组合最优化

中国科学院大学

数学科学学院

郭田德

电话：88256412

Email: tdguo@ucas.ac.cn

第 二 章　预备知识

2.1 记号

2.2 图论与网络优化

2.3 几种规划之间的关系

2.4 最优化问题

2.5 邻域

2.1 记 号

$$c=\begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{pmatrix}=(c_1,c_2,\cdots,c_n)^T \in R^n$$

$$A=\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix}=(P_1,P_2,\cdots,P_n)=\begin{pmatrix} a_1^T \\ a_2^T \\ \vdots \\ a_m^T \end{pmatrix} \in R^{m\times n}$$

已知　$A \in R^{m\times n}, b \in R^{m}$，则

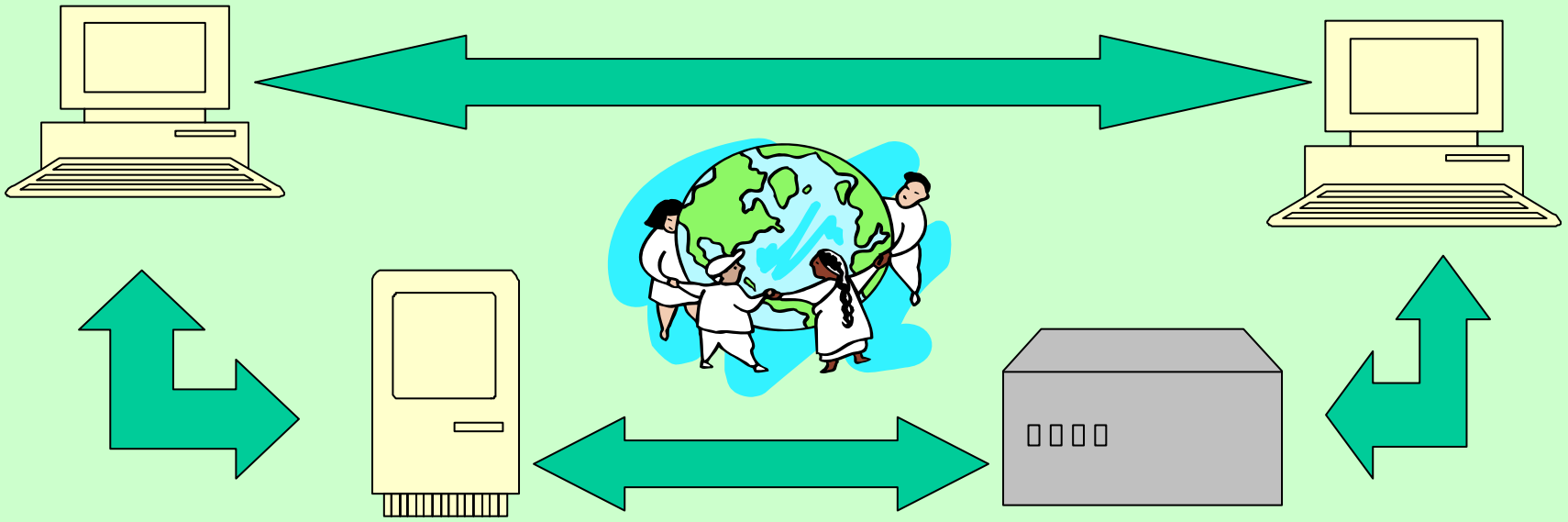
$$Ax = b \Leftrightarrow \begin{pmatrix} a_1^T \\ a_2^T \\ \vdots \\ a_m^T \end{pmatrix} x = b \Leftrightarrow a_i^T x = b_i (i = 1,2,\cdots,m)$$

$$A^T y \le c \Leftrightarrow (P_1, P_2, \cdots, P_n)^T y \le c \Leftrightarrow \begin{pmatrix} P_1^T \\ P_2^T \\ \vdots \\ P_n^T \end{pmatrix} y \le c$$

$$\Leftrightarrow P_j^T y \le c_j (j = 1,2,\cdots,n)$$

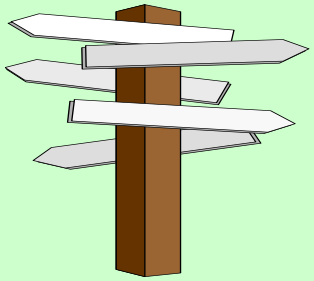
2.2 图论与网络

- 网络：网络社会 ---- 计算机信息网络？



电话通信网络　　运输服务网络
能源和物质分派网络　人际关系网络　　等等

网络优化就是研究如何有效地计划、管理和控制网络系统，使之发挥最大的社会和经济效益



网 络 优 化 简 介

- 网络：数学模型、数学结构 ---- 图

•网络优化就是研究与（赋权）图有关的最优化问题

从若干可能的安排或方案中寻求某种意义下的最优安排或方案，数学上把这种问题称为（最）优化 **(Optimization)** 问题

•运筹学（**Operations Research - OR**）的一个分支

网络优化问题的例子

例2.1 公路连接问题

某一地区有若干个主要城市，现准备修建高速公路把这些城市连接起来， 使得从其中任何一个城市都可以经高速公路直接或间接到达另一个城市. 假定已经知道了任意两个城市之间修建高速公路的成本，那么应如何决定在哪些城市间修建高速公路，使得总成本最小？

例2.2 最短路问题（SPP-Shortest Path Problem）

一名货柜车司机奉命在最短的时间内将一车货物从甲地运往乙地. 从甲地到乙地的公路网纵横交错，因此有多种行车路线，这名司机应选择哪条线路呢？假设货柜车的运行速度是恒定的，那么这一问题相当于需要找到一条从甲地到乙地的最短路.

网络优化问题的例子

例2.3 运输问题(Transportation Problem)

某种原材料有*M*个产地，现在需要将原材料从产地运往*N*个使用这些原材料的工厂．假定*M*个产地的产量和*N*家工厂的需要量已知，单位产品从任一产地到任一工厂的的运费已知，那么如何安排运输方案可以使总运输成本最低？

例2.4 指派问题(Assignment Problem)

一家公司经理准备安排*N*名员工去完成*N*项任务，每人一项. 由于各员工的特点不同，不同的员工去完成同一项任务时所获得的回报是不同的. 如何分配工作方案可以使总回 报最大？

网络优化问题的例子

例2.5 中国邮递员问题(CPP-Chinese Postman Problem)

一名邮递员负责投递某个街区的邮件. 如何为他(她)设计一条最短的投递路线(从邮局出发，经过投递区内每条街道至少一次，最后返回邮局)？由于这一问题是我国管梅谷教授1960年首先提出的，所以国际上称之为中国邮递员问题.

