图与网络模型

§1　图与网络的基本概念

§2　最短路问题

§3　最小生成树问题

§4　最大流问题

§5　最小费用最大流问题

§1　图与网络的基本概念

图论中图是由点和边构成，可以反映一些对象之间的关系。

例如：在一个人群中，对相互认识这个关系我们可以用图来表示，图1就是一个表示这种关系的图。

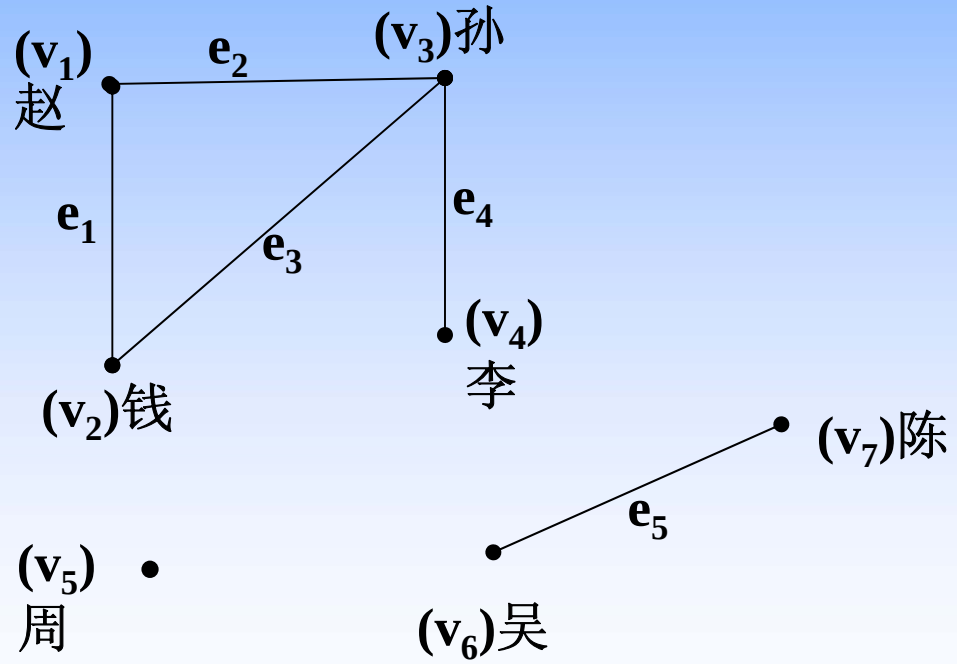


图1

§1 图与网络的基本概念

当然图论不仅仅是要描述对象之间关系，还要研究特定关系之间的内在规律，一般情况下图中点的相对位置如何、点与点之间联线的长短曲直，对于反映对象之间的关系并不是重要的，如对赵等七人的相互认识关系我们也可以用图2来表示，可见图论中的图与几何图、工程图是不一样的。

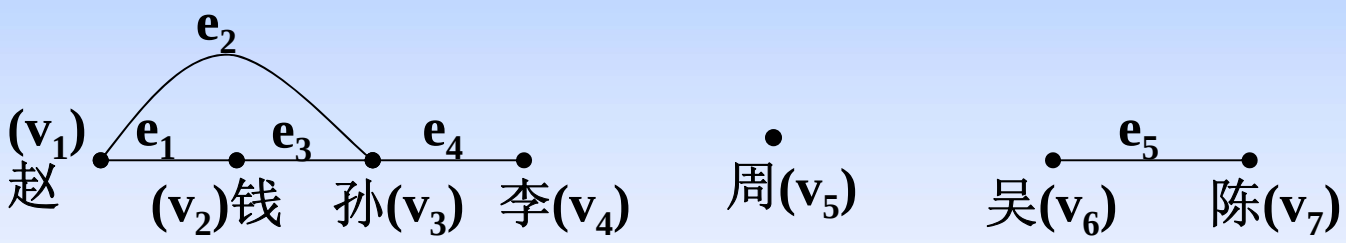


图2

§1　图与网络的基本概念

如果我们把上面例子中的“相互认识”关系改为“认识” 的关系，那么只用两点之间的联线就很难刻画他们之间的关系了，这是我们引入一个带箭头的联线，称为弧。图3就是一个反映这七人“认识”关系的图。相互认识用两条反向的弧表示。

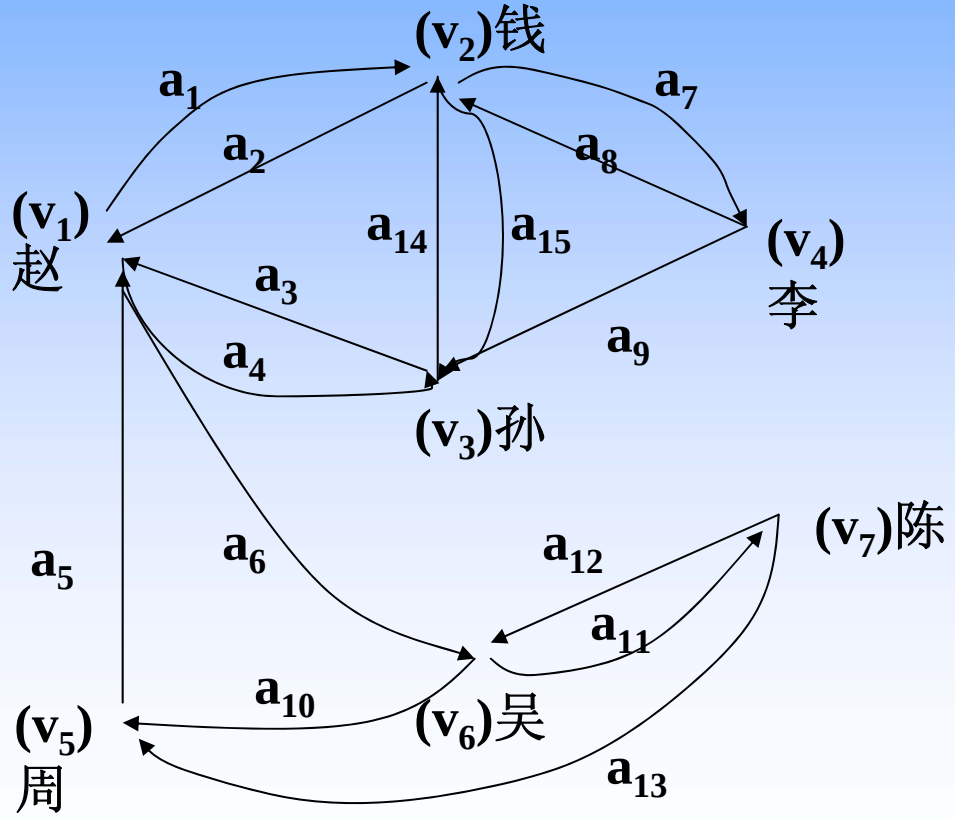


图3

- 无向图：点和边构成的图，记作G=（V，E）。
- 有向图： 由点和弧构成的图，记作D=（V，A）。

连通图：

对无向图G，若任何两个不同的点之间，至少存在一条链，则G为连通图。(由两两相邻的点及其相关联的边构成的点边序列称为链) $(v_{i_1}, a_{i_1}, v_{i_2}, a_{i_2}, \cdots, v_{i_{k-1}}, a_{i_{k-1}}, v_{i_k})$

- 回路：
- 对有向图D=（V，A）,从D中去掉所有弧上的箭头,得到的无向图,称为D的基础图. 记为G(D). 点弧交错序列如是G(D)中的链,且 $a_{i_t}=(v_{i_{t-1}}, v_{i_t})$ 称为路

若路的第一个点和最后一个点相同，则该路为回路。

- 赋权图：

对一个无向图G的每一条边(v_i,v_j)，相应地有一个数w_{ij}，则称图G为赋权图，w_{ij}称为边(v_i,v_j)上的权。

- 网络：

在赋权的有向图D中指定一点，称为发点，指定另一点称为收点，其它点称为中间点，并把D中的每一条弧的赋权数称为弧的容量，D就称为网络。

§2.1 最短路问题

最短路的一般提法为：设 $D=(V,A)$ 为有向图,G(D)为连通图，图中各弧 (v_i,v_j) 有权 l_{ij}（ $l_{ij}=+\infty$ 表示 v_i,v_j 之间没有边），v_s,v_t为图中任意两点，求一条路 μ，使它为从v_s到 v_t 的所有路中总权最短。即：

$$L(\mu)=\sum_{(v_i,v_j)\in\mu} l_{ij} \quad \text{最小。}$$

最短路问题：对一个赋权的有向图D中的指定的两个点Vs和Vt找到一条从 Vs 到 Vt 的路，使得这条路上所有弧的权数的总和最小，这条路被称之为从Vs到Vt的最短路。这条路上所有弧的权数的总和被称为从Vs到Vt的距离。

