算法基础及效率分析

可无输入，至少1输出

$O(n)$: < | $\Theta(n)$: = | $\Omega(n)$: >

非递归：$T(n)=C_{op}C(n)$

递归：例：汉诺塔

$$\begin{cases} M(n)=2M(n-1)+1 & (n>1) \\ M(1)=1 \end{cases}$$

$M(n)=2[2(M(n-2)+1)]+1=\cdots$

算法能力极限：

平凡下界：由输入、输出元素数决定

信息论下界：基于比较的排序算法，其紧密下界为 $\Omega(n\log n)$ → 决策树

敌手下界：恶意、又一致　查找 $\Omega(\log_2(n+1))$

恶意：推向最消耗时间路径

一致：迫使与已做选择一致

P与NP：

易解：可在多项式时间内求解 ↔ 难解

P：确定性　多项式时间，判定问题

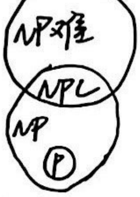
→ 不可判定问题：图灵问题

NP：不确定算法（猜测、验证（多项式））

判定问题

NPC：属于NP，可在多项式时间化简成其　← NP中任何问题

NP难：不一定属于NP，其余同NPC



排序总览

方法	Caver	Cworst	空间	稳定
插入	$O(n^2)$	$O(n^2)$	$O(1)$	✓
希尔	$O(n^{1.3})$	$O(n^2)$	$O(1)$	×
选择	$O(n^2)$	$O(n^2)$	$O(1)$	×
堆	$O(n\log n)$	$O(n\log n)$	$O(1)$	×
冒泡	$O(n^2)$	$O(n^2)$	$O(1)$	✓
快速	$O(n\log n)$	$O(n^2)$	$O(\log n)$	×
归并	$O(n\log n)$	$O(n\log n)$	$O(1)$	✓
计数	$O(n+k)$	$O(n+k)$	$O(n+k)$	✓
桶	$O(n+k)$	$O(n^2)$	$O(n+c)$	✓
基数	$O(nk)$	$O(nk)$	$O(n+k)$	✓

选择排序、冒泡排序（蛮力法）

插入排序（减治法）→ 希尔排序

依次将后面元素插到前面已经排好的数组中

合并排序（分治法）

将数组不断一分为二，直至不可分，随后自底向上合并 → 合并关键

快速排序（分治法）

将数组不断以第一个元素作为中轴(p)进行划分，直至不可划分

→ 划分关键

划分

Lomuto划分　| p | <p | ≥p | ? |　（l, s, i, r）

从左往扫描　若 $A[i]\geq p$　$i=i+1$

若 $A[i]<p$　$s=s+1$　$A[s]\leftrightarrow A[i]$

快速选择（第k小/中位数）　| p | <p | ≥p |

对两侧继续划分　| <p | p | ≥p |

Hoare划分

分别从左右两侧扫描

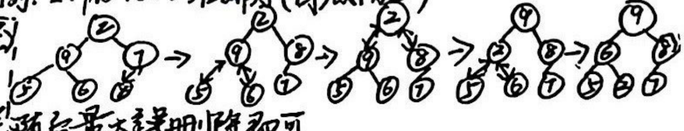
左忽略小值，右忽略大值

| p | 全部≤p | ≥p | … | ≤p | 全部≥p |　（i →, ← j）

不相交 $(i<j)$，$A[i]\leftrightarrow A[j]$，i,j++

只要相交 $(i\geq j)$，停止扫描，$p\leftrightarrow A[j]$

例：2, 9, 7, 5, 6, 8 堆排序（自底向上）



随后最大键删除即可

堆排序

堆的概念：

- 基本完备：二叉树每一层均满（除叶子层最右节点可空）
- 父母优势：父母 ≥ 子女

堆排序（变治法）

① 构造堆 $O(n)$

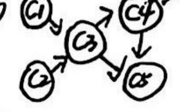
- 自底向上：从右向左，从下到上依次检查父母优势，交换后继续检查换下的父母（交换最大子女）
- 自顶向下：插入一个新节点，比较父母优势，直到满足