例2.6 旅行商问题/货郎（担）问题

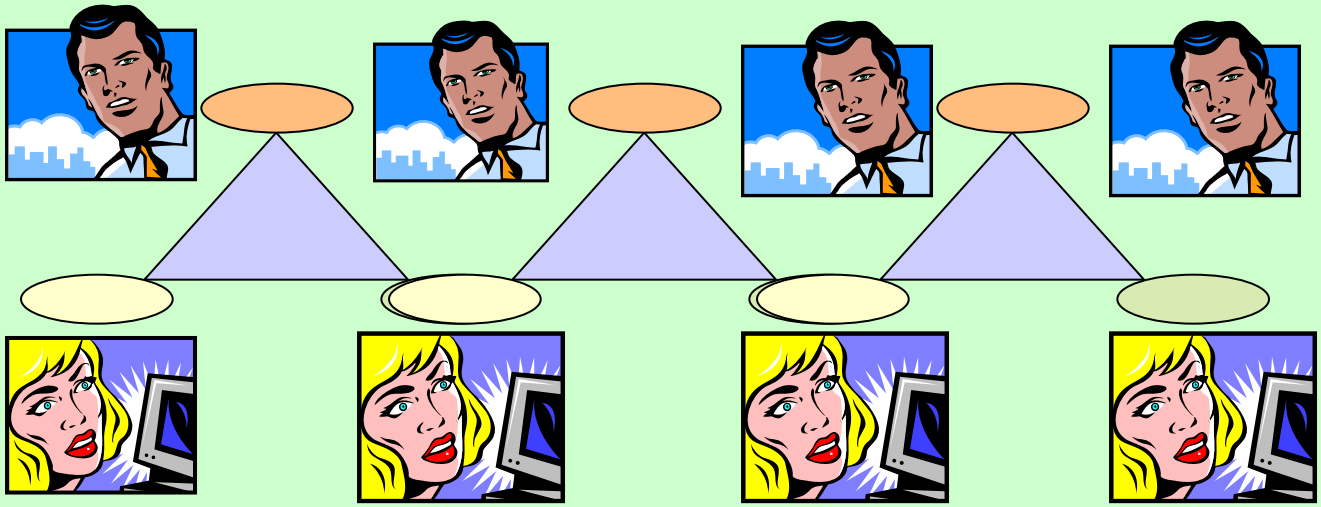
(TSP-Traveling Salesman Problem)

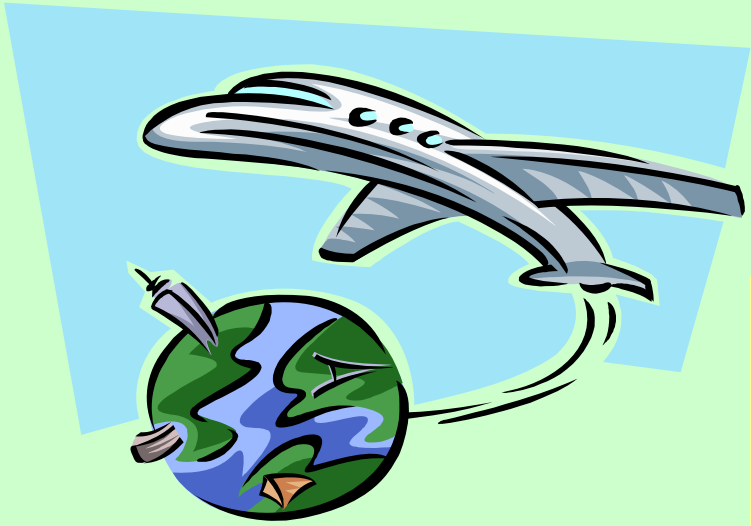
一名推销员准备前往若干城市推销产品. 如何为他(她)设计一条最短的旅行路线(从驻地出发，经过每个城市恰好一次，最后返回驻地)？这一问题的研究历史十分悠久，通常称之为旅行商问题.

网络优化问题的例子

例 稳定婚配问题(**Stable Marriage Problem**)

假设有n个男人和n个女人，每人都希望从n个异性中选择一位自己的配偶．假设每人都对n个异性根据自己的偏好进行了排序，以此作为选择配偶的基础．当给定一种婚配方案(即给每人指定一个配偶)后，如果存在一个男人和一个女人不是互为配偶，但该男人喜欢该女人胜过其配偶，且该女人喜欢该男人也胜过其配偶，则该婚配方案称为不稳定的．安排稳定的婚配方案的问题称为稳定婚配问题。





网络优化问题的例子

•特点：

（1）与图形有关，或易于用图形方式表示

（2）优化问题：从若干可能的安排或方案中寻求某种意义下的最优安排或方案

- 《网络优化》或《网络流》（Network Flows)

或《网络规划》（Network Programming)

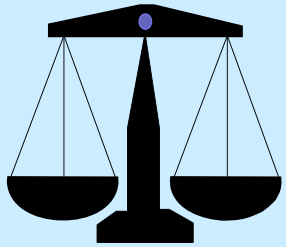
- 与图论课程的联系与区别

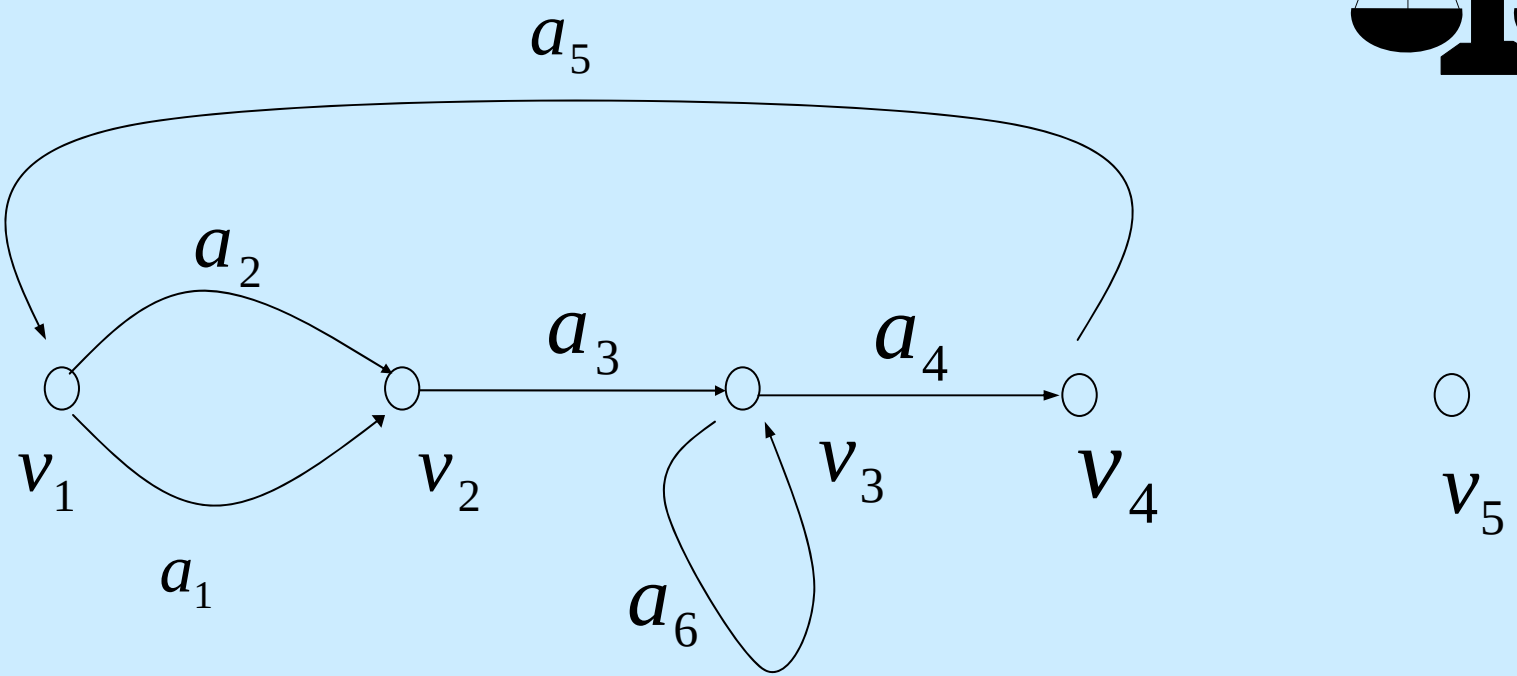
图与网络 – 定义

•定义2.1 一个有向图(Directed Graph 或 Digraph) G是由一个非空有限集合$V(G)$和$V(G)$中某些元素的有序对集合$A(G)$构成的二元组，记为有向图$G=(V(G), A(G))$. 其中$V(G)$ 称为图G的顶点集(Vertex Set)或节点集(Node Set)，$V(G)$中的每一个元素称为该图的一个顶点(Vertex)或节点(Node)； $A(G)$称为图G的弧集(Arc Set)，$A(G)$中的每一个元素(即$V(G)$中某两个元素的有序对(V_i, V_j)) 称为该图的一条从V_i到V_j的弧 (Arc). 在不引起混淆的情况下，记号$V(G)$和$A(G)$中也可以省略G，即分别记顶点集、弧集为V和A，而记有向图$G=(V, A)$.

