HW2

Page22: 2.3-1，2.3-5

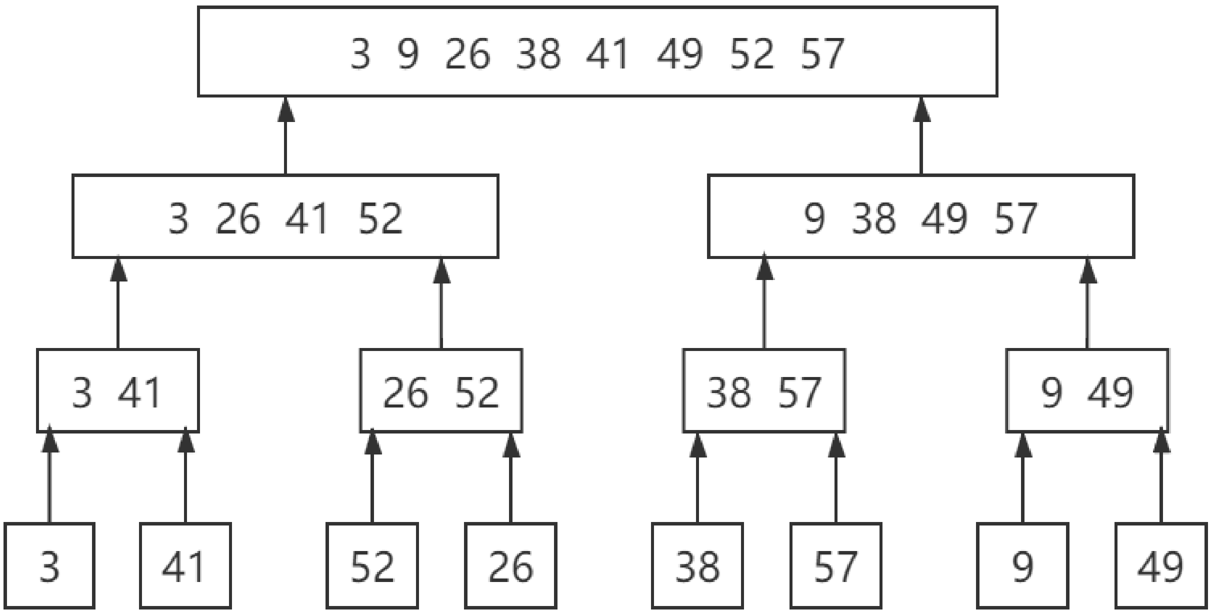
Page30: 3.1-2，3.1-3，3.1-6

Page34: 3.2-3，3.2-5

2.3

Page22：2.3-1，2.3-5

2.3-1 使用图 2-4 作为模型，说明归并排序在数组 $A=\langle 3, 41, 52, 26, 38, 57, 9, 49\rangle$ 上的操作。



2.3-5 回顾查找问题(参见练习 2.1-3)，注意到，如果序列 A 已排好序，就可以将该序列的中点与 v 进行比较。根据比较的结果，原序列中有一半就可以不用再做进一步的考虑了。**二分查找**算法重复这个过程，每次都将序列剩余部分的规模减半。为二分查找写出迭代或递归的伪代码。证明：二分查找的最坏情况运行时间为 $\Theta(\lg n)$。

2.1-3 考虑以下查找问题：

输入：n 个数的一个序列 $A=\langle a_1, a_2, \cdots, a_n\rangle$ 和一个值 v。

输出：下标 i 使得 $v=A[i]$ 或者当 v 不在 A 中出现时，v 为特殊值 NIL。

写出**线性查找**的伪代码，它扫描整个序列来查找 v。使用一个循环不变式来证明你的算法是正确的。确保你的循环不变式满足三条必要的性质。

二分查找递归伪码：

```
BinarySearch(A, v, low, high):
    # 二分查找递归算法
    # 输入：数组A[1..n]，待查找的值v，查找区间[low,high]
    # 输出：若找到，则返回一个匹配元素的下标i，若v不在A中，则返回NIL

    mid = floor((low+high)/2)
    # 注意是low<=high，v=A[high]时，low==high
    if low <= high:
        if v == A[mid]:
            return mid
        if v > A[mid]:
            BinarySearch(A, v, mid+1, high)
        else:
            BinarySearch(A, v, low, mid-1)
    return NIL
```