② 删除最大键 $O(n\log n)$：根键与最后一个键交换 → 删除最大键

→ 自底向上构造新堆（删除键为降序）

拓扑排序 → 无环有向图才有解（~~减1治法~~）

① 深度遍历



C5 1
C4 2
C3 3
C1 4
C2 5

出栈次序 C5, C4, C3, C1, C2

拓扑排序

C2　C1 → C3 → C4 → C5

② 减1治法

依次删除没有输入边顶点，删除次序即为一个解

C1, C2, C3, C4, C5　（解不唯一）

计数排序（时空权衡，输入增强）

比较计数排序：对列表中每个元素计算小于它元素个数，即为排序序号

例：

	62	31	84	96	19	47
一遍 count	3	0	1	1	0	0

选中一个向后扫，小于自己+1，大于自己对方+1

分布计数：例：对13、11、12、13、12、12排序　{11, 12, 13}

数组值	11	12	13
频率	1	3	2
分布值	1	4	6

分布值：有序数组中元素最后一次出现的位置

从后向前按（分布值−1）存放元素，再将分布值−1

分布计数只用一个桶

桶排序：将待排元素分到k个桶中，如将0-100分为0-10, 10-20, …, 90-99、

基数排序：将数组元素统一位数，从最低位到最高位分别计数排序

基于树的查找 → 二叉查找树

二叉查找树（左节子女 < 父母 < 右子女）　查找 Caver=log n

遍历：前序（根→左→右）　中序（左→根→右）

后序（左→右→根）

最优二叉查找树构造（最少平均查找次数）

例：

Key	A	B	C	D
P	0.1	0.2	0.4	0.3

主表

C	0	1	2	3	4
1	0	0.1	0.4		
2		0	0.2		
3			0	0.4	
4				0	0.3
5					0

根表

A	0	1	2	3	4
1		1	2		
2			2		
3				3	
4					4
5					

$C(i,j)$ → 从i到j构成查找树的最少平均 → 关注 $C(1,n)$

$A(i,j)$ → 从i到j构成最优查找树的根节点 → 回溯写结构

$$C(i,j)=\min_{i\leq k\leq j}\{C(i,k-1)+C(k+1,j)\}+\sum_{s=i}^{j}p_s$$

$$C(1,2)=\min\begin{cases} k=1: C(1,0)+C(2,2)+0.3=0.5 \\ k=2: C(1,1)+C(3,2)+0.3=0.4 \end{cases}=0.4$$