一、求解最短路的Dijkstra算法(双标号法）

算法步骤：

1.给始点v_s以P标号 $P(v_s)=0$，这表示从v_s到 v_s的最短距离为0，其余节点均给T标号，$T(v_i)=+\infty \quad (i=2,3,\cdots,n)$ 。

2.设节点 v_i 为刚得到P标号的点，考虑点v_j，其中 $(v_i,v_j)\in A$ ，且v_j为T标号。对v_j的T标号进行如下修改：

$$T(v_j)=\min[T(v_j),P(v_i)+l_{ij}]$$

3.比较所有具有T标号的节点，把最小者改为P标号，即：

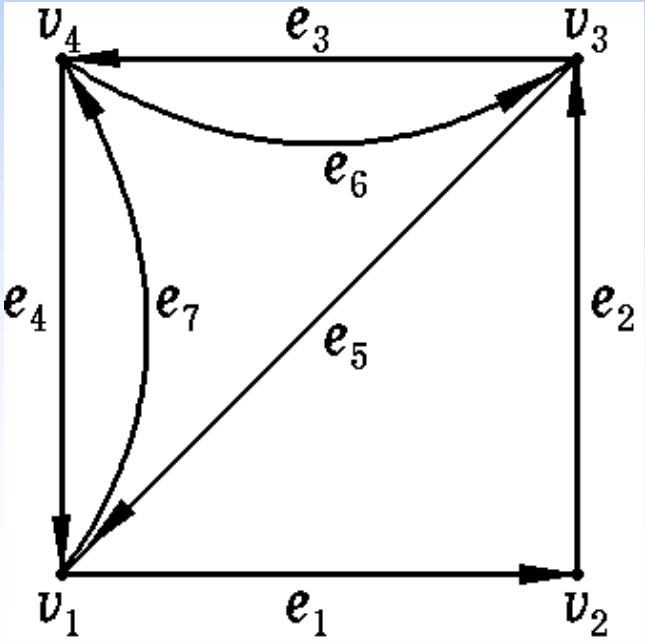
$$P(v_k)=\min[T(v_i)]$$

当存在两个以上最小者时，可同时改为P标号。若全部节点均为P标号，则停止，否则用v_k代替v_i，返回步骤（2）。

图的矩阵表示

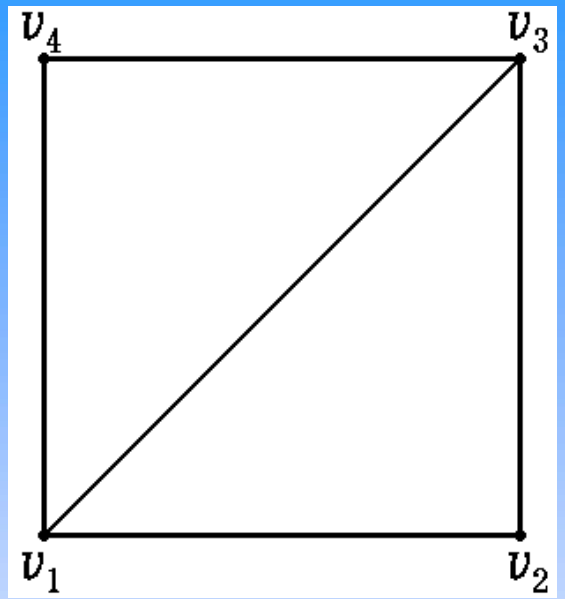
(1) 邻接矩阵　邻接矩阵表示了点与点之间的邻接关系. 一个n阶图G的邻接矩阵$A=(a_{ij})_{n\times n}$，其中

$$a_{ij}=\begin{cases}1, & v_{ij}\in E;\\ 0, & v_{ij}\notin E.\end{cases}$$



$$\boldsymbol{A}=\begin{pmatrix}0&1&0&1\\0&0&1&0\\1&0&0&1\\1&0&1&0\end{pmatrix}$$

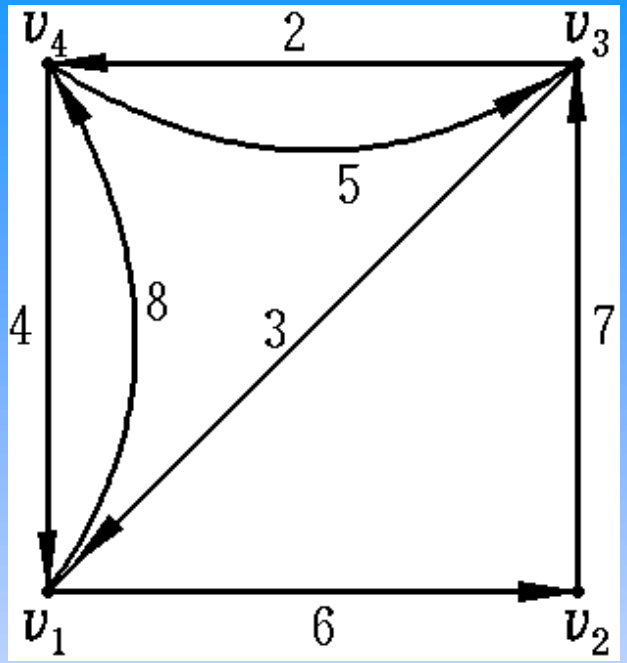
无向图G的邻接矩阵A是一个对称矩阵.



$$A=\begin{pmatrix}0&1&1&1\\1&0&1&0\\1&1&0&1\\1&0&1&0\end{pmatrix}$$

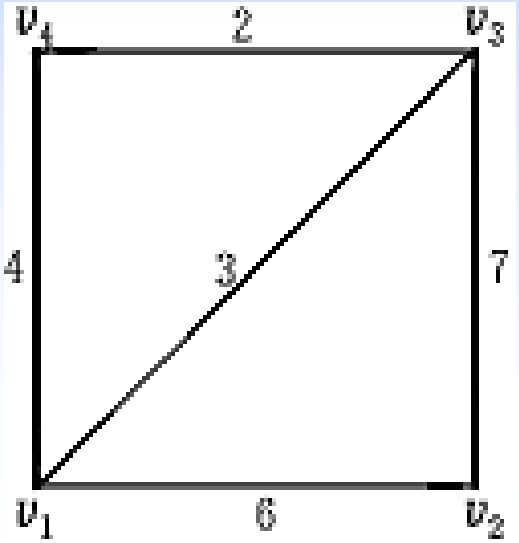
(2) 权矩阵　一个n阶赋权图$G=(V,E,F)$的权矩阵$A=(a_{ij})_{n\times n}$，其中

$$a_{ij}=\begin{cases}F(v_iv_j), & v_{ij}\in E;\\ 0, & i=j;\\ \infty, & v_{ij}\notin E.\end{cases}$$



$$A=\begin{pmatrix} 0 & 6 & \infty & 8 \\ \infty & 0 & 7 & \infty \\ 3 & \infty & 0 & 2 \\ 4 & \infty & 5 & 0 \end{pmatrix}$$

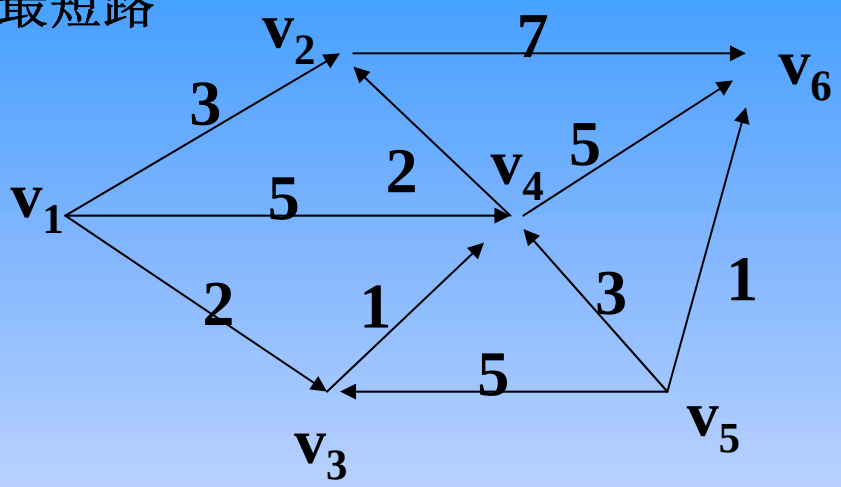
无向图G的权矩阵A是一个对称矩阵.



