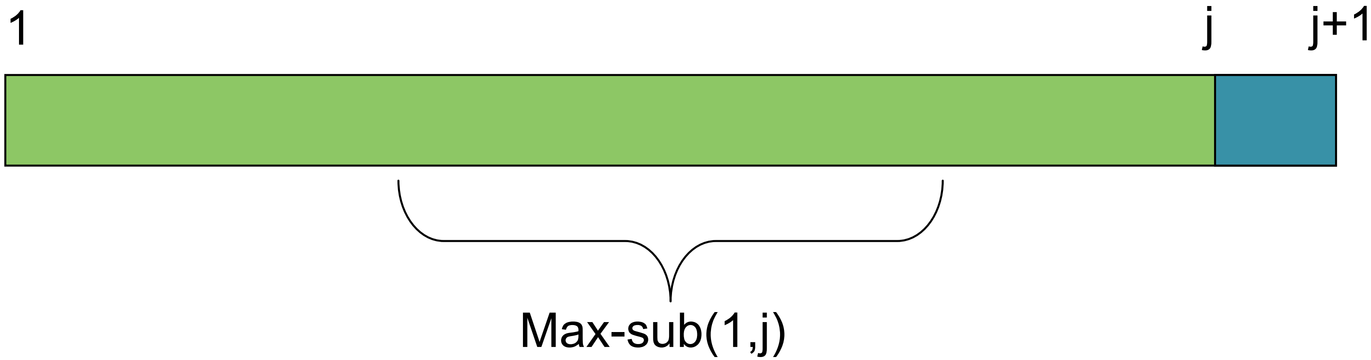
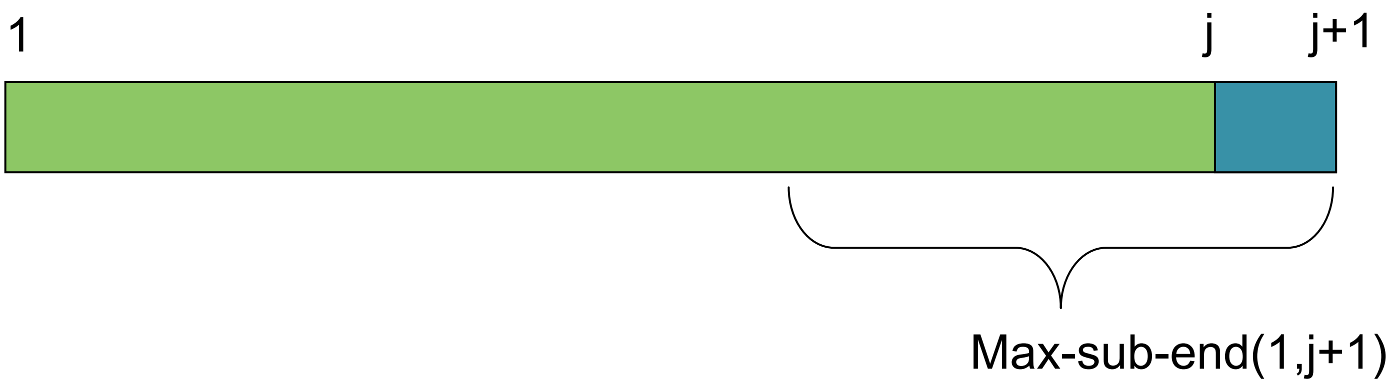
反馈与讨论

