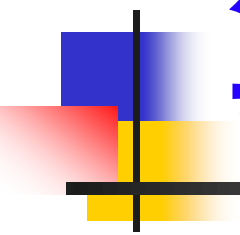
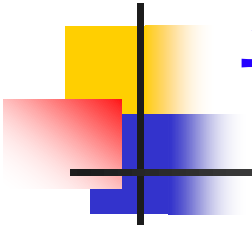


算法分析与设计

Analysis and Design of Algorithm

第11次课



课程回顾

- 贪心算法的基本概念
  - 贪心选择性质
  - 局部最优和全局最优
- 贪心算法的应用
  - 活动安排问题
  - 最优装载问题
  - 哈夫曼编码

哈夫曼编码及其应用

- 哈夫曼编码是广泛用于数据文件压缩的编码方法，其使用字符在文件中出现的频率表来建立一个用0,1串表示各个字符的最优表示方式。

Google

Draco 3D compressor

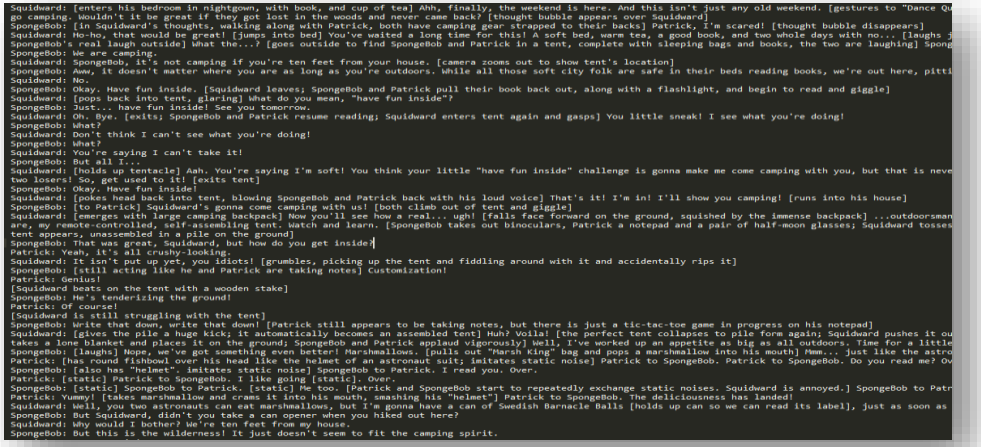
facebook

Zstandard

中国MOOC 8.2 最优前缀码及哈夫曼算法
8.3 哈夫曼算法的正确性证明

哈夫曼编码-实例

- 《海绵宝宝》的剧本
  - 剧集：The Camping Episode（第三季17集）
  - 共12953个字符
  - 占13KB　**如何压缩？**



Squidward: [enters his bedroom in nightgown, with book, and cup of tea] Ahh, finally, the weekend is here. And this isn't just any old weekend. [gestures to "Dance Qu
go camping. Wouldn't it be great if they got lost in the woods and never came back? [thought bubble appears over Squidward]
SpongeBob: [in Squidward's thoughts, walking along with Patrick, both have camping gear strapped to their backs] Patrick, I'm scared! [thought bubble disappears]
Squidward: Ho-hum, that would be great! [jumps into bed] You've waited a long time for this! A soft bed, warm tea, a good book, and two whole days with no... [laughs j
SpongeBob's real laugh outside] What the...? [goes outside to find SpongeBob and Patrick in a tent, complete with sleeping bags and books, the two are laughing] Spong
SpongeBob: We are camping.
Squidward: SpongeBob, it's not camping if you're ten feet from your house. [camera zooms out to show tent's location]
SpongeBob: Aww, it doesn't matter where you are as long as you're outdoors. While all those soft city folk are safe in their beds reading books, we're out here, pitti
Squidward: No.
SpongeBob: Okay. Have fun inside. [Squidward leaves; SpongeBob and Patrick pull their book back out, along with a flashlight, and begin to read and giggle]
Squidward: [pops back into tent, glaring] What do you mean, "have fun inside"?
SpongeBob: Just... have fun inside! See you tomorrow.
Squidward: Oh. Bye. [exits; SpongeBob and Patrick resume reading; Squidward enters tent again and gasps] You little sneak! I see what you're doing!
SpongeBob: What?
Squidward: Don't think I can't see what you're doing!
SpongeBob: What?
Squidward: You're saying I can't take it!
SpongeBob: But all I...
Squidward: [holds up tentacle] Aah. You're saying I'm soft! You think your little "have fun inside" challenge is gonna make me come camping with you, but that is neve
two losers! So, get used to it! [exits tent]
SpongeBob: Okay. Have fun inside!
Squidward: [pokes head back into tent, blowing SpongeBob and Patrick back with his loud voice] That's it! I'm in! I'll show you camping! [runs into his house]
SpongeBob: [to Patrick] Squidward's gonna come camping with us! [both climb out of tent and giggle]
Squidward: [emerges with large camping backpack] Now you'll see how a real... ugh! [falls face forward on the ground, squished by the immense backpack] ...outdoorsman
are, my remote-controlled, self-assembling tent. Watch and learn. [SpongeBob takes out binoculars, Patrick a notepad and a pair of half-moon glasses; Squidward tosses
tent appears, unassembled in a pile on the ground]
SpongeBob: That was great, Squidward, but how do you get inside?
Patrick: Yeah, it's all crushy-looking.
Squidward: It isn't put up yet, you idiots! [grumbles, picking up the tent and fiddling around with it and accidentally rips it]
SpongeBob: [still acting like he and Patrick are taking notes] Customization!
Patrick: Genius!
[Squidward beats on the tent with a wooden stake]
SpongeBob: He's tenderizing the ground!
Patrick: Of course!
[Squidward is still struggling with the tent]
SpongeBob: Write that down, write that down! [Patrick still appears to be taking notes, but there is just a tic-tac-toe game in progress on his notepad]
Squidward: [gives the pile a huge kick; it automatically becomes an assembled tent] Huh? Voila! [the perfect tent collapses to pile form again; Squidward pushes it ou
takes a lone blanket and places it on the ground; SpongeBob and Patrick applaud vigorously] Well, I've worked up an appetite as big as all outdoors. Time for a little
SpongeBob: [laughs] Nope, we've got something even better! Marshmallows. [pulls out "Marsh King" bag and pops a marshmallow into his mouth] Mmm... just like the astro
Patrick: [has round fishbowl over his head like the helmet of an astronaut suit; imitates static noise] Patrick to SpongeBob. Patrick to SpongeBob. Do you read me? Ov
SpongeBob: [also has "helmet"; imitates static noise] SpongeBob to Patrick. I read you. Over.
Patrick: [static] Patrick to SpongeBob. I like going [static]. Over.
SpongeBob: [static] SpongeBob to Patrick. [static] Me too. [Patrick and SpongeBob start to repeatedly exchange static noises. Squidward is annoyed.] SpongeBob to Patr
Patrick: Yummy! [takes marshmallow and crams it into his mouth, smashing his "helmet"] Patrick to SpongeBob. The deliciousness has landed!
Squidward: Well, you two astronauts can eat marshmallows, but I'm gonna have a can of Swedish Barnacle Balls [holds up can so we can read its label], just as soon as
SpongeBob: But Squidward, didn't you take a can opener when you hiked out here?
Squidward: Why would I bother? We're ten feet from my house.
SpongeBob: But this is the wilderness! It just doesn't seem to fit the camping spirit.

回顾：图像的变位压缩存储

- 变位压缩的思想：灰度值序列分段都用小于8位的数字表示一个灰度值

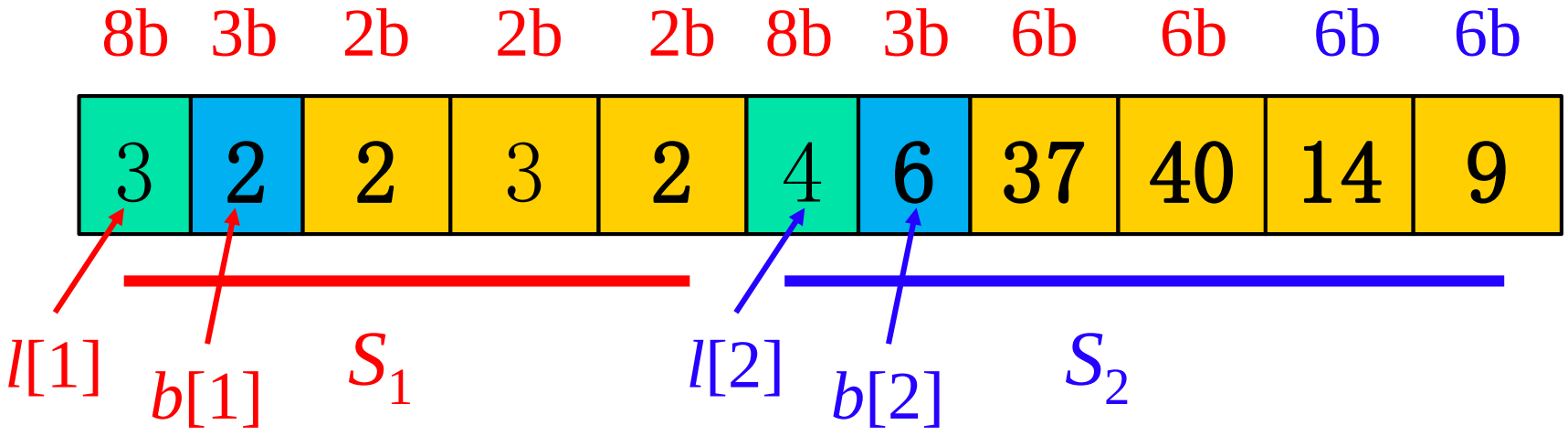
2	3	2	37	40	14	9
2 bits	2 bits	2 bits	6 bits	6 bits	4 bits	4 bits