$$A=\begin{pmatrix} 0 & 6 & 3 & 4 \\ 6 & 0 & 7 & \infty \\ 3 & 7 & 0 & 2 \\ 4 & \infty & 2 & 0 \end{pmatrix}$$

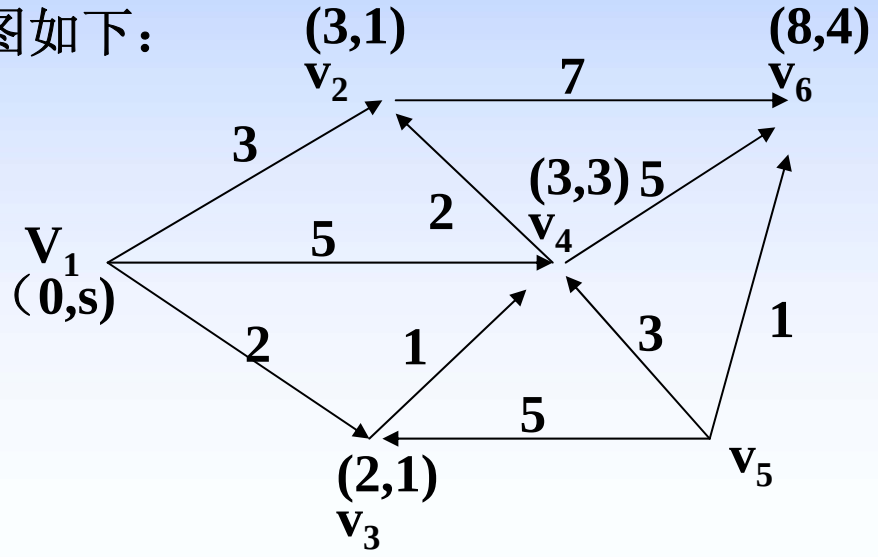
§2.1 最短路问题

例1 求下图中v_1到v_6的最短路



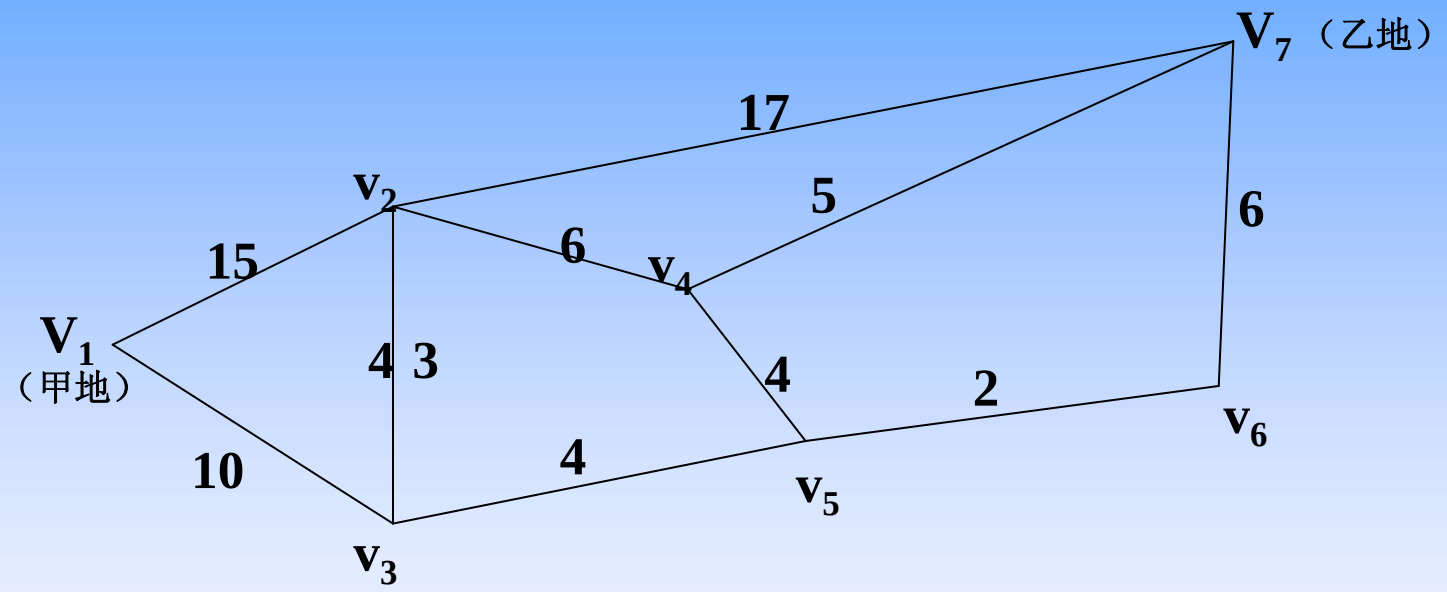
解：采用Dijkstra算法，可解得最短路径为$v_1 \to v_3 \to v_4 \to v_6$

