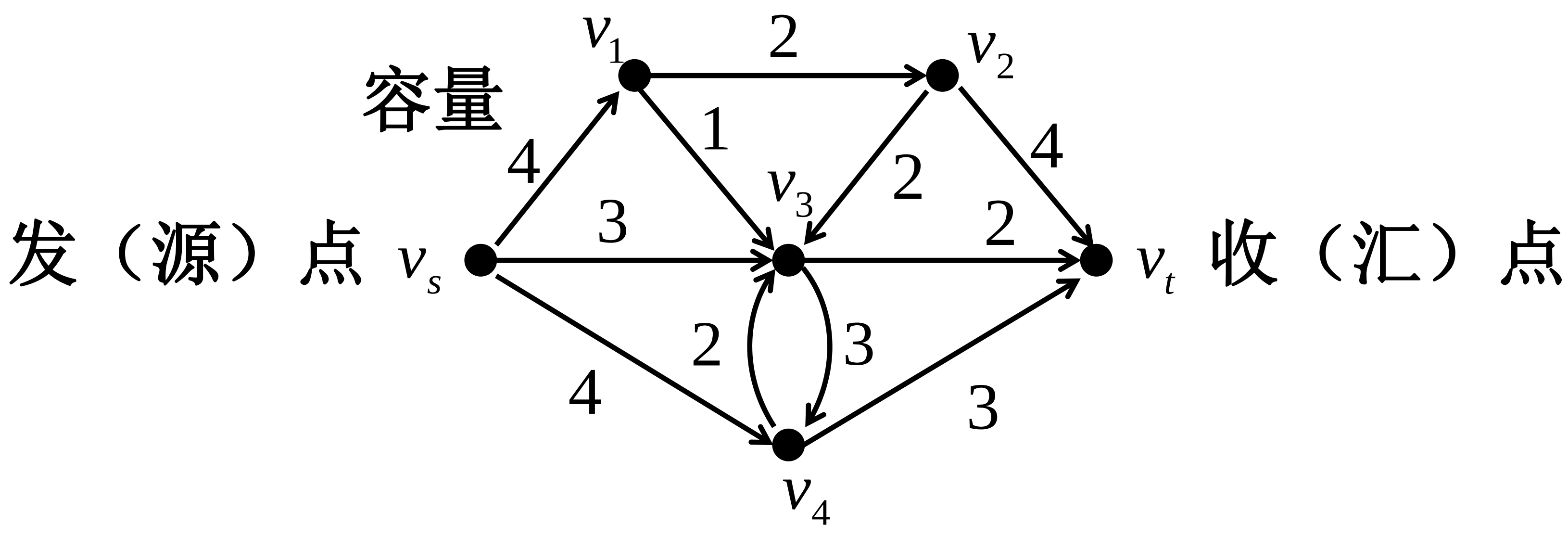

$v_2(4)$
$v_4(9)$
$v_6(13)$ 不变
$v_1(0)$
$v_8[17]$
$v_3(6)$
$v_5(8)$
$v_7[14]$
5
9
4
4
7
4
5
1
6
4
5
7
6

$k=6$


$v_2(4)$
$v_4(9)$
$v_6(13)$
$v_1(0)$
$v_8[15]$
$v_3(6)$
$v_5(8)$
$v_7(14)$ 不变
5
9
4
4
7
4
5
1
6
4
5
7
6

要点：最大流问题描述

例　最大输油量问题



目标：从发点到收点的总输油量最大

约束：1）容量约束，各边流量不大于容量

　　　2）流量平衡约束，各点进出流量总和相等

容量网络

有向连通图 $G=(V,E)$ 各边 (v_i,v_j) 有非负容量 c_{ij}，仅有一个入次为 0 的点 v_s，称为发（源）点，一个出次为 0 的点 v_t，称为收（汇）点，常记为 $G=(V,E,C)$，其中 $C=\{c_{ij}\}$

可行流 满足流量平衡约束和容量约束的 $X=\{x_{ij}\}$

$$\sum_{(v_i,v_j)\in E} x_{ij} = \sum_{(v_k,v_i)\in E} x_{ki}, \forall v_i \in V,\ i\neq s,t \text{（流量平衡约束）}$$

$$0\le x_{ij}\le c_{ij}, \forall (v_i,v_j)\in E \quad \text{（容量约束）}$$

可行流的网络总流量 $W=\sum_{(v_s,v_j)\in E} x_{sj} = \sum_{(v_k,v_t)\in E} x_{kt}$

最大流问题

确定使网络总流量达到最大的可行流

数学规划模型

$$\max \quad W$$

$$\text{s.t.} \quad \sum_{(v_i,v_j)\in E} x_{ij} - \sum_{(v_j,v_i)\in E} x_{ji} = \begin{cases} W & \text{if} \quad i = s \\ 0 & \text{if} \quad i \neq st \\ -W & \text{if} \quad i = t \end{cases}$$

$$0 \leq x_{ij} \leq c_{ij}, \forall (v_i, v_j) \in E$$

最大流问题是一种特殊的线性规划问题，存在有效的网络优化算法

要点：最大流问题的矩阵表示

用矩阵表示图

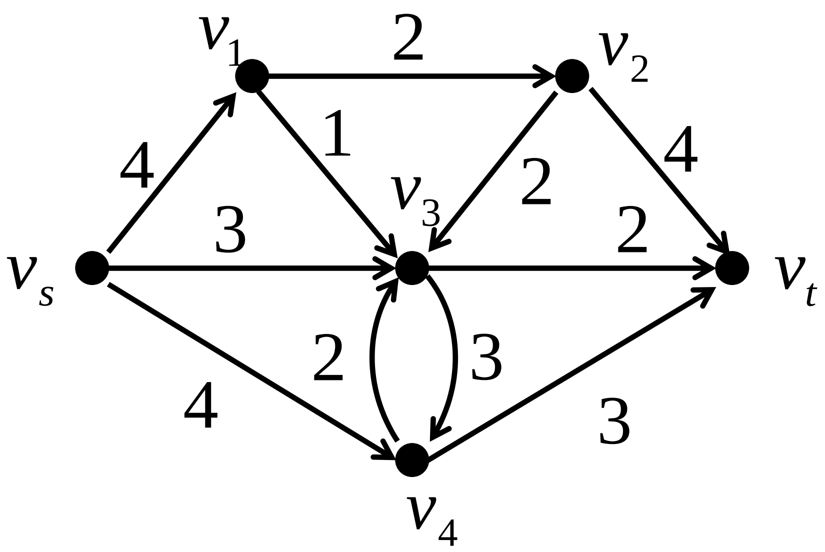
$$\max \quad W$$

$$\text{s.t.} \quad \sum_{(v_i,v_j)\in E} x_{ij} - \sum_{(v_j,v_i)\in E} x_{ji} = \begin{cases} W & \text{if} \quad i = s \\ 0 & \text{if} \quad i \neq st \\ -W & \text{if} \quad i = t \end{cases}$$

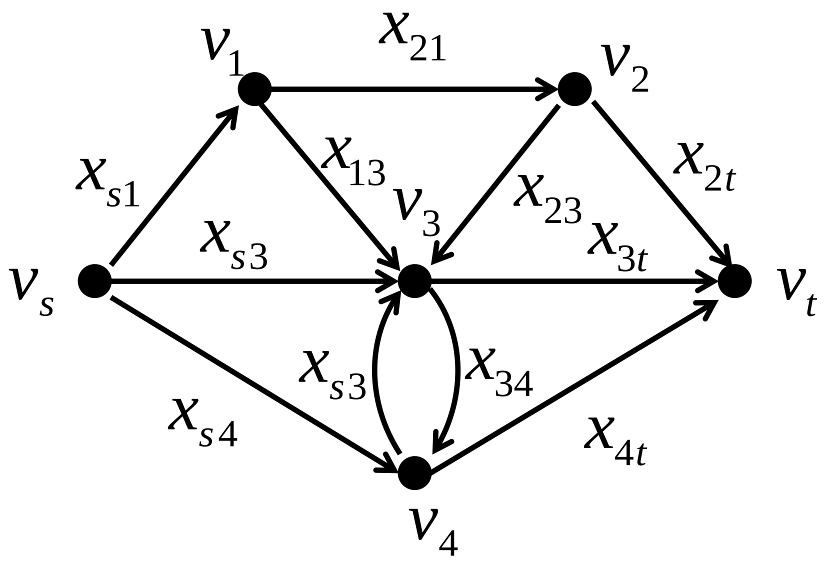
$$0 \le x_{ij} \le c_{ij}, \forall (v_i, v_j) \in E$$

$$\text{s.t.} \quad AX = B$$

$$0 \le X \le C$$



可利用有向图的关联矩阵表示流量平衡约束



$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & -1 & -1 & 1 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & -1 & 1 \\ 0 & 0 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -1 \end{pmatrix} X = \begin{pmatrix} W \\ 0 \\ 0 \\ 0 \\ 0 \\ -W \end{pmatrix}$$