压缩前占用比特数：8×7=56

实际需要比特数：2+2+2+6+6+4+4=26

回顾：图像的变位压缩存储

- 变位压缩存储
  - 优点：减少表示一个元素所需的位数
  - 缺点：额外的存储空间（$l[i]$和$b[i]$）



哈夫曼编码—问题描述

- 用字符在文件中出现的频率表来建立一个用0，1串表示各字符的最优表示方式。
  - 给出现频率高的字符较短的编码
  - 给出现频率较低的字符以较长的编码

采用变长码
总码长减少25%

	a	b	c	d	e	f
频率（千次）	45	13	12	16	9	5
定长码	000	001	010	011	100	101
变长码	0	101	100	111	1101	1100

二元前缀码：对每一个字符规定一个0，1串作为其代码并要求任一字符的代码都不是其他字符代码的前缀。

000101100111 => 0|0|0|101|100|111 => aaabcd

二元前缀码

- 二元前缀码
  - 用0-1字符串作为代码表示字符，要求任何字符的代码都不能作为其他字符代码的前缀
  - 非前缀码的例子

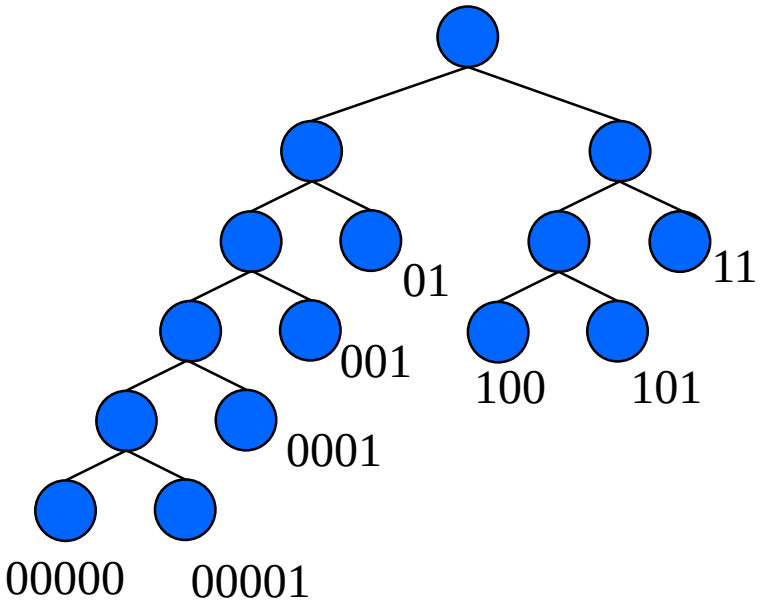
    a: 001, b: 00, c:010, d:01
  - 解码的歧义，例如字符串0100001
    - 解码1： 01, 00, 001　d, b, a
    - 解码2： 010, 00, 01　c, b, d

前缀码的二叉树表示

- 前缀码：

  {00000, 00001, 0001, 001, 01, 100, 101, 11}

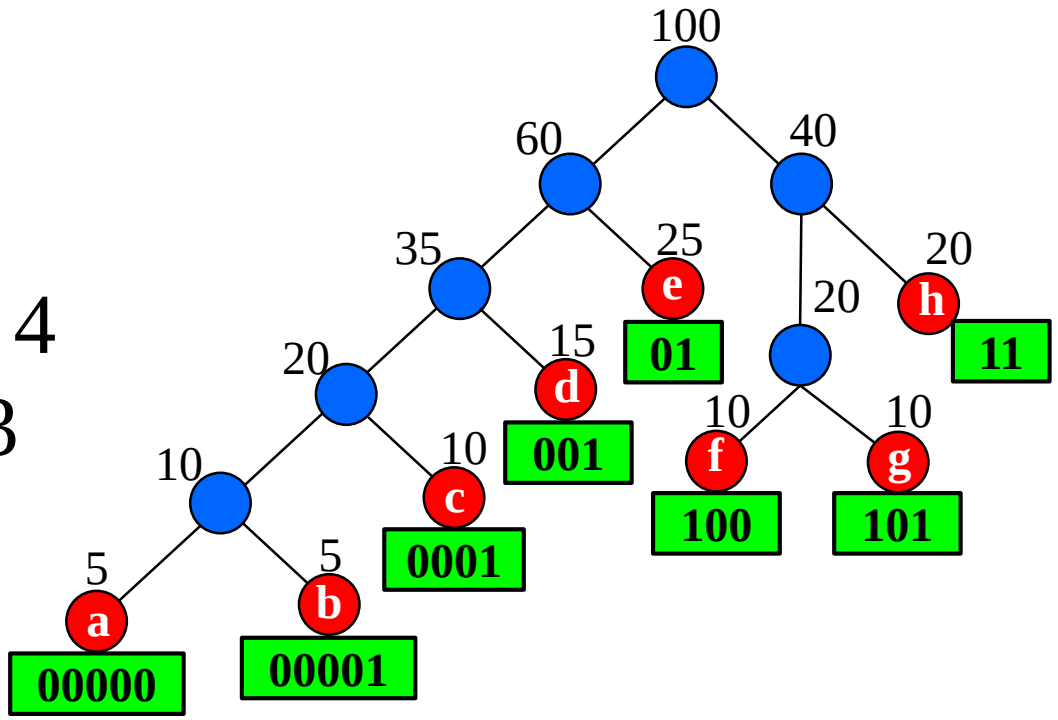
- 构造树：
  - 0-左子树
  - 1-右子树
  - 码对应一片树叶
  - 最大位数为树深



平均码长

$$B=\sum_{i=1}^{n} f(x_i)\,d(x_i)$$

频率　深度



$$B=[(5+5)\times 5+10\times 4+(15+10+10)\times 3+(25+20)\times 2]\div 100=2.85$$

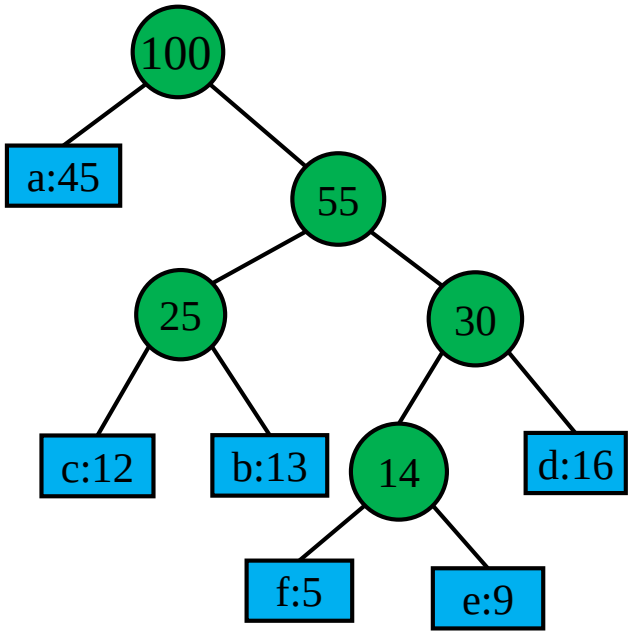
问题：给定字符集$C=\{x_1,x_2,...,x_n\}$和每个字符的频率$f(x_i)$，$i=1,2,...,n$。求关于C的一个最优前缀码（平均码长最小）。

