第2次作业

4.1-5 使用如下思想为最大子数组问题设计一个非递归的、线性时间的算法。从数组的左边界开始，由左至右处理，记录到目前为止已经处理过的最大子数组。若已知 $A[1..j]$的最大子数组，基于如下性质将解扩展为 $A[1..j+1]$的最大子数组：$A[1..j+1]$的最大子数组要么是 $A[1..j]$的最大子数组，要么是某个子数组 $A[i..j+1]$($1\leqslant i\leqslant j+1$)。在已知 $A[1..j]$的最大子数组的情况下，可以在线性时间内找出形如 $A[i..j+1]$的最大子数组。

阅读4.1-5题面及以下程序，然后写出你对这个算法的理解。

```
MAX-SUBARRAY-LINEAR(A)
  n = A.length
  max-sum = -∞
  ending-here-sum = -∞
  for j = 1 to n
      ending-here-high = j
      if ending-here-sum > 0
          ending-here-sum = ending-here-sum + A[j]
      else ending-here-low = j
          ending-here-sum = A[j]
      if ending-here-sum > max-sum
          max-sum = ending-here-sum
          low = ending-here-low
          high = ending-here-high
  return (low, high, max-sum)
```

ending-here-sum>0，继续向后扩展到j、计算到j位置的子序列和。不用担心A[j]<0的情况发生，因为即使A[j]<0导致ending-here-sum变小，其前已经找到的更大的子序列和也已经在全局量max-sum中有了记载，这是一个递推的过程，max-sum不受A[j]<0的影响，而一旦找到更大的ending-here-sum，则会在后面的if语句里修正max-sum，从而可以找到和更大的连续子序列。

如果ending-here-sum<=0，则ending-here-sum +A[j]只会不比A[j]更大，甚至还不如A[j]本身大，所以直接调整为“ending-here-low = j； ending-here-sum=A[j]”、ending-here-low==ending-here-high

这一步比较显然，当找到更大的连续子序列和，修正全局的max-sum和下标low、high，以记录当前以求出的最大连续子序列和和下标区间。

- 4.2-1

  P=48

  Q=72

  R=6

  S=-10

  T=8

  U=-84

  V=-12

  C11=18

  C12=4

  C21=62

  C22=66

4.3-2 证明 $T(n) = T(\lceil n/2 \rceil) + 1$ 的解是lgn

因为包含上取整函数，又要找一个上界，所以我们要用一些技巧：

假设：$T(n) \le c \lg(n - b)$,对$\lceil n/2 \rceil$成立，进而，我们有：

$$
\begin{aligned}
T(n) &\le c \lg(\lceil n/2 \rceil - b) + 1 \\
&\le c \lg(n/2 - b + 1) + 1 \\
&= c \lg\left(\frac{n - 2b + 2}{2}\right) + 1 \\
&= c \lg(n - 2b + 2) - c \lg 2 + 1 \\
&\le c \lg(n - b)
\end{aligned}
$$

最后一个不等号成立需要，$b \ge 2, c \ge 1$

所以，得证！

- 4.3-6

  $T(n)=O(n\lg n)$,

  证明$T(n)\le c(n-a)\lg(n-a)$

  $$
  \begin{aligned}
  T(n) &\le 2c(\lfloor n/2 \rfloor+17-a)\lg(\lfloor n/2 \rfloor+17-a)+n \\
  &\le 2c(n/2+17-a)\lg(n/2+17-a)+n \\
  &= c(n+36-2a)\lg((n+36-2a)/2)+n \\
  &\le c(n+36-2a)\lg(n+36-2a) \qquad c>1, a\ge 36 \\
  &\le c(n-a)\lg(n-a) \\
  &\le cn\lg n
  \end{aligned}
  $$

