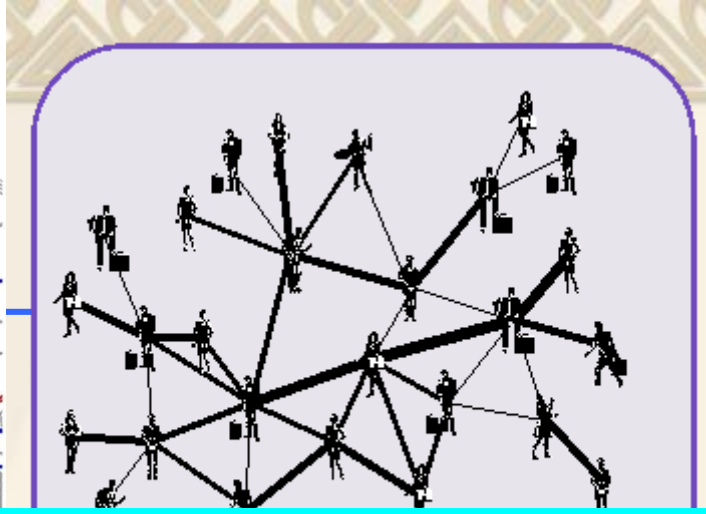
第六讲

网络优化

主讲教师： 朱建明

Email: jmzhu@ucas.ac.cn

Homepage: http://people.gucas.ac.cn/~jianming



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

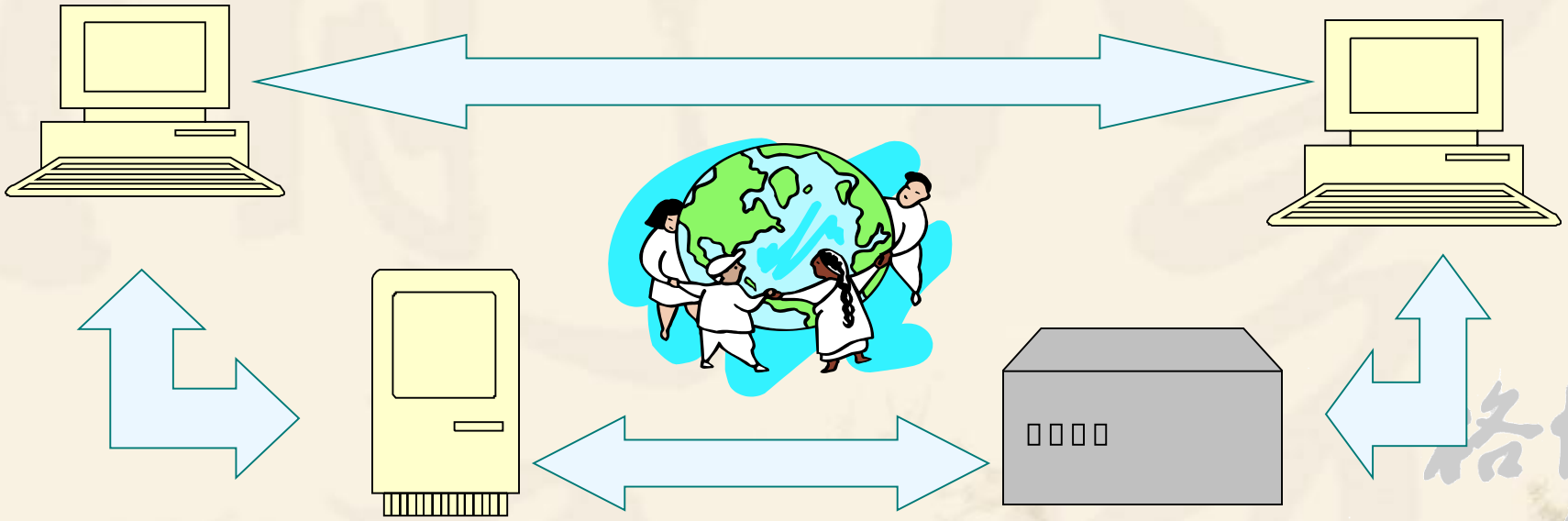
分类

网络优化

网络应用　　网络结构设计

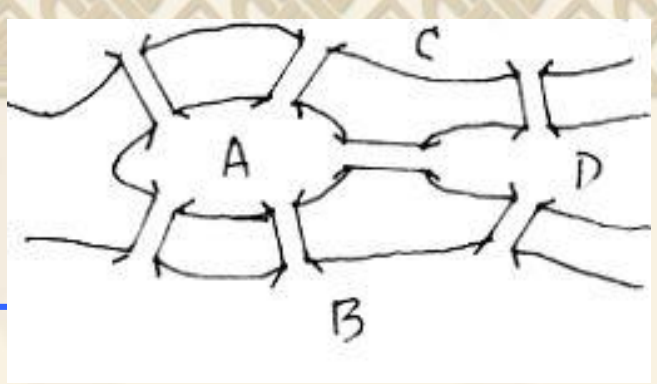
网 络 优 化 简 介

- 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

哥尼斯堡七桥问题



- 普鲁士的哥尼斯堡始建于1254年。
- 整个城市通过七座桥连接它的四个区域（标号为：A，B，C和D）。
- 是否可以找到一条到达每个区域并且通过每座桥恰好一次的游程？每个区域上的到达次数不受限制。

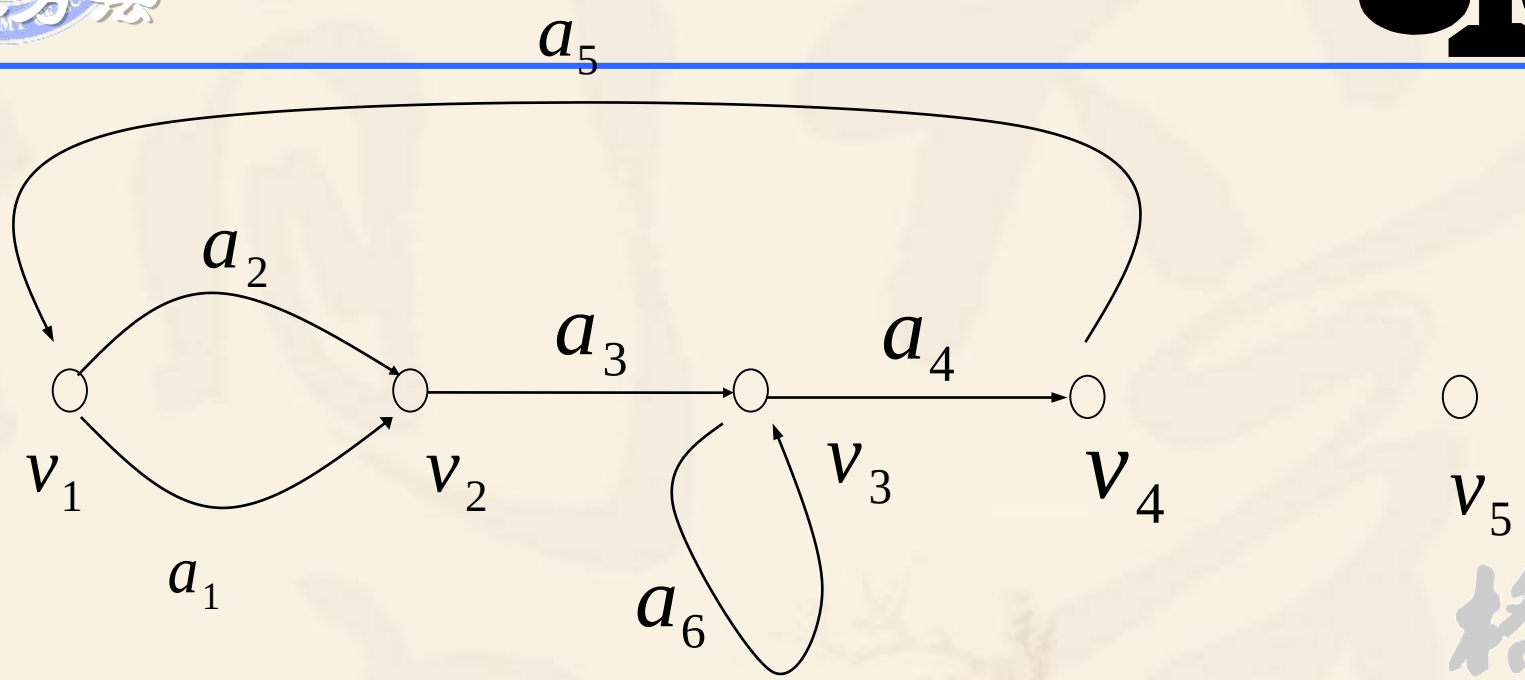
提纲

- ✓网络基本概念
- ✓最小生成树问题
- ✓最短路径问题
- ✓网络流问题
- ✓网络优化的其他应用

网络基本概念

- 无向图（Graph）G=(V,E)；
- 有向图(Directed Graph 或 Digraph) G=(V,A).
  - V称为图G的顶点集(Vertex Set)或节点集(Node Set)
  - A称为图G的弧集(Arc Set)
- 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**有向图或有向网络，简称为网络(Network).