题目要求证明，至少要写出递归式。

最坏情况：v不在A中，或最后一次比较才命中

时间$T(n)=T(n/2)+\theta(1)$，解为$T(n)=\theta(\lg n)$，得证。

3.1

Page31：3.1-2，3.1-3，3.1-6

3.1-2 证明：对任意实常量 a 和 b，其中 $b>0$，有

$$(n+a)^b=\Theta(n^b) \tag{3.2}$$

法**1.**

记$f(n)=(n+a)^b$，$g(n)=n^b$，即证 $n\geq n_0$时，$0\leq c_1g\leq f\leq c_2g$。

> b>0，保证x^b单增。注意a为负的情况。

Note that

$$\begin{aligned} n+a &\leq n+|a| \\ &\leq 2n \qquad \text{when } |a| \leq n \, , \end{aligned}$$

and

$$\begin{aligned} n+a &\geq n-|a| \\ &\geq \frac{1}{2}n \qquad \text{when } |a| \leq \tfrac{1}{2}n \, . \end{aligned}$$

Thus, when $n \geq 2\,|a|$,

$$0 \leq \frac{1}{2}n \leq n+a \leq 2n \, .$$

Since $b > 0$, the inequality still holds when all parts are raised to the power b:

$$0 \leq \left(\frac{1}{2}n\right)^b \leq (n+a)^b \leq (2n)^b \, ,$$

$$0 \leq \left(\frac{1}{2}\right)^b n^b \leq (n+a)^b \leq 2^b n^b \, .$$

Thus, $c_1 = (1/2)^b$, $c_2 = 2^b$, and $n_0 = 2\,|a|$ satisfy the definition.

法**2.** 一个具有启发意义的做法，by No.37, Year 2020

极限方法

由于

$$\lim_{n\to+\infty}\frac{(n+a)^b}{n^b}=\lim_{n\to\infty}\left(1+\frac{a}{n}\right)^b=1$$

即

$$\forall\varepsilon,\exists N,\forall n>N,\ 1-\varepsilon\leqslant\frac{(n+a)^b}{n^b}\leqslant 1+\varepsilon$$

取 $\varepsilon=0.05$，故取 $c_1=0.95,c_2=1.05$ 时，有

$$\exists N,\forall n>N,\ 0.95\leqslant\frac{(n+a)^b}{n^b}\leqslant 1.05$$

即为

$$\exists n_0,\forall n>n_0,\ 0.95\cdot n^b\leqslant(n+a)^b\leqslant 1.05\cdot n^b$$

故 $(n+a)^b=\Theta\left(n^b\right)$

3.1-3 解释为什么“算法 A 的运行时间至少是 $O(n^2)$”这一表述是无意义的。

设算法运行时间为$T(n)$，$T(n)\geq O(n^2)$意味着对于函数集合$O(n^2)$中的某些函数$f(n)$，总是满足$T(n)\geq f(n)$。首先，使用函数集合来描述$T(n)$的下界本身语义模糊。进一步地，由于$g(n)=0$这一函数也在函数集合$O(n^2)$中，而$T(n)\geq 0$总是成立的，所以这一表述实际上没有提供任何有效信息。

3.1-6 证明：一个算法的运行时间为 $\Theta(g(n))$ 当且仅当其最坏情况运行时间为 $O(g(n))$，且其最好情况运行时间为 $\Omega(g(n))$。

$T=\Theta(g)\Leftrightarrow\exists n_0,c_1,c_2:n\geq n_0$时，$c_2g\leq T(n)\leq c_1g.$ (命题1)

最坏时间$O(g)\Leftrightarrow\exists n_0,c_1':n\geq n_1$时，$T_{worst}(n)\leq c_1'g$ (命题2.1)

最好时间$\Omega(g)\Leftrightarrow\exists n_0,c_2':n\geq n_2$时，$T_{best}(n)\geq c_2'g$ (命题2.2)

充分性证明：考虑命题1⇒命题2.1 && 命题2.2.