2014/4/10

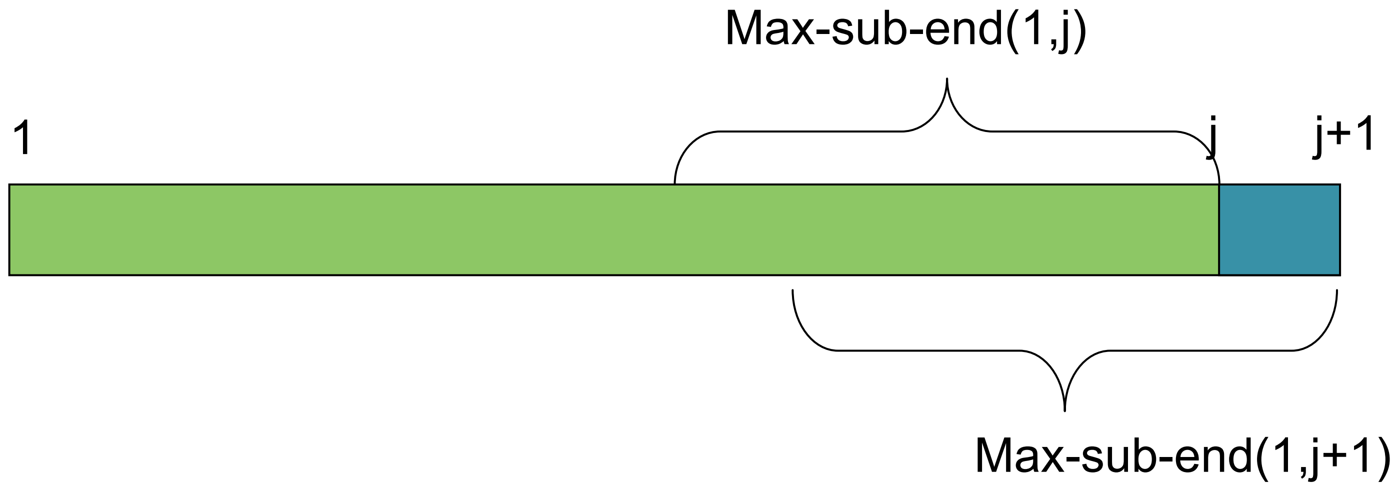
4.1-5

Use the following ideas to develop a nonrecursive, linear-time algorithm for the maximum-subarray problem. Start at the left end of the array, and progress toward the right, keeping track of the maximum subarray seen so far. Knowing a maximum subarray of $A[1..j]$, extend the answer to find a maximum subarray ending at index $j+1$ by using the following observation: a maximum subarray of $A[1..j+1]$ is either a maximum subarray of $A[1..j]$ or a subarray $A[i..j+1]$, for some $1 \le i \le j+1$. Determine a maximum subarray of the form $A[i..j+1]$ in constant time based on knowing a maximum subarray ending at index j.





Max-sub(1,j) = max{Max-sub(1,j), Max-sub-end(1,j+1)}



$$\text{Max-sub-end}(1,j+1) = \begin{cases} \text{Max-sub-end}(1,j) + A[j+1], \text{ if Max-sub-end}(1,j) > 0 \\ A[j+1], \text{ otherwise} \end{cases}$$

```
#include <iostream>
using namespace std;
int main()
{
        int A[5]={9,-1,3,-2,4};
        int MSE[5] = {0,0,0,0,0};//MSE for Max-sub-end
        int MS[5] = {0,0,0,0,0};//MS for Max-sub

        MSE[0] = A[0];
        MS[0] = A[0];

        for (int i = 1;i<5;i++)
        {
                if (MSE[i-1] >0)
                    MSE[i] = MSE[i-1]+A[i];
                else
                    MSE[i] = A[i];

                if (MS[i-1]>MSE[i])
                    MS[i] = MS[i-1];
                else
                    MS[i] = MSE[i];
        }
        cout<<MS[4]<<endl;
}
```

4.3-7

Using the master method in Section 4.5, you can show that the solution to the recurrence $T(n) = 4T(n/3) + n$ is $T(n) = \Theta(n^{\log_3 4})$. Show that a substitution proof with the assumption $T(n) \le cn^{\log_3 4}$ fails. Then show how to subtract off a lower-order term to make a substitution proof work.

Assume $T(n) \le cn^{\log_3 4}$

$$
\begin{aligned}
T(n) &\le 4c\left(\frac{n}{3}\right)^{\log_3 4} + n \\
&= cn^{\log_3 4} + n \\
&\ge cn^{\log_3 4}
\end{aligned}
$$

Proof failed !