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

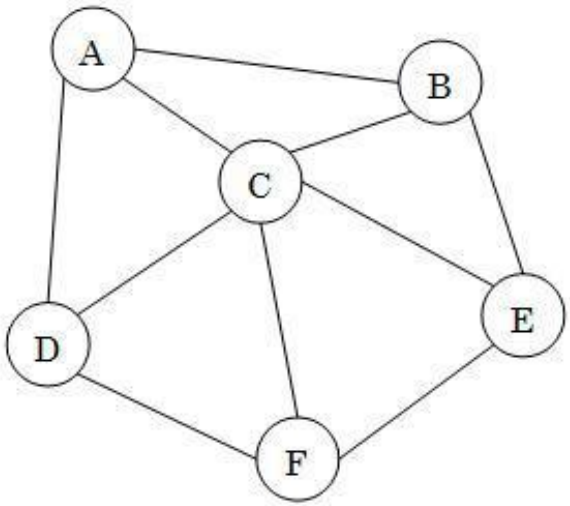
$a_4=(v_3,v_4)$ $a_5=(v_4,v_1)$ $a_6=(v_3,v_3)$

网络基本概念

- 弧的端点(Endpoint)，尾(Tail)，头(Head)；
- 环 (Loop)，孤立点(Isolated Vertex)；
- 重边(multipleedges)；
- **没有环、且没有重边的图称为简单图(Simple Graph)**
- 顶点的度（Degree）；
- 连通图(connected graph)；

网络基本概念

- ✓ 途径(Walk)：点边交错序列；
- ✓ 迹(Trail)：边不重复的途径；
- ✓ 路(path)：顶点不重复的迹；
- ✓ 闭途径，闭迹；
- ✓ 圈(cycle)：中途点不重复的闭迹；



无向图 G

最小生成树问题

例　公路连接问题

某一地区有n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i和j之间修建高速公路的成本（费用）w_{ij} (>0), 那么应任何决定在哪些城市间修建高速公路，使得总成本最小？

✓ 高速公路网正好构成这个完全图G的一个连通的支撑子图T.

✓ 为了使得总建设成本最小, 该子图必须是无圈的

无圈连通图称为树，上面这样一类问题称为最小树问题.

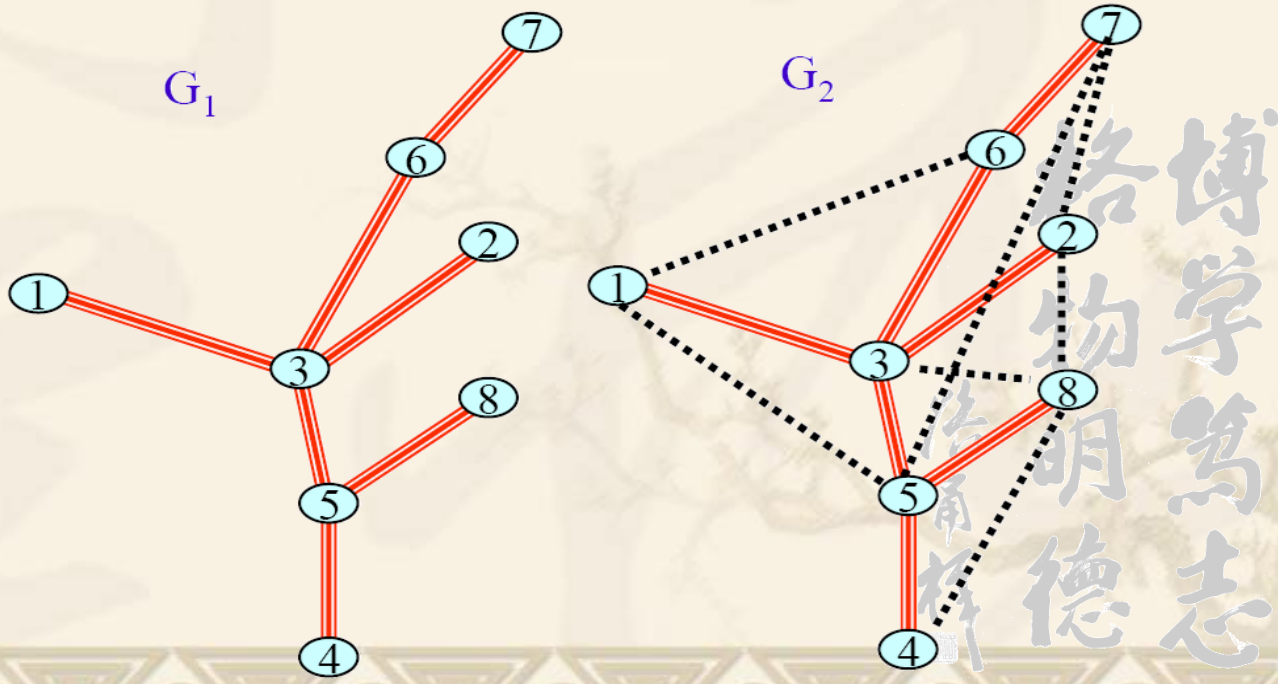
Spanning tree

Definition

A tree is a connected graph without cycles (i.e., acyclic). A spanning tree of a graph is a subgraph that is a tree and contains all the nodes of the original graph.

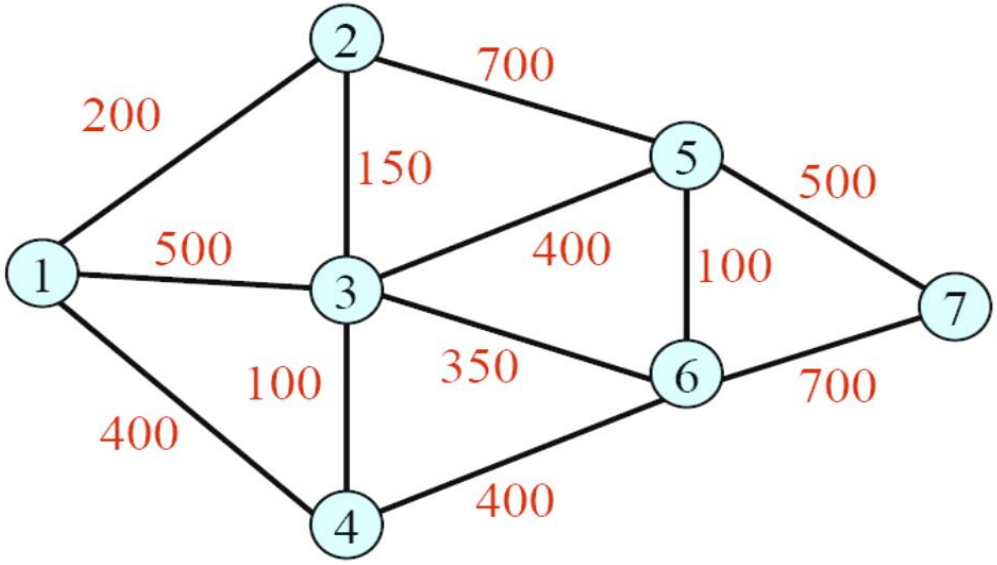
Example

- Graph G_1 is a tree.
- It is also a spanning tree of graph G_2.



Minimum spanning tree

Given a weighted graph $G = (V, E)$, find a spanning tree $T = (V, E_1)$ of G with the minimal total weight of edges in E_1. (E.g., design of communication network.)

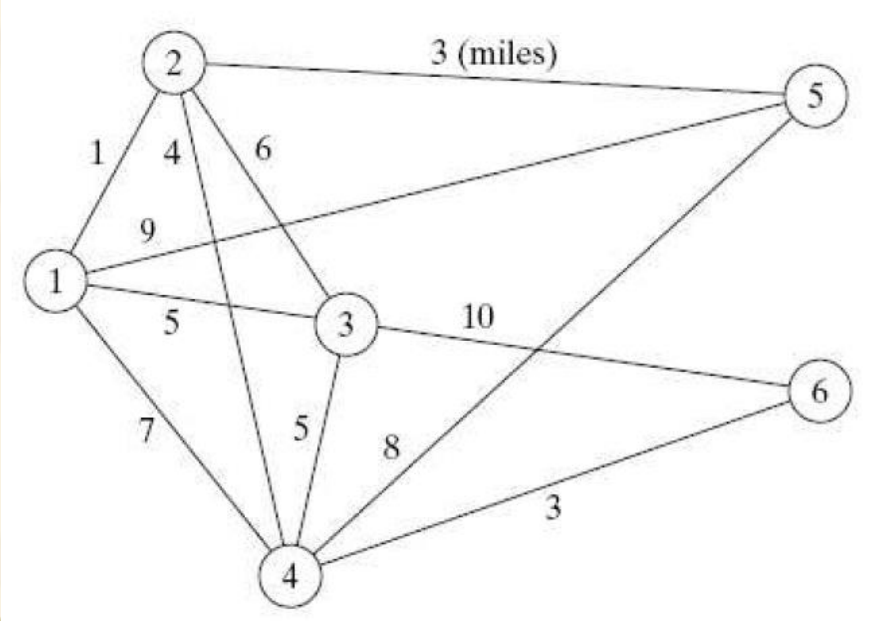


Algorithms

- ✓ Kruskal 算法(1956)
  - 基本思想：每次将一条权最小的边加入子图T 中，并保证不形成圈. （避圈法）
- ✓ Prim 算法（1957）
  - 基本思想：不断扩展一棵子树T=（S，$E0$），直到S包括原网络的全部顶点，得到最小树 T. （边割法）

