习题2-5

TC第4.1节练习**5**

TC第4.3节练习3、7

TC第4.4节练习2、**8**

TC第4.5节练习**4**

TC第4章问题1、**3**、4

4.1-5

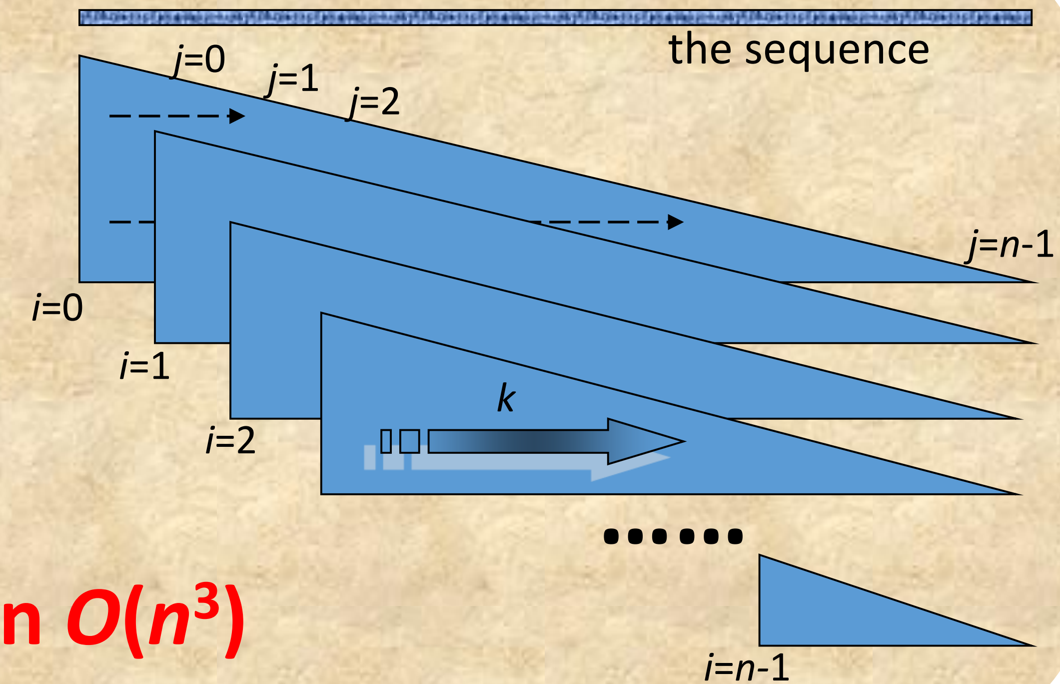
Use the following ideas to develop a nonrecursive, linear-time algorithm for the maximum-subarray problem. Start at the left end of the array, and progress toward the right, keeping track of the maximum subarray seen so far. Knowing a maximum subarray of $A[1 \mathinner{..} j]$, extend the answer to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

Max-sum Subsequence

- The problem: Given a sequence *S* of integer, find the **largest sum** of a consecutive subsequence of *S*. (0, if all negative items)
  - An example: -2, 11, -4, 13, -5, -2; the result 20: (11, -4, 13)

A brute-force algorithm:

```
MaxSum = 0;
 for (i = 0; i < N; i++)
  for (j = i; j < N; j++)
  {
   ThisSum = 0;
   for (k = i; k <= j; k++)
   ThisSum += A[k];
   if (ThisSum > MaxSum)
     MaxSum = ThisSum;
  }
 return MaxSum;
```