如果对有向图G中的每条弧赋予一个或多个实数，得到的有向图称为赋权有向图.

图与网络 – 例





图$G=(V,\ A)$，其中顶点集$V=\{v_1,v_2,v_3,v_4,v_5\}$

弧集$A=\{a_1,a_2,a_3,a_4,a_5,a_6\}$

弧 $a_1=(v_1,v_2)$ $a_2=(v_1,v_2)$ $a_3=(v_2,v_3)$

$a_4=(v_3,v_4)$ $a_5=(v_4,v_1)$ $a_6=(v_3,v_3)$

图与网络 – 基本概念

- 弧的端点(Endpoint)，尾(Tail)，头(Head)；
- 顶点相邻(Adjacent)，邻居(Neighbor)；
- 弧与顶点关联 (Incident)，出弧(Outgoing Arc)，入弧(Incoming Arc)；
- 弧相邻 (Adjacent)；
- 环 (Loop)，孤立点(Isolated Vertex)；
- 顶点的度（Degree），出度，入度，奇点，偶点

没有环、且没有多重弧的图称为简单图(Simple Graph)

图与网络 – 子图

$G=(V,A)$，$V'\subseteq V, A'\subseteq A$，$G'=(V',A')$ 称为G的子图(Subgraph)

- 图的支撑子图 (Spanning Subgraph，又称生成子图)是包含G的所有顶点的子图

图的支撑子图一般是不唯一的

- 有向图中的途径(Walk)是该图的一些顶点 $i_1,i_2,\cdots,i_r$ 和弧 $a_1,a_2,\cdots,a_{r-1}$ 所组成的子图，这些顶点与弧可以交错排列成点弧序列 $i_1,a_1,i_2,a_2,\cdots,a_{r-1},i_r$ 其中 $a_k=(i_k,i_{k+1})$ 或 $a_k=(i_{k+1},i_k)(\forall k=1,2,\cdots,k-1)$

- 如果该序列中的所有弧都指向同一方向，即

$$a_k=(i_k,i_{k+1})(\forall k=1,2,\cdots,k-1)\quad,$$

则该途径称为有向途径(Directed Walk).

图与网络 – 路、圈

- 有向图中的路（Path)是该图的不包含重复顶点的途径.

方向与路的起点到终点的方向一致的弧称为路的前向弧(Forward Arc)；否则称为后向弧或反向弧(Backward Arc). P, P^+, P^-

- 有向图中的有向路(Directed Path)是该图的不包含重复顶点的有向途径，即不包含反向弧的路.

- 有向图的圈(Cycle) 是起点和终点重合，且不含其他重复顶点的途径. C, C^+, C^-

- 有向图中的有向圈(Directed Cycle)是该图的不包含反向弧的圈.

不包含有向圈的图称为无圈图(Acyclic Graph).

图与网络 – 路、圈

• 对于有向图中的两个顶点，如果在图中至少存在一条路把它们连接起来，则称这两个顶点是连通的(Connected). 如果图中任意两个顶点都是连通的，则称该图为连通的(Connected)；否则称为不连通的(Disconnected).

不连通图可以分解成一些连通分支的并，而连通图只有一个连通分支.

所谓连通分支(Component)，就是图中的极大连通子 图。

•如果有向图中从任意一个顶点出发，都存在至少一条有向路到达任意另一个顶点，则称该图为强连通的(Strongly Connected). 同样可以类似地定义强连通分支.

• 若 $S,T \subseteq V; S,T \neq \varnothing; S \cup T = V; S \cap T = \varnothing$ ，则称两个端点分别位于S，T的弧为一个割（cut），记为

$$[S，\ T] = \{(i,j) \in A \mid i \in S, j \in T\} \cup \{(j,i) \in A \mid i \in S, j \in T\}$$

图与网络 – 无向图

- 定义2.2 一个无向图(Undirected Graph) G是由一个非空有限集合V和V中某些 元素 的无序对集合E 构成的，即$G=(V, E)$. 其中 $V=V(G)$ 仍称为 图G的顶点集(Vertex Set)或 节点集(Node Set)，V中的每一个 元素 称为该图的 一个顶点(Vertex)或 节点(Node)；$E=E(G)$通常称为图G的边集合(Edge Set)，E中的每一个元素(即V 中某两个元素的无序对) 称为该图的一条边(Edge).

  边上赋权的无向图称为赋权无向图.

图与网络 – 两种特殊的图

• 若无向图G的任意两个顶点间有且只有一条边相连，则称其为完全图 (Complete Graph). n阶完全图记为K_n.

K_n的弧的数量为 $n(n-1)/2$

• 若图$G=(V, E)$的顶点集V可以表示成两个互不相交的非空子集S，T的并集，且使得G中每一条边的一个端点在S中，另一端点在T 中，则称G为二部图/二分图(Bibartite Graph). 当$|S|=p$，$|T|=q$，且S与T中任意两对顶点都相邻时，称G为完全二部图，记为$K_{p,q}$.

$K_{p,q}$的弧的数量为 pq

图$G=(V, E)$是二部图的充分必要条件是它不含奇长度的圈。

类似地，也可以定义有向的完全图和有向的完全二部图.

图与网络 – 树

•树：图G=(V, T)是一个连通且无圈的图，则称其为树 (Tree).

• 森林：点不相交的树之集合.

命题：设G=(V, E)是一个图，则下述条件等价：

（1） G是一棵树；

（2） G是连通的且有|V|-1条边（边数比顶点数少1）；

（3） G无圈，但若加到G里任一条边就产生唯一的圈；

•赋权图：图G=(V, E)连同于E映射到 在Z(或Z^+ 或Z^-）的一个函数w.

图与网络 – 网络

•网络：图$N=(s,t,V,A,b)$是一个赋权有向图(V,A)，使得有一个点 $s\in V$，其入次为0和一个点 $t\in V$，其出次为0，并且对每一弧 $(u,v)\in A$，有一个容量$b(u,v)\in Z^+$， 则称$N=(s,t,V,A,b)$是一个网络， s为发点， t为收点.