要点：最大流问题的割集

最大流问题的割集

$\{S,\bar{S}\}$ 是 $G=(V,E)$ 的割集，且满足 $v_s\in S$，$v_t\in\bar{S}$

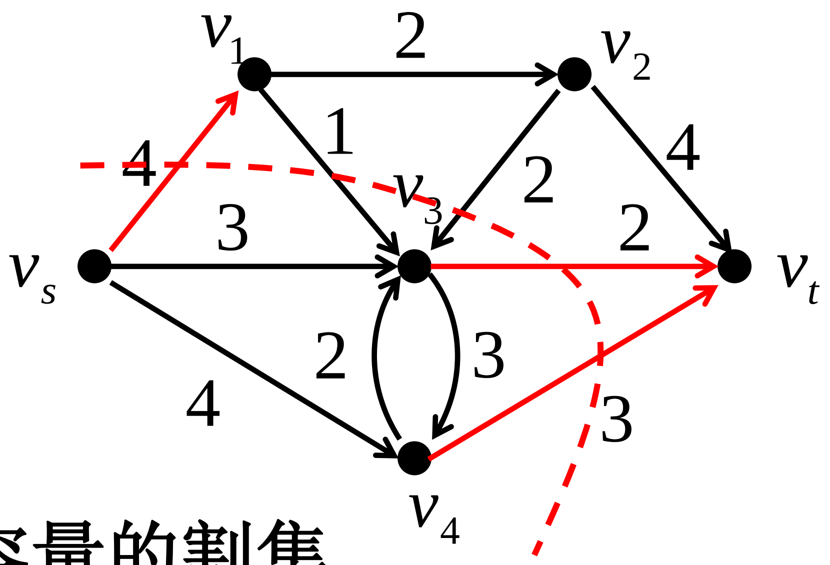
割集容量

割集 $\{S,\bar{S}\}$ 中，所有始点属于 S 、终点属于 $\bar{S}$ 的边的容量和称为 $\{S,\bar{S}\}$ 的割集容量，记为 $C(S,\bar{S})$

例如

$$S=\{v_s\}\Rightarrow C(S,\bar{S})=11$$

$$S=\{v_s,v_3,v_4\}\Rightarrow C(S,\bar{S})=9$$



最小割集容量　具有最小容量的割集

要点：网络的最大流等于最小割集容量

对于任意的可行流 $X=\{x_{ij}\}$ 和割集 $\{S,\bar{S}\}$ $v_s\in S$，$v_t\in\bar{S}$

$$\sum_{(v_i,v_j)\in E} x_{ij}-\sum_{(v_j,v_i)\in E} x_{ji}=\begin{cases} W & \text{if} \quad i=s \\ 0 & \text{if} \quad i\neq st \\ -W & \text{if} \quad i=t \end{cases} \quad 0\le x_{ij}\le c_{ij},\forall(v_i,v_j)\in E$$

$$W=\sum_{v_i\in S}\left(\sum_{(v_i,v_j)\in E} x_{ij}-\sum_{(v_j,v_i)\in E} x_{ji}\right)=\sum_{v_i\in S}\left(\sum_{\substack{v_j\in S\\(v_i,v_j)\in E}} x_{ij}-\sum_{\substack{v_j\in S\\(v_j,v_i)\in E}} x_{ji}\right)$$

$$\Rightarrow \quad +\sum_{v_i\in S}\left(\sum_{\substack{v_j\in \bar{S}\\(v_i,v_j)\in E}} x_{ij}-\sum_{\substack{v_j\in \bar{S}\\(v_j,v_i)\in E}} x_{ji}\right)=\sum_{v_i\in S}\left(\sum_{\substack{v_j\in \bar{S}\\(v_i,v_j)\in E}} x_{ij}-\sum_{\substack{v_j\in \bar{S}\\(v_j,v_i)\in E}} x_{ji}\right)$$

$$\le\sum_{v_i\in S}\sum_{\substack{v_j\in \bar{S}\\(v_i,v_j)\in E}} x_{ij}=C(S,\bar{S})$$

等于割集容量的可行流一定是最大流

要点：可增广链概念

可增广链

设 μ 是从 v_s 到 v_t 的一条链，定义 μ 的方向为从 v_s 到 v_t 的方向，对于 μ 上的任意边，如果其方向和 μ 相同则称其为**前向边**，否则为**后向边**，用 μ^+ 和 μ^- 分别表示前向边和后向边的集合，如果 $X=\{x_{ij}\}$ 是一个可行流，且满足

$$0 \le x_{ij} < c_{ij} \quad \forall (v_i, v_j) \in \mu^+$$
$$0 < x_{ij} \le c_{ij} \quad \forall (v_i, v_j) \in \mu^-$$

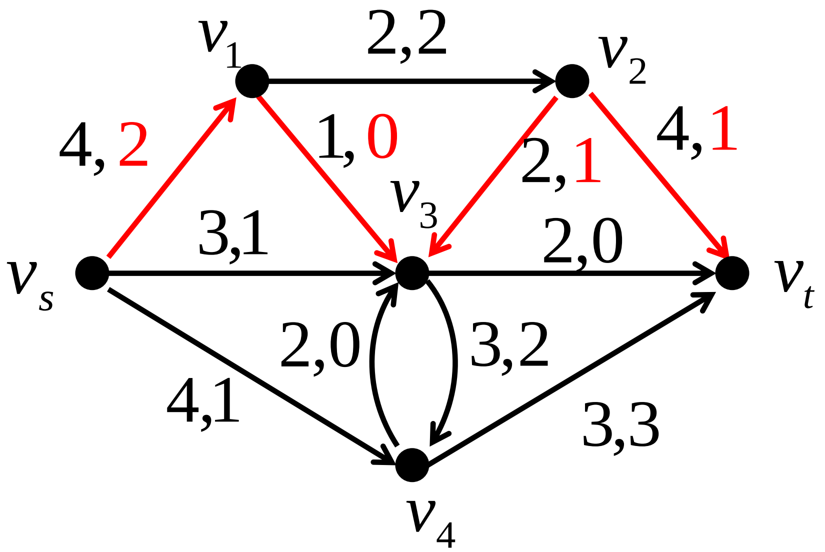
则称 μ 是从 v_s 到 v_t（关于 X ）的**可增广链**

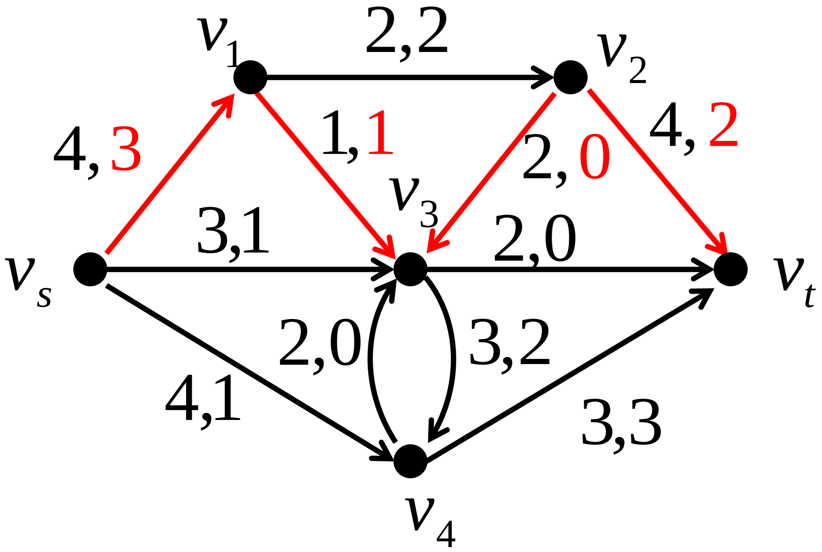
要点：通过可增广链求出最大流

例　下图中每对数字第一个是容量，第二个是一个可行流的流量，下面左图红线是可增广链，右图是对可增广链的流量进行如下调整得到新的可行流

前向边流量加 1，后向边流量减 1

新可行流比原可行流的总流量也加 1





对可增广链流量的一般改进方法

已知条件

$$0 \le x_{ij} < c_{ij} \quad \forall (v_i, v_j) \in \mu^+$$
$$0 < x_{ij} \le c_{ij} \quad \forall (v_i, v_j) \in \mu^-$$