in $O(n^3)$

More Precise Complexity

$$\text{The total cost is}: \sum_{i=0}^{n-1}\sum_{j=i}^{n-1}\sum_{k=i}^{j}1$$

$$\sum_{k=i}^{j}1 = j-i+1$$

$$\sum_{j=i}^{n-1}(j-i+1) = 1+2+\ldots+(n-i) = \frac{(n-i+1)(n-i)}{2}$$

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

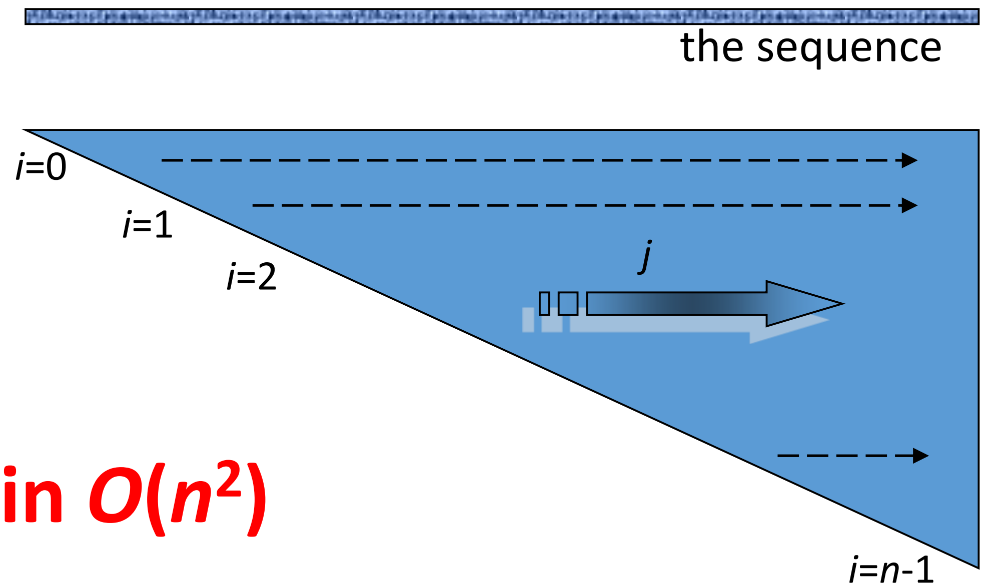
$$= \frac{1}{2}\sum_{i=1}^{n}i^2 - (n+\frac{3}{2})\sum_{i=1}^{n}i + \frac{1}{2}(n^2+3n+2)\sum_{i=1}^{n}1$$

$$= \frac{n^3+3n^2+2n}{6}$$

Decreasing the number of loops

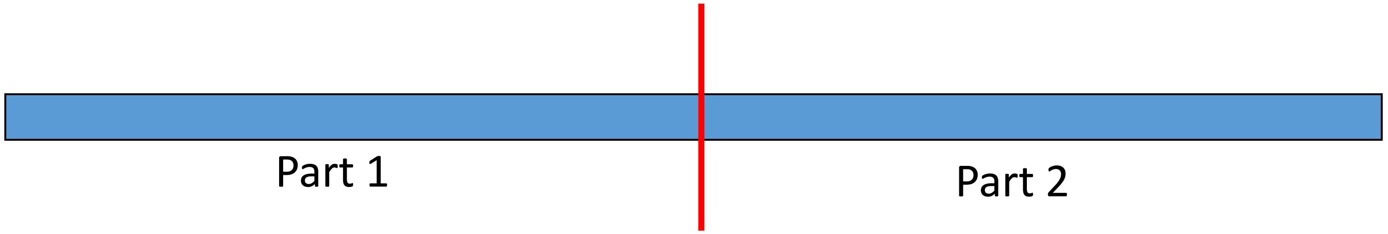
An improved algorithm

```
MaxSum = 0;
 for (i = 0; i < N; i++)
 {
  ThisSum = 0;
  for (j = i; j < N; j++)
  {
   ThisSum += A[j];
   if (ThisSum > MaxSum)
    MaxSum = ThisSum;
  }
 }
 return MaxSum;
```