各点的标号图如下：



§2.1 最短路问题

例2 电信公司准备在甲、乙两地沿路架设一条光缆线，问如何架设使其光缆线路最短？下图给出了甲乙两地间的交通图。权数表示两地间公路的长度（单位：公里）。

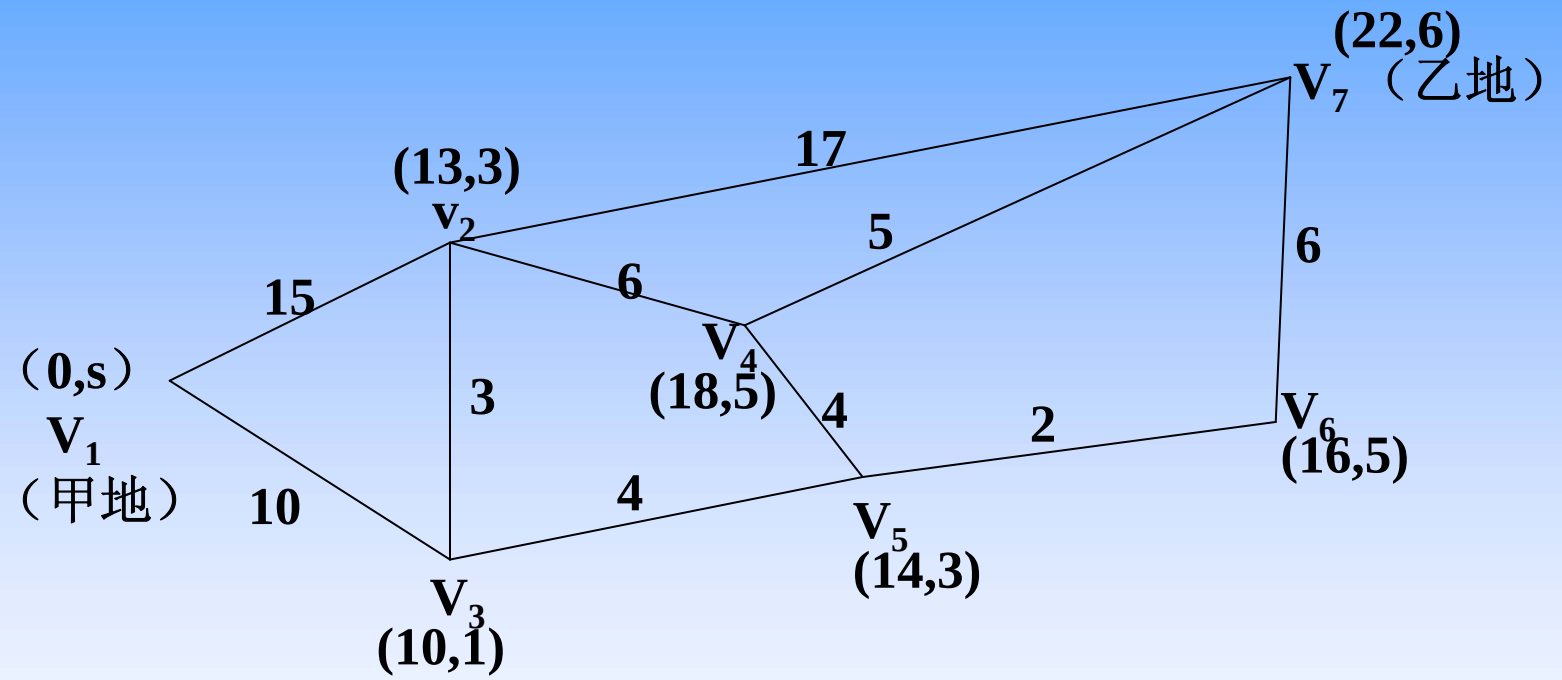


解：这是一个求无向图的最短路的问题。可以把无向图的每一边（v_i,v_j）都用方向相反的两条弧（v_i,v_j）和（v_j,v_i）代替，就化为有向图，即可用**Dijkstra**算法来求解。也可直接在无向图中用**Dijkstra**算法来求解。只要在算法中把从已标号的点到未标号的点的弧的集合改成已标号的点到未标号的点的边的集合即可。

§2.1 最短路问题

例2最终解得：

最短路径$v_1 \to v_3 \to v_5 \to v_6 \to v_7$，每点的标号见下图



§2.1 最短路问题

例3 设备更新问题。某公司使用一台设备，在每年年初，公司就要决定是购买新的设备还是继续使用旧设备。如果购置新设备，就要支付一定的购置费，当然新设备的维修费用就低。如果继续使用旧设备，可以省去购置费，但维修费用就高了。请设计一个五年之内的更新设备的计划，使得五年内购置费用和维修费用总的支付费用最小。

已知：设备每年年初的价格表

年份	1	2	3	4	5
年初价格	11	11	12	12	13

设备维修费如下表

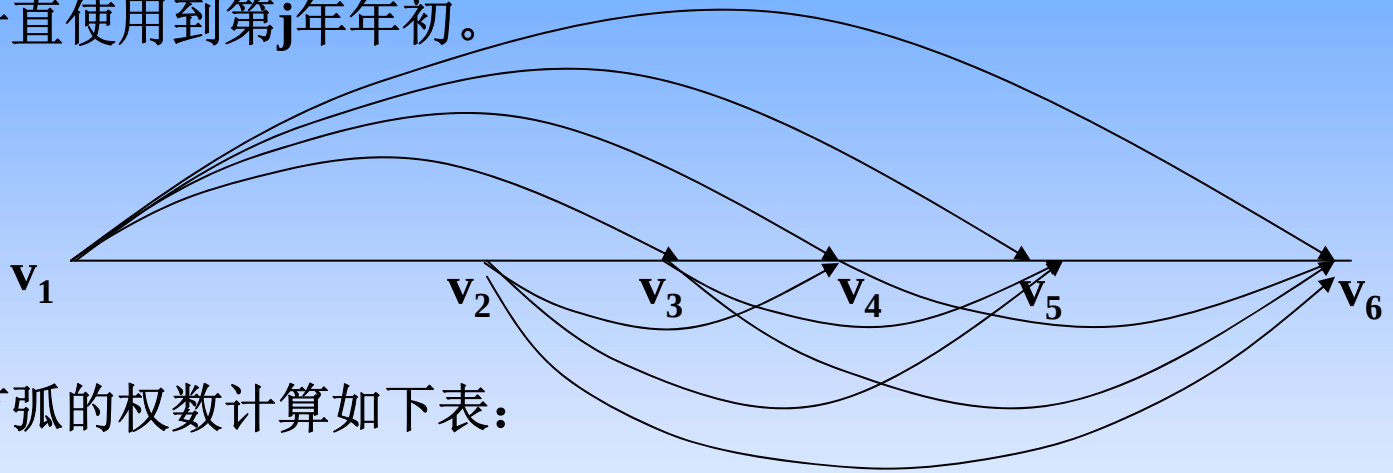
使用年数	0-1	1-2	2-3	3-4	4-5
每年维修费用	5	6	8	11	18

§2.1 最短路问题

例3的解：

将问题转化为最短路问题，如下图：

用v_i表示“第i年年初购进一台新设备”,弧（v_i,v_j）表示第i年年初购进的设备一直使用到第j年年初。

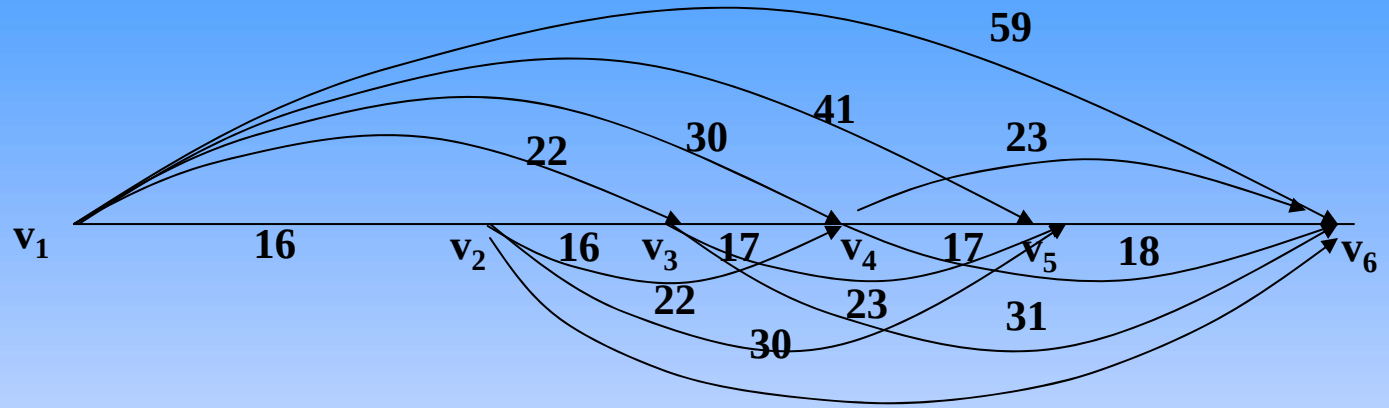


把所有弧的权数计算如下表：

	1	2	3	4	5	6
1		16	22	30	41	59
2			16	22	30	41
3				17	23	31
4					17	23
5						18
6						