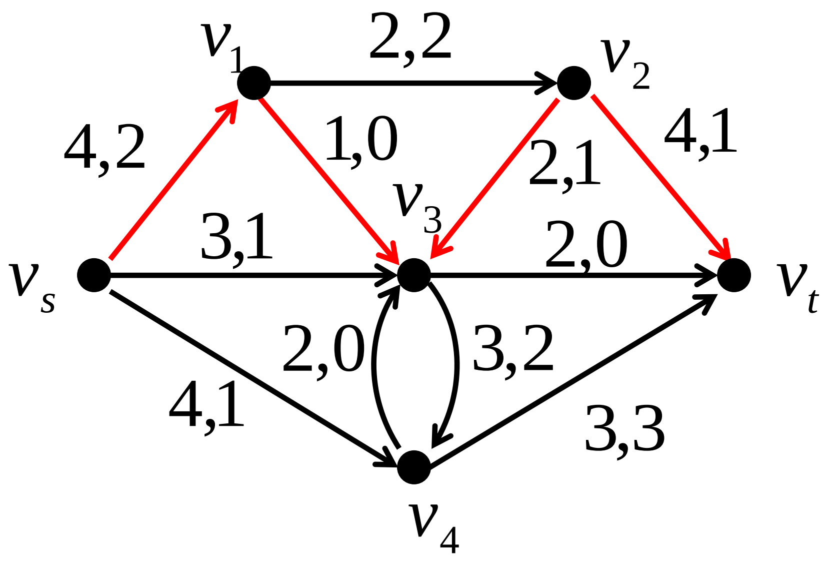
令

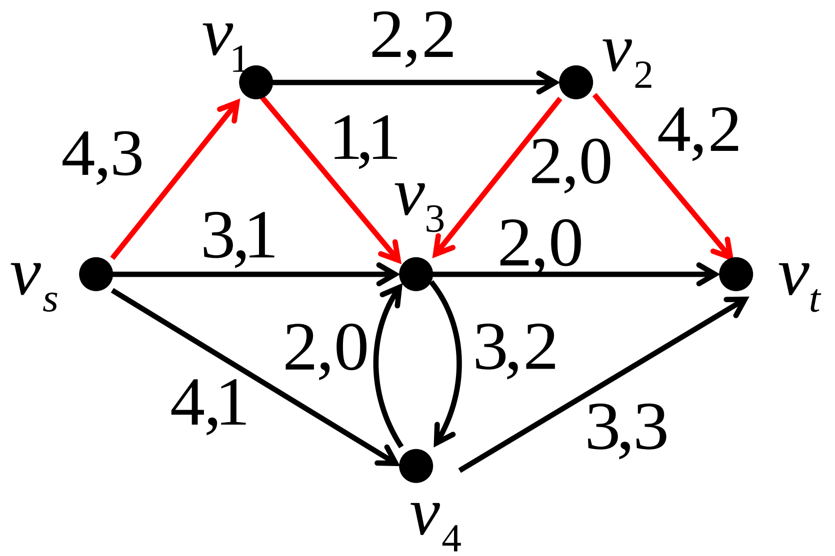
$$\delta_{ij} = c_{ij} - x_{ij} \quad \forall (v_i, v_j) \in \mu^+$$
$$\delta_{ij} = x_{ij} - 0 \quad \forall (v_i, v_j) \in \mu^-$$

$$\Rightarrow \quad \delta = \min_{i,j} \delta_{ij}$$

$$\Rightarrow \quad \hat{x}_{ij} = x_{ij} + \delta \quad \forall (v_i, v_j) \in \mu^+$$
$$\hat{x}_{ij} = x_{ij} - \delta \quad \forall (v_i, v_j) \in \mu^-$$

$$\hat{x}_{ij} = x_{ij}, \forall (v_i, v_j) \notin \mu$$





要点：最大流最小割定理

定理（增广链定理 ） 一个可行流是最大流当且仅当不存在关于它的可增广链

必要性显然成立，下面证明充分性

定义 S 为下述顶点的集合：对任意的 $v_k \in S$ 存在从 v_s 到 v_k 的链，满足可增广链的两个条件，即

如果 (v_i, v_j) 与链的方向相同，则 $0 \le x_{ij} < c_{ij}$

如果 (v_i, v_j) 与链的方向相反，则 $0 < x_{ij} \le c_{ij}$

用 $\bar{S}$ 表示 S 的补集，不存在可增广链 $\Rightarrow$ $v_t \in \bar{S}$

由 S 的定义可知，对于任意的 $v_i \in S, v_j \in \bar{S}$

如果 $(v_i, v_j) \in E$，一定有 $x_{ij} = c_{ij}$，否则 $v_j \in S$

如果 $(v_j, v_i) \in E$，一定有 $x_{ji} = 0$，否则 $v_j \in S$

由以上关系可得

$$W = \sum_{v_i \in S} \sum_{\substack{v_j \in \bar{S} \\ (v_i, v_j) \in E}} \left(x_{ij} - x_{ji}\right) = \sum_{v_i \in S} \sum_{\substack{v_j \in \bar{S} \\ (v_i, v_j) \in E}} c_{ij} = C\left(S, \bar{S}\right)$$

由于任何可行流的流量 $\hat{W}$ 都满足

$$\hat{W} \le C\left(S, \bar{S}\right)$$

所以 X 是最大流

由增广链定理的证明过程可得以下定理

定理　（最大流一最小割定理）

对于任何容量网络 $G=(V,E,C)$，从 v_s 到 v_t 的最大流的流量等于分割 v_s 和 v_t 的最小割集的容量

理由

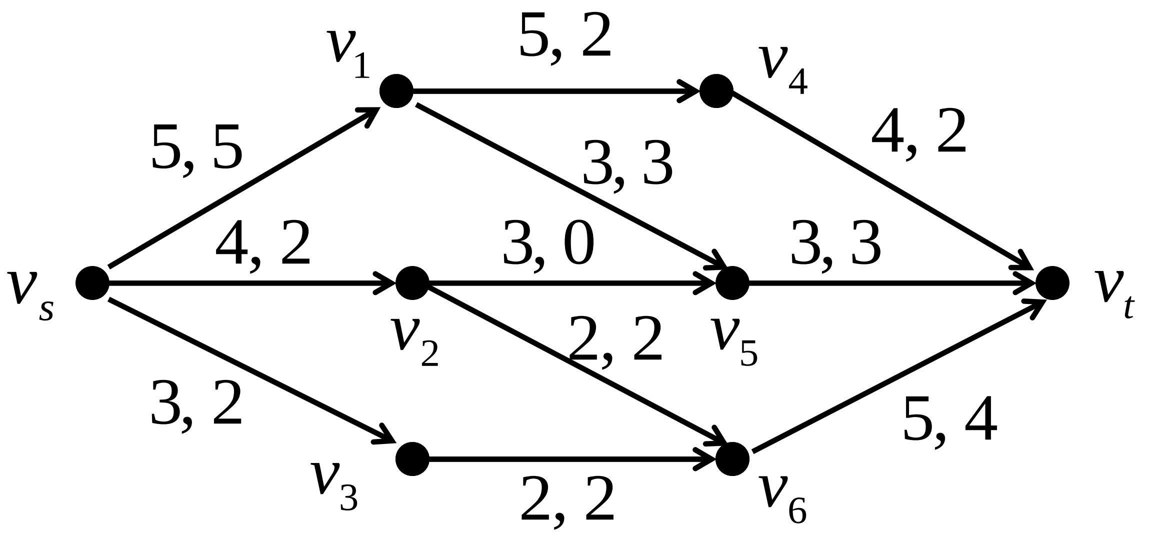
$$\hat{W} \le C\left(S,\bar{S}\right)=W \le C\left(S',\bar{S}'\right)$$