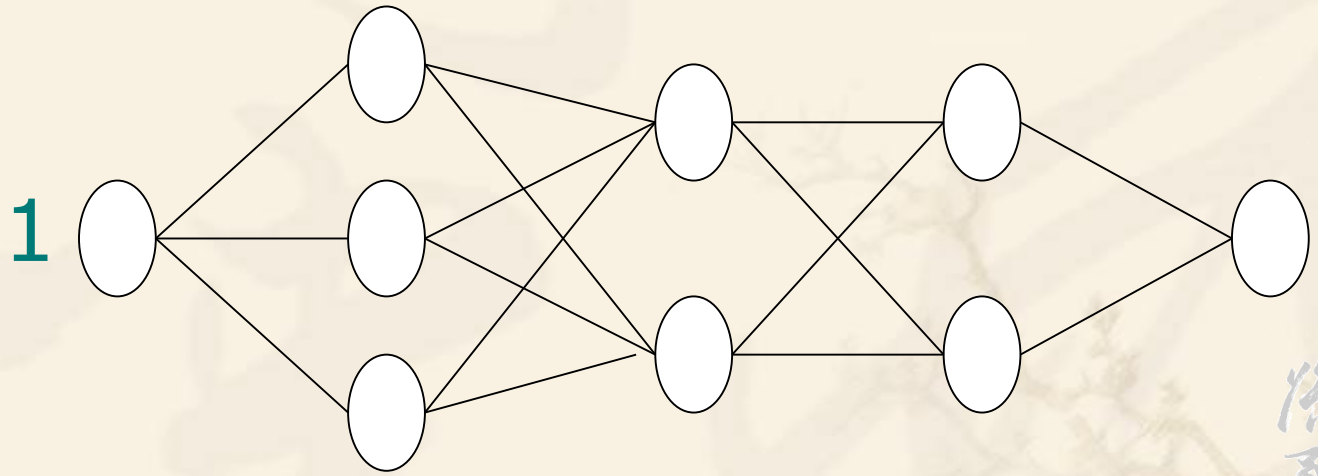
应用

- 某有线电视公司正计划为五个新的居民区提供有线电视服务，图中给出了小区之间可以铺设电缆的情况以及相应的距离。问题是确定最经济的电缆铺设方案，使得五个小区可以连接起来。



最短路问题(Shortest path problem)

- ✓ Given a weighted graph $G = (V,A)$ with n vertices and an $n \times n$ weight matrix $C = [c_{ij}]$. Find the shortest path from vertex 1 to vertex n.

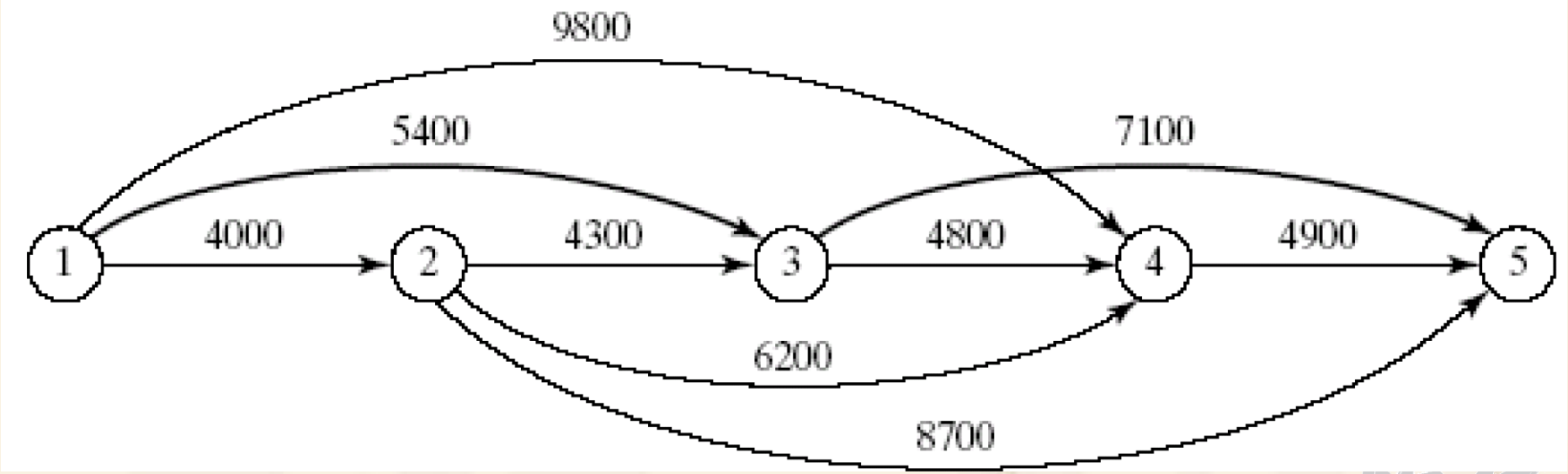


应用实例-设备更换问题

- ✓ RentCar公司开展了一项4年一周期的汽车更换政策。在每一年的开始，消费者都可以选择继续使用购买的汽车，也可以选择更换汽车。一辆汽车最少使用1年，最多使用3年，下表给出了汽车更换的费用，它取决于汽车购买购买的时间和使用的年限。

	给定使用年限的更换费用 (美元)		
开始购买汽车的年数	1	2	3
1	4000	5400	9800
2	4300	6200	8700
3	4800	7100	-
4	4900	-	-

设备更换问题网络模型



Dijkstra算法

- (这里描述的是从节点1开始到各点的dijkstra算法，其中Wa->b表示a->b的边的权值，d(i)即为最短路径值)

  1. 置集合S={2,3,...n}, 数组d(1)=0, d(i)=W1->i(1,i之间存在边) or +无穷大(1.i之间不存在边)

  2. 在S中，令d(j)=min{d(i),i属于S}，令S=S-{j}，若S为空集则算法结束，否则转3

  3. 对全部i属于S,如果存在边j->i，那么置d(i)=min{d(i), d(j)+Wj->i}，转2

算法流程举例

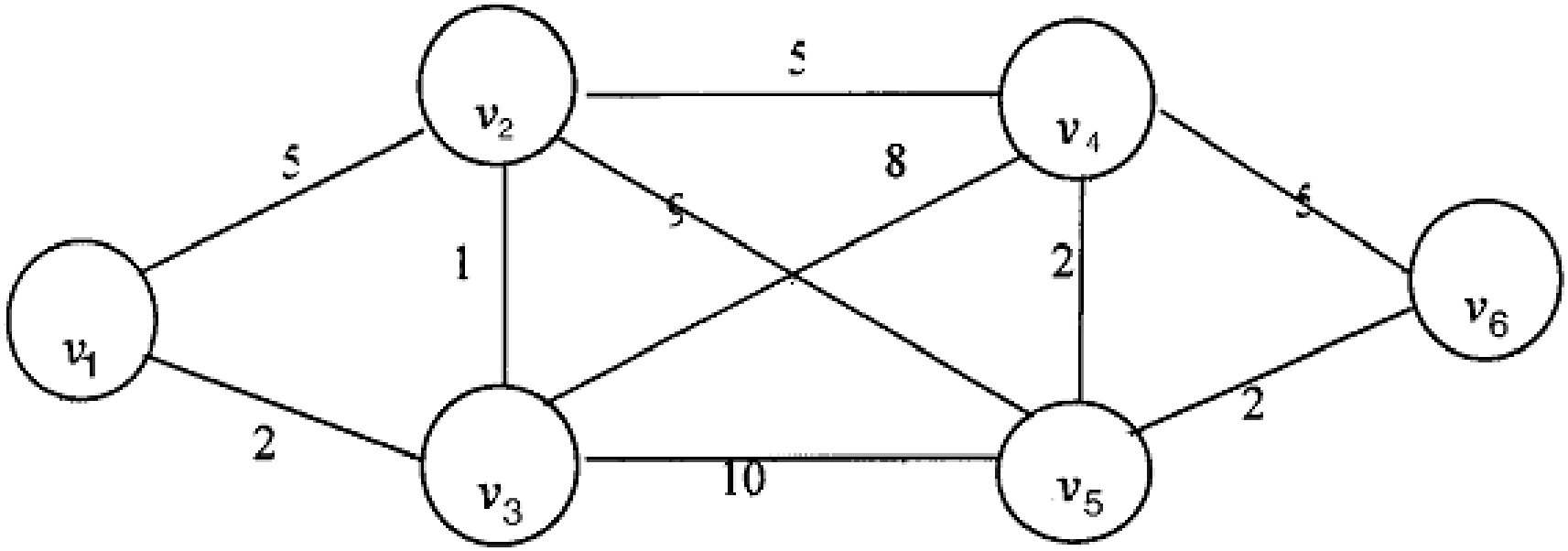
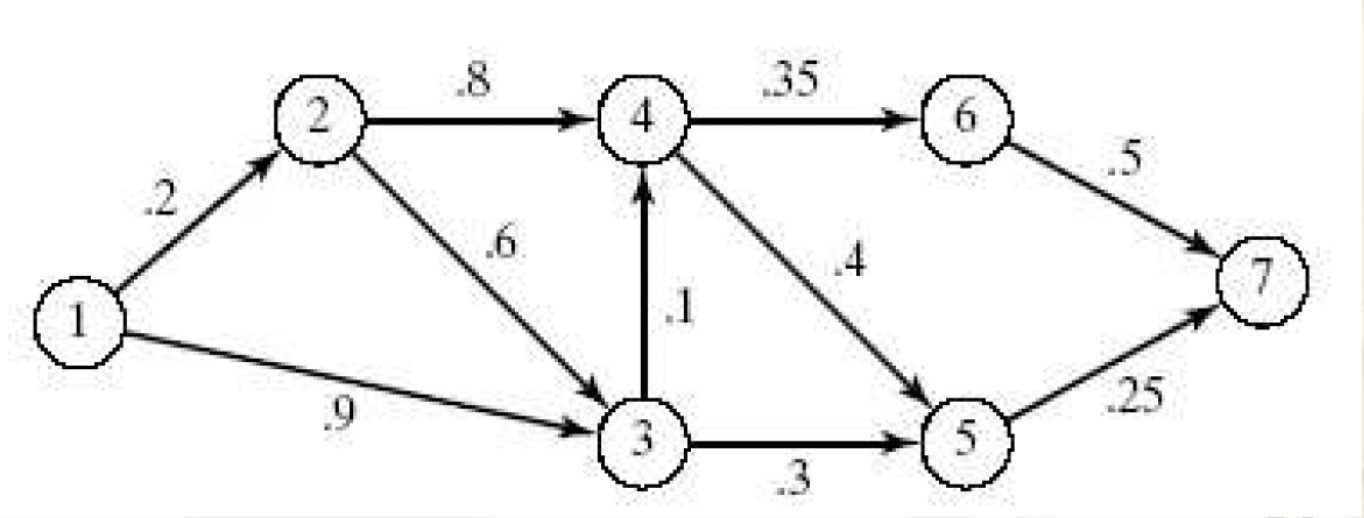