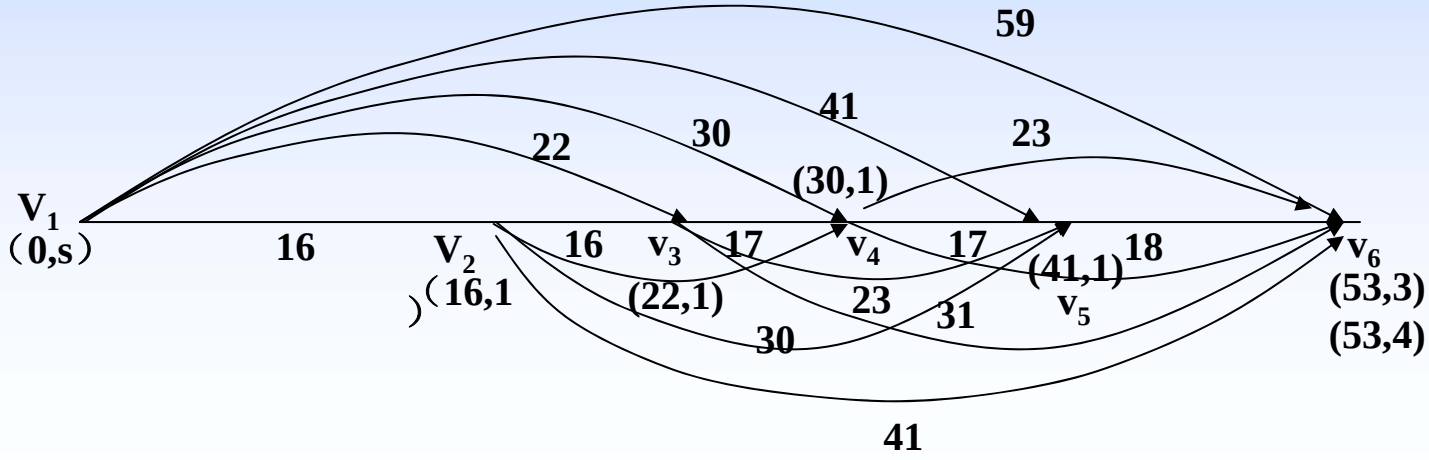
§2.1 最短路问题

(继上页) 把权数赋到图中，再用Dijkstra算法求最短路。



最终得到下图，可知，v_1到v_6的距离是53，最短路径有两条：

$v_1 \to v_3 \to v_6$和 $v_1 \to v_4 \to v_6$



§3 最小生成树问题

- 树是图论中的重要概念，所谓树就是一个无圈的连通图。

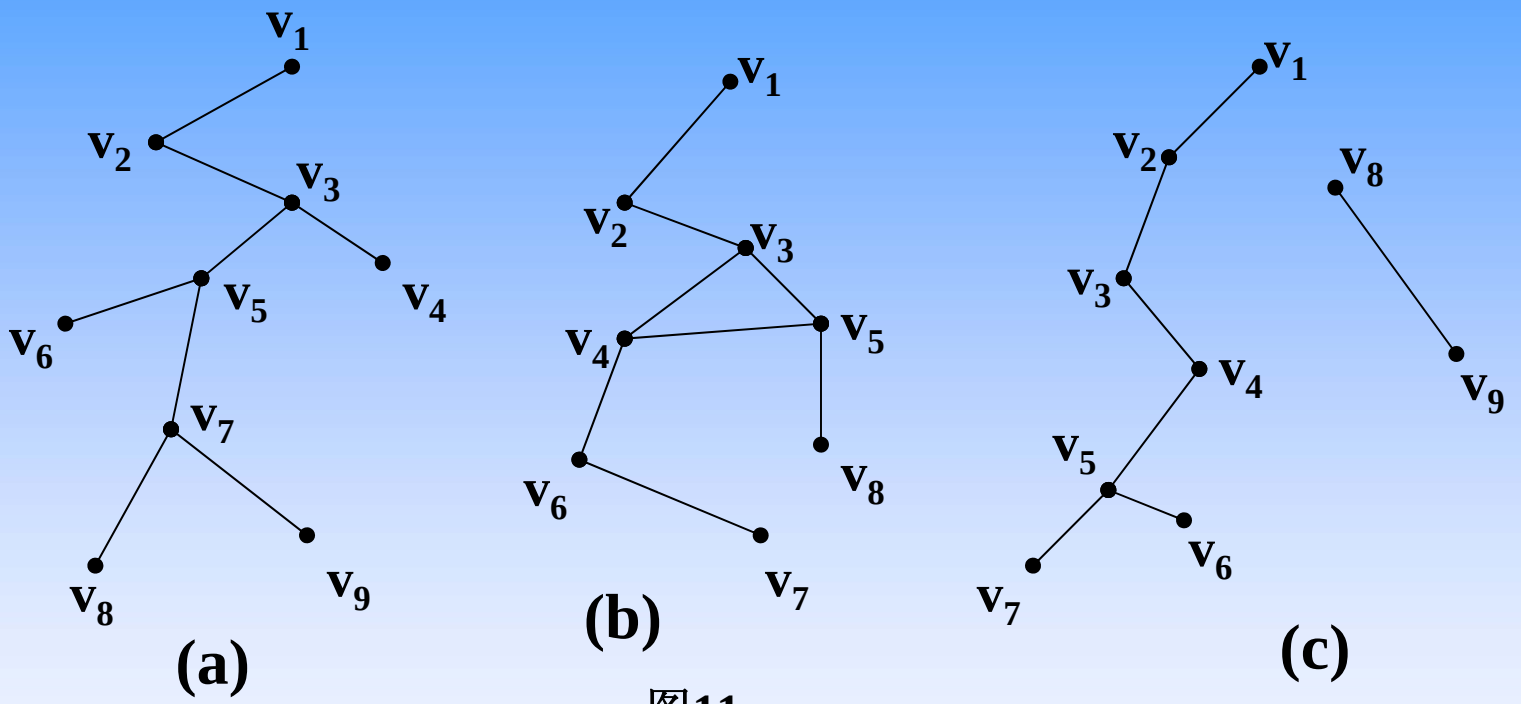


图11

图11中，(a)就是一个树，而(b)因为图中有圈所以就不是树，(c)因为不连通所以也不是树。

§3 最小生成树问题

给了一个无向图G=(V,E)，我们保留G的所有点，而删掉部分G的边或者说保留一部分G的边，所获得的图G，称之为G的生成子图。在图12中，(b)和(c)都是(a)的生成子图。

如果图G的一个生成子图还是一个树，则称这个生成子图为生成树，在图12中，(c)就是(a)的生成树。

最小生成树问题就是指在一个赋权的连通的无向图G中找出一个生成树，并使得这个生成树的所有边的权数之和为最小。

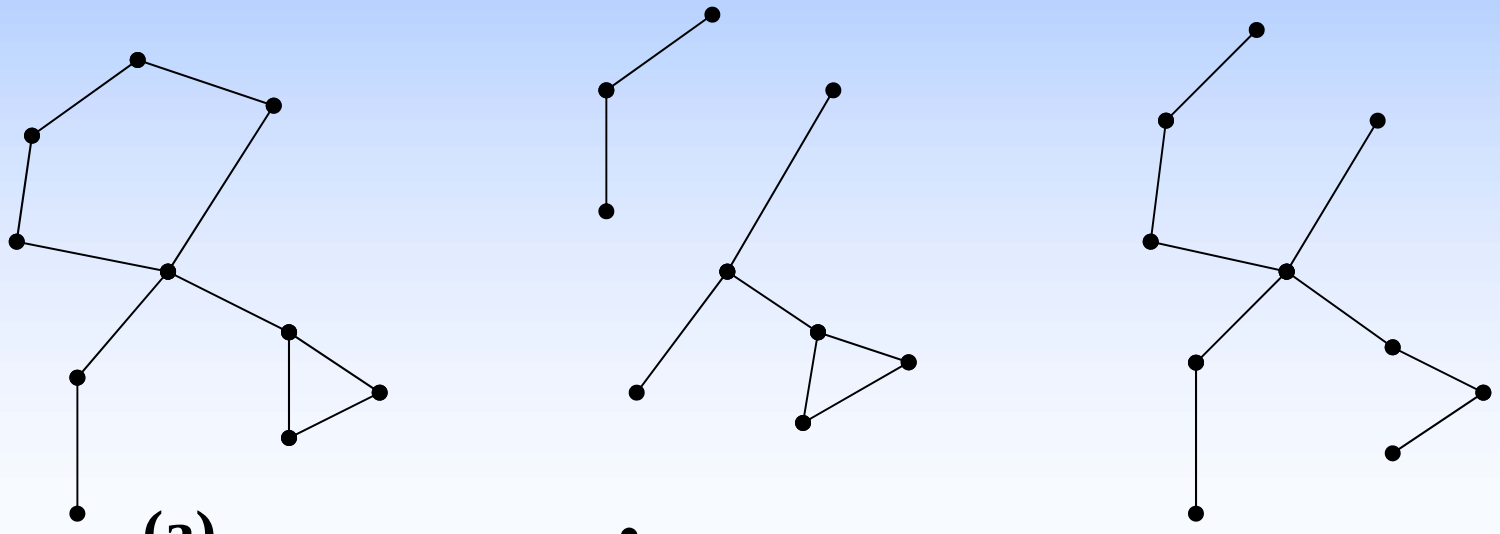


图12

§3 最小生成树问题

一、求解最小生成树的破圈算法

算法的步骤：

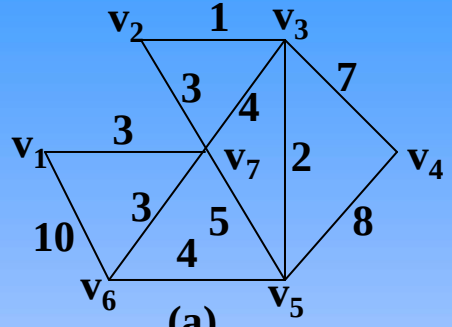
1、在给定的赋权的连通图上任找一个圈。

2、在所找的圈中去掉一个权数最大的边（如果有两条或两条以上的边都是权数最大的边，则任意去掉其中一条）。

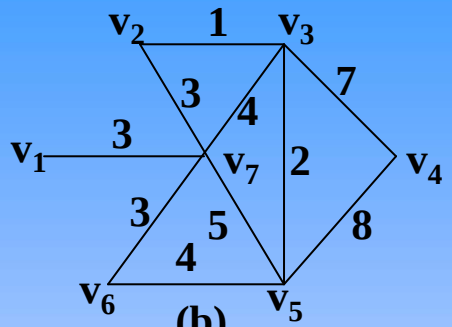
3、如果所余下的图已不包含圈，则计算结束，所余下的图即为最小生成树，否则返回第1步。