其中 $\hat{W}$ 是任意可行流的流量，$\left(S',\bar{S}'\right)$ 是任意分割 v_s 和 v_t 的割集

要点：求最大流的标号算法

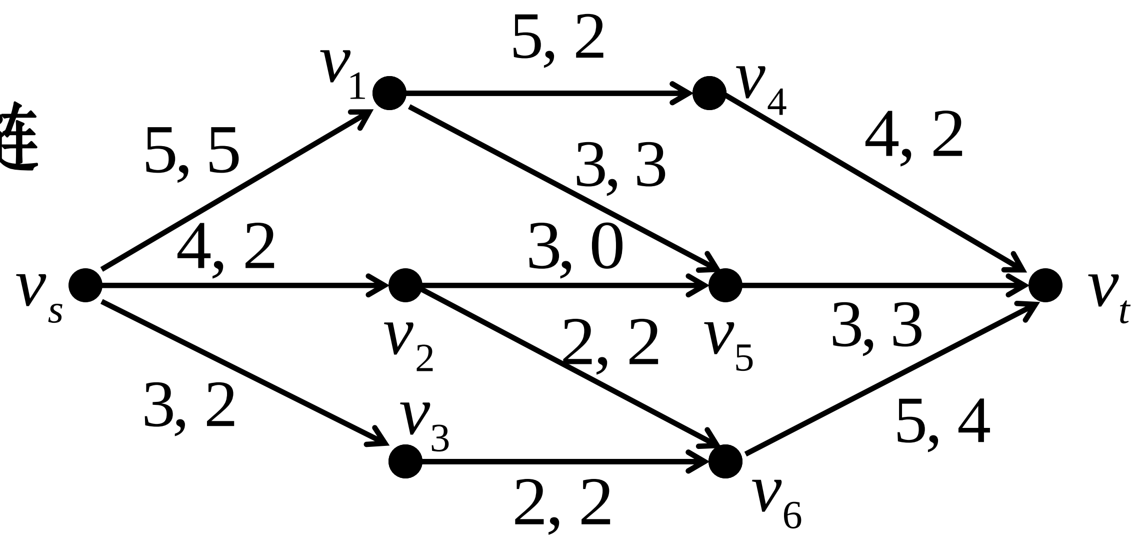
求最大流的标号算法

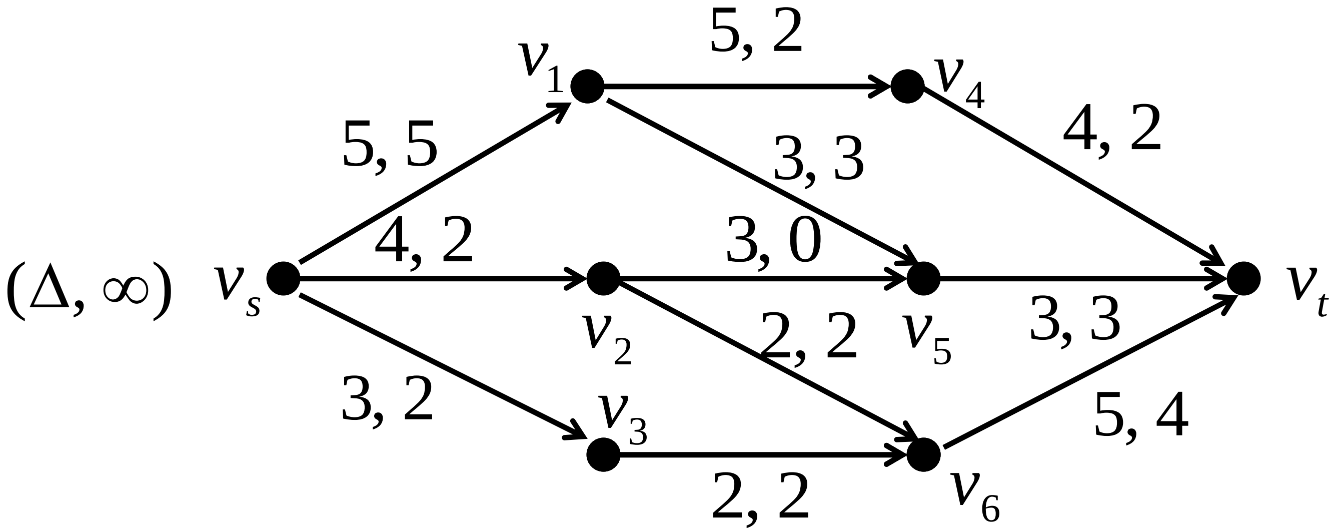
可行流 → 标号寻找可增广链 → 改进的可行流

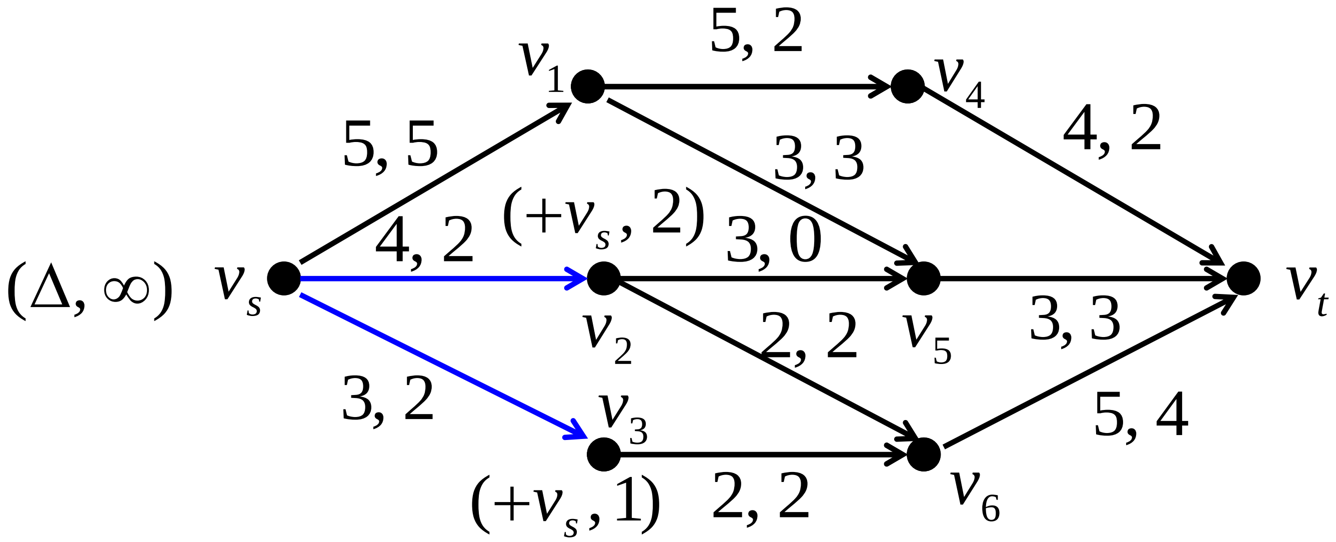