图 1

应用举例-最可靠路径

- 我们每天开车上班时，聪明人能够通过网络分析找到最短的行程。不幸的是在这样的路上行驶往往会被限速，如果你超速的话就会被警察罚款，所以最短的路径不一定是最好的选择。因此需要选择最大概率 不被警察扣罚的路径。



应用举例-最可靠路径（续）

- ✓ 不被扣罚的概率是指：

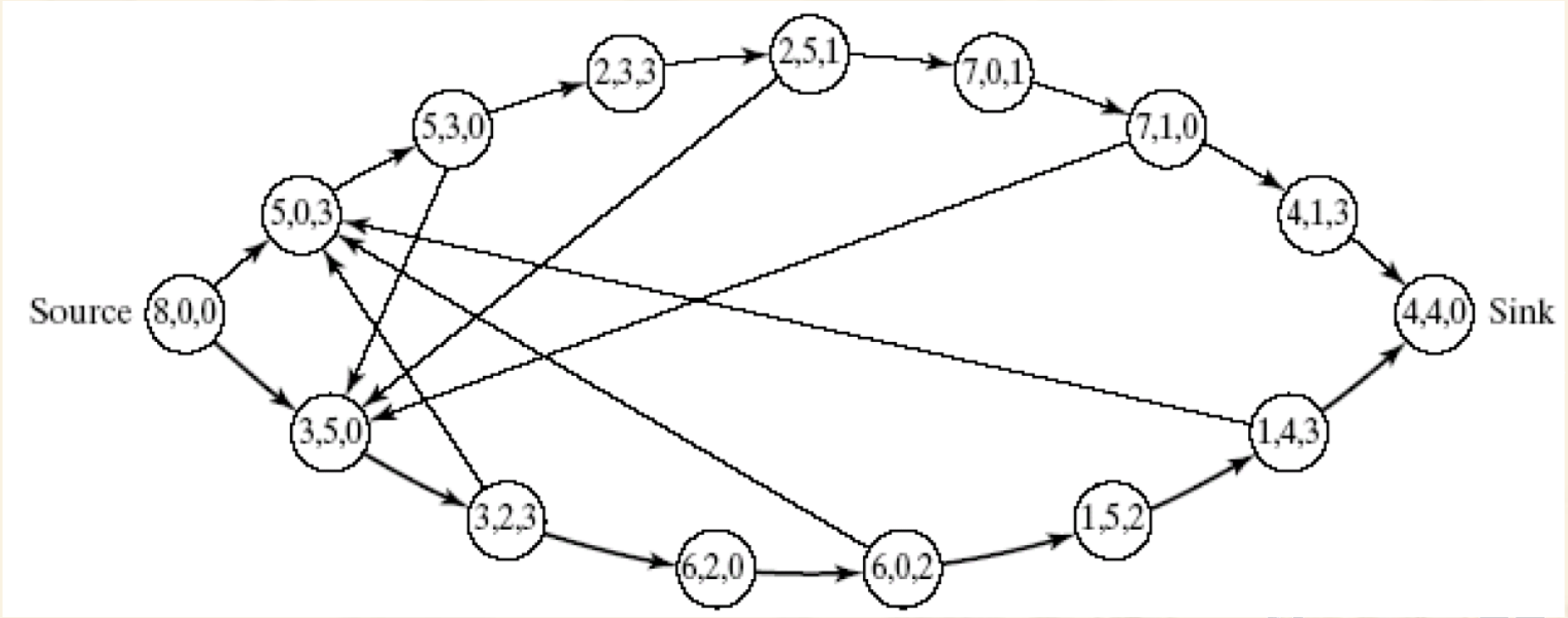
$$p_{1k} = p_1 \times p_2 \times \ldots \times p_k$$

- ✓ 如何将这样一个问题转化为最短路问题？

应用举例-三把水壶问题

- 现在有一把容量为8升装满了水的水壶，借助于一把容量为5升和一把为3升的空水壶将这8升水；平均分为两份。不允许借助其他的工具，如何能用最少次数的转移达到这个目的？
- 用一个节点表示8加仑、5加仑和3加仑容量水壶中的水，这意味着网络是从节点(8,0,0)到我们想得到的节点(4,4,0)。
- 通过从某个水壶向另外一个水壶倒水，我们可以从当前的节点走到新的节点。

应用举例-三把水壶问题(续)



Matching Problem

The marriage problem:

- ✓ We are given disjoint sets P and Q of men and women, and the pairs (p, q) that like each other. The marriage problem is to arrange as many (monogamous) marriages as possible with the restriction that married couple should (at least initially!) like each other.

Matching problem

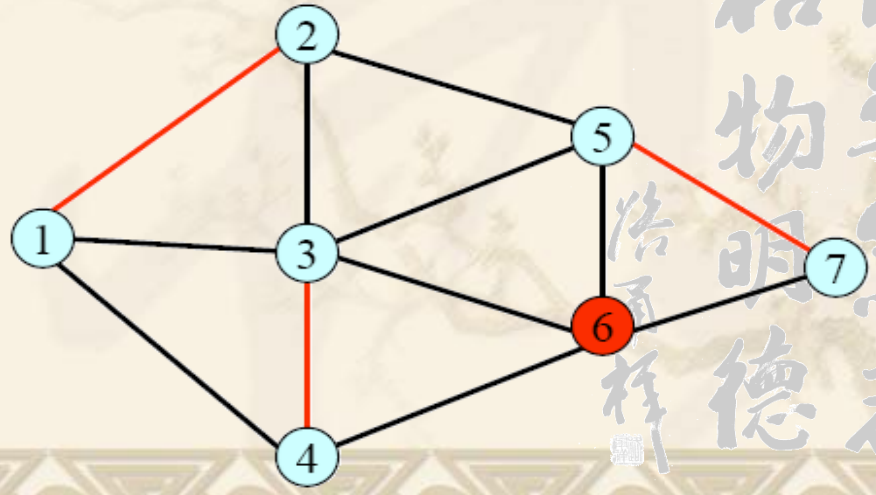
General matching problem

Given a graph $G = (V, E)$, a matching in G is a set M of edges such that no node of G is incident with more than one edge in M.