§3 最小生成树问题

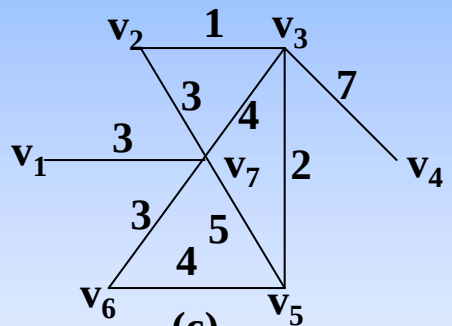
例4 用破圈算法求图（a）中的一个最小生成树



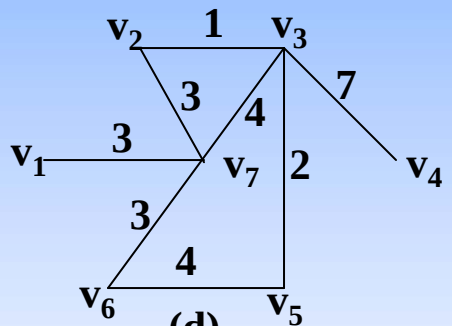
(a)



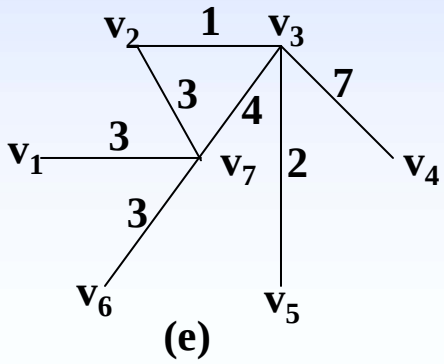
(b)



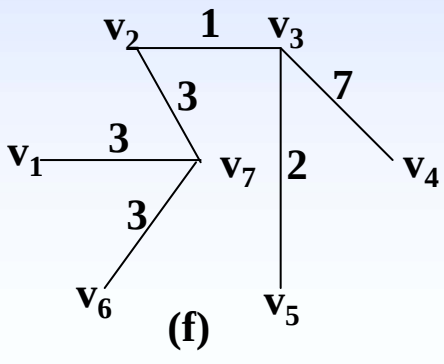
(c)



(d)



(e)



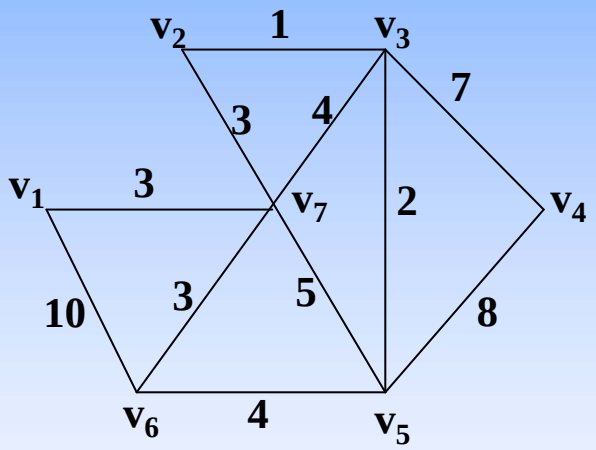
(f)

图13

§3 最小生成树问题

例5、某大学准备对其所属的7个学院办公室计算机联网，这个网络的可能联通的途径如下图，图中$v_1,\ldots,v_7$ 表示7个学院办公室，请设计一个网络能联通7个学院办公室，并使总的线路长度为最短。

图14



解：此问题实际上是求图14的最小生成树，这在例4中已经求得，也即按照图13的(f)设计，可使此网络的总的线路长度为最短，为19百米。有专门的程序可以解决最小生成树问题。

§4、 最大流问题

1、设一个赋权有向图$D=(V, A)$,在V中指定一个发点v_s和一个收点v_t ,其它的点叫做中间点。对于D中的每一个弧$(v_i , v_j) \in A$,都有一个非负数c_{ij},叫做弧的容量。我们把这样的图D叫做一个容量网络，简称网络，记做

$D=(V, A, C)$。

网络D上的流，是指定义在弧集合A上的一个函数

$$f=\{f(v_i, v_j)\}=\{f_{ij}\}$$

其中$f(v_i, v_j)=f_{ij}$ 叫做弧(v_i, v_j)上的流量。

2、称满足下列条件的流为可行流：

（1）容量条件：对于每一个弧（v_i, v_j）$\in A$ 有 $0 \leqslant f_{ij} \leqslant c_{ij}$。

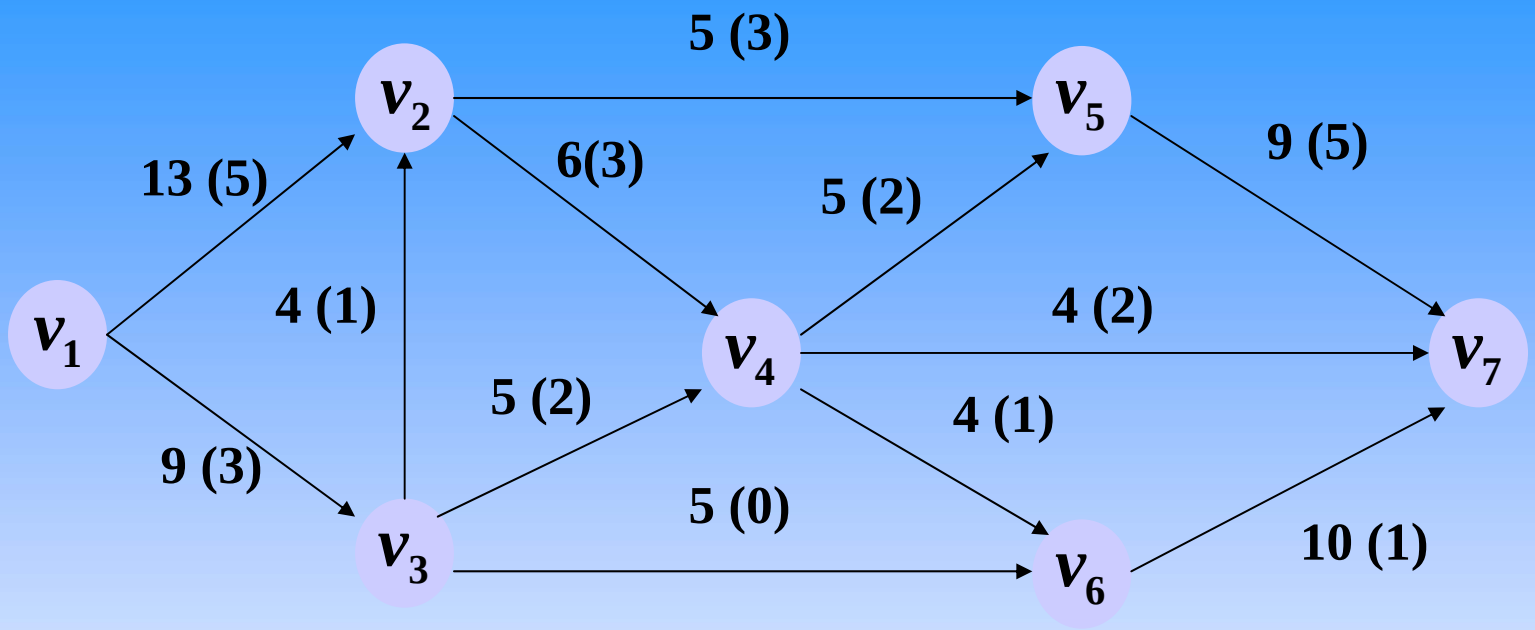
（2）平衡条件：

对于发点v_s，有 $\sum_{(v_s, v_j)\in A} f_{sj} - \sum_{(v_j, v_s)\in A} f_{js} = v(f)$

对于收点v_t，有 $\sum_{(v_t, v_j)\in A} f_{tj} - \sum_{(v_j, v_t)\in A} f_{jt} = -v(f)$

对于中间点，有 $\sum_{(v_i, v_j)\in A} f_{ij} - \sum_{(v_j, v_i)\in A} f_{ji} = 0$

可行流中 $f_{ij}=c_{ij}$ 的弧叫做饱和弧，$f_{ij}<c_{ij}$的弧叫做非饱和弧。$f_{ij}>0$ 的弧为非零流弧，$f_{ij}=0$ 的弧叫做零流弧。



图中 (v_3, v_6) 为零流弧，其余为非饱和弧。

§5 关键路径问题

一项工程任务, 大到建造一座大坝, 一座体育中心, 小至组装一台机床, 一架电视机， 都要包括许多工序. 这些工序相互约束, 只有在某些工序完成之后， 一个工序才能开始. 即它们之间存在完成的先后次序关系, 一般认为这些关系是预知的, 而且也能够预计完成每个工序所需要的时间.