哈夫曼编码—贪心策略

- 回顾压缩编码的思路
  - 给出现频率高的字符较短的编码
  - 给出现频率较低的字符以较长的编码

- 贪心策略
  - 从底向上构建二叉树
  - 优先合并频率较低的字符，生成新的树节点
  - 依次合并，直到完成构建

时间复杂度：$O(n\log n)$



哈夫曼树算法伪码

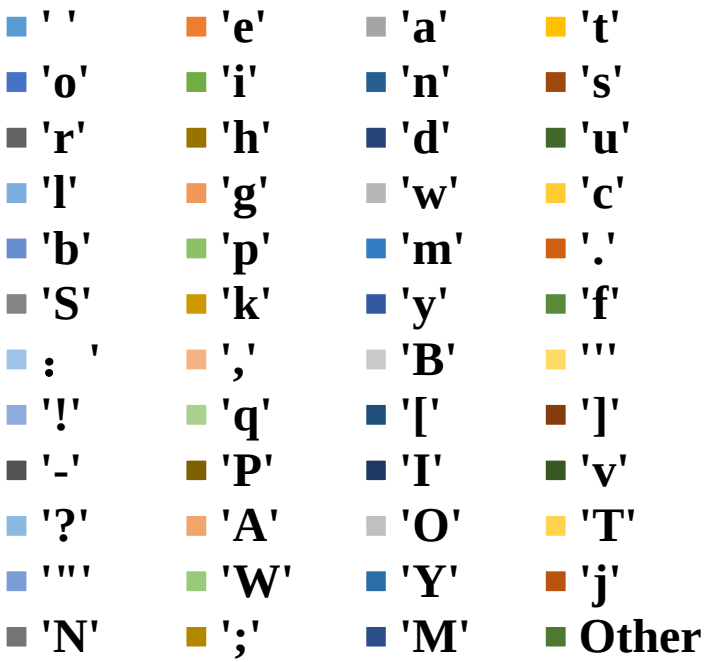
- 算法 Huffman(C)
- 输入：C={x_1,x_2,...,x_n}, $f(x_i)$, i=1,2,...,n
- 输出：Q //队列

```
1.  n ← |C|
2.  Q ← C   //频率递增队列Q
3.  for i ← 1 to n-1 do
4.      z ← Allocate-Node()    //生成结点z
5.      z.left ← 取出Q中最小元    //最小元作为z左儿子，pop()操作
6.      z.right ← 取出Q中最小元  //最小元作为z右儿子，pop()操作
7.      f(z) ← f(x)+f(y)
8.      Insert(Q, z)              //将z插入Q，insert()操作
9.  return Q
```

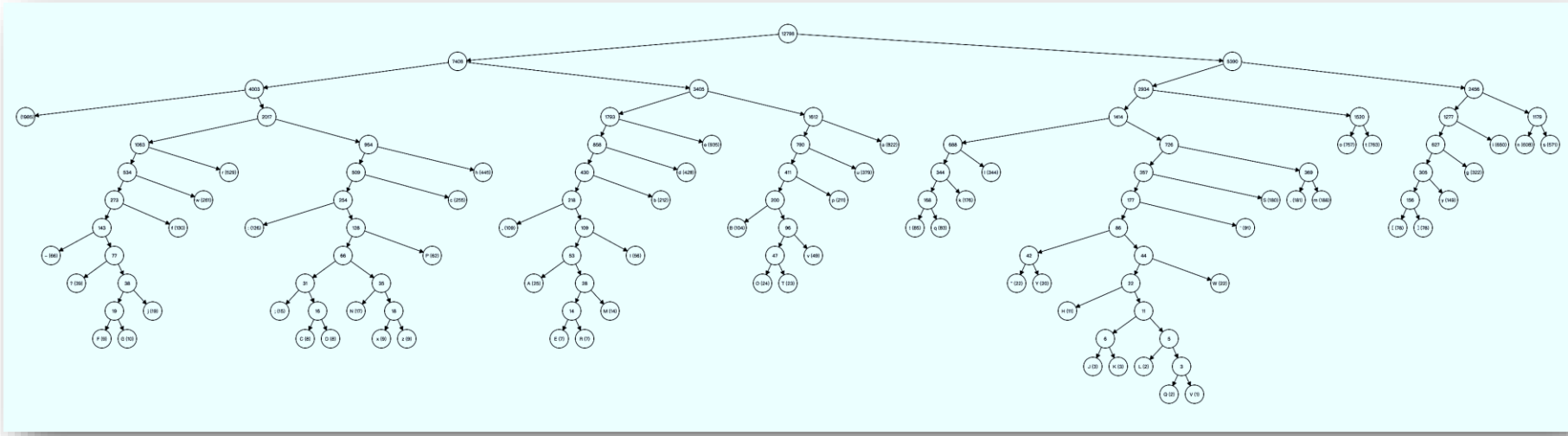
哈夫曼编码-实例的结果

- 《海绵宝宝》的剧本
  - 压缩前占13KB
  - 压缩后占7KB
- 频率排名前十的字符

字符	频率	字符	频率
空格	15.5%	*i*	5.0%
e	7.3%	*n*	4.7%
a	6.4%	*s*	4.4%
t	5.9%	*r*	4.1%
o	5.9%	*h*	3.4%



哈夫曼编码-实例的结果



最短的编码为：000（空格）
最长的编码：1001000101110（Q）和1001000101111（V）

https://people.ok.ubc.ca/ylucet/DS/Huffman.html

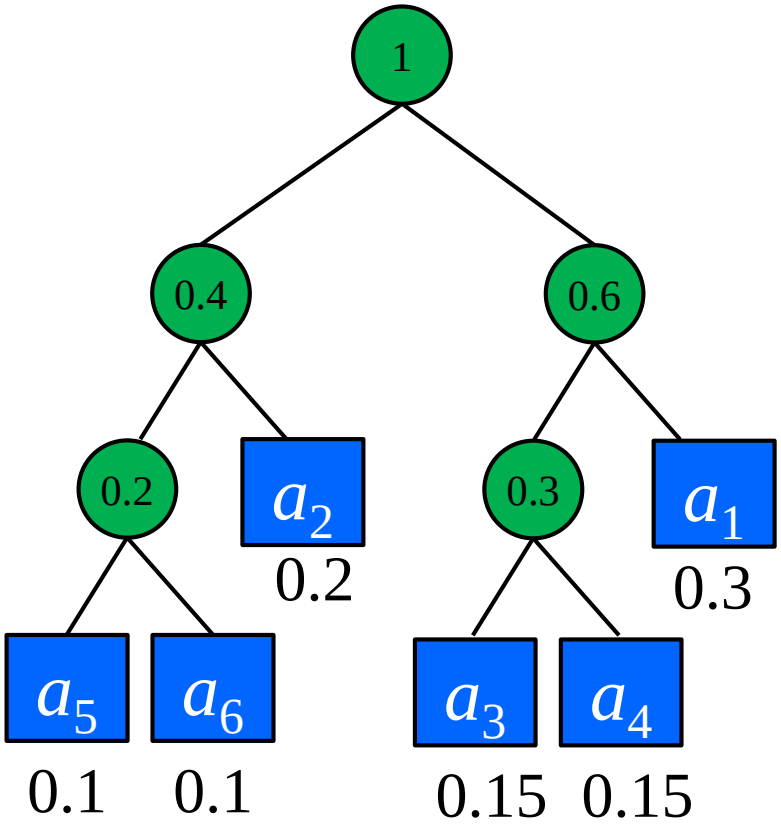
课堂练习

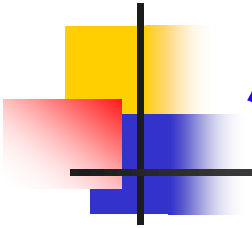
输入：a_1:0.3, a_2:0.2, a_3:0.15, a_4:0.15, a_5:0.1, a_6:0.1

编码：

- $a_1 \rightarrow 11$
- $a_2 \rightarrow 01$
- $a_3 \rightarrow 100$
- $a_4 \rightarrow 101$
- $a_5 \rightarrow 000$
- $a_6 \rightarrow 001$

