算法设计与分析

CS2008 班 U202015533 徐瑞达

2022.03.26

1 2-4: 逆序对

a. 列出 <2,3,8,6,1> 的 5 个逆序对

$(1, 5), (2, 5), (3, 4), (3, 5), (4, 5)$

b. 由 1-n 中的数字构成的什么数组拥有最多的逆序对

数组 $< n, n-1, \ldots, 1 >$ 拥有最多的逆序对，共有 $(n-1) + (n-2) + \ldots + 1 = n(n-1)/2$ 个逆序对。

c. 插入排序的运行时间与输入数组中逆序对的数量之间有什么关系，阐述并证明之

插入排序的运行时间与逆序对数量之间为常数级关系。

令 $N(i)$ 表示在某个 i 下的逆序对个数，则 $\sum_{i=1}^{n} N(i)$ 表示数组的逆序对个数。

在插入排序算法中的 $while$ 循环中，对于每个下标小于 j 但是值大于 $A[j]$ 的元素，该循环都会执行一次，因此，该循环会执行 $N(j)$ 次。而对于在 for 循环中的每次迭代，我们都会进入一次 $while$ 循环，所以插入排序的常量运行次数为 $\sum_{i=1}^{n} N(i)$，也就是 A 的逆序对个数。

d. 在最坏时间复杂度为 $\Theta(n \lg n)$ 的前提下，给出一个确定逆序对数量的算法 (修改归并排序)

Algorithm 1: INVERSIONS(A, p, r)

```
if p<r then
    q = ⌊(p + r)/2⌋
    left=INVERSIONS(A, p, q)
    right=INVERSIONS(A, q + 1, r)
    count=AUXILIARY-FUNCTION(A, p, q, r)+left+right
    return count
```

Algorithm 2: AUXILIARY-FUNCTION(A, p, q, r)

```
n_1 = q - p + 1
n_2 = r - q
let L[1..n_1 + 1] and R[1..n_2 + 1] be new arrays
for i=1 to n_1 do
    L[i] = A[p + i - 1]
for j=1 to n_2 do
    R[j] = A[q + j]
L[n_1 + 1] = ∞
R[n_2 + 1] = ∞
i = 1
j = 1
for k = p  to  r do
    if L[i] ≤ R[j] then
        A[k] = L[i]
        i = i + 1
    else
        count = count + n_1 - i + 1
        A[k] = R[j]
        j = j + 1
return count
```

2 4.1-5: 对算法的理解

由算法得知: 已知 $A[1,...,j]$ 的最大子数组时，$A[1,...,j+1]$ 的最大子数组要么与其相同，要么为 $A[i,...,j+1], 1 \geq i \geq j+1$。在循环中，$j$ 由 1 遍历到 n，每次循环内，更新最大下标为 j，若当前和大于 0，则加上 $A[j]$，否则更新最小下标并更新和，将和与最大和比较并更新最大和及上下标，避免了递归过程，并且仅使用了线性时间完成算法。

3 4.1-2: 证明递归式 $T(n) = T(\lceil n/2 \rceil) + 1$ 的解为 $O(\lg n)$

假设 $T(n) \leq c\lg(n-a)$，

$$\begin{aligned}
T(n) &\leq c\lg(\lceil n/2 \rceil - a) + 1 \\
&\leq c\lg((n+1)/2 - a) + 1 \\
&= c\lg((n+1-2a)/2) + 1 \\
&= c\lg(n+1-2a) - c\lg 2 + 1 \quad (c \geq 1) \\
&\leq c\lg(n+1-2a) \quad (a \geq 1) \\
&\leq c\lg(n-a)
\end{aligned}$$

因此得到递归式 $T(n) = T(\lceil n/2 \rceil) + 1$ **的解为** $O(\lg n)$

4 4.3-9: 求解递归式 $T(n) = 3T(\sqrt{n}) + \log n$

首先

$$\begin{aligned}T(n) &= 3T(\sqrt{n}) + \log n \quad \text{令 } m = \lg n \\ T(2^m) &= 3T(2^{m/2}) + m \\ S(m) &= 3S(m/2) + m.\end{aligned}$$

假设 $S(m) \le cm^{\lg 3} + dm$，**则**

$$\begin{aligned}S(m) &\le 3(c(m/2)^{\lg 3} + d(m/2)) + m \\ &\le cm^{\lg 3} + (\tfrac{3}{2}d + 1)m \quad (d \le -2) \\ &\le cm^{\lg 3} + dm\end{aligned}$$

假设 $S(m) \ge cm^{\lg 3} + dm$，**则**

$$\begin{aligned}S(m) &\ge 3(c(m/2)^{\lg 3} + d(m/2)) + m \\ &\ge cm^{\lg 3} + (\tfrac{3}{2}d + 1)m \quad (d \ge -2) \\ &\ge cm^{\lg 3} + dm\end{aligned}$$

5 4.4-6: 对递归式 $T(n) = T(n/3) + T(2n/3) + cn$ 利用递归树证明其解是 $\Omega(n \log n)$，其中 c 是一个常数。

根据每个结点的最左孩子可以得出从根到叶子结点的最短简单路径，因此可得:

$$cn(\log_3 n + 1) \ge cn \log_3 n = \tfrac{c}{\log_3} n \log n = \Omega(n \log n).)$$

6 4.5-1: 用主方法给出以下递归式的紧确渐近界:

(b)$T(n) = 2T(n/4) + n^{1/2}$

$\Theta(n^{\log_4 2} \lg n) = \Theta(\sqrt{n} \lg n)$

(d)$T(n) = 2T(n/4) + n^2$

$\Theta(n^2)$

7 4.5-4: 主方法能否应用于递归式 $T(n) = 4T(n/2) + n^2 \log n$? 为什么? 给出其渐近上界。

当 $a = 4, b = 2$ **时，有** $f(n) = n^2 \lg n \ne O(n^{2-\epsilon}) \ne \Omega(n^{2+\epsilon})$，**因此不能使用主方法。**

假设 $T(n) \le cn^2(\lg n)^2$，将 n 替换为 $n/2$ 得:

$T(n) = 4T(n/2) + n^2 \lg n$

$\le 4c(n/2)^2(\lg(n/2))^2 + n^2 \lg n$

$= cn^2 \lg(n/2) \lg n - cn^2 \lg(n/2) \lg 2 + n^2 \lg n$

$= cn^2(\lg n)^2 - cn^2 \lg n \lg 2 - cn^2 \lg(n/2) \lg 2 + n^2 lgn$

$= cn^2(\lg n)^2 + (1 - c\lg 2)n^2 \lg n - cn^2 \lg(n/2) \lg 2 \quad (c \ge 1/\lg 2)$

$\le cn^2(\lg n)^2 - cn^2 \lg(n/2) \lg 2$

$\le cn^2(\lg n)^2$

也即渐近上界为 $cn^2(\lg n)^2$.