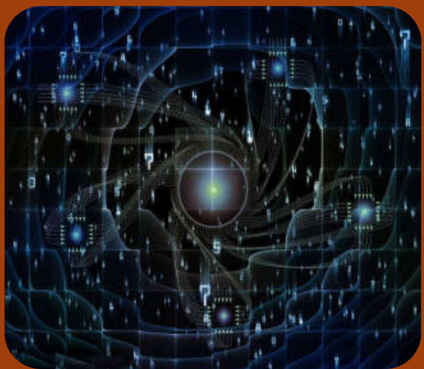
算法与计算理论

课程内容

数据结构

概述	数组与广义表	图	外部排序
线性表	串	查找	
栈与队列	树	内部排序	



算法与计算理论

概述	贪心	……	计算模型
分治	回溯	……	可计算理论
动态规划	……	……	计算复杂性

Contents

本章内容

贪心算法的基本要素

对于一个具体问题, 怎么知道能否用贪心算法

- 贪心选择性质和最优子结构性质
- 贪心选择性质

  对比: 矩阵连乘, 0-1背包, 分数背包

  贪心算法第一基本要素, 与DP主要区别

  自顶向下计算
- OSP（optimal sub strategy）：最优策略的子策略也是最优
- 正确性证明一般过程:

  贪心选择+OSP+数学归纳法

  条件: 子问题与原问题类似, 相对独立

  子问题的最优解和贪心选择联合得整体最优解

构造“贪心”反例

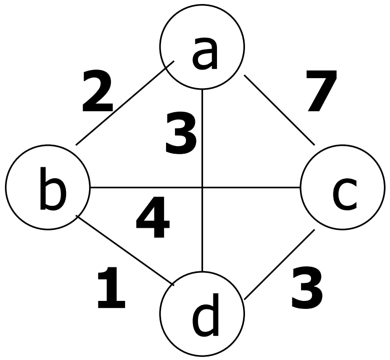
找零问题：一出纳员支付一定数量的现金。假设他手中有各种面值的纸币和硬币，要求他用最少的货币数支付规定的现金

例如，现有4种硬币：它们的面值分别为1分、2分、5分和1角，要支付2角5分。
首先支付2个1角硬币，然后支付一个5分硬币，这就是贪心策略

反例：
三种硬币面值：1分、4分、6分，支付8分

构造“贪心”反例

• 货郎担（TSP）问题：设售货员要到四个城市去售货，最后再回到出发的城市，已知从一个城市到其他城市的费用，求总费用最少的路线



- 结点a为起点：2+1+3+7=13
- 结点b为起点：1+3+7+2=13
- 结点c为起点：3+1+2+7=13
- 结点d为起点：1+2+3+7=13
- 最优解：12

活动安排问题([王]P90)

n个活动申请一个活动室, 各活动起始终止区间(s_i, f_i)

- 输入: n, (s_i, f_i), $i = 1{:}n$
- 输出: 最大相容活动子集(无冲突, 活动个数最多)

最优解可以包含 最早结束的

先给出算法, 再证明正确性(A[i]=true或false)

1. 按终止时间排序 $f_1 \le f_2 \le \ldots \le f_n$. A[1] = true; pt = 1;
2. 对 i = 2:n
3. 若 s[i] >= f[pt], 则 A[i] = true, pt = i,
4. 否则 A[i] = false

活动安排算法正确性证明

n个活动申请一个活动室, 活动起始终止区间(s_i, f_i)

输入: n, (s_i, f_i), i = 1:n, 输出: 最大相容活动子集

11个活动已按结束时间排序:

i	1	2	3	4	5	6	7	8	9	10	11
start_time$_i$	1	3	0	5	3	5	6	8	8	2	12
finish_time$_i$	4	5	6	7	8	9	10	11	12	13	14

活动安排算法正确性证明

n个活动申请一个活动室, 活动起始终止区间(s_i, f_i)

输入: n, (s_i, f_i), i = 1:n, 输出: 最大相容活动子集

第一步 证明贪心选择性质:

存在最优解包含相容最早结束的活动1

第二步 证明最优子结构性质:

若A是包含活动1的最大相容活动集(最优策略),

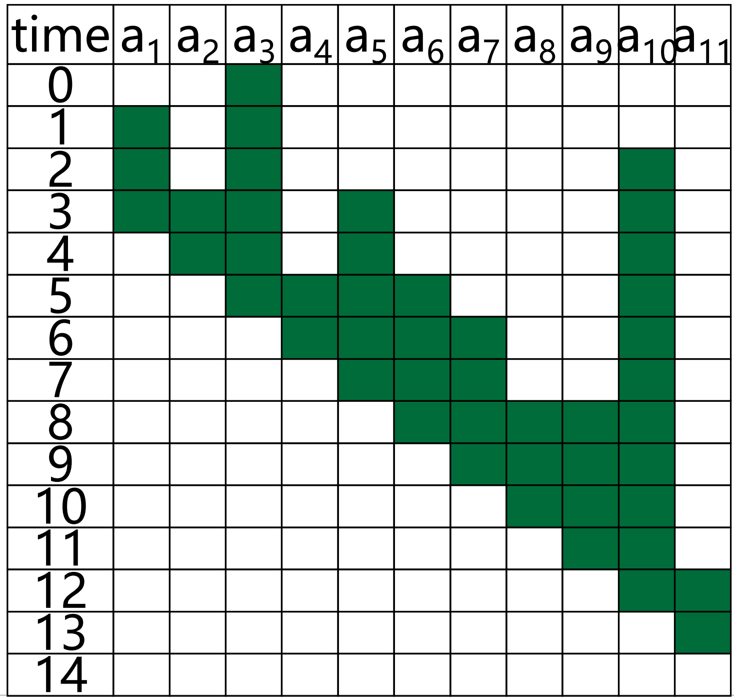
则A-{1}(子决策)是$\{i \mid s_i >= f_1\}$ (子问题)上的最大相容活动集