(4.3-9) 利用改变变量的方法求解递归式 $T(n) = 3T(\sqrt{n}) + \log n$ 。得到的解应是紧确的。

令$n=2^m$，$m=\log n$

$T(n)=3T(2^{m/2}) + m$

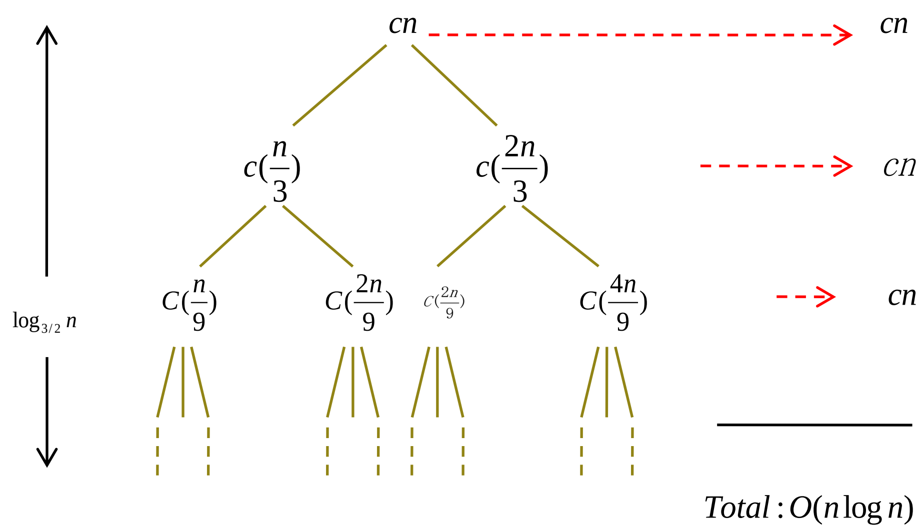
令$S(m) = T(n)$

$S(m)=3S(m/2)+m$

则 $S(m)=O(m^{\log_2 3})$

$=O(\log n^{\log_2 3})$ (或$3^{\log_2 \log n}$)

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



讨论其左分支

左分支深度：$\log_3 n$

0~$\log_3 n$每层的代价是cn

故，$T(n)\geq(\log_3 n+1)cn=\Omega(n\log n)$

注：$\log_3 n=\log n/\log 3\approx\log n$

进行一定的证明

(4.5-1) 用主方法来给出下列递归式的紧确渐近界:

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

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

根据主方法分情况讨论

b) $T(n) = O(n^{1/2}\log n)$

d) $T(n)=O(n^2)$

(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？给出此递归式的渐近上界。

这里a=4,b=2，$n^{\log_b a} = n^2$

而 $f(n)/ n^{\log_b a}=\log n<n^{\varepsilon}$

不能用主方法

$$
\begin{aligned}
T(n) &= 4T(n/2) + n^2 \log n \\
&= 4(4T(n/4) + (n/2)^2 \log(n/2)) + n^2 \log n \\
&= 4^2 T(n/2^2) + n^2 \log n - n^2 + n^2 \log n \\
&= 4^2 T(n/2^2) + 2n^2 \log n - n^2 \\
&= 4^2 (4T(n/2^3) + (n/2^2)^2 \log(n/4)) + 2n^2 \log n - n^2 \\
&= 4^3 T(n/2^3) + n^2 \log n - 2n^2 + 2n^2 \log n - n^2 \\
&= 4^3 T(n/2^3) + 3n^2 \log n - 2n^2 - n^2 \\
&= \ldots \\
&= 4^k T(n/2^k) + kn^2 \log n - n^2 \sum_{i=1}^{k-1} i \\
&= n^2 + kn^2 \log n - n^2 (k-1)k/2 \\
&= n^2 + n^2 \log^2 n - (n^2/2) \log^2 n + n^2 \log n / 2 \\
&= O(n^2 \log^2 n)
\end{aligned}
$$

4.3

a.

a=4,b=3

$\log_b a = \log_3 4$

n^($\log_3$4)=nn^$\log_3$(4/3)

lgn=O(n^Ɛ)

T(n)=Θ(n^($\log_3$4))

c. $T(n) = 4T(n/2) + n^2\sqrt{n}$

We have $f(n) = n^2\sqrt{n} = n^{5/2}$ and $n^{\log_b a} = n^{\log_2 4} = n^2$. Since $n^{5/2} = \Omega(n^{2+\epsilon})$ for $\epsilon = 1/2$, we look at the regularity condition in case 3 of the master theorem. We have $af(n/b) = 4(n/2)^2\sqrt{n/2} = n^{5/2}/\sqrt{2} \le cn^{5/2}$ for $1/\sqrt{2} \le c < 1$. Case 3 applies, and we have $T(n) = \Theta(n^2\sqrt{n})$.

i. $$T(n) = \sum_{i=0}^{\frac{n}{2}-1} \frac{1}{\lg(n-2i)} + \Theta(1)$$

$$\le \sum_{i=2}^{\infty} \frac{1}{lgi} = \Theta(lglgn)$$