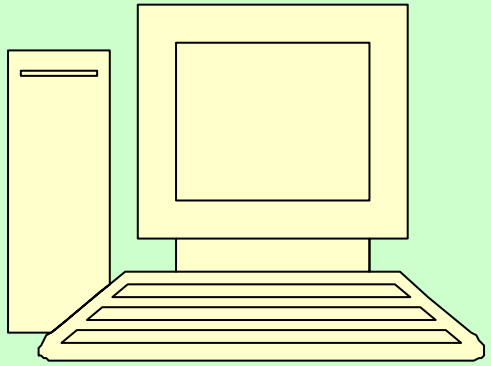
N 中的一个流f 是$R^{|A|}$中一个向量（每一个弧 $(u,v)\in A$ 对应一个分量 $f(u,v)$，使得它满足：

（1） 对一切 $(u,v)\in A$ 有 $0\le f(u,v)\le b(u,v)$

（2）对一切 $v\in V-\{s,t\}$，有 $\sum_{(u,v)\in A} f(u,v)=\sum_{(v,u)\in A} f(u,v)$

•f 的值定义为： $|f|=\sum_{(s,u)\in A} f(s,u)$

图与网络的数据结构



$G=(V，A)$是一个简单有向图

$|V|=n，|A|=m$

邻接矩阵(Adjacency Matrix)表示法

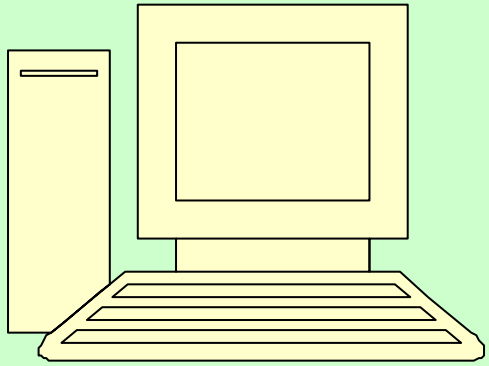
图$G=(V，A)$的邻接矩阵C是如下定义的：C是一个 $n\times n$ 的0-1矩阵，即

$$C=(c_{ij})_{n\times n}\in\{0,1\}^{n\times n},\qquad c_{ij}=\begin{cases}0, & (i,j)\notin A,\\ 1, & (i,j)\in A.\end{cases}$$

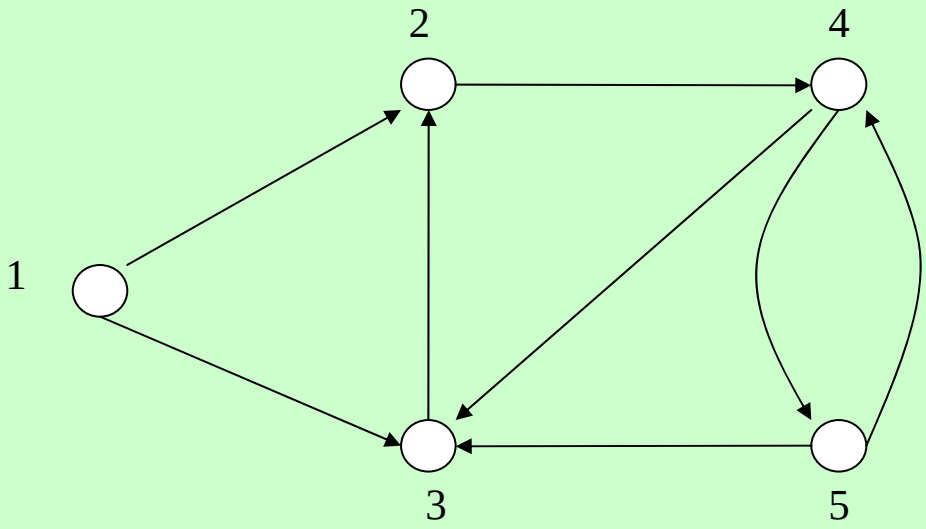
每行元素之和正好是对应顶点的出度， 每列元素之和正好是对应顶点的入度.

在邻接矩阵的所有元素中，只有m个为非零元.

图与网络的数据结构



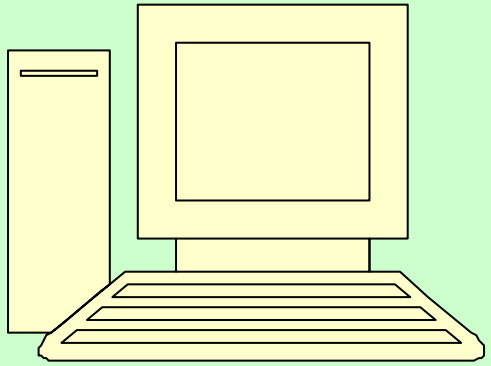
邻接矩阵表示法



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

对于网络中的权，也可以用类似邻接矩阵的矩阵表示. 只是此时一条弧所对应的元素不再是1，而是相应的权而已. 如果网络中每条弧赋有多种权，则可以用多个矩阵表示这些权.

图与网络的数据结构



$G=(V，A)$是一个简单有向图

$|V|=n，|A|=m$

关联矩阵(Incidence Matrix)表示法

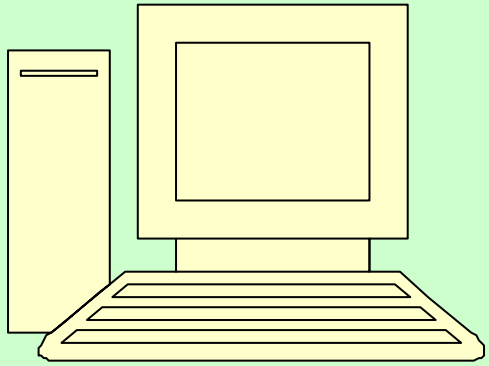
图$G=(V，A)$的关联矩阵C是如下定义的：C是一个$n\times m$ 的矩阵，即

$$B=(b_{ik})_{n\times m}\in\{-1,0,1\}^{n\times m},\quad b_{ik}=\begin{cases}1, & \exists j\in V,k=(i,j)\in A,\\ -1, & \exists j\in V,k=(j,i)\in A,\\ 0, & \text{其他。}\end{cases}$$

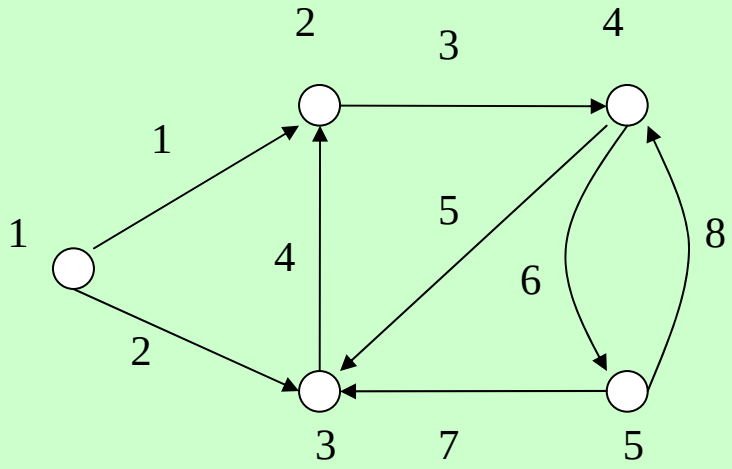
每行元素1的个数正好是对应顶点的出度， 每行元素-1的个数正好是对应顶点的入度.

在关联矩阵的所有元素中，只有$2m$个为非零元.

图与网络的数据结构



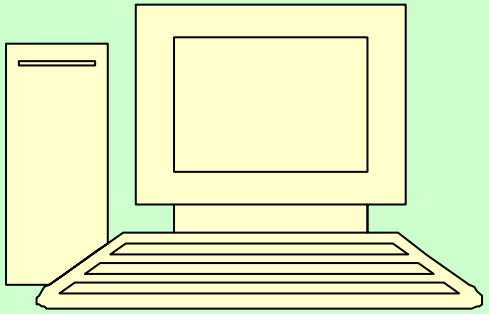
关联矩阵表示法



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

如果网络中每条弧赋有一个权，我们可以把关联矩阵增加一行,把每一条弧所对应的权存贮在增加的行中.如果网络中每条弧赋有多个权，我们可以把关联矩阵增加相应的行数，把每一条弧所对应的权存贮在增加的行中.