由此用数学归纳法, 就可以证明算法正确性

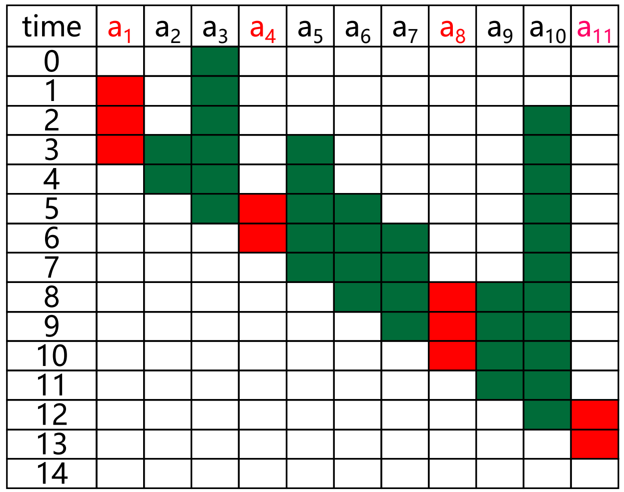
活动安排算法正确性证明

11个活动已按结束时间排序，用贪心算法求解：

i	1	2	3	4	5	6	7	8	9	10	11
$start\_time_i$	1	3	0	5	3	5	6	8	8	2	12
$finish\_time_i$	4	5	6	7	8	9	10	11	12	13	14



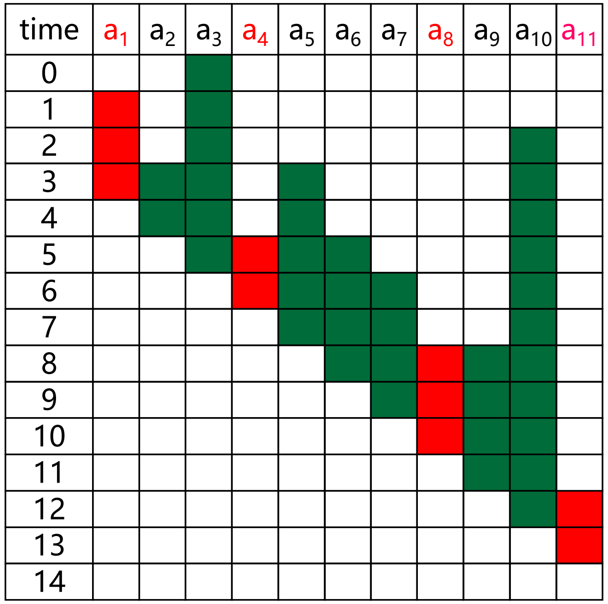
相容活动：$\{a_3, a_9, a_{11}\}$, $\{a_1,a_4,a_8,a_{11}\}$, $\{a_2,a_4,a_9,a_{11}\}$



活动安排算法正确性证明

最优解： $\{a_2,a_4,a_9,a_{11}\}$

$\{a_1,a_4,a_8,a_{11}\}$



第一步 证明贪心选择性质:

存在最优解包含相容最早结束的活动1

第二步 证明最优子结构性质:

若A是包含活动1的最大相容活动集(最优策略)，则A-{1}(子决策)是$\{i \mid s_i >= f_1\}$ (子问题)上的最大相容活动集

- 一、最小的1可以在里面，{1, 4, 9, 11}
- 则A-{1}={4, 9, 11}是子问题{4,6,7,8,9,11}的最优解

- 二、最小的4可以在里面，{4, 9, 11}
- 则A-{4}={9, 11}是子问题{8,9,11}的最优解

- 三、最小的8可以在里面，{8, 11}
- 则A-{8}={11}是子问题{11}的最优解

分数背包

已知一个容量大小为M重量的背包和n种物品，物品i的重量为w_i,假定物品i的一部分x_i放入背包会得到v_ix_i这么大的收益，这里，$0 \le x_i \le 1$，$v_i > 0$，采用怎样的装包方法才会使装入背包的物品总效益最大?

考虑以下情况下的背包问题

$n=3, M=20$,

$(v_1, v_2, v_3) = (25,24,15)$

$(w_1, w_2, w_3) = (18,15,10)$

$(v_1/w_1, v_2/w_2, v_3/w_3) = (125/90, 144/90, 135/90)$

分数背包贪心法证明

$Case\ 1$：所有 $x_i = 1$。显然该解就是最优解。

$Case\ 2$：设 $X = (1,\dots,1, x_j, 0,\dots,0)$ $x_j \neq 1, 1 \leq j \leq n$。下证 X 就是最优解。设问题的最优解 $Y = (y_1,\dots,y_n)$，则存在 k 使得 $y_k \neq x_k$ 的最小下标（否则 $Y = X$，得证）。

首先，可以证明 $y_k < x_k$。（反证：若 $y_k > x_k$，则 $x_k \neq 1, \therefore k \geq j$

$$\Rightarrow \sum_{i=1}^{n} w_i y_i \geq \sum_{i=1}^{k} w_i y_i > \sum_{i=1}^{k} w_i x_i = \sum_{i=1}^{j} w_i x_i = c \therefore Y \text{不是可行解，矛盾）}$$