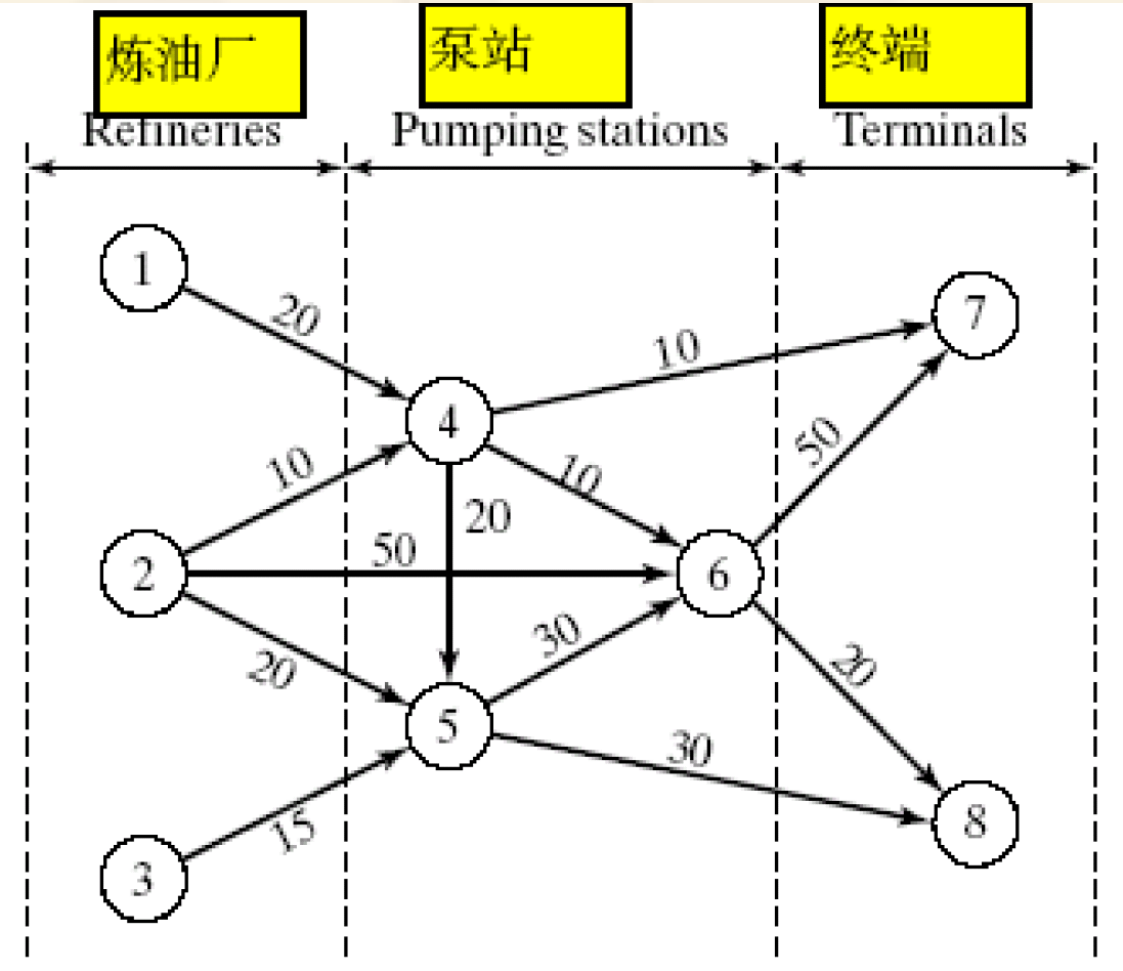
Given a matching M, we say that M covers a node v (or that v is M-covered) if some edge of M is incident with v. Otherwise, v is M-exposed. A maximum matching is one of maximum cardinality, or equivalently, one that has the fewest exposed nodes. A perfect matching is one that covers all the nodes.

$\nu(G)$: cardinality of maximum matching

$\mathrm{def}(G) = |V| - 2\nu(G)$: the minimum number of exposed nodes in G



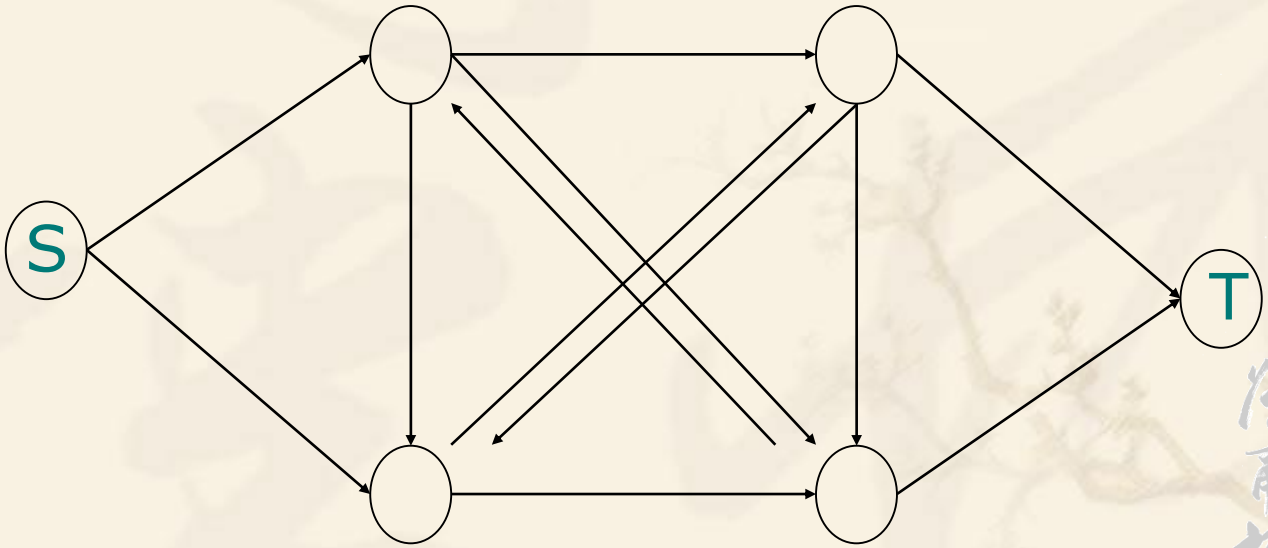
网络流


炼油厂
Refineries
泵站
Pumping stations
终端
Terminals
1
2
3
4
5
6
7
8
20
10
10
10
20
50
50
20
30
20
30
15

网络流（续）

- ✓ 许多实际问题都可以转化为最大流问题
- ✓ 例如

公路交通网络：车流流量



网络流-基本概念

- 有向图$G=(V，A)$上定义权函数：

（1）L：$A\to R$为弧上的权函数，弧(i,j) 对应的权 $L\ (i,j)$ 记为l_{ij}，称为弧 (i,j)的容量下界（lower bound）；

（2）U：$A\to R$为弧上的权函数，弧(i,j)对应的权$U(i,j)$记为u_{ij}，称为弧(i,j) 的容量上界，或直接称为容量（capacity）；

（3）D：$V\to R$为顶点上的权函数，节点i对应的权$D(i)$记为d_i，称为顶点i的供需量（supply/demand）；

- 此时网络可称为流网络(Flow Network,一般仍简称为网络),记为 $\boldsymbol{N=(V,A,L,U,D)}$.

网络流-基本概念（续）

- ✓ d_i >0:供应点(supply node)或源(source)、起始点或发货点
- ✓ d_i <0:需求点(demand node)或汇(sink) 、终止点或吸收点
- ✓ d_i =0:转运点(transshipment node)或平衡点、中间点

网络流-基本概念（续）

- 对于流网络$N=（V，A，L，U，D）$，其上的一个流(flow) x是指从N的弧集A到实数集合R的一个函数$x：A\to R$, 即对每条弧(i,j) 赋予一个实数 (称为弧(i,j)上的流).
- 如果流x满足

$$x_{ij}=-x_{ji},\qquad \forall i,j\in V,\qquad (6.0)$$

$$\sum_{j:(i,j)\in A}x_{ij}-\sum_{j:(j,i)\in A}x_{ji}=d_i,\qquad \forall i\in V,\qquad (6.1)$$

流量守恒/平衡条件

$$l_{ij}\le x_{ij}\le u_{ij},\qquad \forall (i,j)\in A,\qquad (6.2)$$

容量约束/可行条件

- 则称x为可行流(feasible flow).

最大流模型

- ✓ 最大流问题(Maximum Flow Problem)就是在N=（s, t,V,A,U）中找到流值最大的s-t可行流 (即最大流).

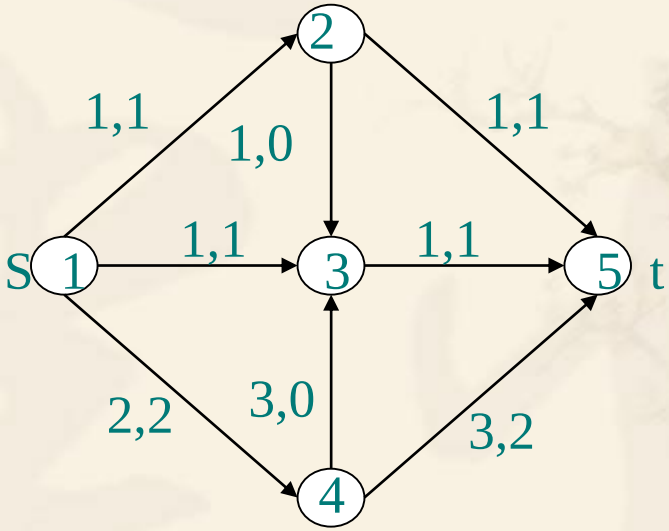
$$\max \quad v$$

$$s.t. \quad \sum_{j:(i,j)\in A} x_{ij} - \sum_{j:(j,i)\in A} x_{ji} = \begin{cases} v, & i = s, \\ -v, & i = t, \\ 0, & i \neq s,t, \end{cases}$$

$$0 \le x_{ij} \le u_{ij}, \qquad \forall (i,j) \in A$$

最大流模型（续）

定义 在流网络$N=(V,A,U,D)$中，对于流x，如果某条弧(i,j)上的流量等于其容量（ $x_{ij}=u_{ij}$ ），则称该弧为饱和弧（saturated arc）；如果某条弧（i，j）上的流量为0($x_{ij}=0$)，则称该弧为空弧（void arc）.