平均码长？



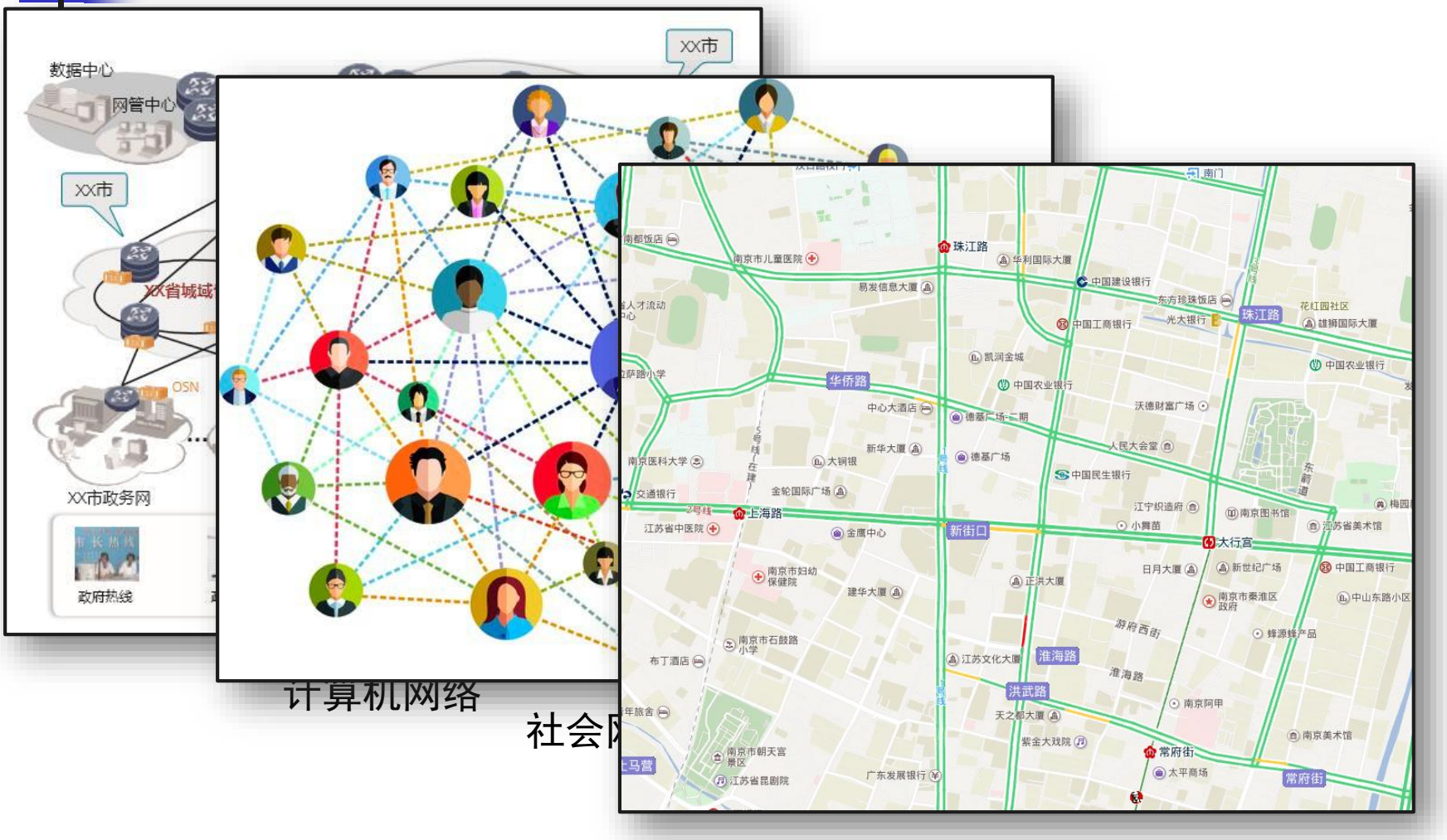


小结

- 哈夫曼编码的应用
- 二叉前缀码及其性质
- 哈夫曼编码的基本流程
  - 贪心选择性质
- 哈夫曼编码的实例

图问题中的贪心算法

图问题的应用



计算机网络

社会网

在线地图

最小生成树

无向连通带权图

$$G=(V,E,W)$$

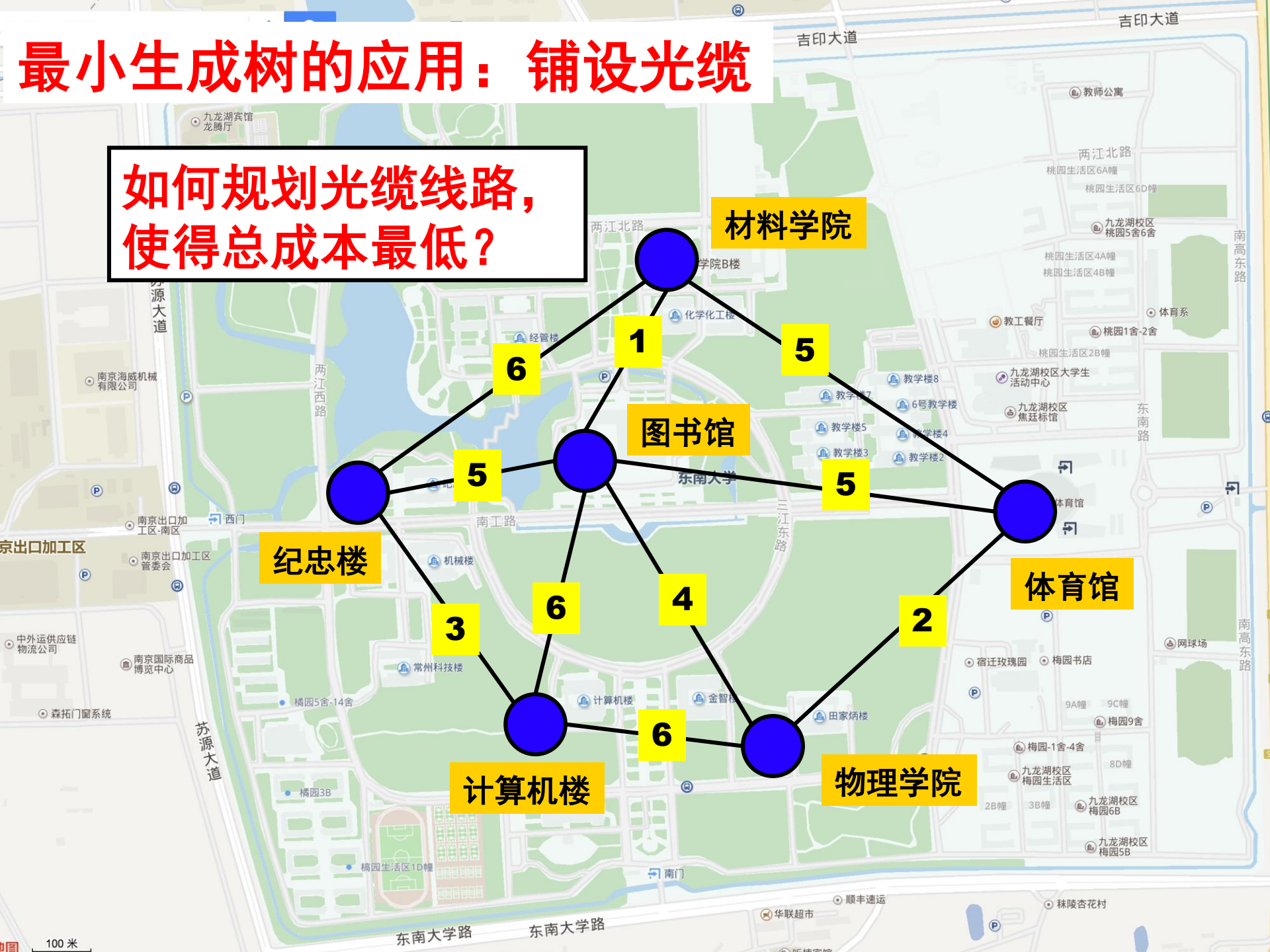
其中$w(e)\in W$是边e的权值。

G的一棵生成树T是包含了G所有顶点的树，树中各边的权值之和$W(T)$称为树的权，具有最小权的生成树称为G的最小生成树。

中国MOOC 8.4 最小生成树
8.5 Prim算法
8.6 Kruskal算法

最小生成树的应用：铺设光缆

如何规划光缆线路，使得总成本最低？

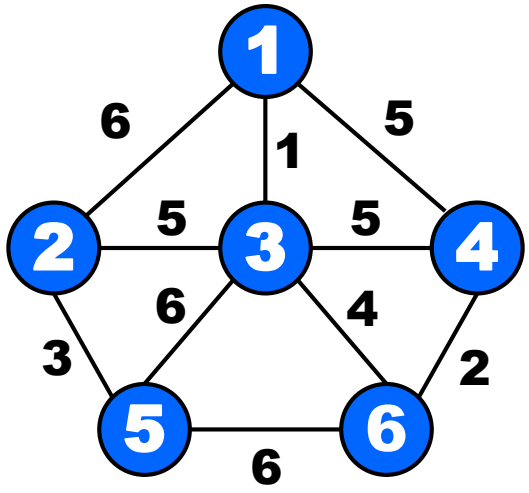


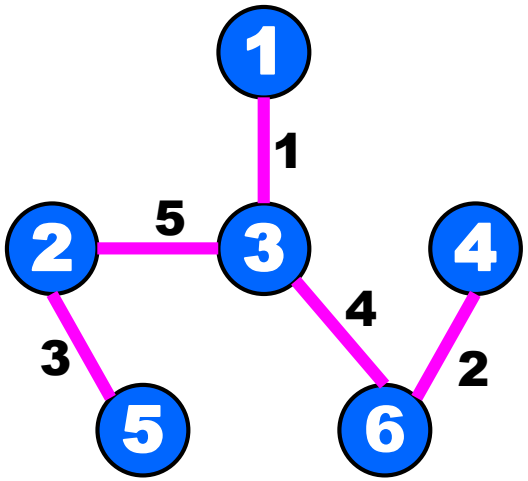
最小生成树的实例

$G=(V, E, W)$, $V=\{1,2,3,4,5,6\}$, W如图所示。

$E=\{\{1,2\},\{1,3\},\{1,4\},\{2,3\},\{2,5\},$

$\{3,4\},\{3,5\},\{3,6\},\{4,6\},\{5,6\}\}$





求最小生成树

- 问题：
  给定连通带权图$G=(V,E,W)$, $W(e)\in W$是边e的权。求G的一棵最小生成树。

- 贪心法：
  - Prim算法
  - Kruskal算法

生成树在网络中有着重要应用

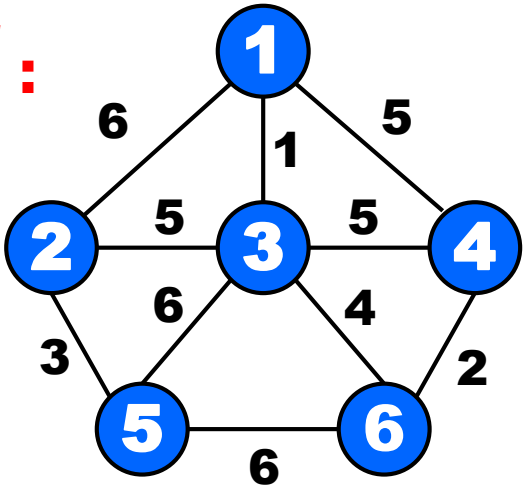
Prim算法

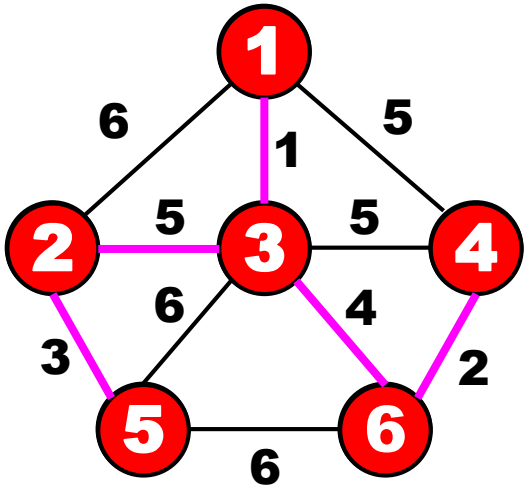
- 设计思想
  - 输入：图$G=(V,E,W)$, V={1,2,...,n}
  - 输出：最小生成树T
  - 步骤：
    - 初始S={1};
    - 选择连接S与V-S集合的最短边$e=\{i,j\}$，其中i∈S, j∈V-S。将e加入树T，j加入S；
    - 继续执行上述过程，直到S=V为止。