下面改造 Y 成为新解 $Z = (z_1,\dots,z_n)$，并使 Z 仍为最优解。将 y_k 增加到 x_k，从 $(y_{k+1},\dots,y_n)$ 中减同样的重量使总量仍是 c。即，

$$z_i = x_i \quad i = 1,2,\dots,k; \quad \text{和} \quad \sum_{i=k+1}^{n} w_i y_i - w_k(z_k - y_k) = \sum_{i=k+1}^{n} w_i z_i$$

分数背包贪心法证明

$$\therefore \sum_{i=1}^{n} w_i z_i = \sum_{i=1}^{k-1} w_i z_i + w_k z_k + \sum_{i=k+1}^{n} w_i z_i$$

$$= \sum_{i=1}^{k-1} w_i y_i + w_k z_k + \left(\sum_{i=k+1}^{n} w_i y_i - w_k (z_k - y_k) \right) = \sum_{i=1}^{n} w_i y_i = c$$

$$\therefore \sum_{i=1}^{n} v_i z_i = \sum_{i=1}^{n} v_i y_i + (z_k - y_k) v_k - \sum_{i=k+1}^{n} (y_i - z_i) v_i \quad // \because Z\text{由}Y\text{变得的}$$

$$= \sum_{i=1}^{n} v_i y_i + (z_k - y_k) w_k v_k / w_k - \sum_{i=k+1}^{n} (y_i - z_i) w_i v_i / w_i$$

$$\geq \sum_{i=1}^{n} v_i y_i + \left[(z_k - y_k) w_k - \sum_{i=k+1}^{n} (y_i - z_i) w_i \right] v_k / w_k \quad // \text{利用} v_i / w_i \downarrow$$

$$= \sum_{i=1}^{n} v_i y_i$$

$\therefore Z$也是最优解，且$z_i = x_i \; i = 1, \ldots, k$； 重复上面过程 $\Rightarrow$ X为最优解。

分数背包贪心法证明（举例）

$n=4,\ M=16$,

$(v_1, v_2, v_3, v_4)=(20,12,6,6)$

$(w_1, w_2, w_3, w_4)=(10,12,6,6)$

$X=(1, 1/2, \quad 0, \quad 0)$

$Y=(1, 1/3, 1/6, 1/6)$ 并且 $k=2, j=2$

代换 $Z=(1, 1/2, \quad z_3, \quad z_4)$

$w_3(y_3 - z_3)+ w_4(y_4 - z_4) = w_2(1/2 - 1/3)$

$6(1/6 - z_3)+ 6(1/6 - z_4) = 12*1/6$

$z_3+ z_4 = 0$

即推得 $Z=(1, 1/2, 0, 0)=X$

有限期的任务安排问题

用贪心法求解有限期的任务安排问题：假设只能在同一台机器上加工n个任务，每个任务i完成时间均是一个单位时间，又设每个任务i都有一个完成期限$d_i>0$,当且仅当任务i在它的期限截止以前被完成时，任务i才能获得p_i的效益，每个任务的期限从整个工序的开工开始计时，问应如何安排加工顺序，才能获得最大效益?

n=6,

$(p_1,p_2,p_3,p_4,p_5,p_6)=(5,25,20,30,10,15)$,

$(d_1,d_2,d_3,d_4,d_5,d_6)=(1,5,2,3,3,2)$

有限期的任务安排问题

- 任务i完成时间均是一个单位时间，当且仅当任务i在它的期限截止以前被完成时，i才能获得p_i的效益

$n=6,(p_1,p_2,p_3,p_4,p_5,p_6)=(5,25,20,30,10,15),$

$(d_1,d_2,d_3,d_4,d_5,d_6)=(1, 5, 2, 3, 3, 2)$

- 类似单价排序?
- 法一：按效益从大到小排序

任务	0	1	2	3	4	5	6
p_i	0	30	25	20	15	10	5
d_i	0	3	5	2	2	3	1

有限期的任务安排问题

任务i完成时间均是一个单位时间，当且仅当任务i在它的期限截止以前被完成时，i才能获得p_i的效益?

法二：按期限从大到小排序

任务	0	1	2	3	4	5	6
p_i	0	25	30	10	20	15	5
d_i	0	5	3	3	2	2	1

有限期的任务安排问题

- 任务i完成时间均是一个单位时间，当且仅当任务i在它的期限截止以前被完成时，i才能获得p_i的效益

法三：时间槽 [C]

按效益从大到小排序

任务	1	2	3	4	5	6
p_i	30	25	20	15	10	5
d_i	3	5	2	2	3	1

1	2	3	4	5

最优装载

最优装载

- 输入: n物品重W[1:n],
  背包容量C
- 输出: 装包使得件数最多.
- 每件物品只能取或不取

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

$$\sum_{i=1}^{n} W_i x_i \le C$$

$$x_i \in \{0,1\}, 1 \le i \le n$$

n =8, [w1, ... ,w8] = [100,200,50,90,150,50,20,80], C=400　从剩下的货箱中，选择重量最小的货箱。

贪心选择性质: 最优解可以包含重量最小的

OSP: 最优解([1:n],C)去掉重量最小([2:n],C-W[1])仍是最优解

最优装载贪心法证明

北京理工大学 BEIJING INSTITUTE OF TECHNOLOGY

设集装箱已依其重量从小到大排序，$(x_1,x_2,\cdots,x_n)$是最优装载问题的一个最优解。又设$k=\min\limits_{1\leqslant i\leqslant n}\{i\mid x_i=1\}$。易知，如果给定的最优装载问题有解，则$1\leqslant k\leqslant n$。