$k=2$：$A[1,2]$

数组查找

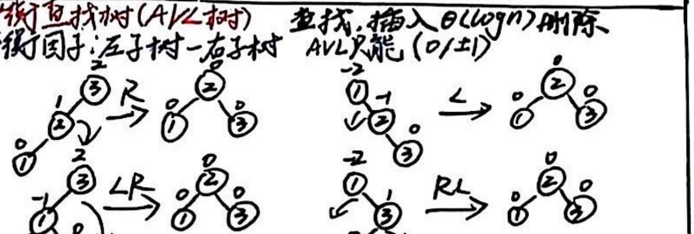
顺序查找（蛮力）　折半查找（减半治）→ 数组必须预排序

$m=\lfloor(l+r)/2\rfloor$ → 中点计算　　折半 $\Theta(\log n)$

平衡查找树（AVL树）

查找、插入 $\Theta(\log n)$ 删除

平衡因子：左子树 − 右子树　　AVL只能（0/±1）

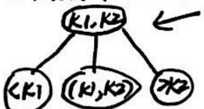


有多个±2节点，旋转以最靠近新插入的±2节点为根的树

平衡查找树（2-3树）

查找、插入、删除 $\Theta(\log n)$　高度平衡特性

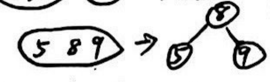
2节点同二叉树，3节点



插入新节点时插到叶子中

如果有3个键，将中间键提为根

例：



$\log_3(n+1)-1\leq h\leq\log_2(n+1)-1$

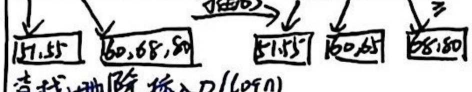
B树（磁盘管理）

次数为m

① 根有2-m子女

② 中间节点有 $\lceil\frac{m}{2}\rceil$-m个子女（$\lceil\frac{m}{2}\rceil-1$ — m−1）个键

③ 高度平衡



查找、删除、插入 $O(\log n)$

散列表查找

散列函数给出元素的散列地址

开散列（分离链）：分配到同一单元格的用链表表示 $\Theta(1)$

闭散列：线性探查解决碰撞，检查碰撞后继单元格，到尾部则折返至头部（接近满时性能恶化）

字符串匹配 Horspool 算法

根据模式最后一个字符在文本中对应的c生成 $t(c)$（移动距离）

$$t(c)=\begin{cases} \text{模式长度} m \text{（c不在前} m-1 \text{个中）} \\ \text{最右边c到最后距离（其他）} \end{cases}$$

BARBER　$t(c)=6$

BARBER　$t(c)=3$

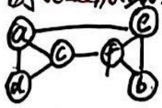
字符串匹配之Boyer-Moore算法 与Horspool不同
坏符号 用K来记录右侧已匹配的字符数
$d_1 = \max\{t(c)-K, 1\}$ 注意c为文本中从右向左第一个不匹配
好后缀 例：对 $\overline{AB}\underline{CBAB}$，K=4情况下 $d_2=4$（注意前缀）
移动距离 $d = \begin{cases} d_1, & K=0 \\ \max\{d_1, d_2\}, & K>0 \end{cases}$
例：……CAB……
$\overline{B}AD\,B\underline{AB}$
$\Rightarrow d_1 = t(C)-2 = 4$
$d_2 = 5$
$d = \max\{4, 5\} = 5$

Warshall（动态规划） $\Theta(n^3)$
计算有向图传递闭包（即存在从i到j路径，矩阵对应R(i,j)为1）
$R^{(0)}$为邻接矩阵，一系列构造出$R^{(n)}$传递闭包

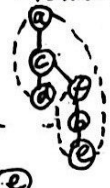
$R^{(K-1)} = \begin{bmatrix} 0&0&0&1 \\ 1&0&0&0 \\ 0&0&0&0 \\ 0&1&0&0 \end{bmatrix} \Rightarrow R^{(K)} = \begin{bmatrix} 0&0&0&1 \\ 1&0&0&0 \\ 0&0&0&0 \\ 0&1&0&0 \end{bmatrix}$

Floyd（动态规划） 计算有向图完全最短路径 $\Theta(n^3)$ 无向也可
从$D^{(0)}$权重矩阵开始，经一系列构造出$D^{(n)}$→距离矩阵
例：$D^{(0)} = \begin{bmatrix} 0&\infty&3&\infty \\ 2&0&\infty&\infty \\ \infty&7&0&1 \\ 6&\infty&\infty&0 \end{bmatrix} \Longrightarrow D^{(1)} = \begin{bmatrix} 0&\infty&3&\infty \\ 2&0&5&\infty \\ \infty&7&0&1 \\ 6&\infty&9&0 \end{bmatrix}$ 对应相加如更小则替代

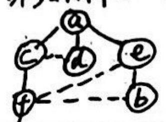
图的遍历（蛮力） DFS（栈） DFS森林



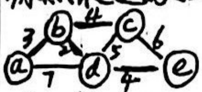
e6,2
b5,3
f4,4
d3,1
c2,5
a1,6



BFS（队列）
$a_1 c_2 d_3 e_4 f_5 b_6$
BFS森林



Dijkstra（贪婪）
单起点最短路径算法
图中不可含负权重
由近至远，依次找出离源点最近的顶点、

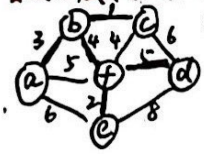


a(-,0) | b(a,3) c(-,∞) d(a,7) e(-,∞)
b(a,3) | c(b,3+4) d(b,3+2) e(-,∞)
d(b,5) | c(b,7) e(d,5+4)
c(b,7) | e(d,9)
e(d,9)
生成一系列路径
标记含义：b(a,3) a为树中父母，3为从源点到此最短路径