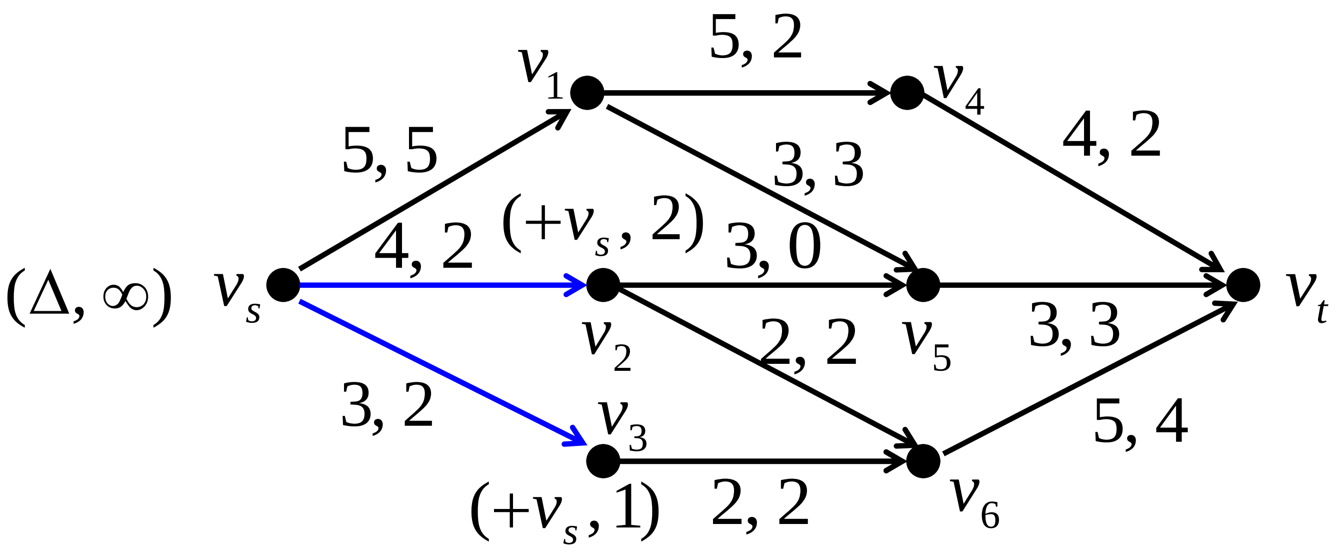
可行流

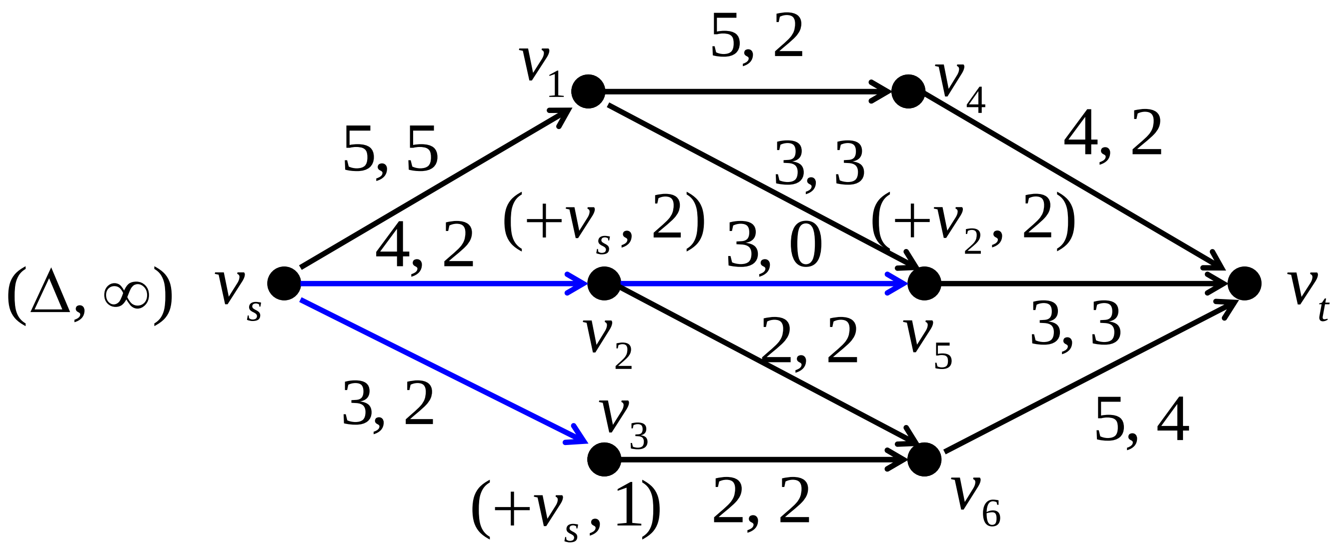
标号寻找可增广链

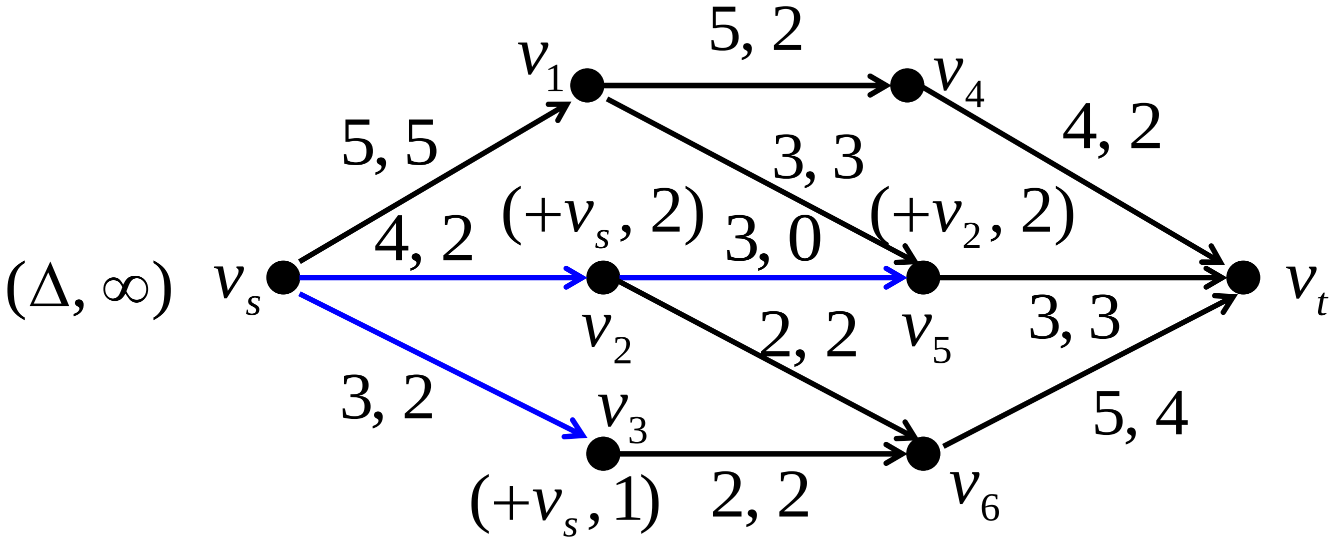


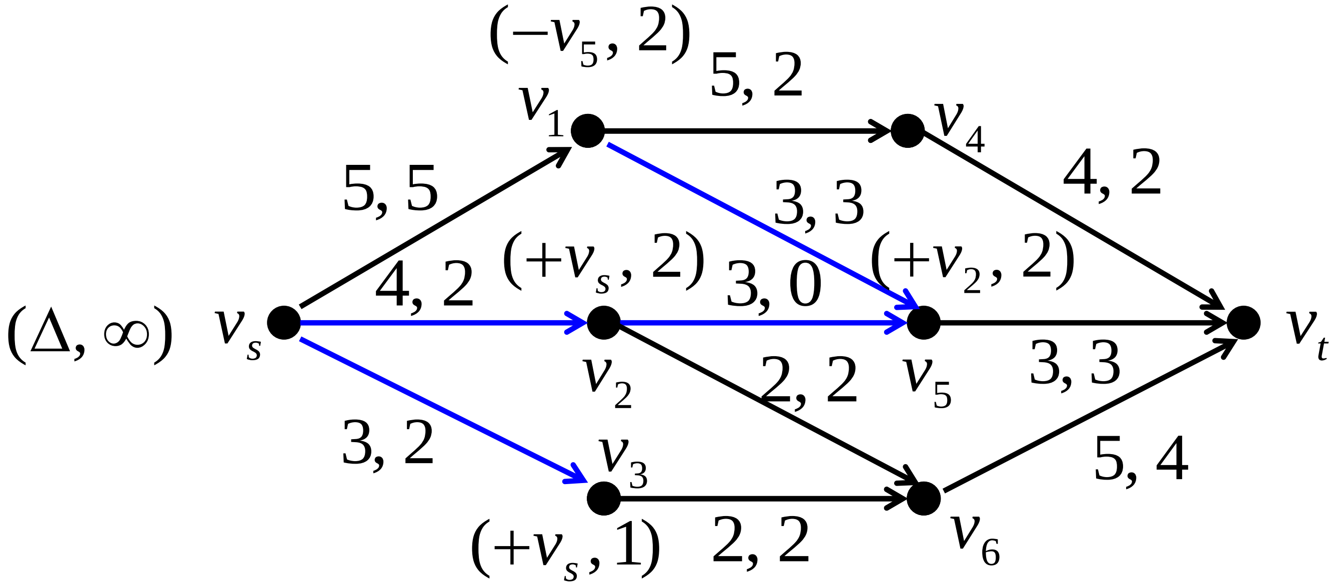

(Δ, ∞)
v_s
v_1
v_2
v_3
v_4
v_5
v_6
v_t
5, 2
5, 5
3, 3
4, 2
4, 2
3, 0
2, 2
3, 3
3, 2
5, 4
2, 2

