渐进符号与数学基础

$O.\ \Omega.\ \Theta$

O：如果$\exists c, n_0 > 0$，对$\forall n > n_0$，有$0 \le f(n) \le c g(n)$，则$f(n) = O(g(n))$

$\lim_{n\to\infty} f(n)/g(n) \le c$

Ω：如果$\exists c, n_0 > 0$，对$\forall n > n_0$，有$0 \le c g(n) \le f(n)$，则$f(n) = \Omega(g(n))$

$\lim_{n\to\infty} f(n)/g(n) \ge c$

Θ：如果$\exists c_1, c_2, n_0 > 0$，对$\forall n > n_0$，有$0 \le c_1 g(n) \le f(n) \le c_2 g(n)$，则$f(n) = \Theta(g(n))$

$\lim_{n\to\infty} f(n)/g(n) \le c$ 且 $\lim_{n\to\infty} g(n)/f(n) \le c$

积分近似求和：

$$\int_{m-1}^{n} f(x)\,dx \le \sum_{k=m}^{n} f(k) \le \int_{m}^{n+1} f(x)\,dx$$

$$n! \sim \sqrt{2\pi n}\left(\frac{n}{e}\right)^n \qquad \sum_{i=1}^{n} \frac{1}{i} \sim \log n.$$

多重对数：$\log^{(i)} n \qquad \log^{*} n = \min\{i \ge 1, \log^{(i)} n \le 1\}$

递归与分治策略

求递归式
- 代换法　猜解 $\to$ 数归证明　有时需要增强假设．适当时进行变量替换．
- 迭代法　递归式 $\to$ 求和，找求和式界
  $\downarrow$
- 递归树法

主方法 $T(n)=aT(\frac{n}{b})+f(n)$

分治：
- 最大子数组 left | right | crossing.
- 大整数乘法 $ad+bc=(a-b)(d-c)+ac+bd$
- Strassen算法

概率分析与随机算法

指示器变量，随机排列算法等等.

排序.

比较排序算法

↓

时间下界：$\Omega(n\log n)$

用二元决策树可证明

- 堆排序：max-heapify → $\underline{\text{build-maxheap}}^{O(n)}$
  heap-sort 从小到大排序是建最大堆.
  不稳定.
- 快速排序：选择pivot的策略
  不稳定.

非比较排序算法：
- 计数排序 稳定 $O(n+k)$
  ↓ 定义域大小（k）
- ↓
- 基数排序

直接插

桶排序　平均: O(n)　最差: O(n

表插

中位数与顺序统计量　最大值.最小值　下界 $\lceil 3n/2 \rceil - 2$

第k大数
- 期望线性时间.
- 最坏线性时间
  5个1组，找中位数的中位数x. 使用x作partition

摊还分析
- 聚合分析
- 核算分析　actual + credit.
- 势能分析

二项堆　势能为树的棵数.

斐波那契堆　势能为 T+

动态规划算法

算法示例
- 多段图规划
- 矩阵链乘
- 最长公共子序列
- 0-1背包问题

适用于动态规划的优化问题中的两要素
- 最优子结构，
- 重叠子问题

动态规划的运行时间依赖于
- 子问题个数
- 每个子问题有多少种选择

设计技巧
- 阶段的划分
- 状态的表示
- 储存表的设计

图算法

图的表示方法

- 邻接表
- 邻接矩阵

图的遍历方法

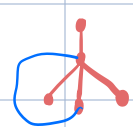
- BFS
  - 记录：前驱 π，深度 d.
  - 计算无权图单源最短路
- DFS
  - 记录：
    - 访问：白→灰→黑
    - $v.d$：发现时间
    - $v.f$：完成时间
    - $v.\pi$
  - 性质：括号化定理