(1)　当$k=1$时，$(x_1,x_2,\cdots,x_n)$是一个满足贪心选择性质的最优解。

(2)　当$k>1$时，取$y_1=1;y_k=0;y_i=x_i,1<i\leqslant n,i\neq k$，则，

$$\sum_{i=1}^{n} w_i y_i = w_1 - w_k + \sum_{i=1}^{n} w_i x_i \leqslant \sum_{i=1}^{n} w_i x_i \leqslant c$$

因此，$(y_1,y_2,\cdots,y_n)$是所给最优装载问题的一个可行解。

另一方面，由$\sum\limits_{i=1}^{n} y_i=\sum\limits_{i=1}^{n} x_i$知，$(y_1,y_2,\cdots,y_n)$是一个满足贪心选择性质的最优解。所以，最优装载问题具有贪心选择性质。

设$(x_1,x_2,\cdots,x_n)$是最优装载问题的一个满足贪心选择性质的最优解，则易知，$x_1=1$，$(x_2,\cdots,x_n)$是轮船载重量为$c-w_1$且待装船集装箱为$\{2,3,\cdots,n\}$时相应最优装载问题的一个最优解。也就是说，最优装载问题具有最优子结构性质。

哈夫曼编码([王]P96)

某通讯系统使用5种字符a、b、c、d、e，使用频率分别为0.1，0.14，0.2，0.26，0.3，利用二叉树设计不等长编码

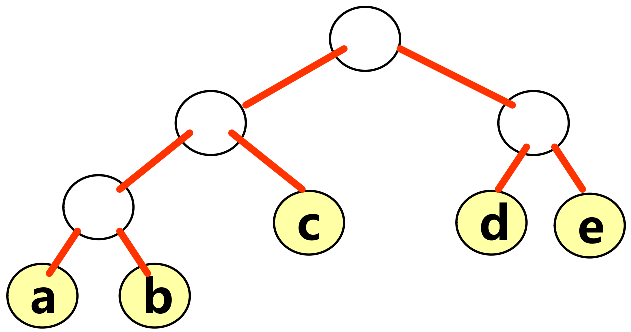
1）构造以 a、b、c、d、e为叶子的二叉树；

2）将该二叉树所有左分枝标记0，所有右分枝标记1；

3）从根到叶子路径上标记作为叶子结点所对应字符的编码；

哈夫曼编码([王]P96)

- 5种字符a、b、c、d、e，使用频率分别为0.1，0.14，0.2，0.26，0.3



a: 000　b: 001
c: 01　d: 10
e: 11

30000-(1000+1400)*3-(2000+2600+3000)*2
=7600

哈夫曼编码贪心选择性质

Let a and b be two characters that are sibling leaves of maximum depth in T. Without loss of generality, we assume that $f[a] \le f[b]$ and $f[x] \le f[y]$. Since $f[x]$ and $f[y]$ are the two lowest leaf frequencies, in order, and $f[a]$ and $f[b]$ are two arbitrary frequencies, in order, we have $f[x] \le f[a]$ and $f[y] \le f[b]$. As shown in Figure 16.6, we exchange the positions in T of a and x to produce a tree T', and then we exchange the positions in T' of b and y to produce a tree T''. By equation (16.5), the difference in cost between T and T' is

$$\begin{aligned}
B(T) - B(T') &= \sum_{c \in C} f(c) d_T(c) - \sum_{c \in C} f(c) d_{T'}(c) \\
&= f[x] d_T(x) + f[a] d_T(a) - f[x] d_{T'}(x) - f[a] d_{T'}(a) \\
&= f[x] d_T(x) + f[a] d_T(a) - f[x] d_T(a) - f[a] d_T(x) \\
&= (f[a] - f[x])(d_T(a) - d_T(x)) \\
&\ge 0 ,
\end{aligned}$$

because both $f[a] - f[x]$ and $d_T(a) - d_T(x)$ are nonnegative. More specifically, $f[a] - f[x]$ is nonnegative because x is a minimum-frequency leaf, and $d_T(a) - d_T(x)$ is nonnegative because a is a leaf of maximum depth in T. Similarly, exchanging y and b does not increase

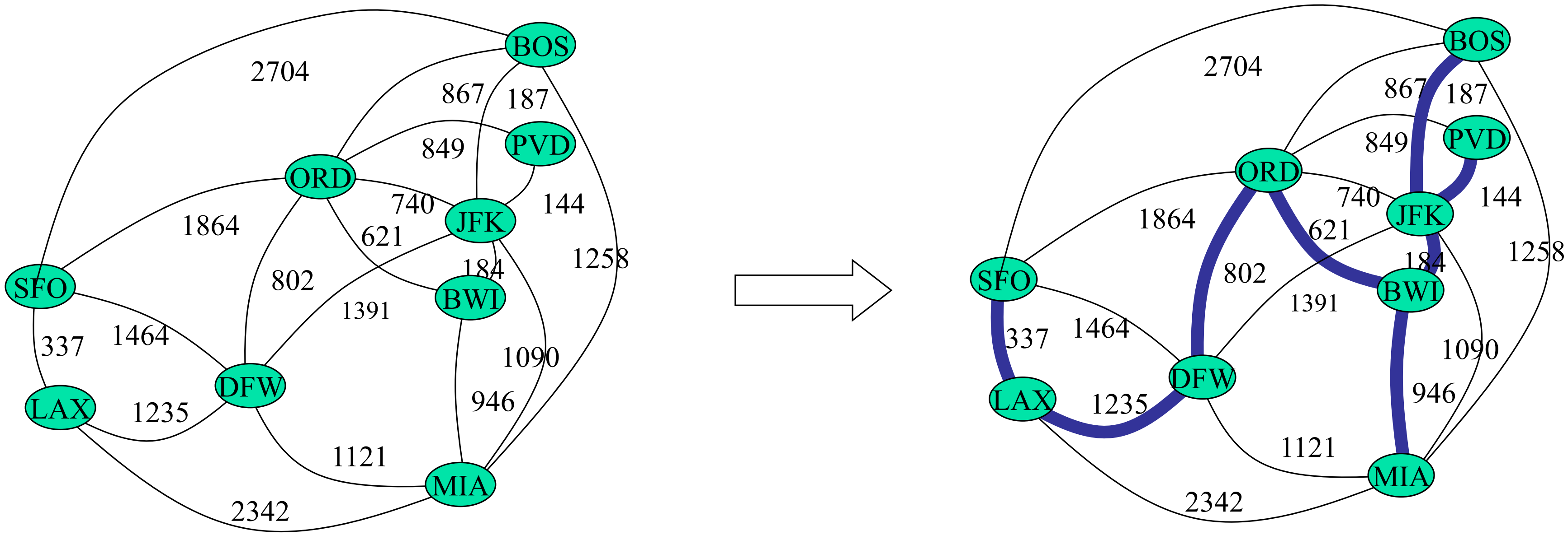
最小生成树(MST[王]P103)