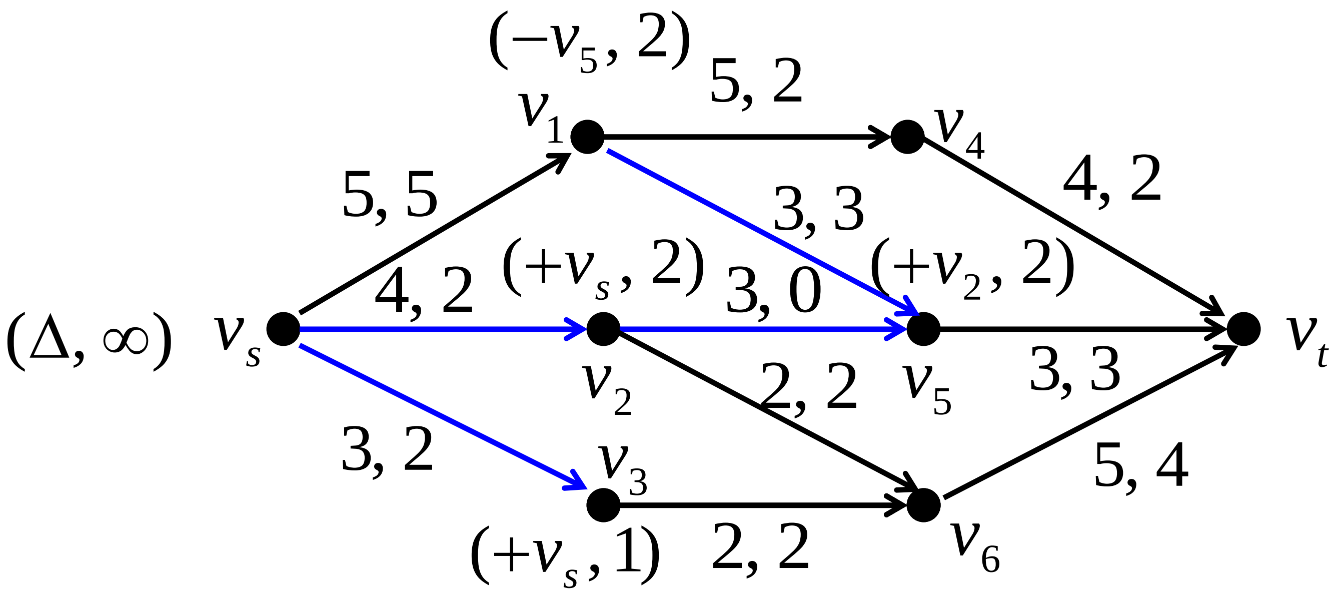

(Δ, ∞)
v_s
v_1
v_2
v_3
v_4
v_5
v_6
v_t
5, 2
5, 5
3, 3
4, 2
(+v_s, 2)
4, 2
3, 0
2, 2
3, 3
3, 2
5, 4
(+v_s, 1)
2, 2

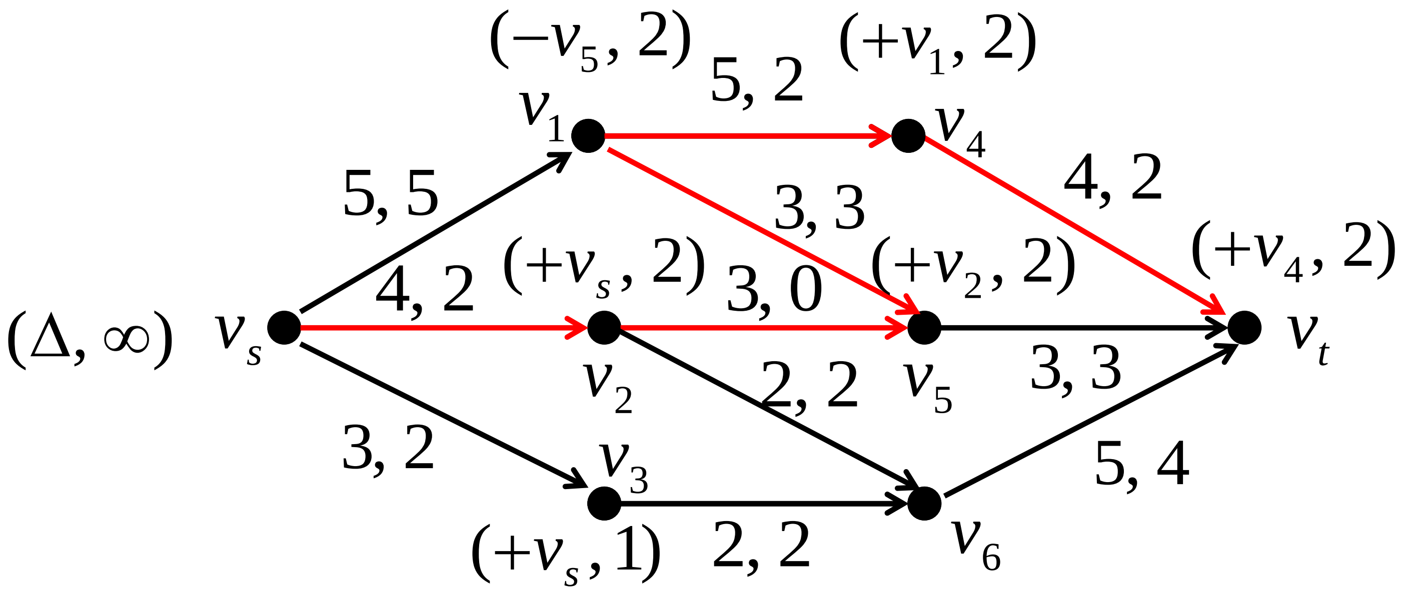

(Δ, ∞)
vs
v1
v2
v3
v4
v5
v6
vt
5, 5
4, 2
3, 2
5, 2
3, 3
3, 0
2, 2
2, 2
4, 2
3, 3
5, 4
(+vs, 2)
(+vs, 1)

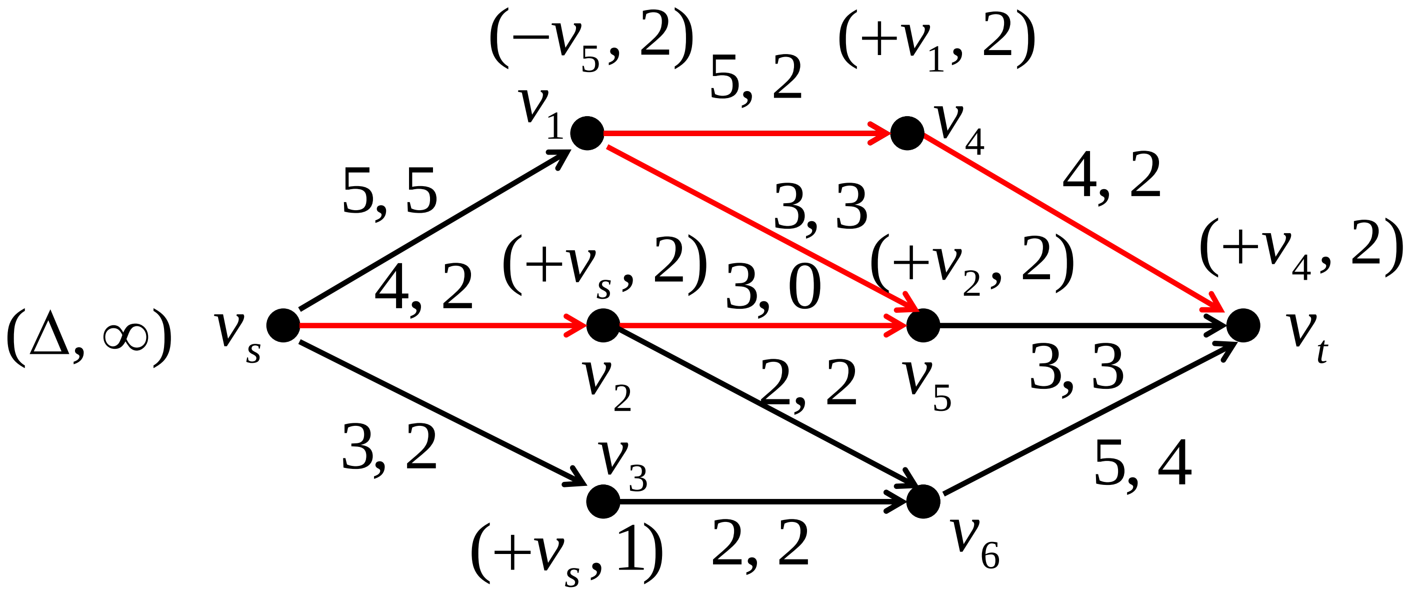

(Δ, ∞)
vs
v1
v2
v3
v4
v5
v6
vt
5, 5
4, 2
3, 2
5, 2
3, 3
3, 0
2, 2
2, 2
4, 2
3, 3
5, 4
(+vs, 2)
(+v2, 2)
(+vs, 1)


(Δ, ∞)
v_s
v_1
v_2
v_3
v_4
v_5
v_6
v_t
5, 5
4, 2
3, 2
5, 2
3, 3
3, 0
2, 2
2, 2
4, 2
3, 3
5, 4
$(+v_s, 2)$
$(+v_2, 2)$
$(+v_s, 1)$