这时工程领导人员迫切希望了解最少需要多少时间才能够完成整个工程项目， 影响工程进度的要害工序是哪几个?

PT(Potentialtask graph)图

在PT(Potentialtask graph)图中, 用结点表示工序, 如果工序 i 完成之后工序 j 才能启动, 则图中有一条有向边(i, j), 其长度w_i表示工序 i 所需的时间.

这种图必定不存在有向回路, 否则某些工序将在自身完成之后才能开始, 这显然不符合实际情况.

在PT图中增加两个虚拟结点v_0和v_n, 使所有仅为始点的结点都直接与v_0联结, v_0为新增边的始点, 这些新增边的权都设为0; 使所有仅为终点的结点都直接与v_n联结, v_n为新增边的终点. 这样得到的图G仍然不存在有向回路.

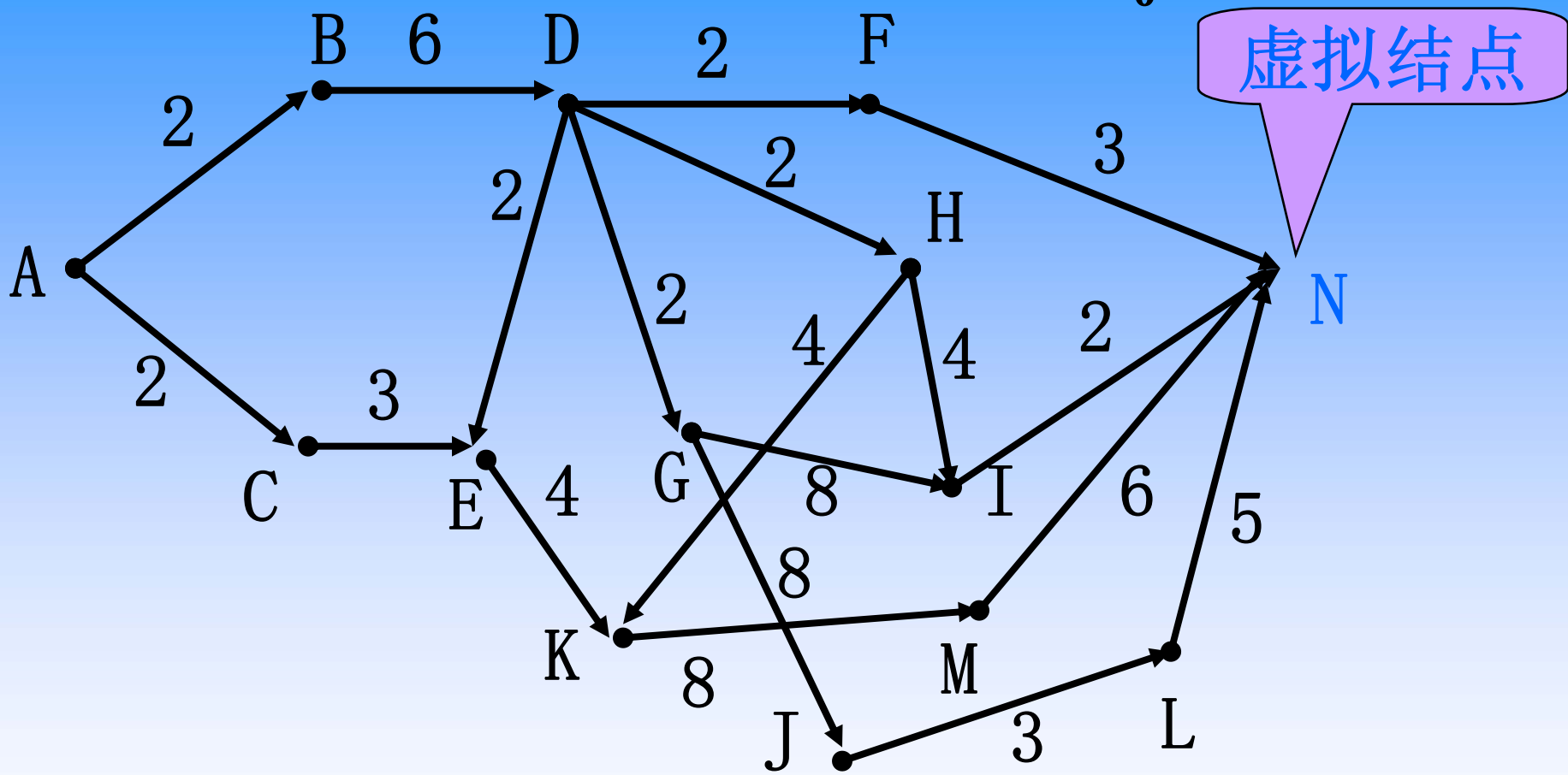
例 一项工程由13道工序组成，所需时间(单位：天)及先行工序如下表所示

工序序号	A	B	C	D	E	F	G	H	I
所需时间	2	6	3	2	4	3	8	4	2
先行工序	—	A	A	B	C,D	D	D	D	G,H

工序序号	J	K	L	M
所需时间	3	8	5	6
先行工序	G	H,E	J	K

试问这项工程至少需要多少天才能完成？那些工程不能延误？那些工程可以延误？最多可延误多少天？

先作出该工程的**PT**图. 由于除了工序A外，均有先行工序, 因此不必虚设虚拟结点v_0.



在**PT**图中,容易看出各工序先后完成的顺序及时间.

关键路径(最长路径)算法

定理 若有向图G中不存在有向回路, 则可以将G 的结点重新编号为$u_1, u_2, \ldots, u_n$, 使得对任意的边$u_i u_j \in E(G)$, 都有$i < j$.

各工序最早启动时间算法步骤:

① 根据定理对结点重新编号为$u_1, u_2, \ldots, u_n$.

