HW1

范潇　2254298

2024 年 5 月 2 日

题目 1. (1-1) 证明基于键值对比较的排序算法时间复杂性下界是 $n \log n$.

证明. 不妨设参与比较的元素的关键字互异。

对于参与排列的 n 个元素，其最终可能的排列个数即全排列数 $n!$. 现将键值比较的过程抽象为一棵二叉树，其中的结点有以下性质：

1. 每个结点代表着若干个排列
2. 若为叶子节点，则只代表一个排列
3. 否则，选取两个元素进行比较，该二叉树对应的算法将该结点代表的排列依据比较结果进一步分为两类，分别用左孩子和右孩子代表

不同排序算法所对应的二叉树形状不同，每个结点所代表的元素以及用于比较的元素都可能不同。但对于一个正确的排序算法，应该能在有限步内对任意合法输入给出正确的排序结果，也就是对于一个给定的输入，从根节点开始，根据节点上的比较操作移动至孩子结点，最终能够在有限步内达到某个叶子结点，其对应的排序就是该算法的输出。因此，一个正确的排序算法对应至少应该保证在足够大的深度内，$n!$ 个所有的可能排列都至少出现在叶子节点上一次。由于高度为 h 的二叉树（根节点高度设为 0）至多有 $2^{h+1}-1$ 个结点，所以对于一个正确的排序算法，当输入有 n 个元素时，对应的二叉树有不等式

$$2^{h+1} - 1 \geq n!$$

整颗二叉树的深度便对应着最坏情况下所需要的比较次数，又因为

$$h \geq \lg(n! + 1) - 1 = \Theta(n \lg n)$$

所以基于键值对比较的排序算法时间复杂性下界是 $n \log n$. □

题目 2. (1-2) 证明主定理

设常数 $a \geq 1, b > 1$，$f(n)$ 为一个定义域和值域均为非负实数的函数，并令 $T(n)$ 为一个定义域为正实数的函数，且

$$T(n) = \begin{cases} aT(n/b) + f(n) & n > 1 \\ \Theta(1) & n \leq 1 \end{cases},$$

那么 $T(n)$ 有如下渐进界：

1. 如果 $f(n)=O(n^{\log_b a-\epsilon})$，其中 ϵ 为某正数，则 $T(n)=\Theta(n^{\log_b a})$。
2. 如果存在常数 $k\geq 0$，使得 $f(n)=\Theta(n^{\log_b a}\log_b^k n)$，则 $T(n)=\Theta(n^{\log_b a}\log_b^{k+1} n)$。
3. 如果 $f(n)=\Omega(n^{\log_b a+\epsilon})$，其中 ϵ 为某正数，同时当 n 足够大时，对于某个常数 $c<1$ 有 $af(n/b)\leq cf(n)$，则 $T(n)=\Theta(f(n))$。

证明. 因为 $n=b^{\log_b n}<b^{\lfloor\log_b n\rfloor+1}$，所以 $n/b^{\lfloor\log_b n\rfloor+1}<1$，从而当 n 足够大时有

$$\begin{aligned} T(n)=aT(n/b)+f(n)=\cdots &= a^{\lfloor\log_b n\rfloor+1}T(n/b^{\lfloor\log_b n\rfloor+1})+\textstyle\sum_{i=0}^{\lfloor\log_b n\rfloor}a^i f(n/b^i)\\ &= \Theta(a^{\lfloor\log_b n\rfloor+1})+\textstyle\sum_{i=0}^{\lfloor\log_b n\rfloor}a^i f(n/b^i)\\ &= \Theta(a^{\log_b n})+\textstyle\sum_{i=0}^{\lfloor\log_b n\rfloor}a^i f(n/b^i)\\ &= \Theta(n^{\log_b a})+\textstyle\sum_{i=0}^{\lfloor\log_b n\rfloor}a^i f(n/b^i)\end{aligned}$$

当 $f(n)=O(n^{\log_b a-\epsilon})$，其中 ϵ 为某正数时，为了证明 $T(n)=\Theta(n^{\log_b a})$，由于 $n^{\log_b a}=\Theta(n^{\log_b a})$，我们只需证明

$$\sum_{i=0}^{\lfloor\log_b n\rfloor}a^i f(n/b^i)=O(n^{\log_b a}).$$

而

$$\sum_{i=0}^{\lfloor\log_b n\rfloor}a^i f(n/b^i)=\sum_{i=0}^{\lfloor\log_b n\rfloor}a^i O((n/b^i)^{\log_b a-\epsilon})=\sum_{i=0}^{\lfloor\log_b n\rfloor}O(a^i n^{\log_b a-\epsilon}(b^\epsilon)^i/a^i)=n^{\log_b a-\epsilon}\sum_{i=0}^{\lfloor\log_b n\rfloor}O((b^\epsilon)^i)$$

从而

$$\sum_{i=0}^{\lfloor\log_b n\rfloor}a^i f(n/b^i)=O(n^{\log_b a})\Theta(1)=O(n^{\log_b a}).$$

当存在常数 $k\geq 0$，使得 $f(n)=\Theta(n^{\log_b a}\log_b^k n)$ 时，为了证明 $T(n)=\Theta(n^{\log_b a}\log_b^{k+1} n)$，由于 $n^{\log_b a}=o(n^{\log_b a}\log_b^{k+1} n)$，我们只需证明

$$\sum_{i=0}^{\lfloor\log_b n\rfloor}a^i f(n/b^i)=\Theta(n^{\log_b a}\log_b^{k+1} n).$$