    ↓

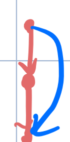
  - 应用：拓扑排序 按 $v.f$ 由后往前排
    - 分解强连通分量（证明用收缩图）
      对 G^T 按 $v.f$ 由后往前深搜

eg：边的分类

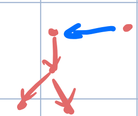
- 树边：访问时白色.
- 后向边：灰色



- 前向边：黑色



- 横向边：黑色



最小生成树：

- 核心思想：找轻量边
- 具体算法
  - Prim：每次找离点集V最近的点加入
    - 二叉堆实现：$O(V\log V + E\log V)$
    - 斐波那契堆：$O(V\log V + E)$
  - Kruskal：选择两个端点不在同一连通分量的边加入
    - 并查集 $O(E\log E)$ or $O(E\log V)$

单源最短路径

- 基本操作：relax (u, v, w)
- 算法
  - 非负权 → Dijkstra：
    首先将源点d值设为0，然后不断extractmin，再从当前点出发relax
    - 时间复杂度分析：
      - 二叉堆 $O(E\log V)$
      - 斐堆 $O(V\log V + E)$
  - 负权 → Bellman-Ford
    对所有边进行$|V|-1$轮relax，最后再进行一轮relax，如果还有点的d值减小，则存在负权环。
    - 时间复杂度 $O(VE)$

    ↓ 优化

    对于有向无环负权图

对于有向无环负权图

进行拓扑排序，按照拓扑排序顺序对每个节点的 E_{out} relax

时间复杂度 $O(V+E)$

所有结点对的最短路径

Floyd-Warshall

进行 $|V|$ 轮遍历。在第 i 轮遍历中，遍历每个结点对，考虑是否用 i 作中间结点。

时间复杂度：$O(V^3)$

对于非负权图：

对所有结点运行一次 Dijkstra

时间复杂度：$O(V^2 \log V + VE)$

对于负权无负权环图：Johnson 算法

加入源点 s，运行 Bellman-Ford 算法，对 $w(u,v)$ 重新赋权：$\hat{w}(u,v) = w(u,v) + u.d - v.d$，再对所有结点运行 Dijkstra.

时间复杂度 $O(VE + E + V^2 \lg V + VE) = O(V^2 \lg V + VE)$

图越稀疏优势越大.

P与NP

P：多项式时间可解　　NP：多项式时间可验证

$P \subseteq NP$

NP完全问题　　定义：对于 $\forall X \in NP$，X都可在多项式时间归约到Y，即 $X \le_p Y$，则Y为NP完全问题

实例：SAT（布尔表达式是否可满足）

$\Updownarrow$

3SAT（析取子句的合取，每个子句有3个不同文字）

$\downarrow$ 归约：将不同子句中不矛盾的文字连起来

团问题 $\Leftrightarrow$ 独立集

$\Updownarrow$

G存在大小为k的团

$\Updownarrow$

G'存在大小为n-k的顶点覆盖

顶点覆盖

$\downarrow$

哈密顿回路

$\downarrow$

图G存在哈密顿回路

$\Updownarrow$

图H存在代价为0的TSP回路

旅行商问题

H为完全图 $w_{ij} = \begin{cases} 0 & (i,j) \in G \\ 1 & (i,j) \notin G \end{cases}$

解决方法
- 对于某些特殊情况，寻找多项式时间算法
- 输入规模小，用指数级别算法
- 近似算法

近似算法

性能比 $\rho(n)$: $\max\left\{\frac{C^*}{C}, \frac{C}{C^*}\right\} \le \rho(n)$

相对误差界 $\varepsilon(n)$ $\frac{|C-C^*|}{C^*} \le \varepsilon(n)$

顶点覆盖问题: 任意选边 将顶点加入点集并删除已覆盖的边，直到边集为空.

满足三角不等式的TSP问题: DF遍历最小生成树 遍历序列为近似最优 近似比为2.

3SAT: 随机赋值 近似比期望 $\frac{8}{7}$

TSP无近似算法

证明如果有近似算法 → 哈密顿回路为P

构造 $c(u,v) = \begin{cases} 1 & \text{if } (u,v) \in E \\ \rho|V|+1 & \text{else.} \end{cases}$

∃哈密顿回路 近似算法返回值 $\le \rho|V|$

不存在 近似算法返回值 $\ge \frac{\rho|V|+|V|}{\rho}$