图与网络的数据结构



$G=(V，A)$是一个简单有向图

$|V|=n$，$|A|=m$

弧表（Arc List)表示法

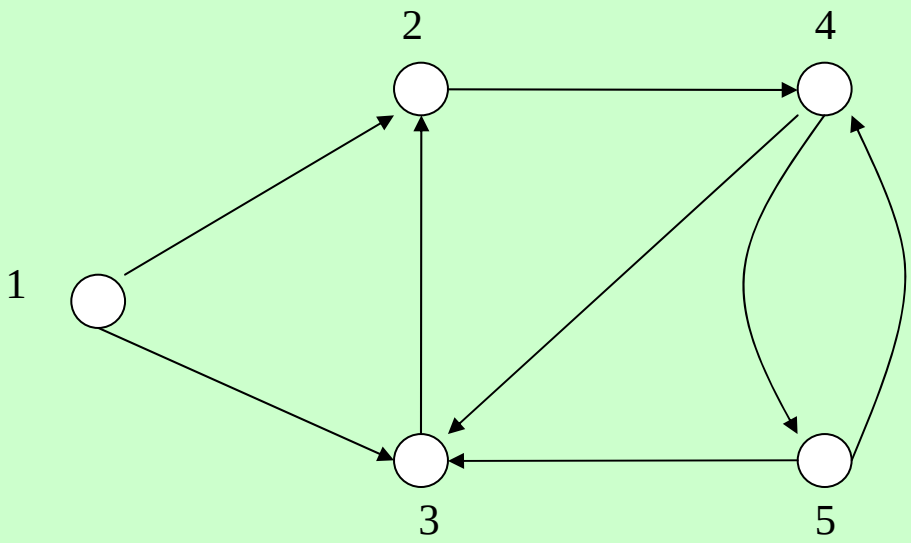
所谓图的弧表，也就是图的弧集合中的所有有序对.

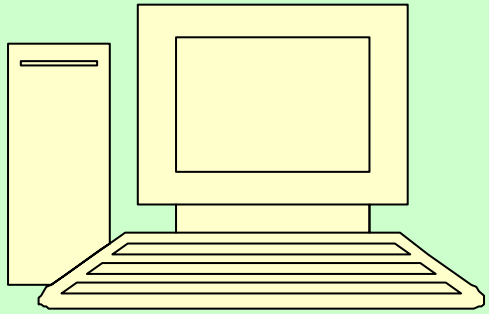
弧表表示法直接列出所有弧的起点和终点.

特点:

共需$2m$个存贮单元，当网络比较稀疏时比较方便.

图与网络的数据结构

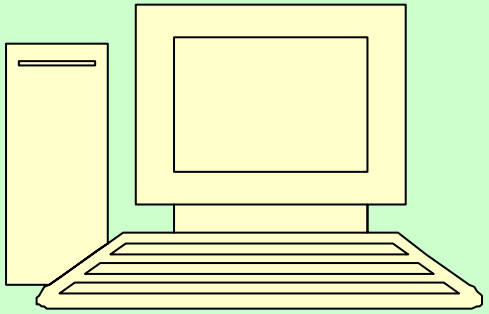




弧表表示法

起点	1	1	2	3	4	4	5	5
终点	2	3	4	2	3	5	3	4
权 (例)	8	9	6	4	0	3	6	7

图与网络的数据结构



$G=(V，A)$是一个简单有向图

$|V|=n$，$|A|=m$

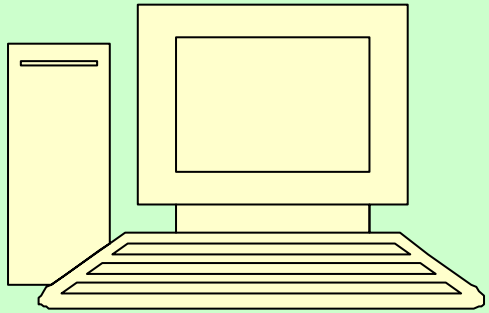
邻接表 (Adjacency Lists)表示法

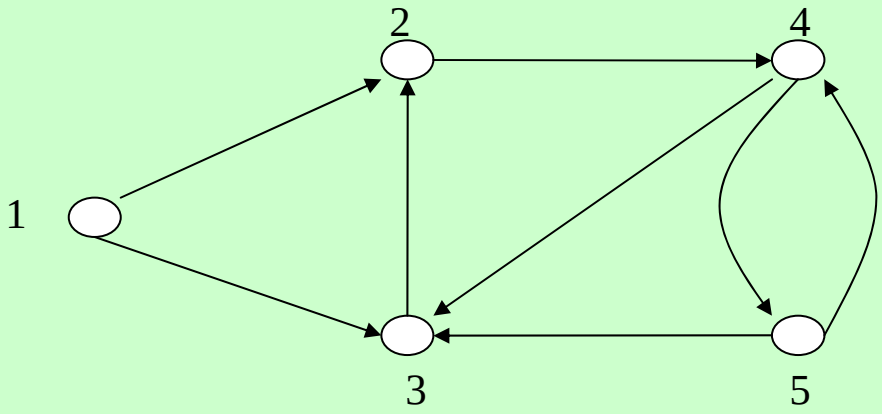
图的邻接表是图的所有节点的邻接表的集合;

而对每个节点，它的邻接表就是它的所有出弧.

对于有向图$G=（V，A）$，一般用$A（i）$表示节点i的邻接表，即节点i的所有出弧构成的集合或链表（实际上只需要列出弧的另一个端点，即弧的头）.

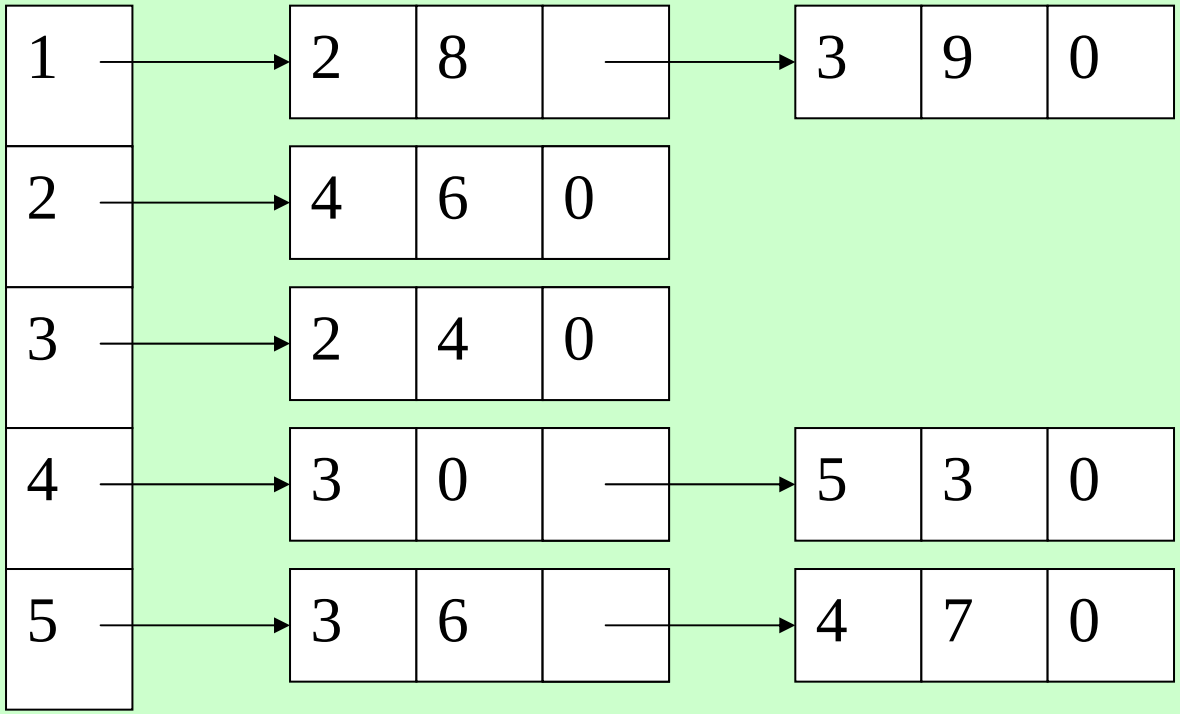
图与网络的数据结构





1.3.4 邻接表表示法

单向链表(指针数组)