最大流最小割定理, Ford, Fulkerson, 1956

定义 设$[S，T]$是网络$N=(s,t,V,A,U)$中给定的一个割，且$s\in S$，$t\in T$，则称割$[S，T]$为s-t割. s-t割$[S，T]$中的弧$(i，j)(i\in S，j\in T)$称为割的前向弧，弧$(i,j)(j\in S，i\in T)$称为割的反向弧. s-t割$[S，T]$的容量定义为前向弧的容量之和，记为 $U(S,T)=\sum_{\substack{(i,j)\in A,\\ i\in S,j\in T}} u_{ij}$

一个网络中容量最小的s-t割称为最小割.

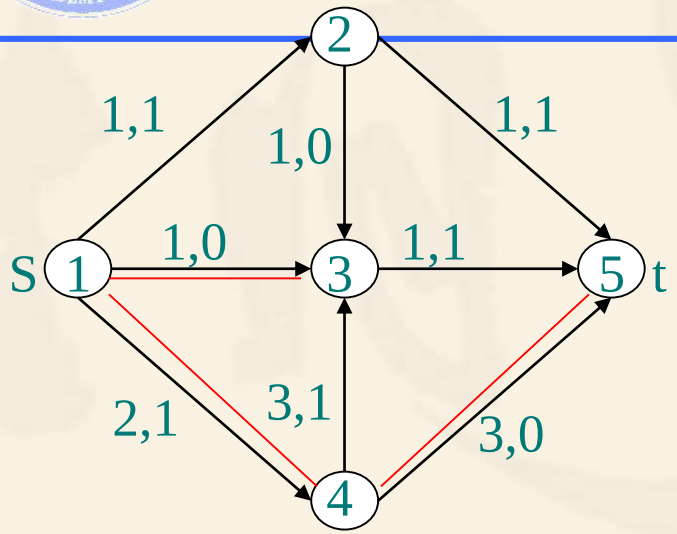
定理 (最大流最小割定理) 任给一个网络$N=(s,t,V,A,U)$，最大流的流量等于最小割的容量.

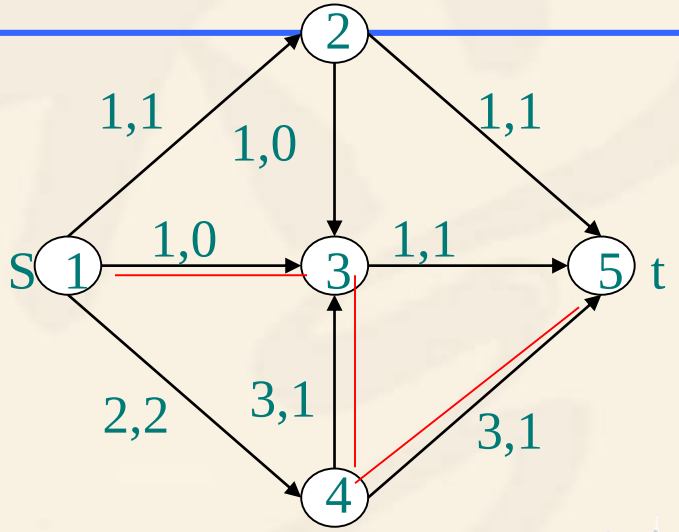
Ford-Fulkerson标号算法

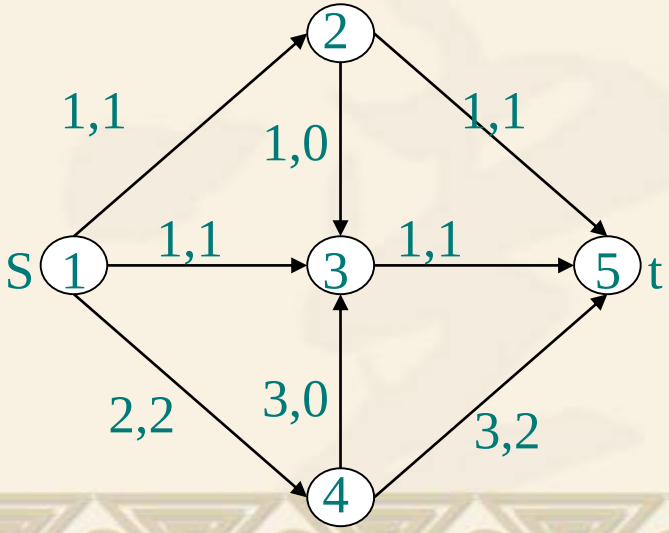
- ✓ 基本思想：从任何一个可行流开始,沿增广路对流进行增广，直到网络中不存在增广路为止.

- ✓ 详细标号算法过程

example







最大流流值为4

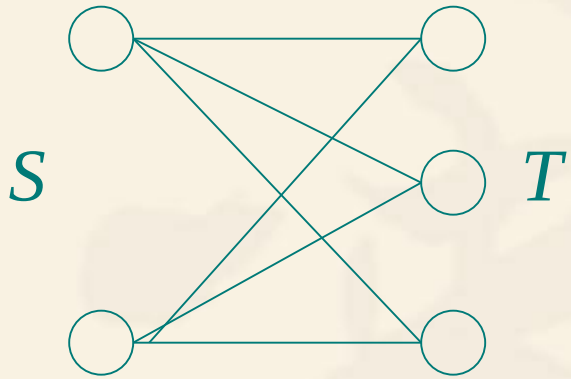
应用举例-产品生产计划

- 有四家工厂从事于生产四种玩具。下表给出了每个工厂可以生产的玩具种类。所有种类的玩具生产一件需要的工时和原料基本相同。四个工厂每天的产量分别是200,180,300和100个玩具。对四种玩具的需求量分别是200,150,350和100个。求工厂可以最大满足对四种玩具需求的生产方案。

工厂	可以生产的玩具种类
1	1,2,3
2	2,3
3	1,4
4	3,4

应用举例-运输问题

- 有一批货物从货源地运往目的地，假设货源集合为S，目的地集合为T. 货源i可提供的货物数量为a_i个单位，目的地j对货物的需求量为b_j个单位。



$$\sum_{j:(i,j)\in A} x_{ij} = a_i, \qquad i \in S,$$

$$\sum_{i:(i,j)\in A} x_{ij} = b_j, \qquad j \in T,$$

$$x_{ij} \geq 0, \qquad i \in S, j \in T$$

运输费用 c_{ij}

最小费用流

- 流网络$N=(V,A，L,U,D)$ 上增加如下的权函数：

C： $A\to R$为弧上的权函数，弧（i，j）对应的权 C（i，j）记为c_{ij}，称为弧（i，j）的单位流量的成本或费用（cost）；

- 此时网络可称为容量-费用网络 (一般仍可简称为网络),记为 $N=(V,A,C,L,U,D)$.

最小费用流问题就是在网络中寻找总费用最小的可行流.

最小费用路算法

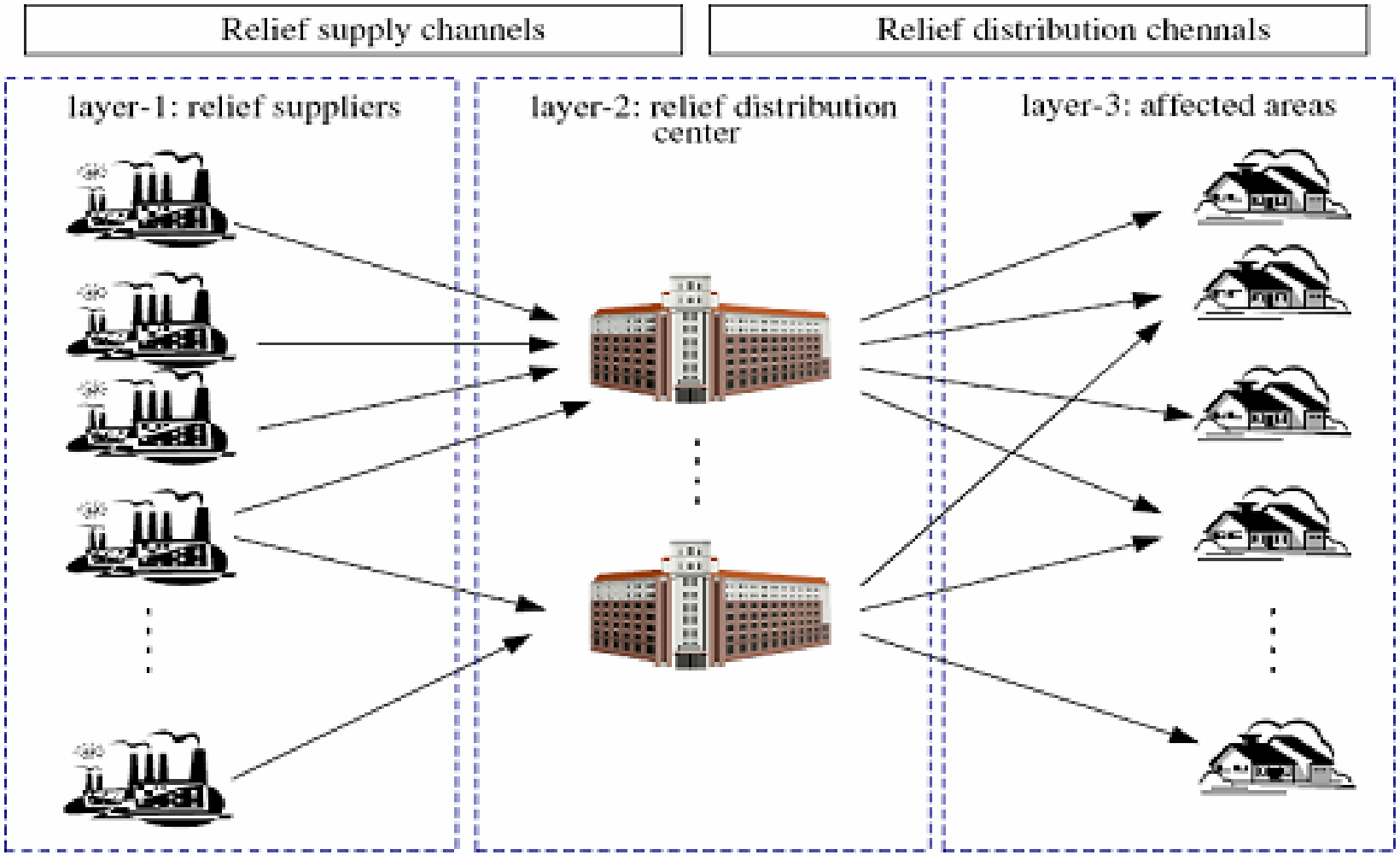
- ✓ 为了获得最小费用流，我们希望沿费用最小的增广路对当前流x进行增广，以最小的费用增加获得流值更大的s-t可行流y。