而

$$\sum_{i=0}^{\lfloor\log_b n\rfloor}a^i f(n/b^i)=\sum_{i=0}^{\lfloor\log_b n\rfloor}a^i\Theta((n/b^i)^{\log_b a}\log_b^k(n/b^i))=n^{\log_b a}\Theta(\sum_{i=0}^{\lfloor\log_b n\rfloor}\log_b^k(n/b^i)).$$

又因为

$$\sum_{i=0}^{\lfloor\log_b n\rfloor}\log_b^k(n/b^i)=\sum_{i=0}^{\lfloor\log_b n\rfloor}(\log_b n-i)^k\leq\sum_{i=0}^{\lfloor\log_b n\rfloor}(\log_b n)^k\leq(\log_b n)^k(\log_b n+1)\leq 2(\log_b n)^{k+1}$$

$$(\log_b n/2)^{k+1}\leq\sum_{i=0}^{\lfloor\log_b n/2\rfloor}(\log_b n-i)^k\leq\sum_{i=0}^{\lfloor\log_b n\rfloor}\log_b^k(n/b^i)=\sum_{i=0}^{\lfloor\log_b n\rfloor}(\log_b n-i)^k$$

从而

$$\sum_{i=0}^{\lfloor\log_b n\rfloor}a^i f(n/b^i)=n^{\log_b a}\Theta(\sum_{i=0}^{\lfloor\log_b n\rfloor}\log_b^k(n/b^i))=n^{\log_b a}\Theta(\log_b^{k+1} n)=\Theta(n^{\log_b a}\log_b^{k+1} n).$$

当 $f(n) = \Omega(n^{\log_b a+\epsilon})$, 其中 ϵ 为某正数, 同时当 n 足够大时, 对于某个常数 $c < 1$ 有 $af(n/b) \leq cf(n)$。为了证明 $T(n) = \Theta(f(n))$，由于 $\sum_{i=0}^{\lfloor \log_b n \rfloor} a^i f(n/b^i) = \Omega(f(n))$ ，我们只需证明

$$\sum_{i=0}^{\lfloor \log_b n \rfloor} a^i f(n/b^i) = O(f(n))$$

而当 n 足够大时,

$$\sum_{i=0}^{\lfloor \log_b n \rfloor} a^i f(n/b^i) \leq \Theta(1) + \Theta(1)\sum_{i=0}^{\infty} c^i f(n) = \Theta(1) + O(f(n)) = O(f(n))$$

其中 $\Theta(1)$ 为无法使用 $af(n/b) \leq cf(n)$ 时产生的项。

从而有

$$\sum_{i=0}^{\lfloor \log_b n \rfloor} a^i f(n/b^i) = O(f(n))$$

□

题目 3. (1-3) 求解递归方程（用渐进符号表示），c 是常数

1. $T(n) = T(n-1) + cn$
2. $T(n) = 2T(n/2) + cn$
3. $T(n) = T(n/3) + T(2n/3) + cn$
4. $T(n) = 7T(n/2) + cn$
5. $T(n) = 9T(n/3) + n\log n$

解答.

1. 直接展开可得 $T(n) = c(n + \cdots + 2) + T(1) = \Theta(n^2) + \Theta(1) = \Theta(n^2)$
2. 符合主定理中的第二种情况，解得 $T(n) = \Theta(n \lg n)$
3. 下面用归纳法证明存在常数 $d > 0$ 使得，$T(n) \leq dn \lg n$。因为由归纳假设可得

$$\begin{aligned} T(n) = T(n/3) + T(2n/3) + cn &\leq \tfrac{1}{3}dn(\lg n - \lg 3) + \tfrac{2}{3}dn(\lg n - \lg \tfrac{3}{2}) + cn \\ &= dn\lg n - dn(\tfrac{1}{3}\lg 3 + \tfrac{2}{3}\lg \tfrac{3}{2}) + cn \end{aligned}$$

所以只需要取足够大的 d 便可以使得 $T(n) \leq n \lg n$ 在 n 足够大时成立。从而有 $T(n) = O(n \lg n)$

类似地，下面用归纳法证明存在常数 $d > 0$ 使得，$T(n) \geq dn \lg n$。因为由归纳假设可得

$$\begin{aligned} T(n) = T(n/3) + T(2n/3) + cn &\geq \tfrac{1}{3}dn(\lg n - \lg 3) + \tfrac{2}{3}dn(\lg n - \lg \tfrac{3}{2}) + cn \\ &= dn\lg n + cn - dn(\tfrac{1}{3}\lg 3 + \tfrac{2}{3}\lg \tfrac{3}{2}) \end{aligned}$$

所以只需要取足够大的 便可以使得 $T(n) \geq dn \lg n$ 在 n 足够大时成立。从而有 $T(n) = \Omega(n \lg n)$

综上，$T(n) = \Theta(n \lg n)$

4. 符合主定理的第一种情况，解得 $T(n) = \Theta(n^{\lg 7})$
5. 符合主定理的第一种情况，解得 $T(n) = \Theta(n^2)$