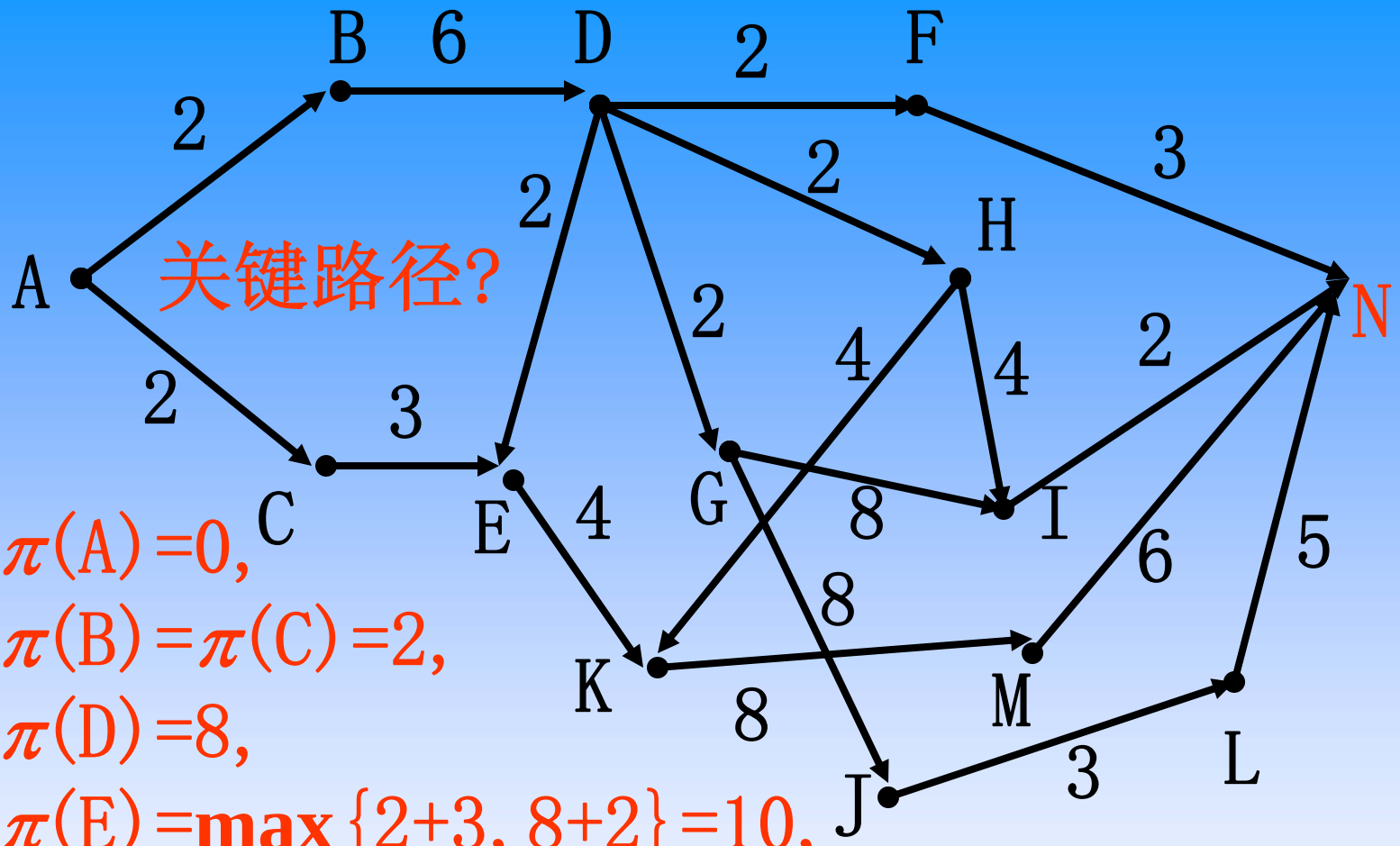
② 赋初值 $\pi(u_1) = 0$.

③ 依次更新 $\pi(u_j), j = 2, 3, \ldots, n$.

$$\pi(u_j) = \max\{\pi(u_i) + \omega(u_i, u_j) \mid u_i u_j \in E(G)\}.$$

④ 结束.

其中$\pi(u_j)$表示工序 u_j 最早启动时间, 而$\pi(u_n)$即$\pi(v_n)$是整个工程完工所需的最短时间.



此例不必重新编号，只要按字母顺序即可.

$\pi(A)=0,$

$\pi(B)=\pi(C)=2,$

$\pi(D)=8,$

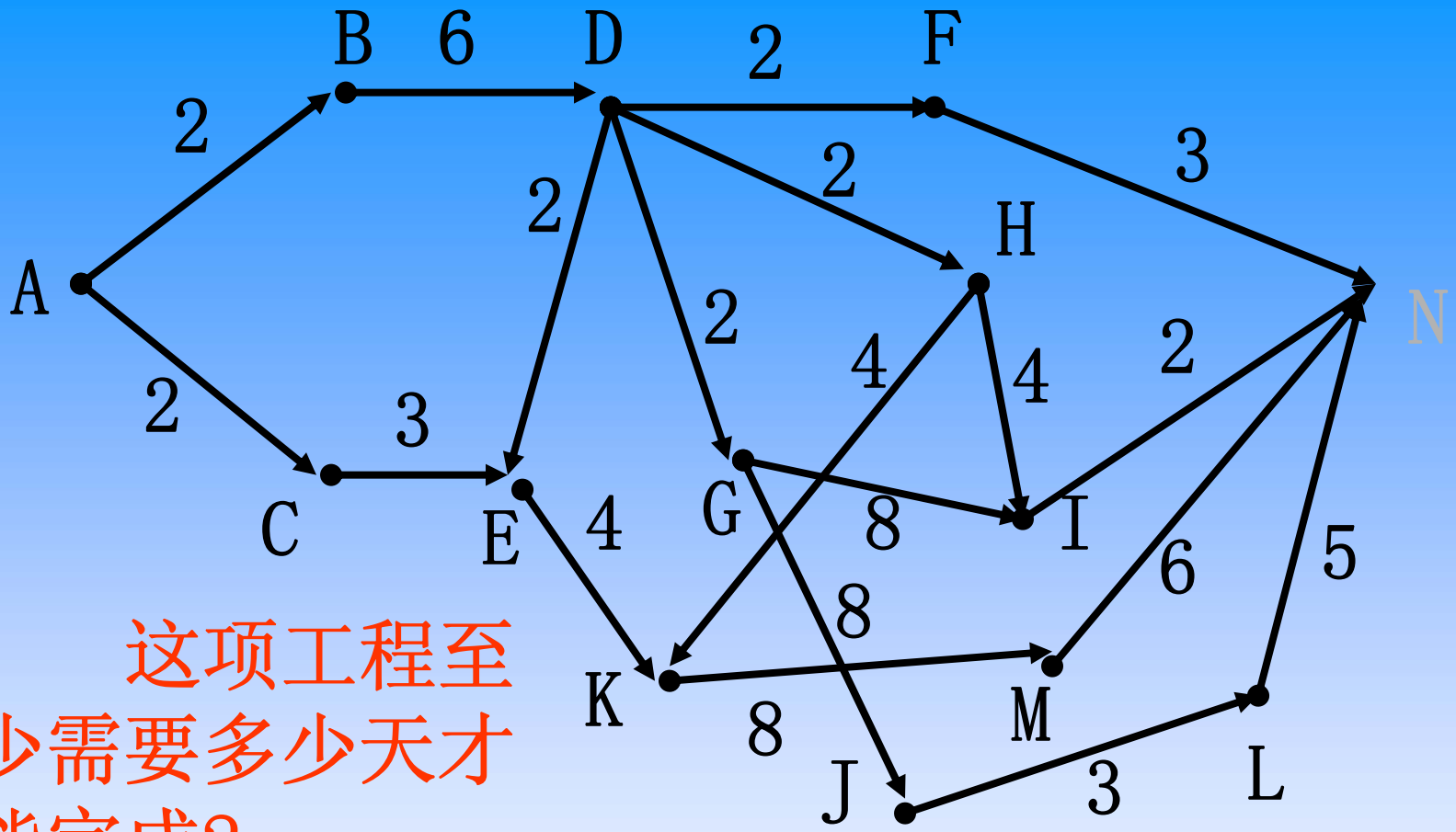
$\pi(E)=\mathbf{max}\{2+3,\ 8+2\}=10,$

$\pi(F)=\pi(G)=\pi(H)=\pi(D)+2=10,$ $\pi(J)=\pi(G)+8=18,$

$\pi(I)=\mathbf{max}\{\pi(G)+8,\ \pi(H)+4\}=18,$ $\pi(L)=\pi(J)+3=21,$

$\pi(K)=\mathbf{max}\{\pi(E)+4,\ \pi(H)+4\}=14,$ $\pi(M)=\pi(K)+8=22,$

$\pi(N)=\mathbf{max}\{\pi(F)+3,\ \pi(I)+2,\ \pi(L)+5,\ \pi(M)+6\}=28.$



这项工程至少需要多少天才能完成?

就是要求A到N的最长路,此路径称为关键路径.

那些工程不能延误? 那些工程可以延误? 最多可延误多少天?

关键路径上的那些工程不能延误.

通过以上计算表明：

这项工程至少需要28天才能完成.

关键路径(最长路径)：

A→B→D→E→K→M→N

A→B→D→H→K→M→N

工序A, B, D, E, H, K, M不能延误, 否则将影响工程的完成.

但是对于不在关键路径上的工序，是否允许延误？如果允许，最多能够延误多长时间呢？

各工序允许延误时间$t(u_j)$等于各工序最晚启动时间$\tau(u_j)$减去各工序最早启动时间$\pi(u_j)$.

即 $t(u_j)=\tau(u_j)-\pi(u_j)$.