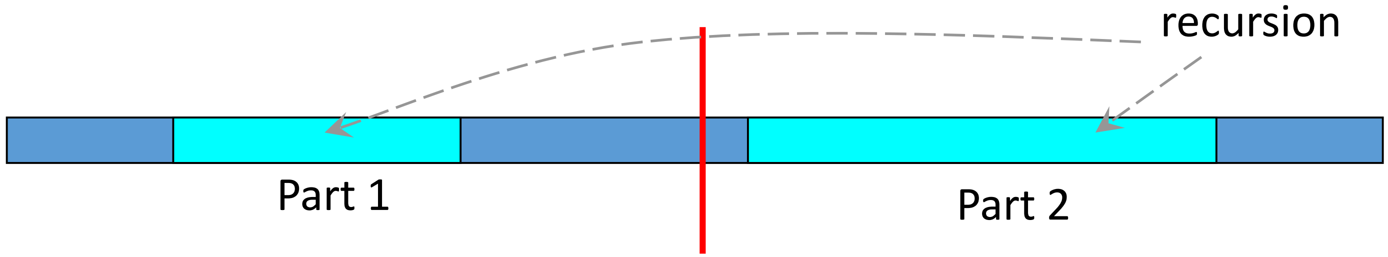


in $O(n^2)$

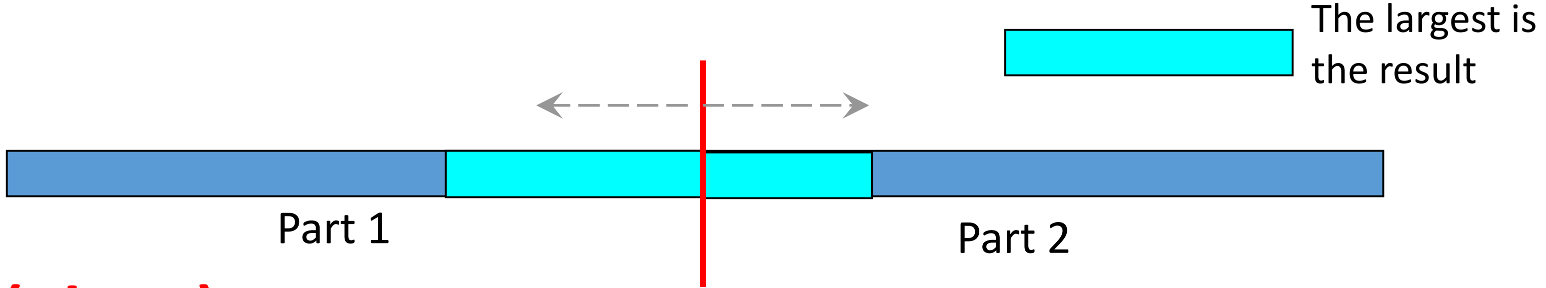
Power of
Divide and Conquer



the sub with largest sum may be in:



or:



in *O*(*n*log*n*)

Power of Divide and Conquer

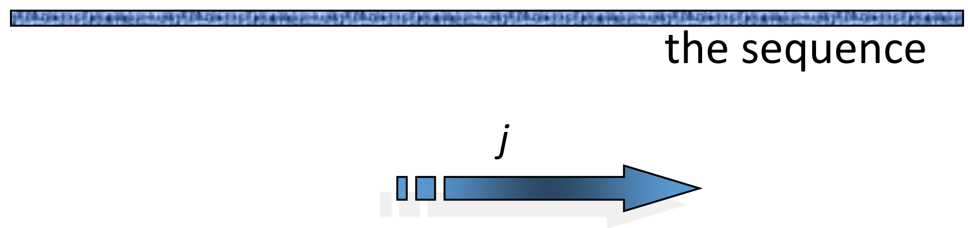
```
Center = (Left + Right) / 2;
 MaxLeftSum = MaxSubSum(A, Left, Center);  MaxRightSum = MaxSubSum(A, Center + 1, Right);

 MaxLeftBorderSum = 0; LeftBorderSum = 0;
 for (i = Center; i >= Left; i--)
 {
   LeftBorderSum += A[i];
   if (LeftBorderSum > MaxLeftBorderSum)    MaxLeftBorderSum = LeftBorderSum;
 }

 MaxRightBorderSum = 0; RightBorderSum = 0;
 for (i = Center + 1; i <= Right; i++)
 {
   RightBorderSum += A[i];
   if (RightBorderSum > MaxRightBorderSum)  MaxRightBorderSum = RightBorderSum;
 }
 return Max3(MaxLeftSum, MaxRightSum,
        MaxLeftBorderSum + MaxRightBorderSum);
```

Note: this is the core part of the procedure, with base case and wrap omitted.

A Linear Algorithm

First scan the array to eliminate the case of “all negative integers”

```
ThisSum = MaxSum = 0;
  for (j = 0; j < N; j++)
  {
    ThisSum += A[j];
    if (ThisSum > MaxSum)
      MaxSum = ThisSum;
    else if (ThisSum < 0)
      ThisSum = 0;
  }
  return MaxSum;
```



This is an example of “online algorithm”

Negative item or subsequence cannot be a prefix of the subsequence we want.

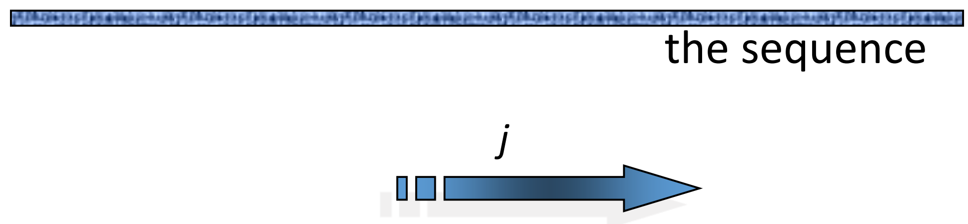
in *O*(*n*)

A Linear Algorithm

First scan the array to eliminate the case of "all negative integers"