Prim算法伪码

- 算法Prim(G,E,W)
  1. S ← {1}
  2. while V–S≠ ∅ do
  3. 从V-S中选择*j*使得*j*到S中顶点的边权最小
  4. S ← S∪{ *j* }

- 实例：





Prim算法的复杂度

- 算法步骤执行$O(n)$次
- 每次执行$O(n)$时间：
  - 找连接S与V-S的最短边
  - 参考课本第104页
- 算法时间：$T(n)=O(n^2)$
- 如果引入堆数据结构，时间复杂度如何？

Kruskal算法

- 设计思想
  - 输入：图G=(V,E,W), V={1,2,...,*n*}
  - 输出：最小生成树T
  - 步骤：
    - 按照长度从小到大对边进行排序；
    - 依次考察当前最短边*e*，如果*e*与T的边不构成回路，则把*e*加入到树T，否则跳过*e*。直到选择了*n*-1条边为止。

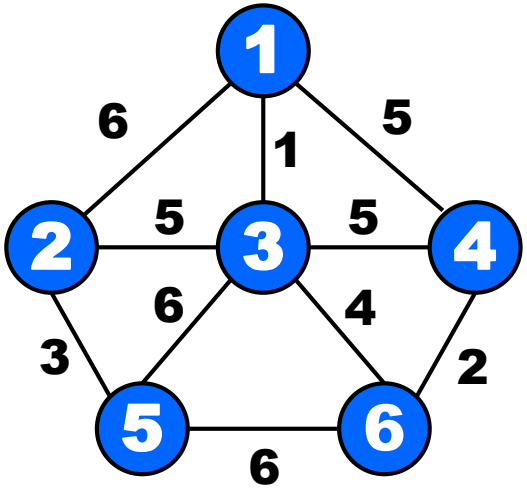
Kruskal算法伪码

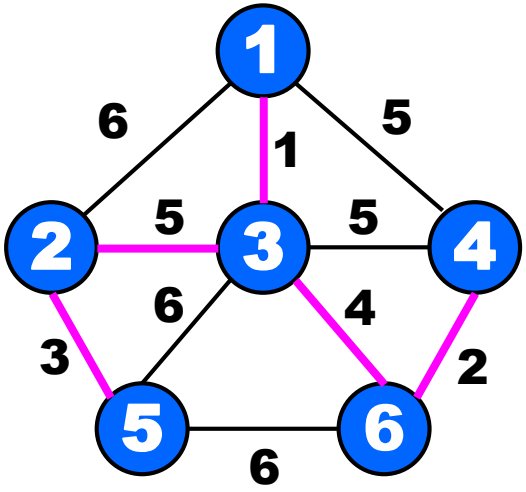
- 输入：图G　//顶点数n，边数m
- 输出：最小生成树T

1. 权从小到大排序E的边，E=$\{e_1,e_2,\ldots,e_m\}$
2. T←∅
3. repeat
4. 　　e←E中的最短边
5. 　　if e的两端点不在同一连通分支
6. 　　then T←T∪$\{e\}$
7. 　　E←E−$\{e\}$
8. until T包含了n-1条边

Kruskal算法

- 实例





Kruskal算法的复杂度

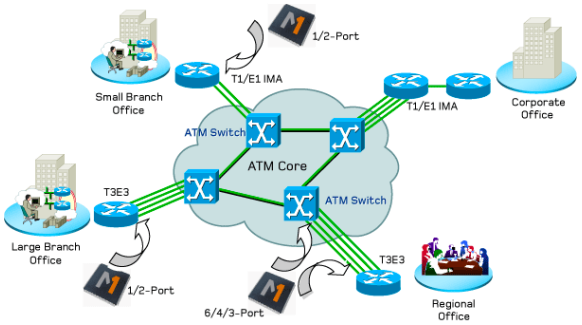
- 对所有的边按权值排序$O(e\log e)$次
- 最多检查e条边，花费$O(e)$时间
- 算法时间：$T(n)=O(e\log e)$

哪些情况用Prim算法，
哪些情况用Kruskal算法？

最短路径问题及其应用

- 最短路径问题是图论研究中的一个经典算法问题，旨在寻找图（由结点和路径组成的）中两结点之间的最短路径。

地图导航



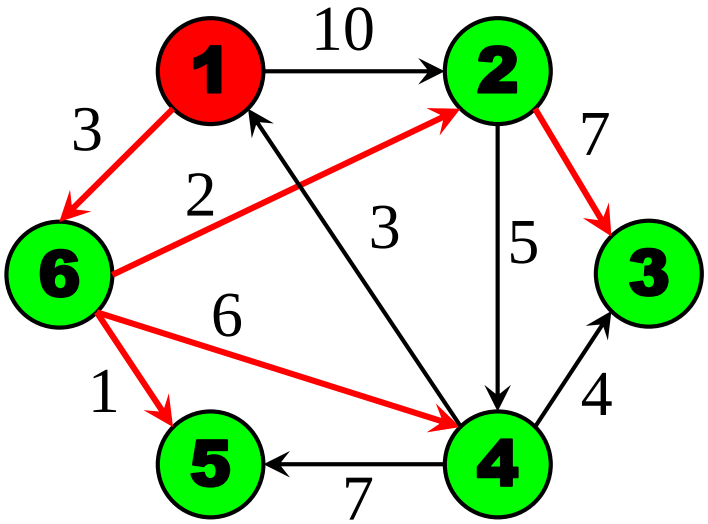
网络路由

Latex排版

单源最短路径问题

给定带权有向图$G=(V,E,W)$，每条边$e=<i,j>$的权$w(e)$为非负实数，表示i到j的距离。源点$s \in V$。

求：从s出发到达其他结点的最短路径。



- 源点：1
- 1→6→2: *short*[2]=5
- 1→6→2→3 : *short*[3]=12
- 1→6→4 : *short*[4]=9
- 1→6→5 : *short*[5]=4
- 1→6 : *short*[6]=3

Dijkstra算法有关概念

- $x \in S \Leftrightarrow x \in V$且从$s$到$x$的最短路径已经找到
- 初始：S={s}，S=V时算法结束
- 从s到u相对于S的最短路径：从s到u且仅经过S中顶点的最短路径
- $dist[u]$：从s到u相对S最短路径的长度
- $short[u]$：从s到u的最短路径的长度
- $dist[u] \geqslant short[u]$

算法的设计思想

- 输入：有向图$G=(V,E,W)$, $V=\{1,2,...,n\}$, $s=1$
- 输出：从s到每个顶点的最短路径
- 步骤：
  1. 初始S={1}；
  2. 对于$i \in$V-S，计算1到i的相对S的最短路径，长度记为$dist[i]$；
  3. 选择V-S中$dist$值最小的j，将j加入到S，修改V-S中的顶点的$dist$值；
  4. 继续上述过程，直到S=V为止。