最小生成树之Kruskal（贪婪）
将边的权重大小排序，按序访问每条边，如该边加入后不产生回路，则加入最小生成树中

bc	ef	ab	bf	cf	af	df	ae	cd	de
1	2	3	4	4	5	5	6	6	8

最小生成树之Prim（贪婪）：不断将不在树中的最近顶点添加到树中

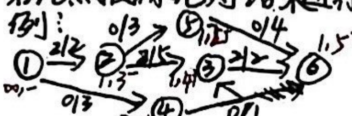


a(-,-) | b(a,3) c(-,∞) d(-,∞) e(a,6) f(a,5)
b(a,3) | c(b,1) d(-,∞) e(a,6) f(b,4)
c(b,1) | d(c,6) e(a,6) f(b,4)
f(b,4) | d(f,5) e(f,2)
e(f,2) | d(f,5)
d(f,5)
节点标记含义：
b(a,3)：a为最近树中顶点、3为边权重

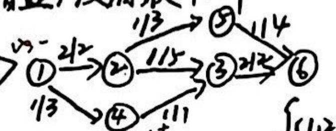
二分图最大匹配（迭代改进）

增益 $\Longrightarrow$
队列：2
2 5 1 从6增益
BFS方式 直到找到另一个集合中自由顶点、
增益路径：一条属于已有匹配，一条不属于，交替进行两端在两个集合的自由顶点、

最大流量（迭代改进：最短增益路径法）
前向边：正的未使用容量 r_{ij}
后向边：正的流量 x_{ij}
先标记先扫描：建立一个队列存放节点、
标记方法 ① 前向边：(l_j, v_i^+) $l_j = \min\{l_i, r_{ij}\}$
② 后向边：(l_j, v_i^-) $l_j = \min\{l_i, x_{ij}\}$
对每个队列中节点、1) 前向边扫描：若未被标记且 $r_{ij}>0$，标记入队
2) 后向边扫描：若未被标记且 $x_{ij}>0$，标记入队
若汇点被标记则结束进行增益，没有取下一个
例：



1 4 3 2 5 6 沿143256增益1



1 4
{(1,2), (4,3)} 最小割
注意增益对反向边是减
最大流量值＝最小割容量；最后一次迭代，从已标记到未标记构成

稳定婚姻（迭代改进）→不存在受阻对 等级矩阵
①求婚：自由男士向其优先列表下一位求婚
②回应：如果被求者自由则接受；不自由与当前配偶比，若更喜欢则换，反之拒绝

哈夫曼树及编码（贪婪） 变长编码
依次找到概率最小的两个子树组合

回溯法 n皇后 子集和 哈密顿回路
每次构造解的一个分量，眼看向前，如遇死胡同则回到上一个分叉路口
状态空间树：深度优先 再来思想高效

单纯形法（迭代改进）→线性规划
① 标准化 必须最大化、线性方程
max $z = 3x + 5y$
s.t. $\begin{cases} x + y \le 4 \\ x + 3y \le 6 \\ x \ge 0 \quad y \ge 0 \end{cases}$
$\Rightarrow$ max $z = 3x + 5y + 0u + 0v$
s.t. $\begin{cases} x + y + u = 4 \\ x + 3y + v = 6 \\ x, y, u, v \ge 0 \end{cases}$

单纯形表

	x	y	u	v		
u	1	1	1	0	4	$\theta_u = 4$
v（主元行→）	1	3	0	1	6	$\theta_v = 2$
目标行→	-3	-5	0	0	0	

$\Rightarrow$

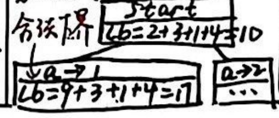
	x	y	u	v	
u	$\frac{2}{3}$	0	1	$-\frac{1}{3}$	2
y	$\frac{1}{3}$	1	0	$\frac{1}{3}$	2
	$-\frac{4}{3}$	0	0	$\frac{5}{3}$	10

当前极点(0, 2, 2, 0)