STEP 0 . 取x为任一s-t可行流、且在同一流值的流中费用最小的流 (如x=0).

STEP 1. 若x的流值达到v, 结束；否则在增量网络$N(x)$中判别最小费用路. 若无这样的路,则流值不可达到v, 结束；否则STEP 2.

STEP 2. 沿该最小费用增广路增广流量(增广后的流值不超过v), 转STEP 1.

应用举例-应急物资调运

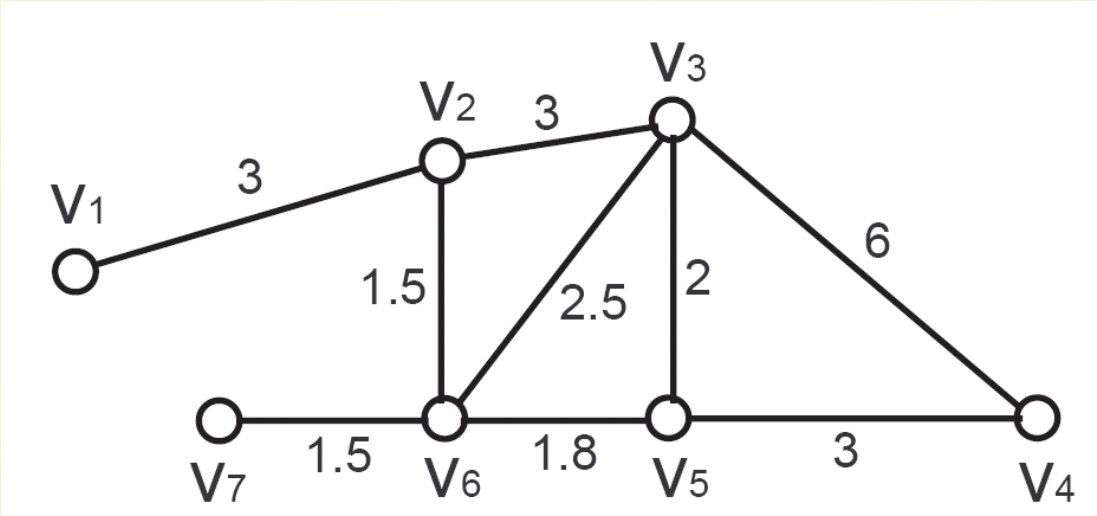

Relief supply channels
Relief distribution chennals
layer-1: relief suppliers
layer-2: relief distribution center
layer-3: affected areas

网络优化的其他应用

- ✓ 图的中心
- ✓ 支配集
- ✓ 覆盖集

应用实例-消防队选址

- 设某市有7个化工厂，已知这些工厂间的公路以及（单位：km）如图。现拟建一个消防队负责这些工厂的消防工作。问消防队应设在哪个工厂，以便任一个工厂失火时都能及时赶去救火？



图的中心、中位点

- Given a weighted graph $G = (V,A)$ with n vertices and an $n \times n$ weight matrix $C = [cij]$.
- $d(u,v)$表示顶点u到v的最小距离。
- 图G的半径(radius): $\operatorname{rad} G = \min_{u \in V(G)} \max_{v \in V(G)} d(u,v)$
- 图G的直径(diameter):

$$\operatorname{diam} G = \max_{u \in V(G)} \max_{v \in V(G)} d(u,v) = \max\{d(u,v) \mid \forall u,v \in V(G)\}$$

- 图G中顶点u的离心率(eccentricity): $e(u) = \max_{v \in V(G)} d(u,v)$
- 图G的**中心**(center):图G中具有最小离心率的顶点
- 图G的**中位点**(median)：设 $g(u) = \sum_{v \in V(G)} d(u,v)$，使$g(u)$达到最小的顶点称为图$G$的中位点。

应用实例-应急增援中心的选址

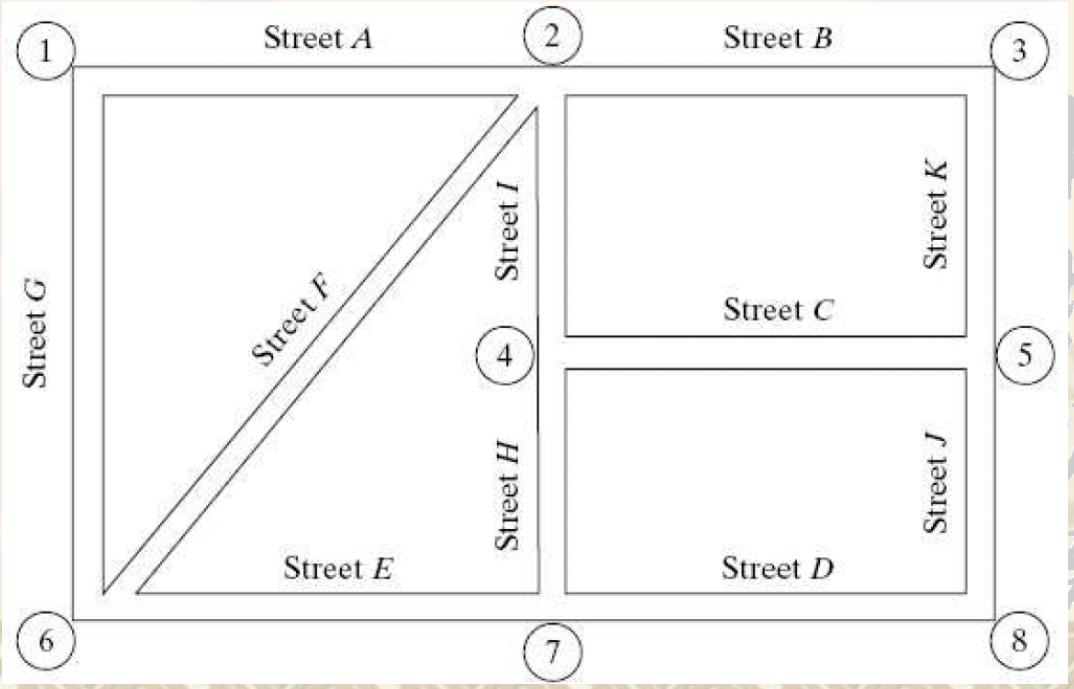
- ✓ 欲在n个地点设置若干个应急中心，使得每个地点都至少一个应急中心相邻（有直达通路）。同时为了减少造价，应急中心的数目要尽可能少。问应设多少个？设置在哪儿？

支配集（domination set）

- 设 $D \subseteq V(G)$,若对 $\forall u \in V(G)$,要么 $u \in D$，要么 u 与 D 中的某些顶点相邻，则称 D 为图 G 的一个支配集。
- 如果一个支配集的任何真子集都不是支配集，则称它为极小支配集。
- 图 G 的含顶点最少的支配集称为最小支配集。最小支配集的顶点个数称为 G 的支配数，记为 $\gamma(G)$ 或 γ 。

应用实例

- ✓ 为了提高校园的安全性，A大学的保安部门决定在校园内部的几个位置安装紧急报警电话。保安部希望在校园的每条主要街道上都至少有一部电话的情况下，使得安装的总电话数目最少。



点覆盖（vertex covering set）

- 设 $F \subset V(G)$,若 G 的每条边至少有一个端点属于 F，则称 F 是 G 的一个点覆盖集。若对 $\forall v \in F, F-\{v\}$ 都不再是 G 的点覆盖，则称点覆盖 F 是一个极小点覆盖。图 G 的含点数最小的点覆盖称为最小点覆盖，其点数称为 G 的点覆盖数，记为 β 或 $\beta(G)$