Assume $T(n) \le cn^{\log_3 4} - dn$

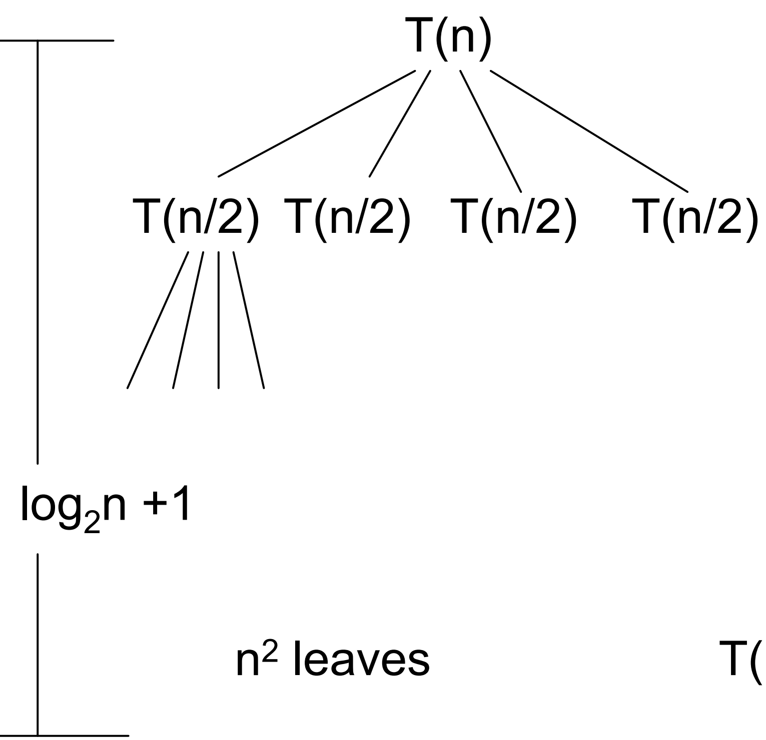
$$
\begin{aligned}
T(n) &\le 4c\left(\frac{n}{3}\right)^{\log_3 4} - \frac{4}{3}dn + n \\
&= cn^{\log_3 4} - dn - \left(\frac{1}{3}d - 1\right)n \\
&\le cn^{\log_3 4} - dn \qquad (c \ge 0, d \ge 3)
\end{aligned}
$$

4.5-4

Can the master method be applied to the recurrence $T(n) = 4T(n/2) + n^2 \lg n$? Why or why not? Give an asymptotic upper bound for this recurrence.

$$a = 4, b = 2, \log_b a = 2, f(n) = \Omega(n^2)$$

$\boxed{f(n) = \Omega(n^{2+\epsilon})}$ $\epsilon > 0$ is not exist! Master method failed!



$n^2 \lg n$

$4*(n/2)^2 \lg(n/2) = n^2 \lg(n/2)$

$4^i (n/2^i)^2 \lg(n/2^i) = n^2 \lg(n/2^i)$

$$T(n) = n^2(\underbrace{\lg n + \lg n - \lg 2 + \lg n - 2\lg 2 + \ldots \lg n - (\log_2 n - 1)\lg 2}_{\log_2 n}) + n^2$$

$n^2 \lg^2 n$

4-1 Recurrence examples

Give asymptotic upper and lower bounds for $T(n)$ in each of the following recurrences. Assume that $T(n)$ is constant for $n \le 2$. Make your bounds as tight as possible, and justify your answers.

a. $T(n) = 2T(n/2) + n^4$.

b. $T(n) = T(7n/10) + n$.

c. $T(n) = 16T(n/4) + n^2$.

d. $T(n) = 7T(n/3) + n^2$.

e. $T(n) = 7T(n/2) + n^2$.

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

g. $T(n) = T(n-2) + n^2$.

Theorem 4.1 (Master theorem)

Let $a \geq 1$ and $b > 1$ be constants, let $f(n)$ be a function, and let $T(n)$ be defined on the nonnegative integers by the recurrence

$$T(n) = aT(n/b) + f(n) ,$$

where we interpret n/b to mean either $\lfloor n/b \rfloor$ or $\lceil n/b \rceil$. Then $T(n)$ has the following asymptotic bounds:

1. If $f(n) = O(n^{\log_b a - \epsilon})$ for some constant $\epsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$.
2. If $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \lg n)$.
3. If $f(n) = \Omega(n^{\log_b a + \epsilon})$ for some constant $\epsilon > 0$, and if $af(n/b) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$. ■

e. $T(n) = 7T(n/2) + n^2$.

$$a = 7, b = 2, f(n) = n^2 = O(n^{\log_2 7 - \epsilon}) \qquad (0 < \epsilon \leq \log_2 7 - 2)$$

Apply Case 1 of the Master Theorem

$$T(n) = \Theta(n^{\lg 7})$$

c. $T(n) = 16T(n/4) + n^2$.

$$a = 16, b = 4, f(n) = n^2 = \Theta(n^{\log_4 16})$$

Apply Case 2 of the Master Theorem

$$T(n) = \Theta(n^2 \lg n)$$

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

$$a = 2, b = 2, f(n) = n^4 = \Omega(n^{\log_2 2 + 3})$$

$$af(\frac{n}{b}) = 2(\frac{n}{2})^4 = \frac{1}{8}n^4 \le cf(n) \qquad (\frac{1}{8} \le c < 1)$$

Apply Case 3 of the Master Theorem

$$T(n) = \Theta(n^4)$$

4-3 *More recurrence examples*

Give asymptotic upper and lower bounds for $T(n)$ in each of the following recurrences. Assume that $T(n)$ is constant for sufficiently small n. Make your bounds as tight as possible, and justify your answers.

a. $T(n) = 4T(n/3) + n \lg n$.

b. $T(n) = 3T(n/3) + n/\lg n$.