A(1)={2,3}

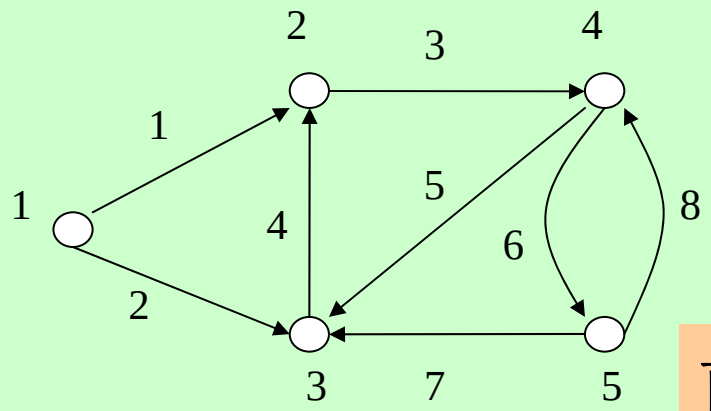
A(2)={4}

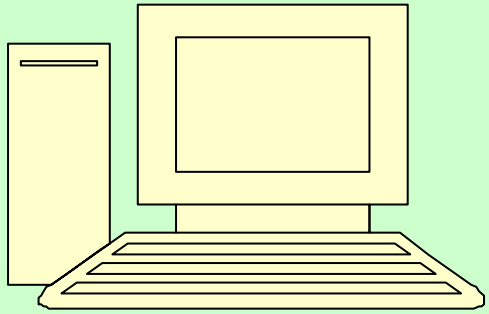
A(3)={2}

A(4)={3,5}

A(5)={3,4}

图与网络的数据结构





星形（Star）表示法

前向星形(Forward Star)表示法

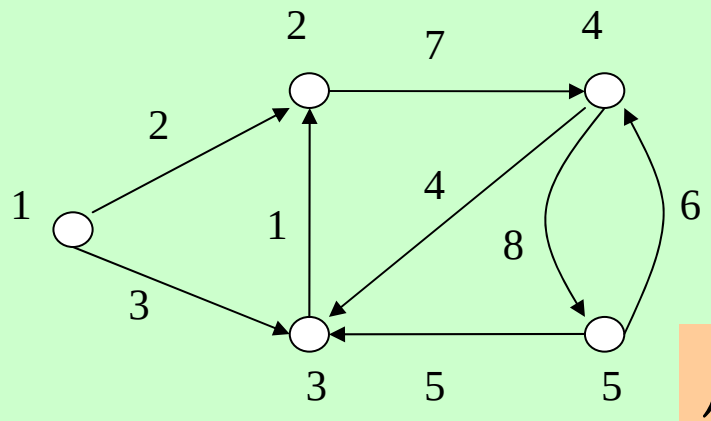
节点对应的出弧的起始地址编号数组（记为*point*）为

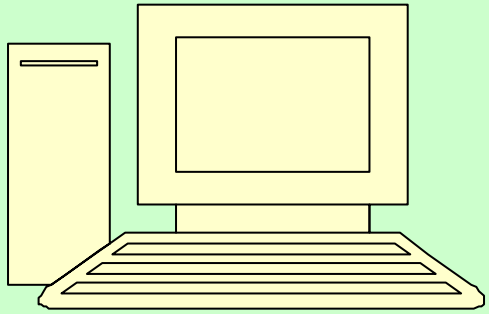
节点号*i*	1	2	3	4	5	6
起始地址*point*(*i*)	1	3	4	5	7	9

记录弧信息的数组为

弧编号	1	2	3	4	5	6	7	8
起点	1	1	2	3	4	4	5	5
终点	2	3	4	2	3	5	3	4
权	8	9	6	4	0	3	6	7

图与网络的数据结构





星形（Star）表示法

反向星形(Reverse Star)表示法

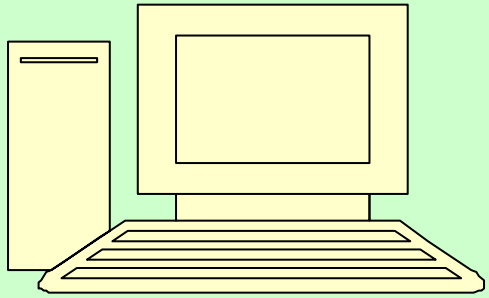
节点对应的入弧的起始地址编号数组（记为*rpoint*）为

节点号*i*	1	2	3	4	5	6
起始地址*rpoint*(*i*)	1	1	3	6	8	9

记录弧信息的数组为

弧编号	1	2	3	4	5	6	7	8
终点	2	2	3	3	3	4	4	5
起点	3	1	1	4	5	5	2	4
权	4	8	9	0	6	7	6	3

图与网络的数据结构



几点说明

（1）星形表示法和邻接表方法常用. 星形表示法的优点是占用的存贮空间较少;邻接表方法增加或删除一条弧所需的计算工作量很少

（2）当网络不是简单图，而是具有平行弧(即多重弧)时，邻接矩阵法是不能采用的. 其他方法则可以推广到可以处理平行弧的情形.

（3）无向图处理类似. 如无向图的关联矩阵只含有元素0和+1. 在邻接表和星形表示中，每条弧会被存贮两次;反向星形表示显然是没有必要的，等等.

2.3 几种规划之间的关系

- 非线性规划
- 凸规划
- 线性规划

- 非线性规划问题：

$$\min f(x)$$
$$\text{s.t. } g_i(x) \geq 0, \quad i = 1, 2, \cdots, m$$
$$\mathrm{h}_j(x) = 0, \quad \mathrm{j} = 1, 2, \cdots, p$$
$$\text{其中} f, g_i, h_j \in R^n.$$