②最优测试：目标行所有单元格非负，得到最优解
③确定主元列（输入变量）：选择目标行中绝对值max负单元格
④确定主元行（分离变量）：对主元列中每个正单元格，用最右单元格除以主元列中的，比率θ最小的为主元行
⑤迭代：将主元行除以主元，其他行减主元行把主元列其他化0，主元行用主元列标识替代

分支界限法：对每个节点部分解，计算其解衍出的最佳边界值
分配问题

	一	二	三	四	
	9	2	7	8	a
	6	4	3	7	b
	5	8	1	8	c
	7	6	9	4	d



背包问题
①物品按价值/重量降序 ②向左包含，向右不包含
③上界＝已有价值＋剩余容量×最高(价值/重量)
例：
w=0, v=0
ub=100
旅行商（可以只规定b在c前→无向图）
下界＝①每个城市与最近的两个城市距离之和，并计算所有总和/2，向上取整 ②包含特定边先选特定边，再选短

背包问题（动态规划） F(i,j)表示i个物品承重量j的背包问题的最大价值
$F(i,j) = \begin{cases} \max\{\underbrace{F(i-1,j)}_{\text{不包括i}}, \underbrace{v_i + F(i-1, j-w_i)}_{\text{包括i}}\}, & j - w_i \ge 0 \\ F(i-1, j) & j - w_i < 0 \end{cases}$ $\Theta(nW)$
$F(0,j) = F(i,0) = 0$ $(i,j \ge 0)$.

找零（dp） F(n)表示需找零为n时的最少硬币数 m种币
$F(n) = \min_{j: n \ge d_j} \{F(n-d_j)\} + 1$ $(n>0)$ $F(0)=0$ $\Theta(nm)$

机器人路径收集（动态规划） F(i,j)表示在(i,j)收集的最大硬币数
$F(i,j) = \max\{F(i-1,j), F(i,j-1)\} + c_{ij}$（当前格） $\Theta(nm)$
$F(i,0) = F(0,j) = 0$ 对于 $i, j \ge 1$

币值最大化（dp） F(n)表示n个硬币可选（不相邻）的最大金额
$F(n) = \max\{\underbrace{F(n-1)}_{\text{不包括n}}, \underbrace{F(n-2)+c_n}_{\text{包括n}}\}$ $(n>1)$ $F(0)=0$ $F(1)=c_1$
$\Theta(n)$

约瑟夫斯（减常因子） J(n)表示n人时幸存者序号
$J(2k) = 2J(k) - 1$ $J(2k+1) = 2J(k) + 1$ $J(1) = 1$

生成组合对象（减治法）
生成排列：①自底向上生成（最小变化）②Johnson-Trotter ③字典序
JT算法（移动元素：箭头指向相邻更小元素）初始化1…n，找最大移动元素，互换后，将大于其方向调转
生成子集：①自底向上（挤压序） ②位串法（a_i属于时，$b_i=1$）

分治法之主定理 $T(n) = aT(n/b) + f(n)$ $f(n) \in \Theta(n^d)$
$T(n) \in \begin{cases} \Theta(n^d) & a < b^d \\ \Theta(n^d \log n) & a = b^d \\ \Theta(n^{\log_b a}) & a > b^d \end{cases}$
大整数乘法 $\Theta(n^{\log_2 3})$
Strassen矩阵 $\Theta(n^{\log_2 7})$

霍纳（变治） $p(x) = 2x^4 - x^3 - 3x^2 + x - 5$

系数	2	-1	-1	1	-5
x=3	2	3×2-1	3×5+3	3×18+1	3×55-5

变治 $\begin{cases} \text{实例化简（预排，AVL，高斯）} \\ \text{改变表现（堆排）} \\ \text{问题化简} \end{cases}$

分治法之最近对：中间一分为二，考虑中线附近 $\Theta(n\log n)$
快包：$\overrightarrow{P_1P_n}$（最左右两端连线）找$\angle P_{max}P_1P_n$最大点、
继续分 $C_{worst} = \Theta(n^2)$ $C_{avg} = \Theta(n)$