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

d. $T(n) = 3T(n/3 - 2) + n/2$.

e. $T(n) = 2T(n/2) + n/\lg n$.

f. $T(n) = T(n/2) + T(n/4) + T(n/8) + n$.

g. $T(n) = T(n-1) + 1/n$.

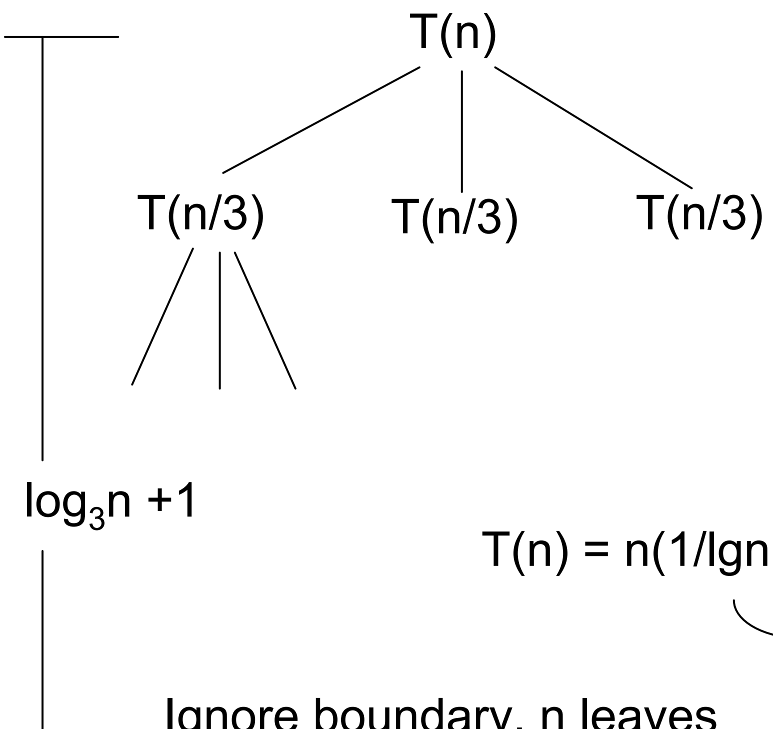
h. $T(n) = T(n-1) + \lg n$.

i. $T(n) = T(n-2) + 1/\lg n$.

j. $T(n) = \sqrt{n}T(\sqrt{n}) + n$.

$b.\ \ T(n) = 3T(n/3) + n/\lg n.$

- a = 3; b=3, f(n) = n/lgn;
- $n^{\log_b a}$ = n; lgn = $O(n^{\varepsilon})$
- Cannot apply Master Theorem, use recursion tree.



n/lgn

3*(n/3)/lg(n/3) = nlg(n/3)

$3^i (n/3^i)/\lg(n/3^i) = n\lg(n/3^i)$

$$T(n) = n\underbrace{(1/\lg n + 1/(\lg n - \lg 3) + 1/(\lg n - 2\lg 3) + \ldots 1/(\lg n - (\log_3 n - 1)\lg 3))}_{\log_3 n} + n$$

$$T(n) = n\underbrace{(1/\lg n+1/(\lg n-\lg 3)+1/(\lg n-2\lg 3)+\ldots 1/(\lg n-(\log_3 n-1)\lg 3))}_{\log_3 n}+n$$