无向连通带权图G=(V,E,w).

G的生成树是G的包含所有顶点的一颗子树

若G的生成树T’在所有G的生成树中各边权总和最小,

则T’称为G的最小生成树(MST, minimum spanning tree)

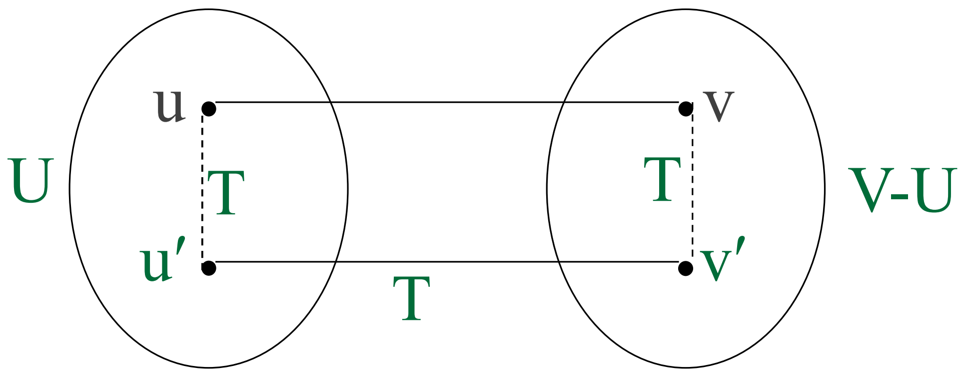


MST性质(贪心选择+归纳)

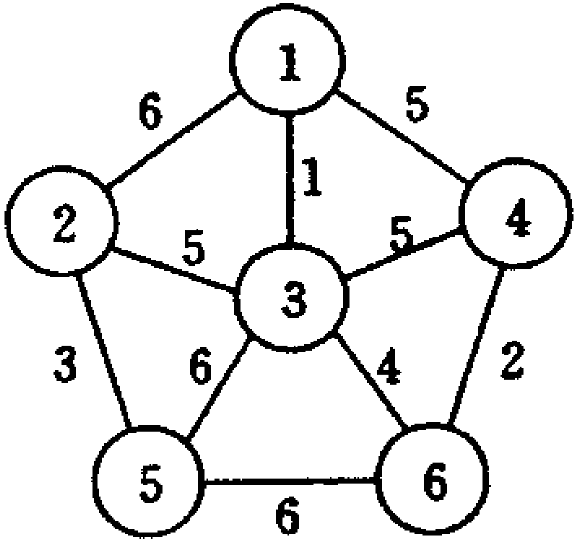
设G=(V,E)是连通带权图，U是V的真子集。若$(u,v)\in E$，且$u\in U$，$v\in V-U$，且在所有这样的边中，(u,v)的权w(u,v)最小，那么一定存在G的一棵最小生成树，它以(u,v)为其中一条边。这个性质有时也称为MST性质。