应用实例-信号干扰

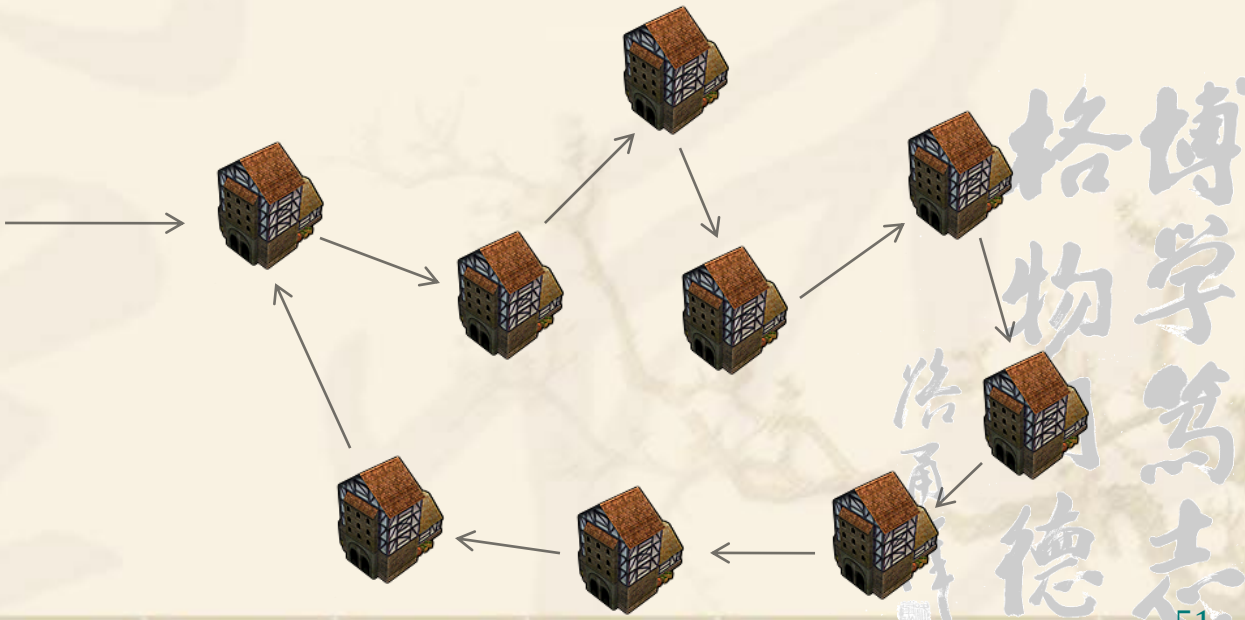
- ✓ 信号干扰问题
- ✓ 课表问题
- ✓ 药品存储问题

独立集（vertex independent set）

- 设 $I \subseteq V(G)$,若 I 中任二顶点均不相邻，则称 I 为图 G 的一个点独立集(简称独立集)；若对 $\forall u \in V(G) \setminus I, I \cup \{u\}$ 都不再是 G 的独立集，则称独立集 I 为图 G 的一个极大点独立集。G 的含点数最多的点独立集称为最大点独立集，它含点的个数称为 G 的独立数，记为 $\alpha(G)$ 或 α 。

TSP

- The **Travelling Salesman Problem** (**TSP**) is a problem in combinatorial optimization studied in operations research and theoretical computer science. Given a list of cities and their pair-wise distances, the task is to find a shortest possible tour that visits each city exactly once.



vehicle routing problem

- The **vehicle routing problem (VRP)** is a combinatorial optimization and integer programming problem seeking to service a number of customers with a fleet of vehicles.

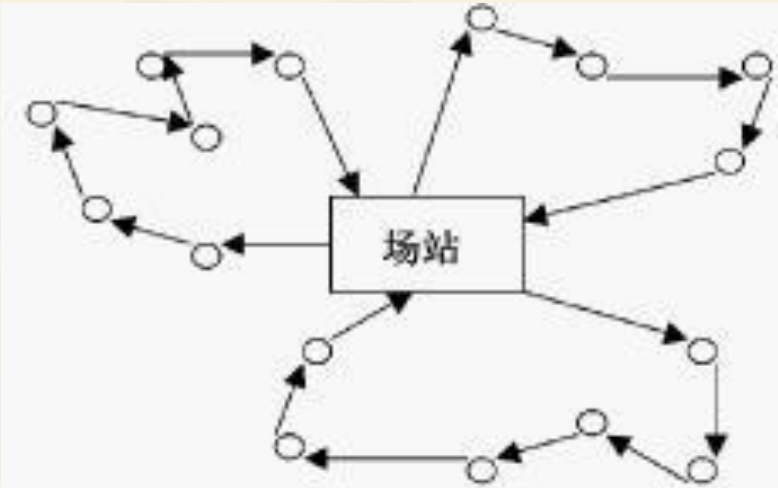


图1 VRP 示意图

heuristic algorithm(启发式算法)

- In computer science, a **heuristic algorithm**, or simply a **heuristic**, is an algorithm that is able to produce an acceptable solution to a problem in many practical scenarios, in the fashion of a general heuristic, but for which there is no formal proof of its correctness. Alternatively, it may be correct, but may not be proven to produce an optimal solution, or to use reasonable resources. Heuristics are typically used when there is no known method to find an optimal solution, under the given constraints (of time, space *etc.*) or at all.

启发式算法的发展

- ✓ 40年代：由于实际需要，提出了启发式算法（快速有效）。
- ✓ 50年代：逐步繁荣，贪婪算法和局部搜索得到关注。
- ✓ 60年代: 反思，发现以前提出的启发式算法速度很快，但是解得质量不能保证，而且对大规模的问题仍然无能为力（收敛速度慢）。
- ✓ 70年代：计算复杂性理论的提出，NP问题。许多实际问题不可能在合理的时间范围内找到全局最优解。发现贪婪算法和局部搜索算法速度快，但解不好的原因主要是他们只是在局部的区域内找解，解没有全局最优性。

启发式算法的发展

- ✓ 1975年holland提出遗传算法 (Genetic Algorithm)
- ✓ 1977年Glouer提出禁忌搜索算法 (Tabu Search)
- ✓ 1982年Kirkpatrick提出模拟退火算法 (Simulated Annealing)
- ✓ 1995年Dorigo提出蚁群算法 (Ant Colony Optimization)

启发式算法的优点与缺点

- ✓ 启发式算法的优点

  1. 模型误差、数据不精确性、参数估计误差等可能造成最优算法的解比启发式算法的解更差；

  2. 复杂问题无法求得最优算法或最优算法太复杂；

  3. 简单易行，直观，程序简单。

- ✓ 启发式算法的缺点

  1. 不能保证最优；

  2. 不稳定；

  3. 依赖于实际问题、设计者经验。

启发式算法的分类

- 简单直观的算法
  - 一步算法：不在两个可行解之间比较，在未终止的迭代过程中，得到的中间解有可能不是可行解；
  - 改进算法：迭代过程是从一个可行解到另一个可行解变换，通过两个解的比较而选择好的解，直到满足一定的要求为止；

启发式算法的分类

- 数学规划算法

用连续优化（如线性规划）的方法求解组合优化问题（如整数线性规划模型），其中包括一些启发式规则。

基于数学规划的理论。

启发式算法的分类

- 现代优化算法
  - 禁忌搜索算法
  - 模拟退火算法
  - 遗传算法
  - 人工神经网络
  - 蚁群算法
  - 粒子群算法
  - 混合算法

特点：

- 基于客观世界中的一些自然现象；
- 建立在计算机迭代计算的基础上；
- 具有普适性，可解决实际应用问题。

启发式算法的性能分析

- 评价算法优劣的指标

  算法的复杂性（计算效率）

  解的偏离程度（计算效果）

  算法的稳健性（不同实例、不同时间、不同起点的差异）

- 评价算法优劣的手段

  最坏情况分析（纯理论）

  概率分析（理论分析）

  计算模拟分析（统计特性）

背包问题的贪婪算法（Greedy algorithm）

贪婪算法：采用逐步构造最优解的方法。

在每个阶段，都作出一个看上去最优的决策（在一定的标准下）。决策一旦作出，就不可再更改。作出贪婪决策的依据称为贪婪准则（**greedy criterion**）。

背包问题的贪婪算法（Greedy algorithm）

STEP 1 对物品以$\frac{c_i}{a_i}$从大到小排列，不妨把排列

记成$\{1,2,\cdots,n\}, k:=1;$

STEP 2 若$\sum_{i=1}^{k-1} a_i x_i + a_k \le b$, 则$x_k = 1$; 否则$x_k = 0$,

$k:=k+1$; 当$k=n+1$时，停止；否则，重复STEP2。

时间复杂性分析　　算法性能分析

顶点覆盖贪婪算法

- ✓ Input: A graph G=(V,E)
- ✓ Output: A node cover C.

```
Begin
    C=empty;
    while E!=empty  do
            choose the node in V that has the largest degree,
            remove it form G and add it to C.
 End
```

算法分析　　性能分析　　改进算法