分治算法的一般描述和分析方法

分治算法的一般性描述

分治算法 **Divide-and-Conquer(P)**

1. **if** $|P| \le c$ **then** $S(P)$
2. **divide** P **into** $P_1, P_2, \ldots, P_k$
3. **for** $i \leftarrow 1$ **to** k
4. $\qquad y_i \leftarrow$ **Divide-and-Conquer(P_i)**
5. **Return Merge** $(y_1, y_2, \ldots, y_k)$

划分

求解子问题

综合解

设计要点

- 原问题可以划分或者归约为规模较小的子问题
  - 子问题与原问题具有相同的性质
  - 子问题的求解彼此独立
  - 划分时子问题的规模尽可能均衡
- 子问题规模足够小时可直接求解
- 子问题的解综合得到原问题的解
- 算法实现：递归或迭代

分治算法时间分析

时间复杂度函数的递推方程

$$W(n)=W(|P_1|)+W(|P_2|)+\ldots+W(|P_k|)+f(n)$$
$$W(c)=C$$

- $P_1, P_2, \ldots, P_k$ 为 划分后产生的 子问题
- $f(n)$为划分子问题以及将子问题的解综合得到原问题解的总工作量
- 规模为c的最小子问题的工作量为C

两类常见的递推方程

$$f(n)=\sum_{i=1}^{k}a_i f(n-i)+g(n) \qquad (1)$$

$$f(n)=af(\frac{n}{b})+d(n) \qquad (2)$$

例子：

Hanoi塔， $W(n)=2W(n-1)+1$

二分检索，$W(n)=W(n/2)+1$

归并排序，$W(n)=2W(n/2)+n-1$

递推方程的求解

方程1 $$f(n)=\sum_{i=1}^{k}a_i f(n-i)+g(n)$$

方程2 $$f(n)=af(\frac{n}{b})+d(n)$$

求解方法

方程1：迭代法、递归树

方程2：迭代法、换元法、递归树、主定理

方程2的解

方程 $T(n) = aT(n/b) + d(n)$

d(n)为常数

$$T(n) = \begin{cases} O(n^{\log_b a}) & a \neq 1 \\ O(\log n) & a = 1 \end{cases}$$

d(n) = cn

$$T(n) = \begin{cases} O(n) & a < b \\ O(n\log n) & a = b \\ O(n^{\log_b a}) & a > b \end{cases}$$

小结

- 分治算法的一般描述

  划分或归约为彼此独立的子问题
  分别求解每个子问题
  给出递归或迭代计算的终止条件
  如何由子问题的解得到原问题解

- 分治算法的分析方法

  求解时间复杂度的递推方程
  常用的递推方程的解