- 凸规划问题：

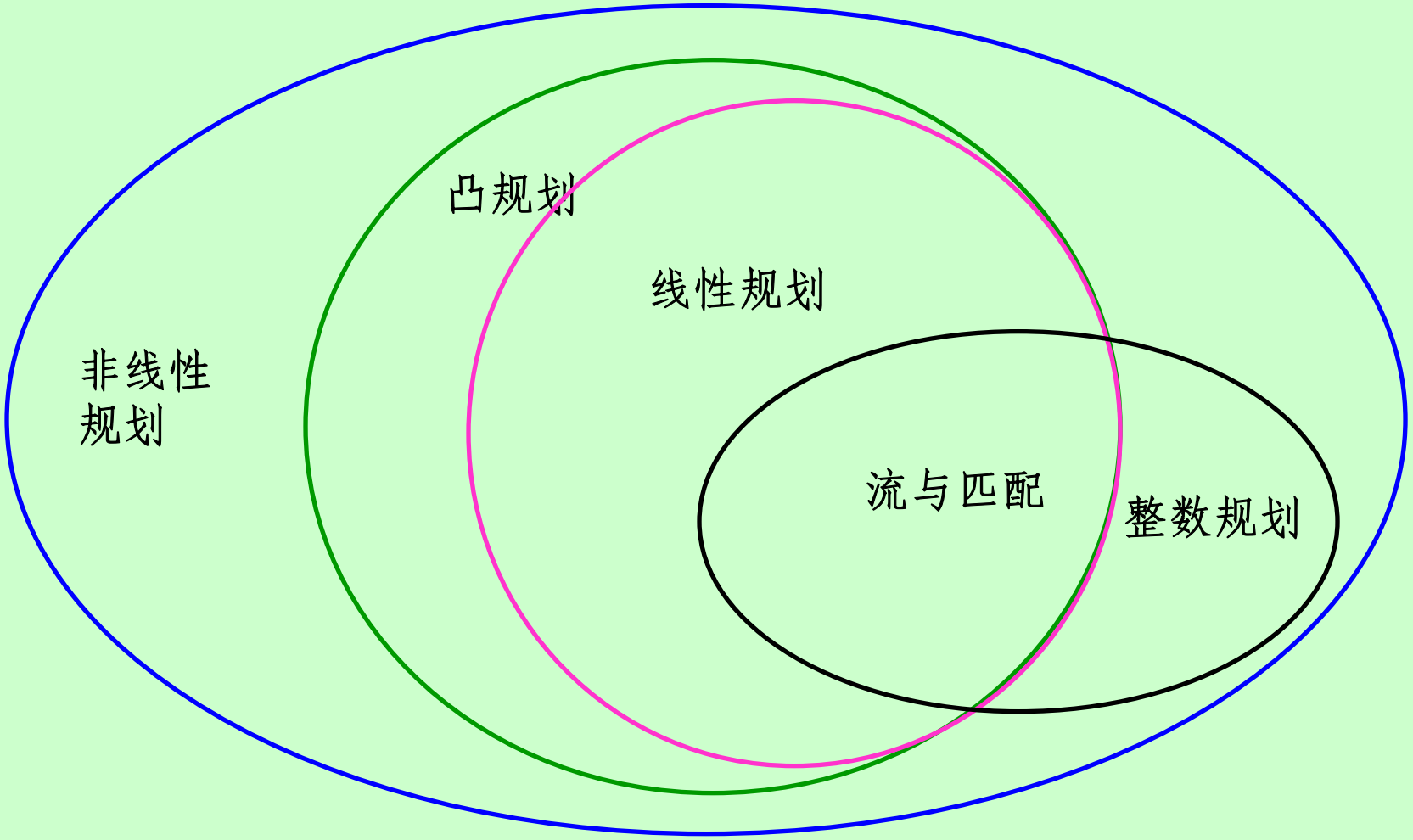
$$f\text{凸}, g_i\text{凹}, h_j\text{线性}.$$

- 线性规划问题：

$$f, g_i, h_j\text{都是线性}.$$

线性规划问题是一个组合优化问题：

- 解某些线性规划问题，如流和匹配问题等，可比解一般的线性规划问题更有效；
- 这些问题又与一些明显困难的问题紧密相关，如：最短路问题有O(n^2)的求解算法，但TSP（旅行商问题）却是一个N-P完备问题。


非线性
规划
凸规划
线性规划
流与匹配
整数规划

2.4 最优化问题

最优化问题的实例

定义：所谓最优化问题的一个实例（或例子）是一对元素 (F, C)，其中 F 是一个集合或可行点的定义域，C 是费用函数或映射：

$$C : F \to R^1$$

问题是求一个 $f \in F$，使得对一切 $y \in F$ 有 $C(y) \geq C(f)$.

这样一个点 f 称为给定实例的整体（或全局）最优解。

定义：一个所谓的最优化问题，就是它的一些实例的集合 I。

最优化问题

例

例2.7　0-1背包问题(knapsack problem)

给定n个容积分别为a_i，价值分别为c_i的物品. 设有一个容积为b的背包，如何以最大的价值装包？用数学规划模型表示为：

$$\max \sum_{i=1}^{n} c_i x_i$$

$$s.t. \quad \sum_{i=1}^{n} a_i x_i \le b$$

$$x_i \in \{0,1\}, \ i = 1,\ldots,n.$$

D= {0,1}^{n}

例2.8 装箱问题(Bin Packing)

以尺寸为1的箱子装进给定的n个尺寸不超过1的物品，如何使所用的箱子个数最少?

最优化问题

例

例2.9 整数线性规划(Integer Linear Programming)

$$\text{(IP)} \quad \begin{aligned} &\min c^T x \\ &s.t.\ \ Ax = b \\ &x \geq 0,\ \ x \in Z^n \end{aligned}$$

•我们假设线性整数规划的参数（约束矩阵和右端项系数）都是整数(或有理数).

•许多组合优化问题可以用整数规划模型表示,但有时不如直接用自然语言描述简洁

最优化问题

例

例2.10 旅行商问题 (TSP)

•TSP的一个实例，是指给定了一个整数 $n > 0$，和 n 个城市中每对城市之间的距离 $(d_{ij})_{n*n}$。一条环游就是经过每一个城市恰好一次的闭路。

•问题：找一条总长度最短的环游。

•取 F = {n 个对象的所有循环排列 π}

• $\pi(j)$：通过城市j后直接去城市 $\pi(j)$。那么一个循环排列就表示一条环游。费用函数C把 π 映射为：

$$\sum_{j=1}^{n} d_{j\pi(j)}$$

最优化问题

例

例2.11最小支撑树(MST: Minimal Spanning Tree)

- MST的一个实例，是指给定了一个整数 $n > 0$，和 $n*n$ 对称矩阵距离 $(d_{ij})_{n*n}$。
- 问题：求一棵n 个节点的支撑树，使得其边的长度之和最小。
- 取 F = {点集V={1, 2, …, n}上的所有支撑树(V, E)}

$$C: \quad (V,E) \rightarrow \quad \sum_{(i,j)\in E} d_{ij}$$

例2.12线性规划(LP)

2.5 邻 域

$f \in F$ 把某种意义上“靠近”于 f 的点之集合定义为 f 的邻域 $N(f)$

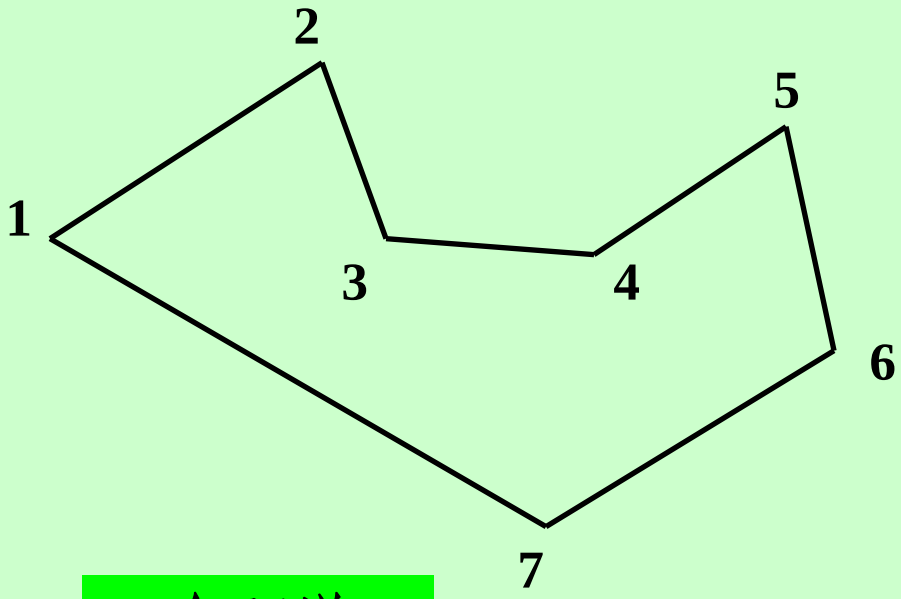
定义：给定一个诸实例为**(*F*,*C*)**的最优化问题，一个邻域就是对每一个实例确定的映射：