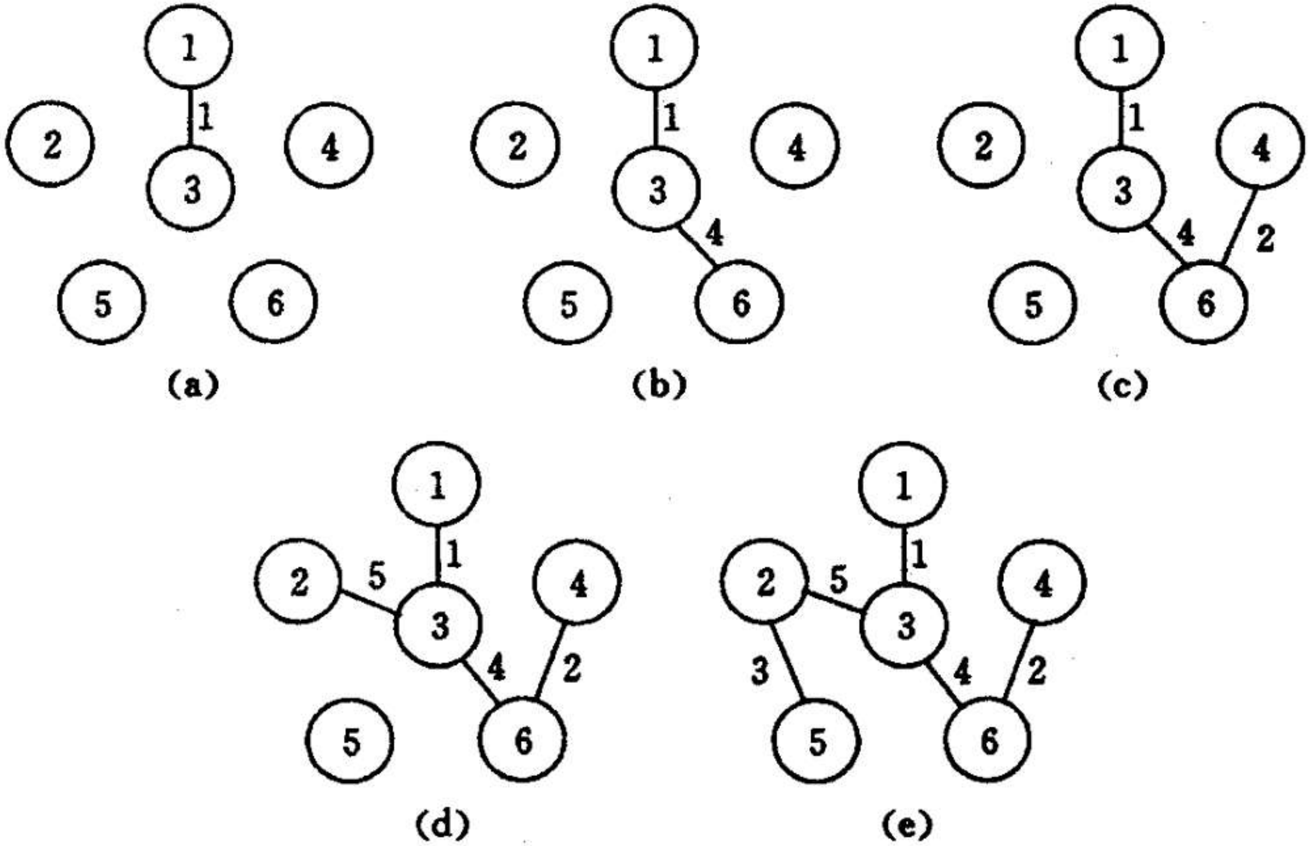
证明: 如图, 设T是MST, $(u,v)\notin T$, $(u',v')\in T$.

则必有 $w(u,v) \le w(u',v')$. 由T去(u',v')添(u,v)得树T'.