运行实例

- 输入：$G=(V,E,W)$, $V=\{1,2,3,4,5,6\}$, 源点1

$S=\{1\}$

$dist[1]=0$

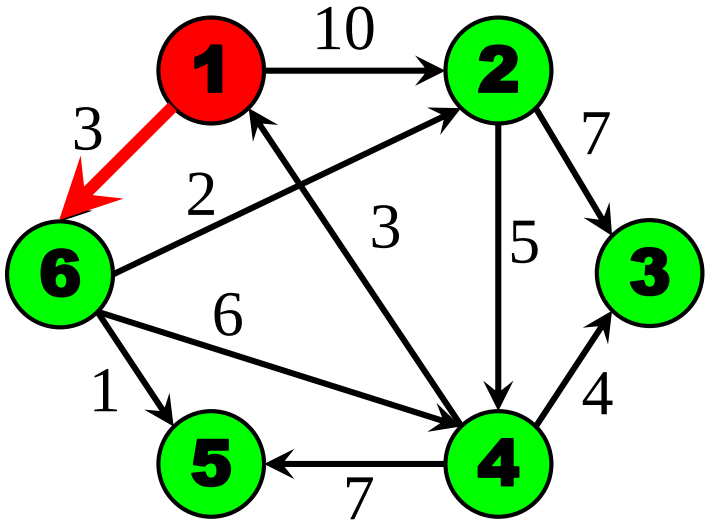
$dist[2]=10$

$dist[6]=3$

$dist[3]=\infty$

$dist[4]=\infty$

$dist[5]=\infty$



运行实例

- 输入：$G=(V,E,W)$, $V=\{1,2,3,4,5,6\}$, 源点1

$S=\{1,6\}$

$dist[1]=0$

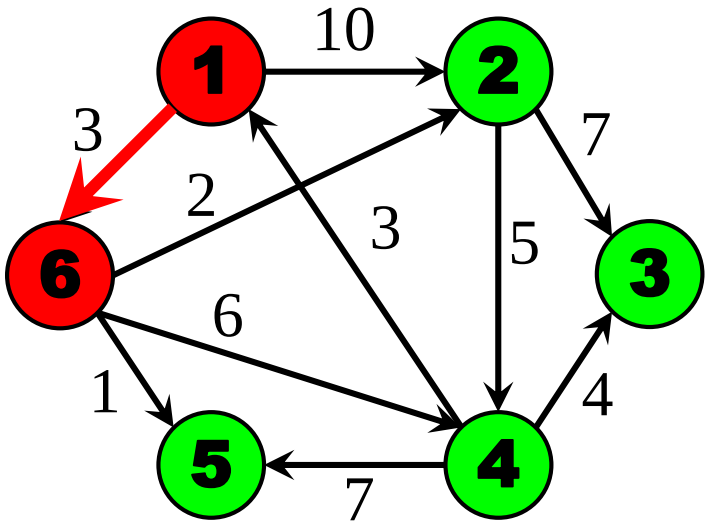
$dist[6]=3$

$dist[2]=5$

$dist[4]=9$

$dist[5]=4$

$dist[3]=\infty$



运行实例

- 输入：$G=(V,E,W)$, $V=\{1,2,3,4,5,6\}$, 源点1

$S=\{1,6,5\}$

$dist[1]=0$

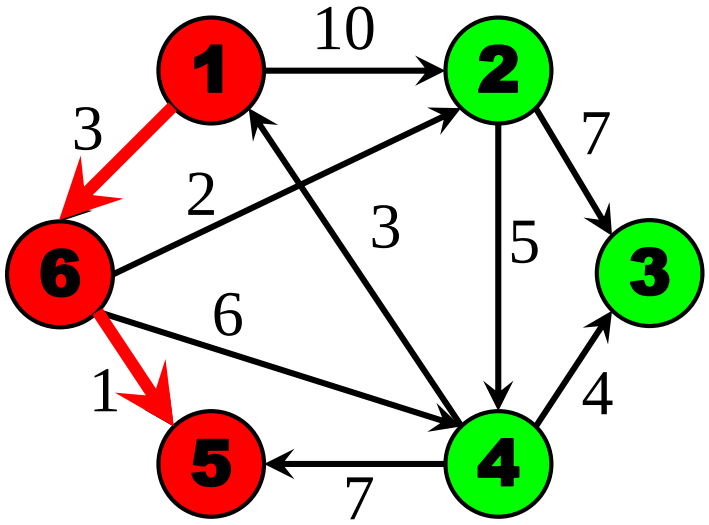
$dist[6]=3$

$dist[5]=4$

$dist[2]=5$

$dist[4]=9$

$dist[3]=\infty$



运行实例

- 输入：$G=(V,E,W)$, $V=\{1,2,3,4,5,6\}$, 源点1

$S=\{1,6,5,2\}$

$dist[1]=0$

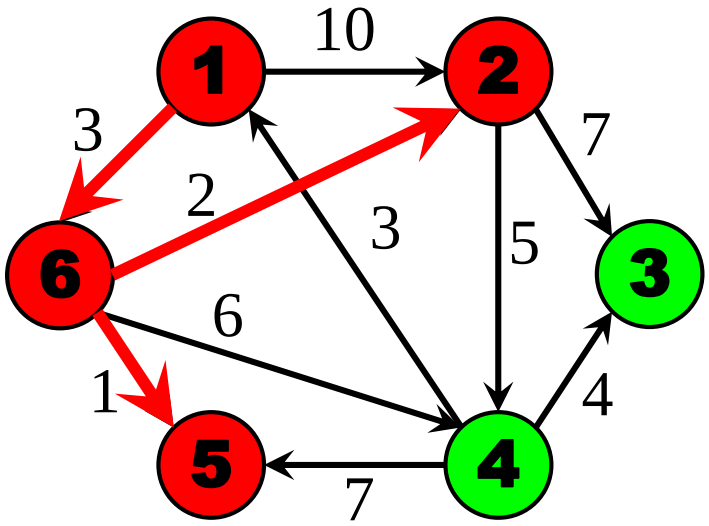
$dist[6]=3$

$dist[5]=4$

$dist[2]=5$

$dist[4]=9$

$dist[3]=12$



运行实例

- 输入：G=(V,E,W), V={1,2,3,4,5,6}, 源点1

S={1,6,5,2,4}

dist[1]=0

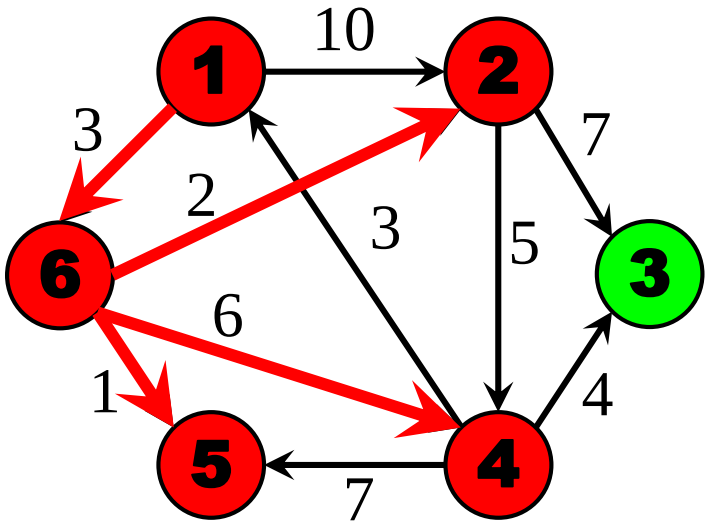
dist[6]=3

dist[5]=4

dist[2]=5

dist[4]=9

dist[3]=12



运行实例

- 输入：$G=(V,E,W)$, $V=\{1,2,3,4,5,6\}$, 源点1

$S=\{1,6,5,2,4,3\}$

dist[1]=0

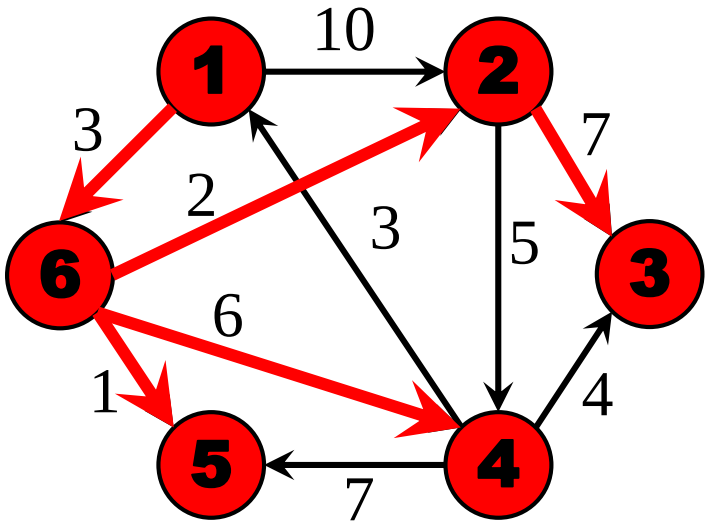
dist[6]=3

dist[5]=4

dist[2]=5

dist[4]=9

dist[3]=12



找到了问题的解！

时间复杂度分析

- 时间复杂度：$O(n^2)$
  - 用数组存储*dist*[]
  - 算法进行n-1步
  - 每步挑选1个具有最小*dist*函数值的结点进入到S，需要$O(n)$时间

- 选用基于堆实现的优先队列的数据结构，可以将算法时间复杂度降低到$O(m\log n)$

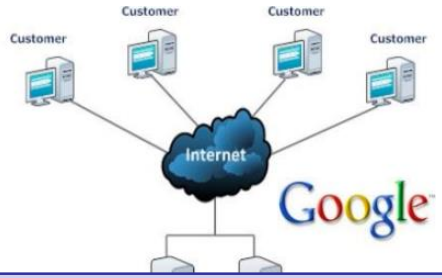
贪心算法求解NP完全问题

回顾：NP完全问题

- NP问题（Non-Deterministic Polynomial）
- P问题的集合
  - 所有可以在多项式时间内求解的判定问题。
- NP完全问题的集合
  - NP中某些问题不确定能否在多项式时间内求解。这些问题中任何一个如果存在多项式时间的算法，那么所有NP问题都是多项式时间可解
- P=NP？
  - 七个“千僖年数学难题”之一