Ignore the boundary condition; let $n = 3^k$, then n= klg3

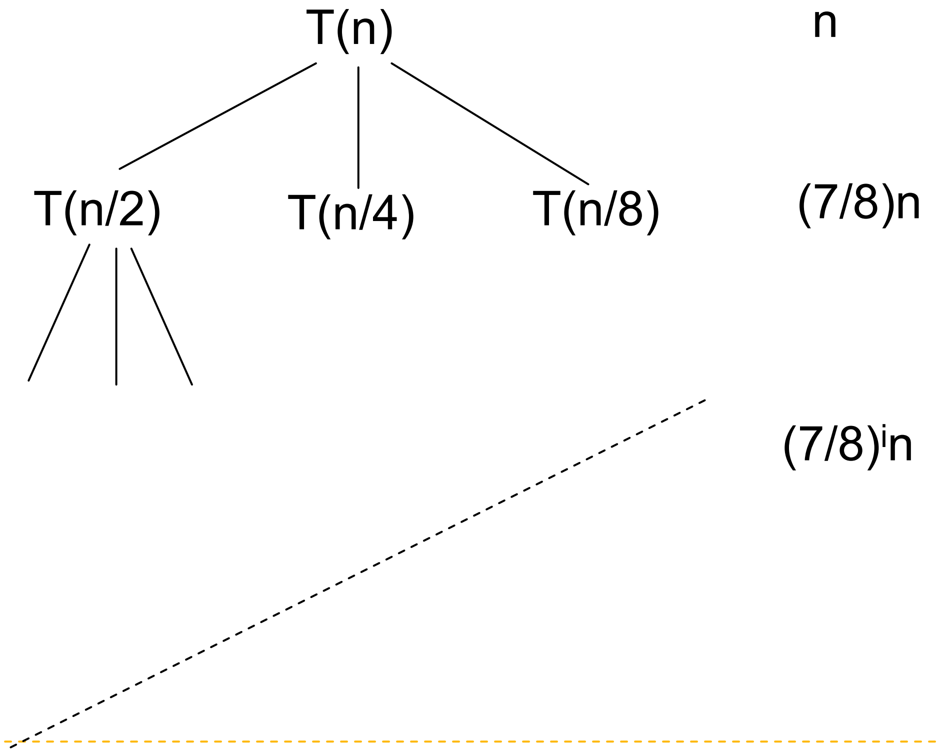
$$T(n) = n/\lg 3(1/k+1/(k-1) +1/(k-2)+\ldots 1/1)+n$$
$$= n/\lg 3^* \Theta(\ln k)+n= \Theta(n\lg\lg n)$$ (Harmonic series property(调和级数性质))

$f.\quad T(n) = T(n/2) + T(n/4) + T(n/8) + n.$

The length of longest path is lg(n)

The length of shortest path is $\log_8(n)$

$$T(n) < \sum_{i=0}^{\lg n - 1} \left(\frac{7}{8}\right)^i n = \frac{1-(7/8)^{\lg n}}{1-(7/8)} n = 8\left(n - n^{1+\lg(7/8)}\right) = O(n)$$



$$T(n) > \sum_{i=0}^{\log_8 n - 1} \left(\frac{7}{8}\right)^i n = \frac{1-(7/8)^{\log_8 n}}{1-(7/8)} n = 8\left(n - n^{1+\log_8(7/8)}\right) = \Omega(n)$$

Can also be proved by guess and substitution.

$$g.\quad T(n) = T(n-1) + 1/n.$$

Use recursion tree, the total time is

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

$i.\quad T(n) = T(n-2) + 1/\lg n.$

- T(n) = 1/lgn+1/lg(n-2)+…+ 1/lg2.
- Let n = 2k,
- T(n) = 1/lg2+1/(lg2+lg2)+1/(lg2+lg3)+…1/(lg2+lgk)
- T(n)>k/(lg2+lgk)= Ω(k/lgk)= Ω(n/lgn)
- T(n) = 1/lg2+1/lg4+1/lg6+…+1/lg2k
- <1/lg2+1/lg3+1/lg4+…+1/lgk < $\int_2^k \frac{1}{\lg x} dx$

$$\int \frac{1}{\ln x} dx = \frac{x}{\ln x} + \int \frac{1}{\ln^2 x} dx$$

T(n) = O(n/lgn) = Θ(n/lgn)

j. $T(n) = \sqrt{n}\,T(\sqrt{n}) + n.$

$$T(n) = \sqrt{n}\,T(\sqrt{n}) + n$$

Diving by n on both sides we get,

$$\frac{T(n)}{n} = \frac{T(\sqrt{n})}{\sqrt{n}} + 1.$$

Renaming $S(n) = T(n)/n$, we get

$$S(n) = S(\sqrt{n}) + 1.$$

$S(n) = S(n^{(1/2)})+1 = S(n^{(1/4)})+2 = S(n^{(1/2^\wedge i)})+i$,
Let $(S(c)= \Theta(1)$, we get $i = \Theta(\lg\lg n)$

$$S(n) = \Theta(\lg\lg n).$$

$$\Rightarrow T(n) = nS(n) = \Theta(n \lg\lg n)$$

- 讨论题：结合例子说明为什么分治策略（Divide and Conquer）可以比暴力法（Brute-Force) 更加高效？