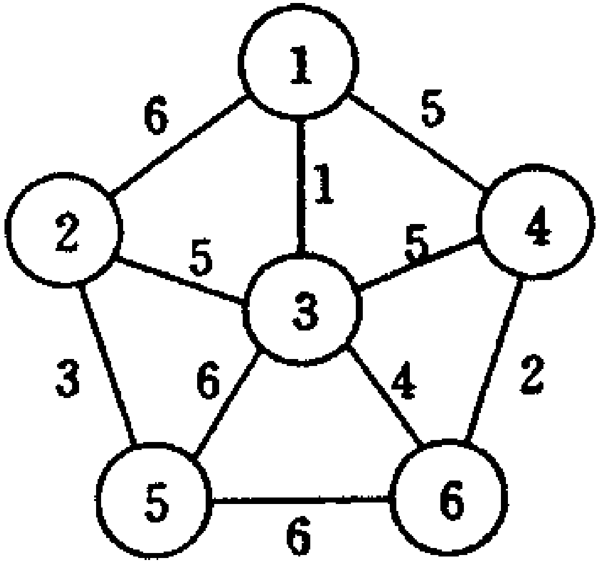


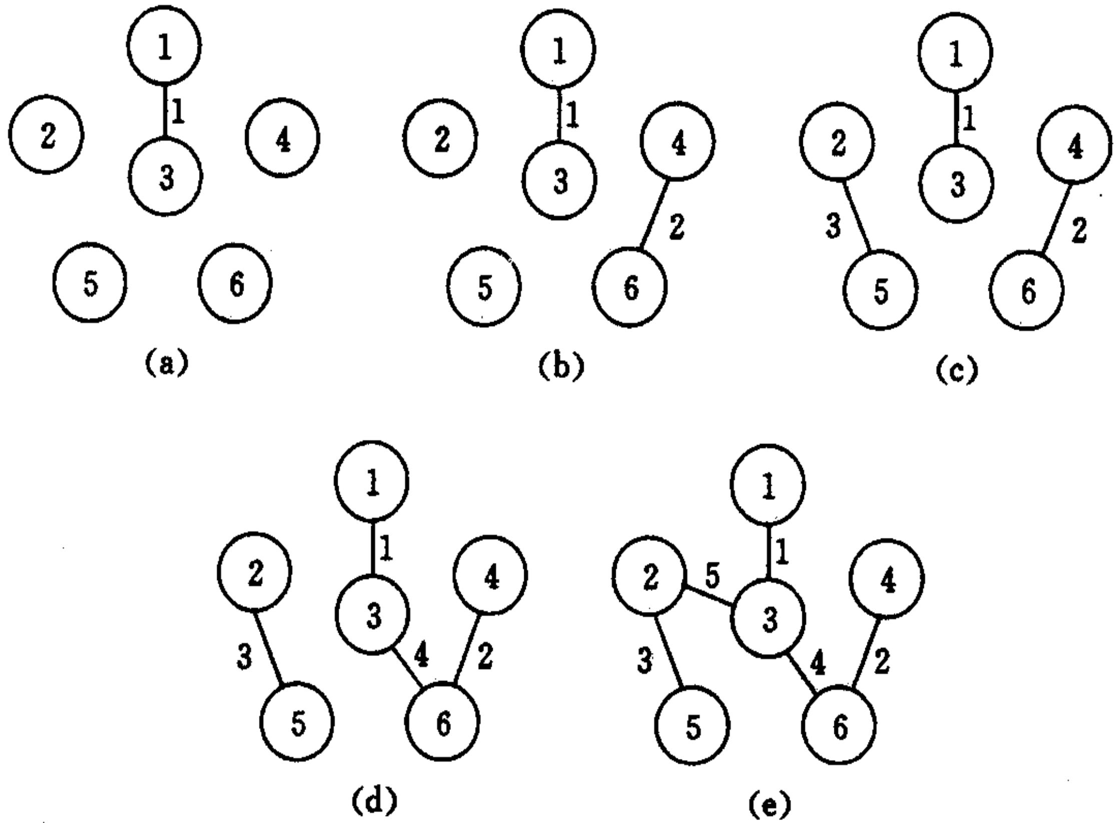
Prim算法举例





Kruskal算法举例





并查集算法(Make-set, Find-set)

p[x]: x的父亲; rank[x]: x的阶; Find(x): 找x的根

init(x)

1 p[x]=x

2 rank[x]=0

Union(x,y)

1 Link(Find(x), Find(y))

Find(x)

1 若 x≠p[x], 则

2 p[x] = Find(p[x])

3 返回 p[x]

路径压缩(pc)技术

Link(x,y) //合并两树根

1 若 rank[x]>rank[y]

2 则 p[y]=x

3 否则 { p[x]=y

4 若 rank[x]=rank[y]

5 则 rank[y]=rank[y]+1 }

加入并查集结构的Kruskal算法

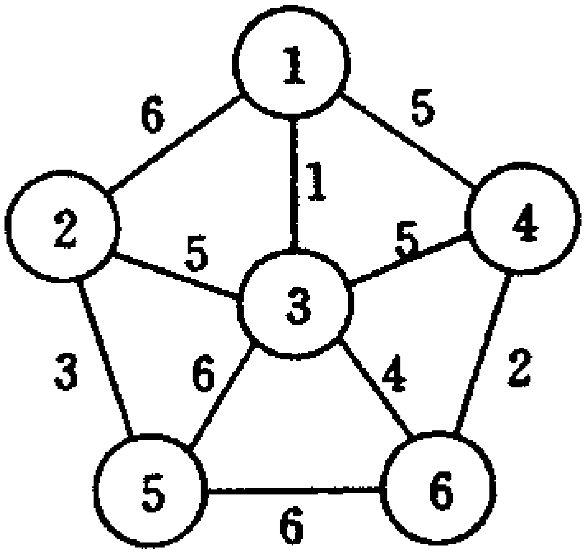
北京理工大学 BEIJING INSTITUTE OF TECHNOLOGY

1. A为空, Q=E按边权升序排列, 每个点是一颗树
2. 当Q非空
3. 顺序取Q中边(u,v)
4. 若u,v在不同树中, 则添(u,v)到A, 合并u,v所在树,
5. 输出A

---

1. A为空, Q=E按边权升序排列, $\forall x$ Make(x)
2. 当Q非空
3. 顺序取Q中边(u,v)
4. 若Find(u)≠Find(v), 则添(u,v)到A, Union(u,v),
5. 输出A

Kruskal: 取边, 查找, 合并



Q={31, 46, 25, 36, 34, 23, 14, 12, 35, 56}

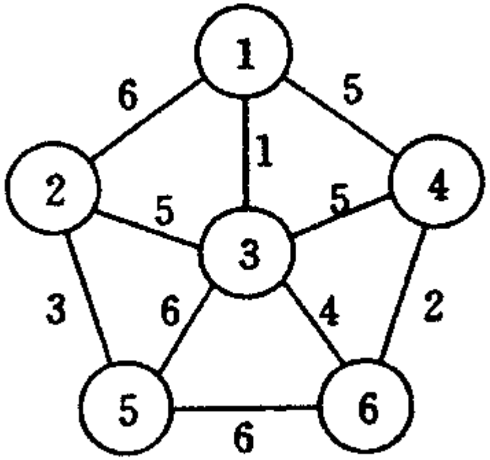
1:0 2:0 3:0 4:0 5:0 6:0

查找合并
(3,1) (4,6) (2,5)

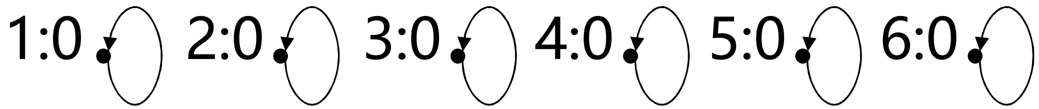
1:1 6:1 5:1

3:0 4:0 2:0

Kruskal: 取边, 查找, 合并

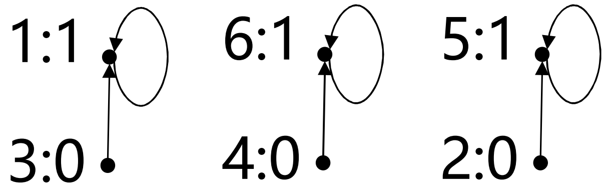


Q={31, 46, 25, 36, 34, 23, 14, 12, 35, 56}

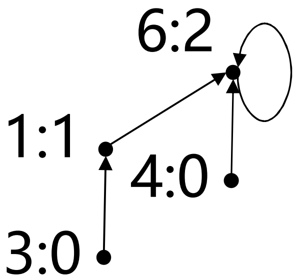


查找合并

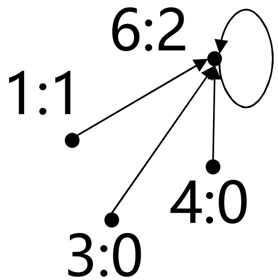
(3,1) (4,6) (2,5)