由于$T_{worst}(n), T_{best}(n) \in T(n)$，故有$T_{worst}(n) \le c_1 g$，$T_{best}(n) \ge c_2 g$成立。

则取$n_1 = n_2 = n_0$，$c_1' = c_1$，$c_2' = c_2$，即可证明命题2.1 && 命题2.2正确。

必要性证明：考虑命题1⇐命题2.1 && 命题2.2.

由于$T_{worst}(n) \ge T(n) \ge T_{best}(n)$，故有$c_2' g \le T(n) \le c_1' g$成立。

则取$n_0 = max\{n_1, n_2\}$，$c_1 = c_1'$，$c_2 = c_2'$，即可证明命题1正确。

3.2

Page34：3.2-3，3.2-5

3.2-3　证明等式(3.19)。并证明 $n! = \omega(2^n)$且 $n! = o(n^n)$。

$$\lg(n!) = \Theta(n \lg n) \tag{3.19}$$

证明：(1)取$c_1 = \frac{1}{8}, c_2 = 1, n_0 = 12$。（注：常数的取法并不唯一）

注意到$\lg(n!) = \lg 1 + \lg 2 + \ldots + \lg n < n \lg n$成立。

将$\lg 1 + \lg 2 + \ldots + \lg n$中大于等于$\lg \lceil \frac{n}{2} \rceil$的项取出(共有$\lfloor \frac{n}{2} \rfloor$项)，并把每一项缩小到$\lg \lceil \frac{n}{2} \rceil$。得到：

$$\begin{aligned}
&\lg 1 + \lg 2 + \ldots + \lg n \\
&> \lfloor \frac{n}{2} \rfloor \lg \lceil \frac{n}{2} \rceil \\
&> \lfloor \frac{n}{2} \rfloor \lg \frac{n}{2} \\
&= \lfloor \frac{n}{2} \rfloor \lg n - \lfloor \frac{n}{2} \rfloor \\
&> \frac{n-1}{2} \lg n - \frac{n}{2} \\
&= \frac{n-2}{4} \lg n + \frac{n}{4} \lg n - \frac{n}{2} \\
&> \frac{n-2}{4} \lg n, \qquad\qquad n > 4, \lg n > 2 \\
&= \frac{n}{8} \lg n + \frac{n-4}{8} \lg n, \qquad n > 12 \\
&> \frac{1}{8} n \lg n
\end{aligned}$$

综上，当$n>12$时，有

$\frac{1}{8}n\lg n<\lg(n!)<n\lg n$

注：此类题目因为可以令n大于一个任意大的常数，所以上述放缩方法仅供参考。即使放缩不紧，只要证出来就可以。

(2)$n!=1*2*3*4*5*...*(n-1)*n$(n足够大时)

对于任意正整数c_0，取$n_0=max\{4,2c_0\}$，则$n>n_0$时，有：

$n!=1*2*3*4*5*...*(n-1)*n>1*2*2*4*2*...*2*2c_0=c_0 2^n$，得证。

(3)对于任意正整数c_0，取$n_0=c_0$，则$n>n_0$时，有：

$c_0 n!=c_0*2*3...*n<n*n*n*...*n=n^n$，得证。

***3.2-5　如下两个函数中，哪一个渐近更大些：$\lg(\lg^* n)$还是 $\lg^*(\lg n)$？**

证明：设$2^{2^{\cdot^{\cdot^{2}}}}$共$k$个2，记作$\overset{k}{2}$。记$r=\lg(\lg^* n)$，$s=\lg^*(\lg n)$。

因为$\overset{k-1}{2}<n\le\overset{k}{2}$时，$\lg^* n=k$，故只要判断$n=\overset{k}{2}$，$r,s$哪个渐进更大即可。

因为$r(n)=\lg k$，$s(n)=\lg^*\overset{k-1}{2}=k-1$，明显$s$渐进更大。严格证明：

$\lim\limits_{k\to\infty}\frac{r(n)}{s(n)}=\lim\limits_{k\to\infty}\frac{\lg k}{k-1}=0$，故$s$渐进更大。