多机调度问题及其应用

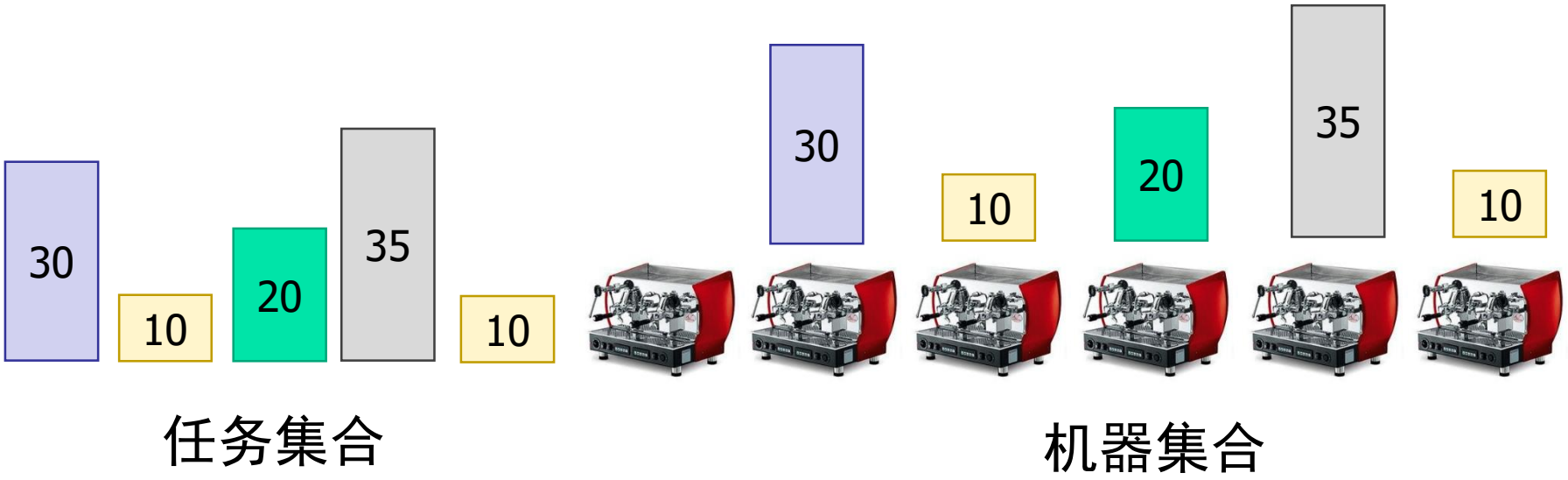
- 多机调度问题要求给出一种作业调度方案，使所给的n个作业，每个作业运行时间为t_i，要求在尽可能短的时间内由m台相同的机器加工处理完成。



多机调度问题是NP完全问题，到目前为止还没有有效的解法。对于这一类问题，用贪心选择策略有时可以设计出较好的近似算法。

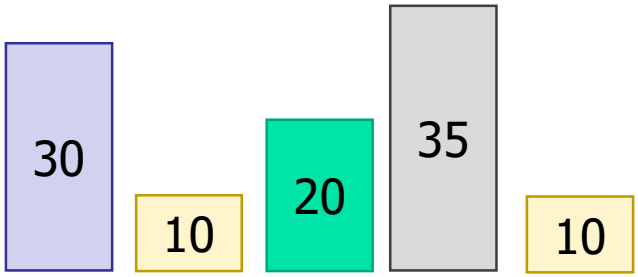
多机调度问题—贪心策略

- 当机器数$m \geq$任务数n时
  - 每台机器放一个任务

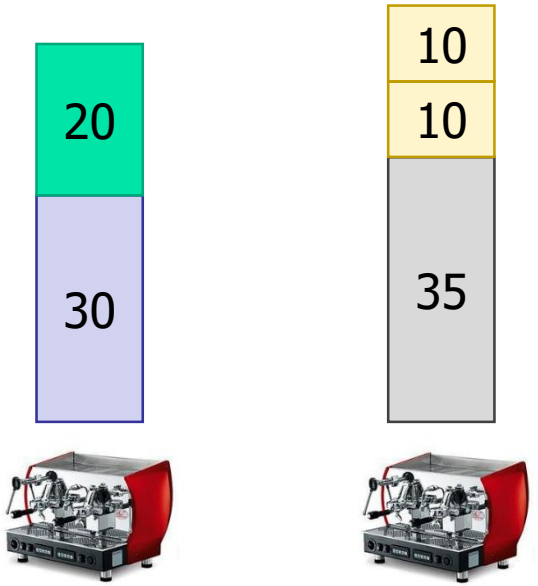


多机调度问题—贪心策略

- 当机器数m<任务数n时
  - 优先放长任务
  - 优先选择负载最轻机器



任务集合

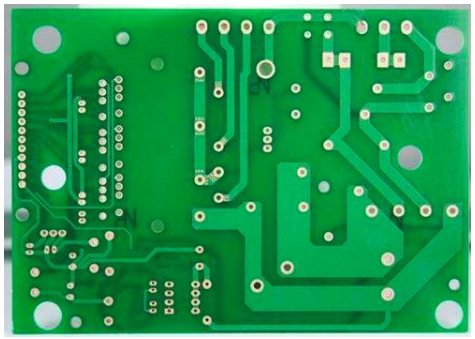


机器集合

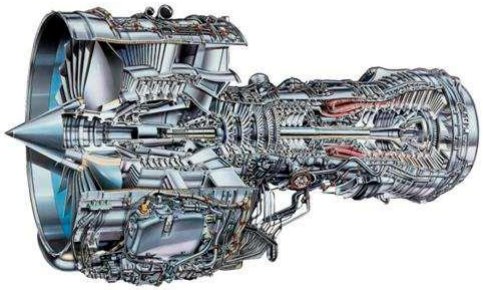
旅行商问题

- 旅行商问题（Travelling Salesman Problem,TSP）：旅行家旅行*n*个城市，要各城市经历且经历一次，然后回到源点，求出最短路程。

规划快递线路



电路板钻洞



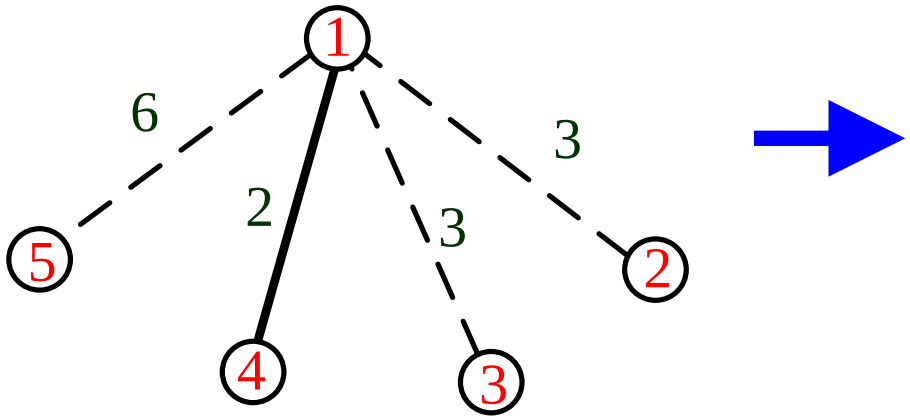
飞机发动机维修

旅行商问题—最短链接策略

- 最近邻点策略：
  - 从任意城市出发，每次在没有到过的城市中选择最近一个，直到经所有城市回到出发点。
- 将5城市的代价矩阵列出：

$$C=\begin{bmatrix} \infty & 3 & 3 & 2 & 6 \\ 3 & \infty & 7 & 3 & 2 \\ 3 & 7 & \infty & 2 & 5 \\ 2 & 3 & 2 & \infty & 3 \\ 6 & 2 & 5 & 3 & \infty \end{bmatrix}$$

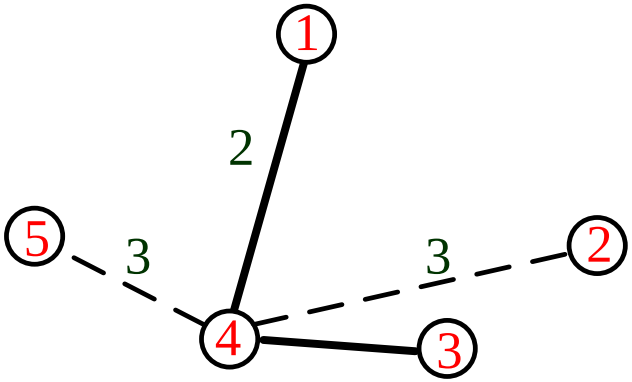
城市1→4：



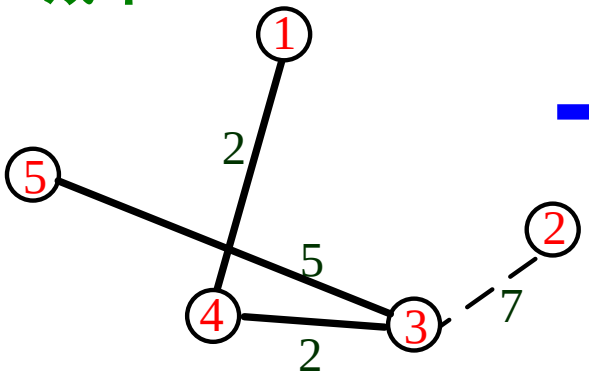
旅行商问题—最短链接策略

$$C=\begin{bmatrix}\infty & 3 & 3 & 2 & 6\\ 3 & \infty & 7 & 3 & 2\\ 3 & 7 & \infty & 2 & 5\\ 2 & 3 & 2 & \infty & 3\\ 6 & 2 & 5 & 3 & \infty\end{bmatrix}$$