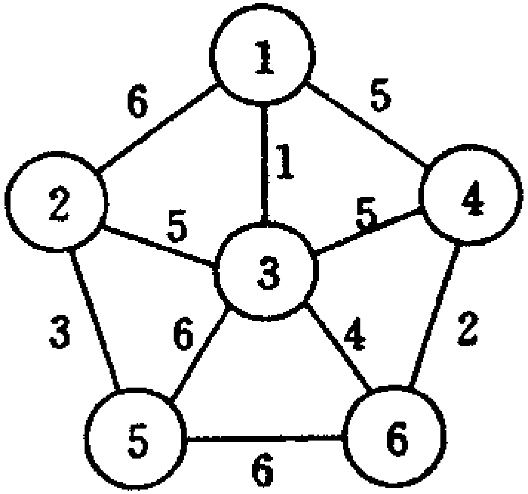
查并(3,6)



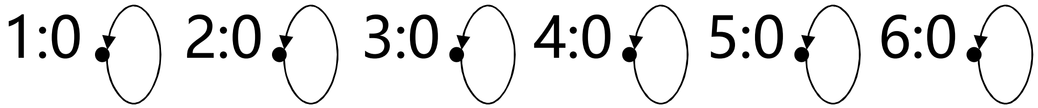
查(3,4)



Kruskal: 取边, 查找, 合并

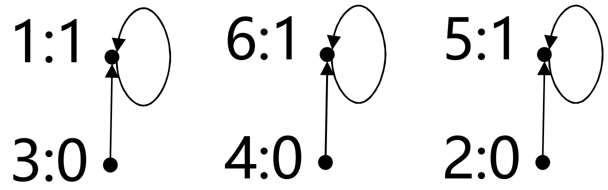


Q={31, 46, 25, 36, 34, 23, 14, 12, 35, 56}

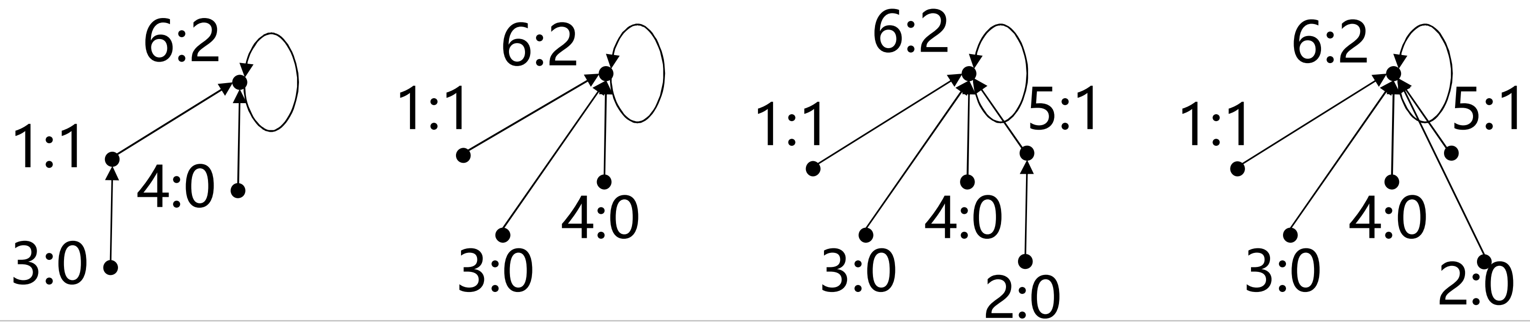


查找合并

(3,1) (4,6) (2,5)



查并(3,6) 查(3,4) 查并(2,3) 查(1,2)



排序之车间作业计划

一台机器、n个零件,使得各加工零件在车间里停留的平均时间最短。

零件	加工时间
1	1.8
2	2.0
3	0.5
4	0.9
5	1.3
6	1.5

若按此顺序:
(1.8+3.8+4.3+5.2+6.5+8)/6=4.9

实际上最短：3, 4, 5, 6, 1, 2
(0.5+1.4+2.7+4.2+6+8)/6=3.8

本章小节

!!贪心不一定正确(0-1背包), 需要证明

典型题目:

1. 活动安排问题
2. 贪心算法的基本要素（分数背包）
3. 有期限的任务安排问题
4. 数量最优装载
5. 哈夫曼编码
6. 并查集的最小生成树

本章作业

1. 字符a~h出现的频率恰好是前8个Fibonacci数，它们的Huffman编码是什么? 将结果推广到n个字符的频率恰好是前n个Fibonacci数的情形.
2. 若在0-1背包问题中, 各物品依重量递增排列时, 其价值恰好降序排列, 对这个特殊的0-1背包问题, 设计一个有效算法找出最优解, 并说明算法的正确性.
3. 将最优装载问题的贪心算法推广到2艘船的情形贪心算法还能产生最优解吗?

4. 最优分解问题.

问题描述:设n是一个正整数,将n分解为若干互不相同的自然数之和,且使这些自然数的乘积最大.

算法设计:对于给定的正整数n,计算最优分解方案.

数据输入:由文件input.txt提供输入数据.

文件只有一行,是正整数n.

结果输出:将计算的最大乘积输出到文件output.txt

例如若n=10, 则最优分解为2+3+5, 最大乘积为30.