$$N: F \longrightarrow 2^F$$

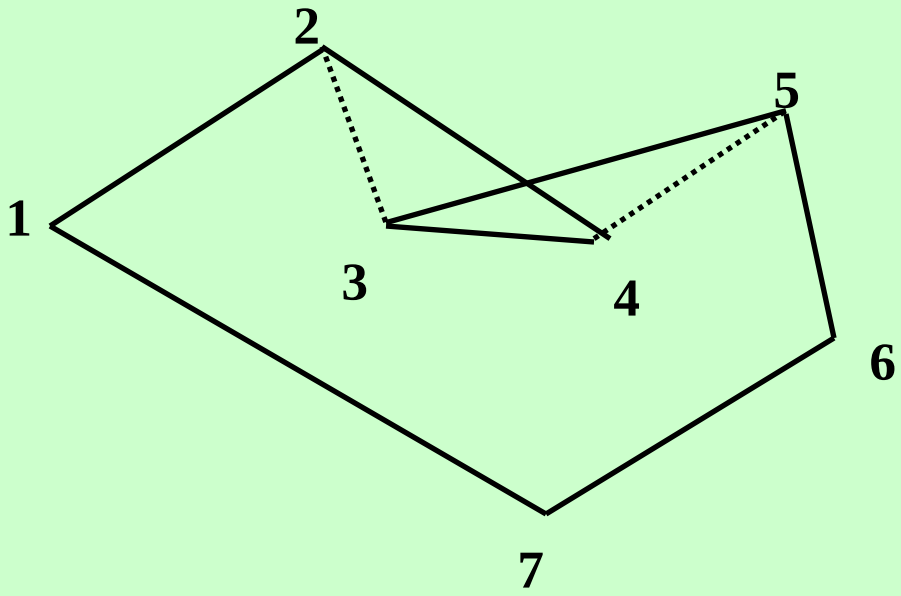
在许多组合问题中， **N**的选取可能主要依赖于***F***的结构。

例：在**TSP**中，一个重要的邻域是2-交换邻域:

$N_2(f)=\{g: g\in F$ 且能按下述方式由f得到：从f中去掉两条边，然后用另外两条边去代替$\}$



一个环游f



另一个环游 $g\in N_2(f)$

k-交换邻域（最多可以交换k条边）

例：在**MST**中，一个重要的邻域是：

$N(f)=\{g:g\in F$ 且能按下述方式由**f**得到：加一条边到**f**里产生一个圈，再去掉圈上的一条边**}**

例：在LP里，可以定义邻域：

$$N_{\varepsilon}(x)=\{y:Ax=b,y\geq 0\text{且}\,\|\,y-x\,\|\leq\varepsilon\}$$

局部最优与整体最优

定义： 给定一个最优化问题的实例**(F,C)**和一个邻域 $f \in F$，如果对一切 $g \in N(f)$ 有 $c(f) \le c(g)$ 则称**f**为关于**N**的局部最优解。

定义： 给定一个可行集为**F**, 邻域为**N**的最优化问题，如果 $f \in F$ 是关于**N**的局部最优解，那么**f**也是整体最优解，则称邻域**N**是精确的.

例：在TSP里，N_2不是精确的，而N_n是精确的，其中n是城市的数目.

凸集与凸函数

凸集：称 $S \subseteq R^n$ 为一个凸集，如果 $\forall x, y \in S, 0 \le \lambda \le 1,$ 必有

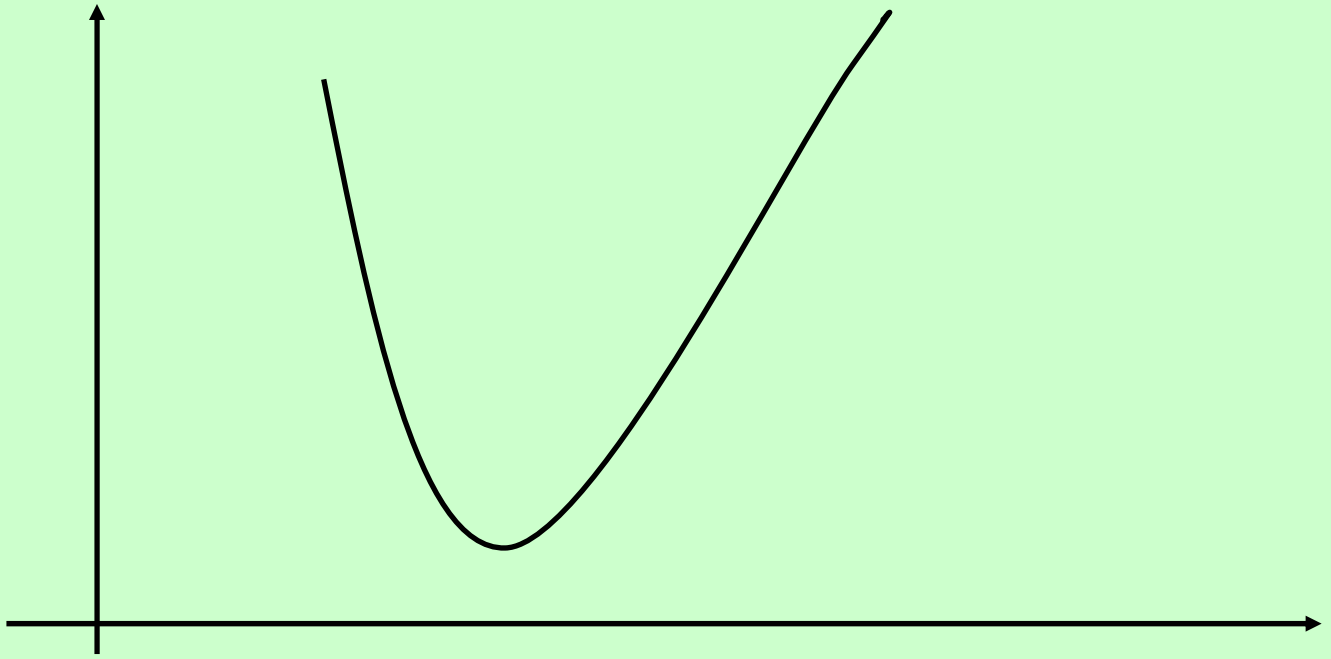
$$\lambda x + (1-\lambda) y \in S$$

凸函数： $S \subseteq R^n$ 为凸集，称 $f : S \to R^1$ ，为凸函数，

如果 $\forall x, y \in S, 0 \le \lambda \le 1,$ 必有

$$f\left(\lambda x + (1-\lambda)\right) y \le \lambda f(x) + (1-\lambda) f(y)$$

凹函数： $-f(x)$ 在 S 上为凸函数，则称 $f(x)$ 凹函数。



定理： 设**(F,c)**是一个最优化问题的一个实例，其中 $F \subseteq R^n$ 是一个凸集，**c**是一个凸函数，那么对 $\forall \varepsilon > 0$, 由欧氏距离定义的邻域

$$N_\varepsilon(x) = \{y : y \in F \text{且} \| y - x \| \leq \varepsilon\}$$

是精确的.