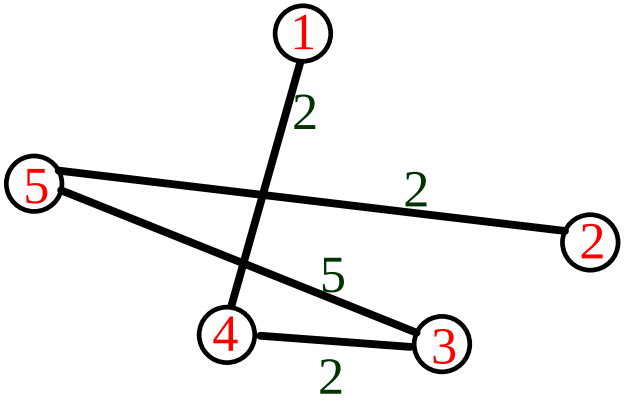
城市4→3：



城市3→5：



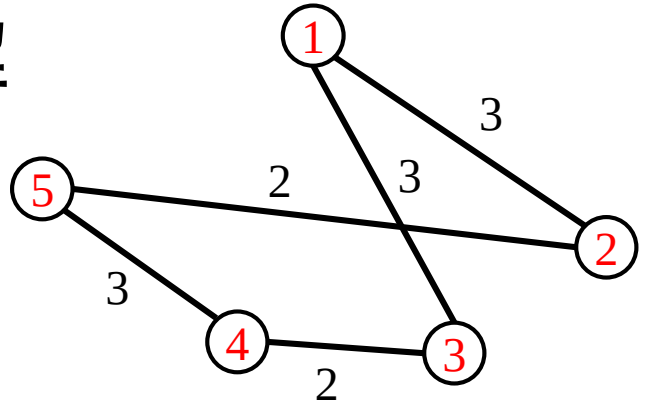
城市5→2→1：



代价为14

旅行商问题—最短链接策略

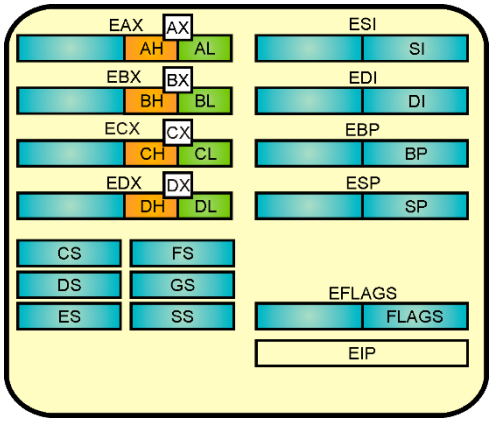
- 时间复杂度$O(n^2)$
  - 因为n-1次贪心选择，每次都找满足条件的最短边。
- 贪心算法不一定是最优解
  - 最优解是1→2→5→4→3→1
  - 最优解代价为13



图着色问题及其应用

- 给定无向连通图$G=(V,E)$，求图G的最小色数k，使得用k种颜色对G中顶点着色，可使任意两个顶点着色不同。
  - k个颜色的集合为{颜色1，颜色2，…，颜色k}。

地图着色



程序编译器的
寄存器分配算法

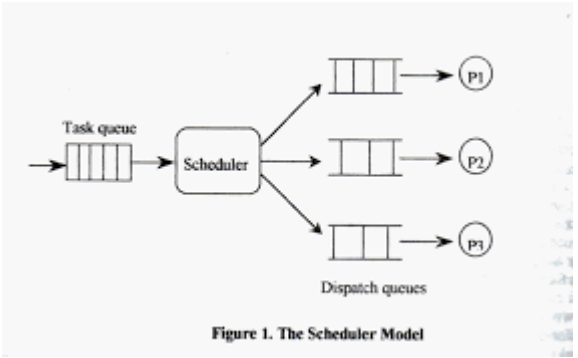


Figure 1. The Scheduler Model

任务调度

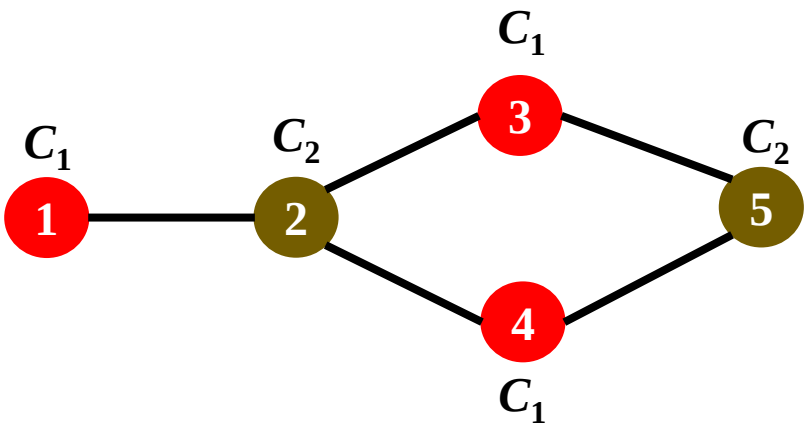

乌兰巴托
Ulaanbaatar
蒙古
阿拉木图
Алматы
什凯克
Бишкек
吉尔吉斯斯坦
拉合尔
Lahore
新德里
New Delhi
尼泊尔
不丹
孟加拉国
印度
那格浦尔
Nagpur
蒙拉
吉大港
缅甸
河内
Hà Nội
老挝
仰光
海得拉巴
孟买
北朝鲜
首尔
南韩
日本
金沢
松江
釜山
山口
大阪
松山
熊本
鹿児島
宮崎
东海
那覇
台湾
香港
澳门
菲律宾海
500 千米
500 英里
Google
地图数据 ©2012 GS(2011)6020 Basarsoft, Google, MapIT, ORION-ME, SK M&C, T

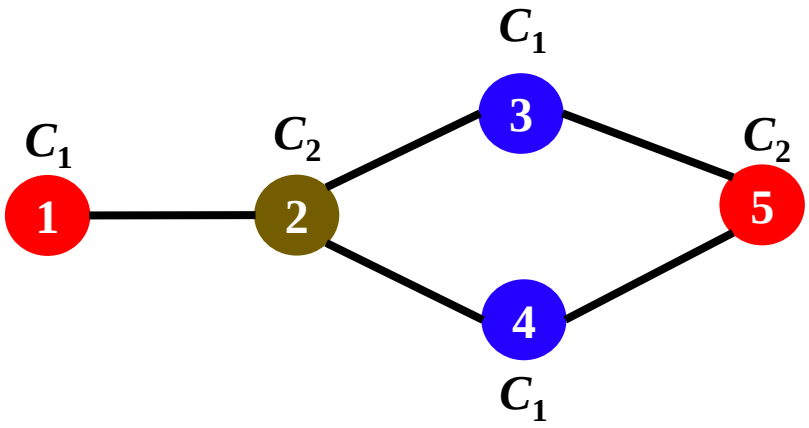
图着色问题

- 贪心策略：以任意顶点作为开始，依次考虑未被着色的每个顶点
  - 如果一顶点可用颜色1着色，即该顶点的邻接点都还未被着色，则用颜色1为该顶点着色；
  - 当没有顶点能以这种颜色着色时，选择颜色2和一个未被着色的顶点作为开始顶点，用第二种颜色尽可能多的顶点着色；
  - 如果还有未着色的顶点，则选颜色3，尽可能多的着色，依此类推。

图着色问题

- 例如：





如果求解顶点顺序是：

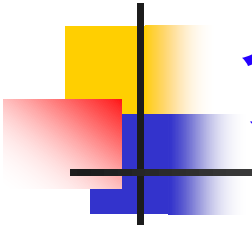
1，2，3，4，5

得到最优解。

如果求解顶点顺序是：

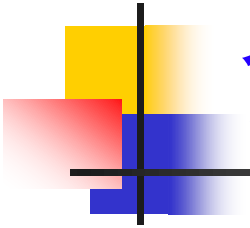
1，5，2，3，4

得到近似解。



贪心法小结

- 贪心法适用于组合优化问题
- 求解过程是多步判断过程，最终的判断序列对应于问题的最优解
- 判断依据某种“短视”贪心选择性质，性质的好坏决定了算法的成败。贪心性质往往依赖于直觉或者经验



贪心法小结(cont.)

- 贪心法正确性证明方法：
  - 直接计算优化函数，贪心法的解恰好取得最优值
  - 数学归纳法（对算法步骤或者问题规模归纳）
  - 交换论证
- 证明贪心策略不对：举反例

贪心法小结(cont.)

- 对于某些不能保证对所有的实例都得到最优解的贪心算法（近似算法），可做参数化分析或者误差分析。
- 贪心算法的优势：
  - 算法简单
  - 时空复杂度低
- 几个著名的贪心算法
  - 最小生成树的Prim算法和Kruskal算法
  - 单源最短路径的Dijkstra算法