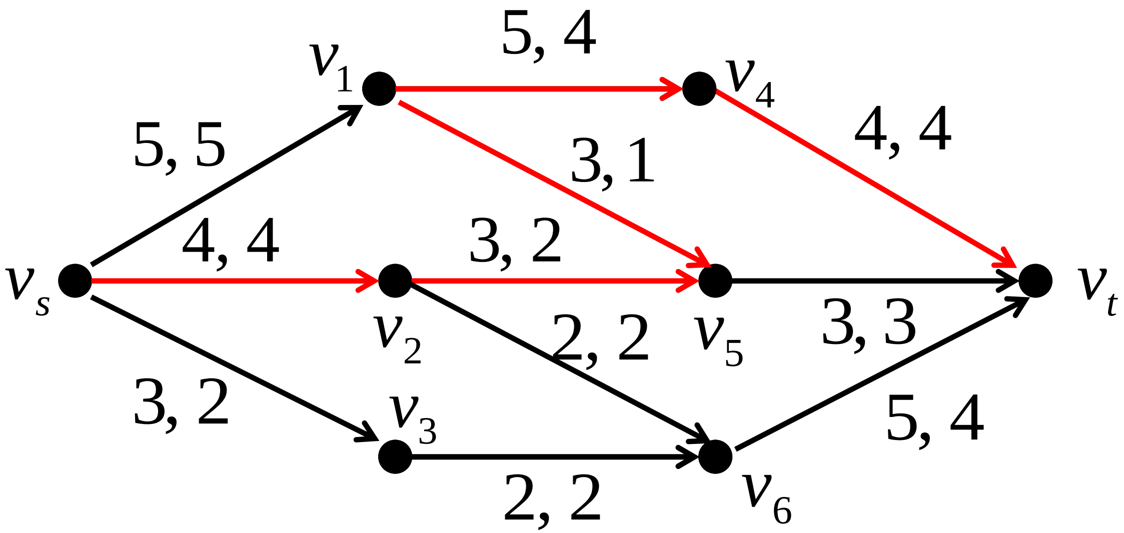

$(-v_5, 2)$
(Δ, ∞)
v_s
v_1
v_2
v_3
v_4
v_5
v_6
v_t
5, 5
4, 2
3, 2
5, 2
3, 3
3, 0
2, 2
2, 2
4, 2
3, 3
5, 4
$(+v_s, 2)$
$(+v_2, 2)$
$(+v_s, 1)$


$(-v_5, 2)$
5, 2
v_1
v_4
5, 5
3, 3
4, 2
$(+v_s, 2)$
3, 0
$(+v_2, 2)$
4, 2
(Δ, ∞)
v_s
v_t
v_2
2, 2
v_5
3, 3
3, 2
v_3
5, 4
v_6
$(+v_s, 1)$
2, 2

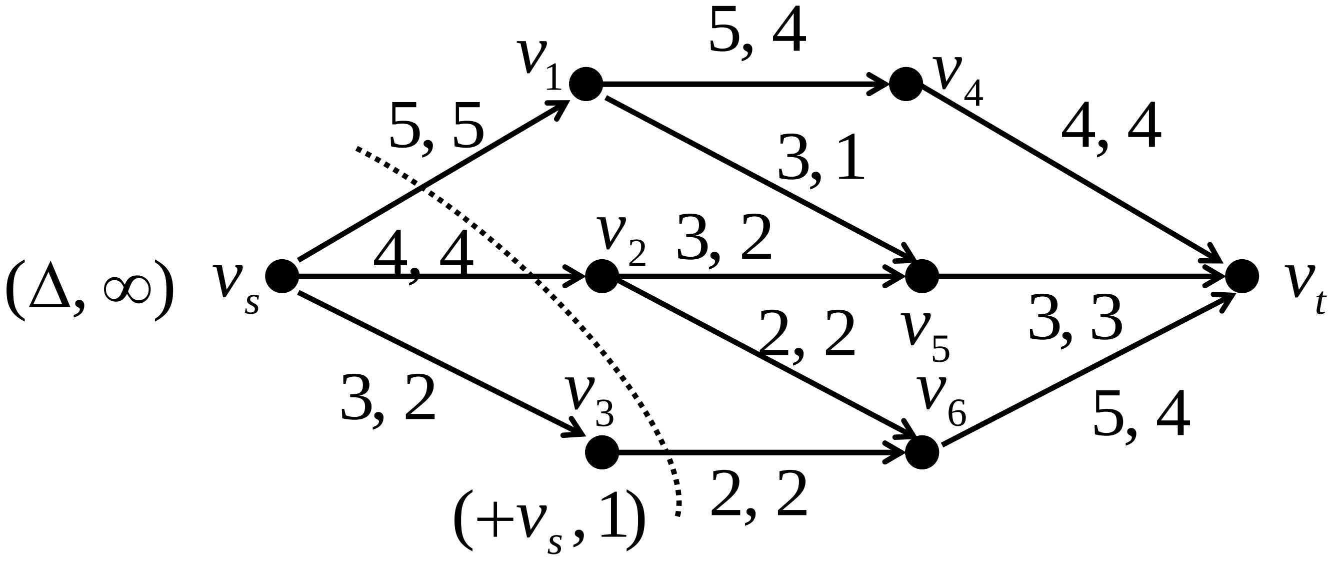

$(-v_5, 2)$
5, 2
$(+v_1, 2)$
v_1
v_4
5, 5
3, 3
4, 2
$(+v_4, 2)$
4, 2
$(+v_s, 2)$
3, 0
$(+v_2, 2)$
(Δ, ∞)
v_s
v_t
v_2
2, 2
v_5
3, 3
3, 2
v_3
5, 4
v_6
$(+v_s, 1)$
2, 2


(−v5, 2)
5, 2
(+v1, 2)
v1
v4
5, 5
3, 3
4, 2
(+vs, 2)
3, 0
(+v2, 2)
(+v4, 2)
4, 2
(Δ, ∞)
vs
vt
v2
2, 2
v5
3, 3
3, 2
v3
5, 4
(+vs, 1)
2, 2
v6

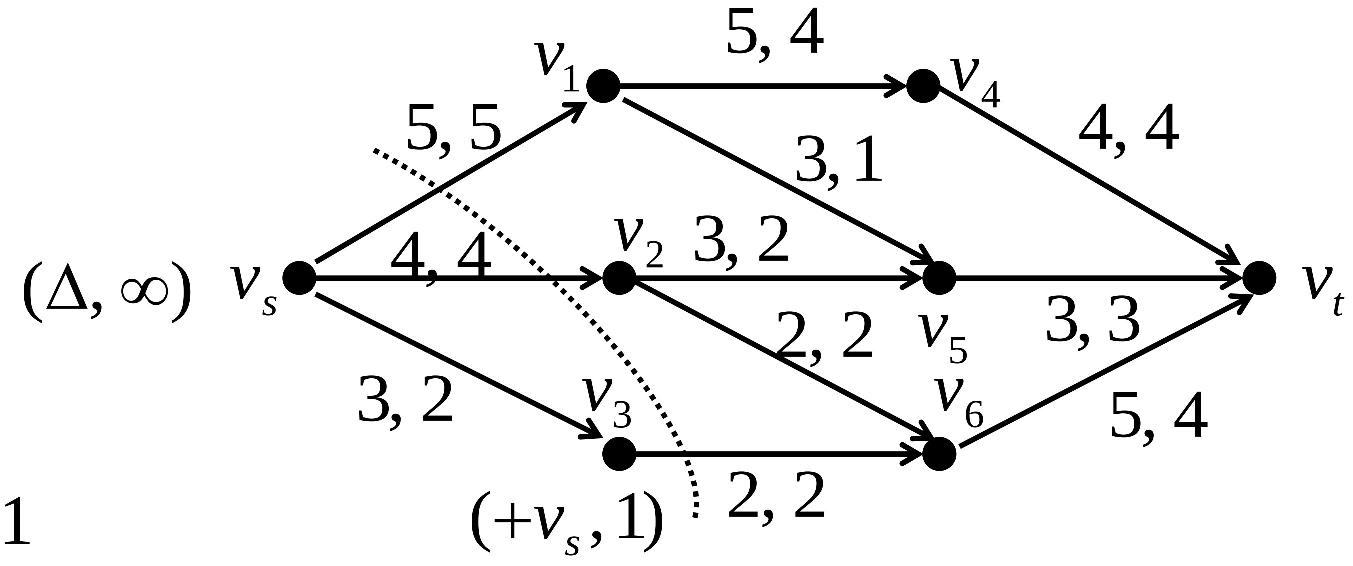
改进可行流


v1
5, 4
v4
5, 5
3, 1
4, 4
4, 4
3, 2
vs
vt
v2
2, 2
v5
3, 3
3, 2
v3
5, 4
2, 2
v6

标号寻找可增广链

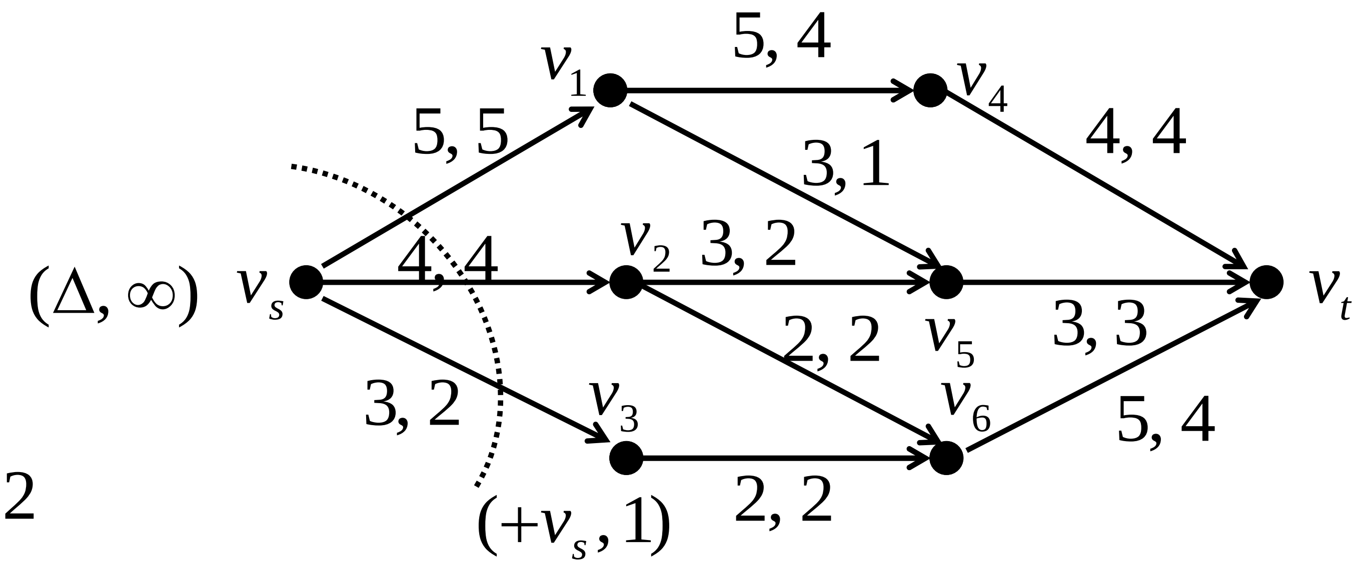


无法标号，停止，此时虚线所示割集容量已经等于可行流的流量，所以已经得到最大流



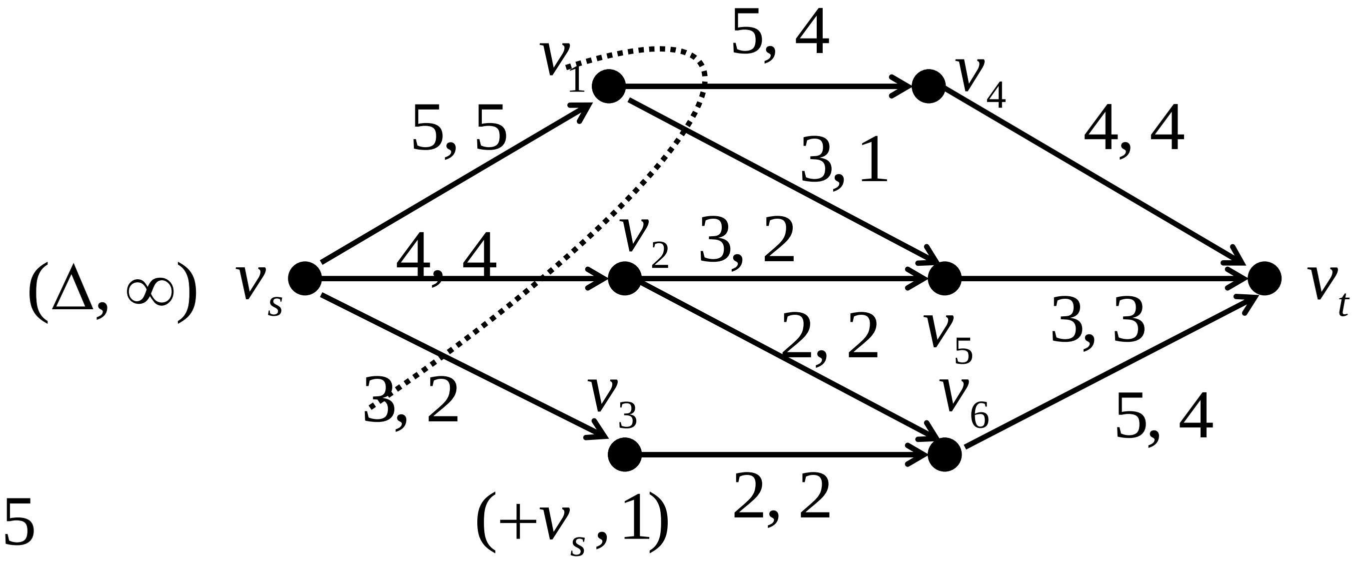
$S=\{v_s, v_3\}$

$\Rightarrow C(S, \bar{S})=11$



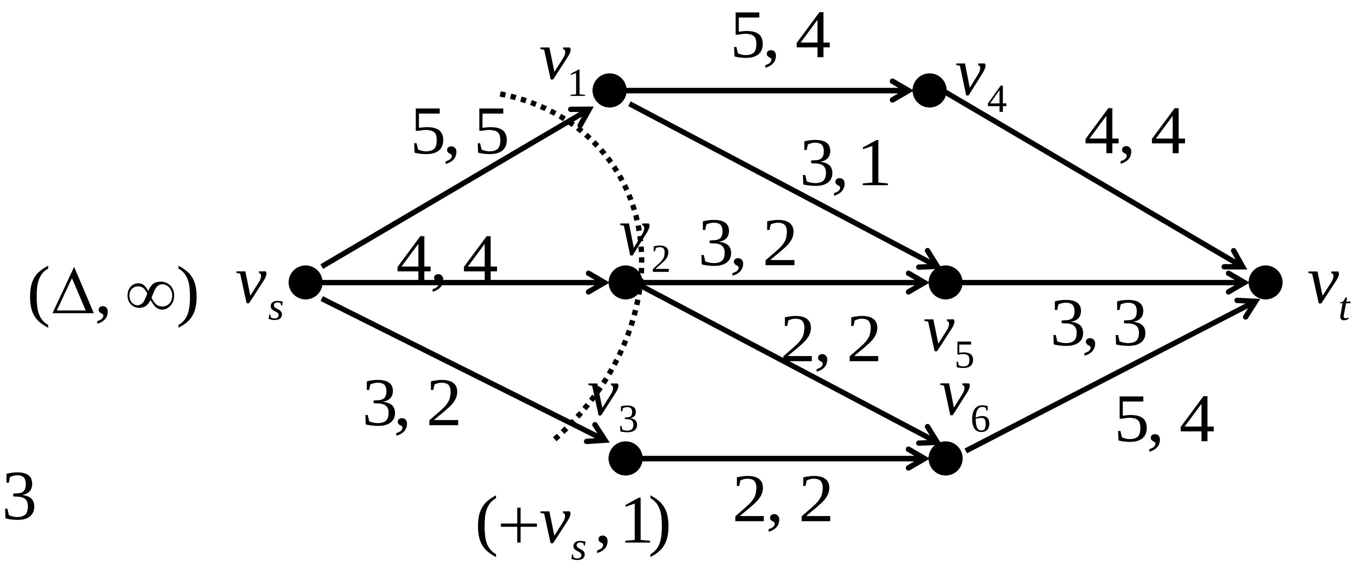
$S=\{v_s\}$

$\Rightarrow C(S, \bar{S})=12$



$S = \{v_s, v_1\}$

$\Rightarrow C(S, \bar{S}) = 15$



$S = \{v_s, v_2\}$

$\Rightarrow C(S, \bar{S}) = 13$

整流定理：整数容量网络存在其所有流量都是整数的最大流

理由：从零流开始，每次按照标号算法改进后的可行流的所有流量一定还是整数