FIND-MAXIMUM-SUBARRAY

```
s[1].left ← 1
S[i] save the left and right index and the sum of a maxium subarray of A[1..n]
s[1].right ← 2
s[i].sum ← a[1]
for i ← 2 to n
if s[i − 1].sum < 0
    then s[i].left ← i
         s[i].right ← i + 1
         s[i].sum ← a[i]
    else s[i].left ← s[i − 1].left
         s[i].right ← i + 1
         s[i].sum ← s[i − 1].sum + a[i]
Max ← s[1]
for i ← 2 to n
if Max.sum < s[i].sum
    then Max ← s[i]
return Max
```



This is an example of "online algorithm"

Negative item or subsequence cannot be a prefix of the subsequence we want.

in $O(n)$

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.

$Assume$: $T(n) \le cn^{\log_3 4} - dn, d > 0$

Then, as $T(n) = 4T\left(\frac{n}{3}\right) + n$, we have

$$T(n) \le 4\left(c\left(\frac{n}{3}\right)^{\log_3 4} - d\left(\frac{n}{3}\right)\right) + n$$

$$= cn^{\log_3 4} - \frac{4dn}{3} + n$$

$$\le cn^{\log_3 4} - dn$$

$$\begin{aligned} T(n) &\le 4c(n/3)^{\log_3 4} + n \\ &\le cn^{\log_3 4} \boxed{+\, n} \end{aligned}$$

$$-\frac{4dn}{3} + n \le -dn$$

$$-\frac{4d}{3} + 1 \le -d$$

$$3 \le d$$

$$\boxed{T(n) = O(n^{\log_3 4})}$$

Next step:

$T(n) = \Omega(n^{\log_3 4})$

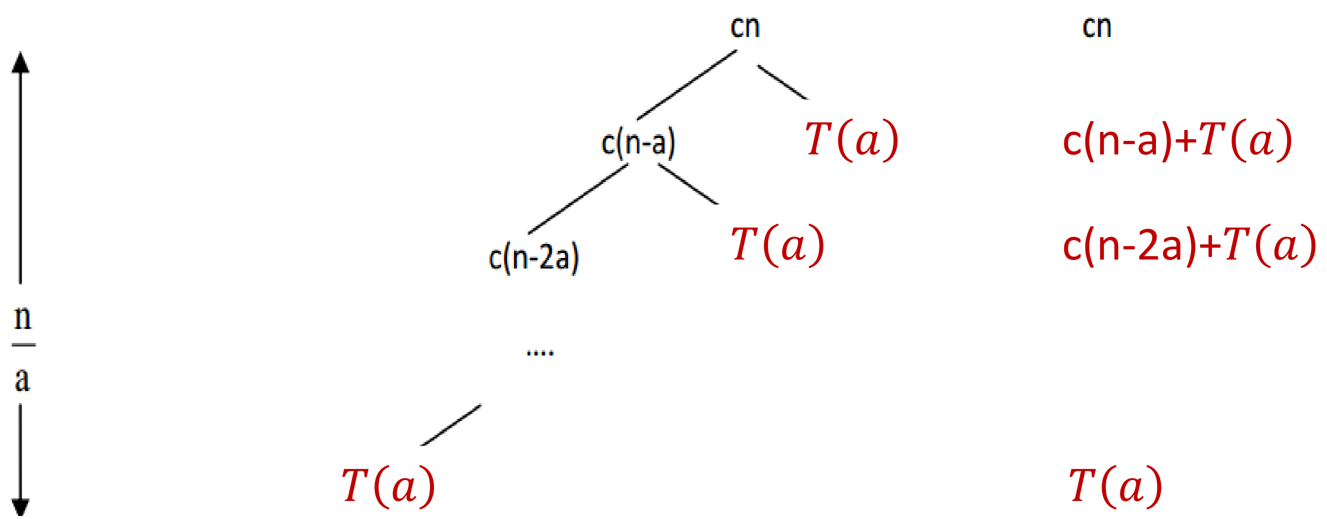
4.4-8

Use a recursion tree to give an asymptotically tight solution to the recurrence $T(n) = T(n-a) + T(a) + cn$, where $a \geq 1$ and $c > 0$ are constants.

T(n)=T(n-a)+T(a)+cn

Assume n=ka, k>=1

$T(a) = ?$



$$T(n) = T(a) + \sum_{i=0}^{k-2} c(n - ia) + (k-1)T(a)$$

$$= cn(k-1) - ca\frac{(k-1)*(k-2)}{2} + kT(a)$$

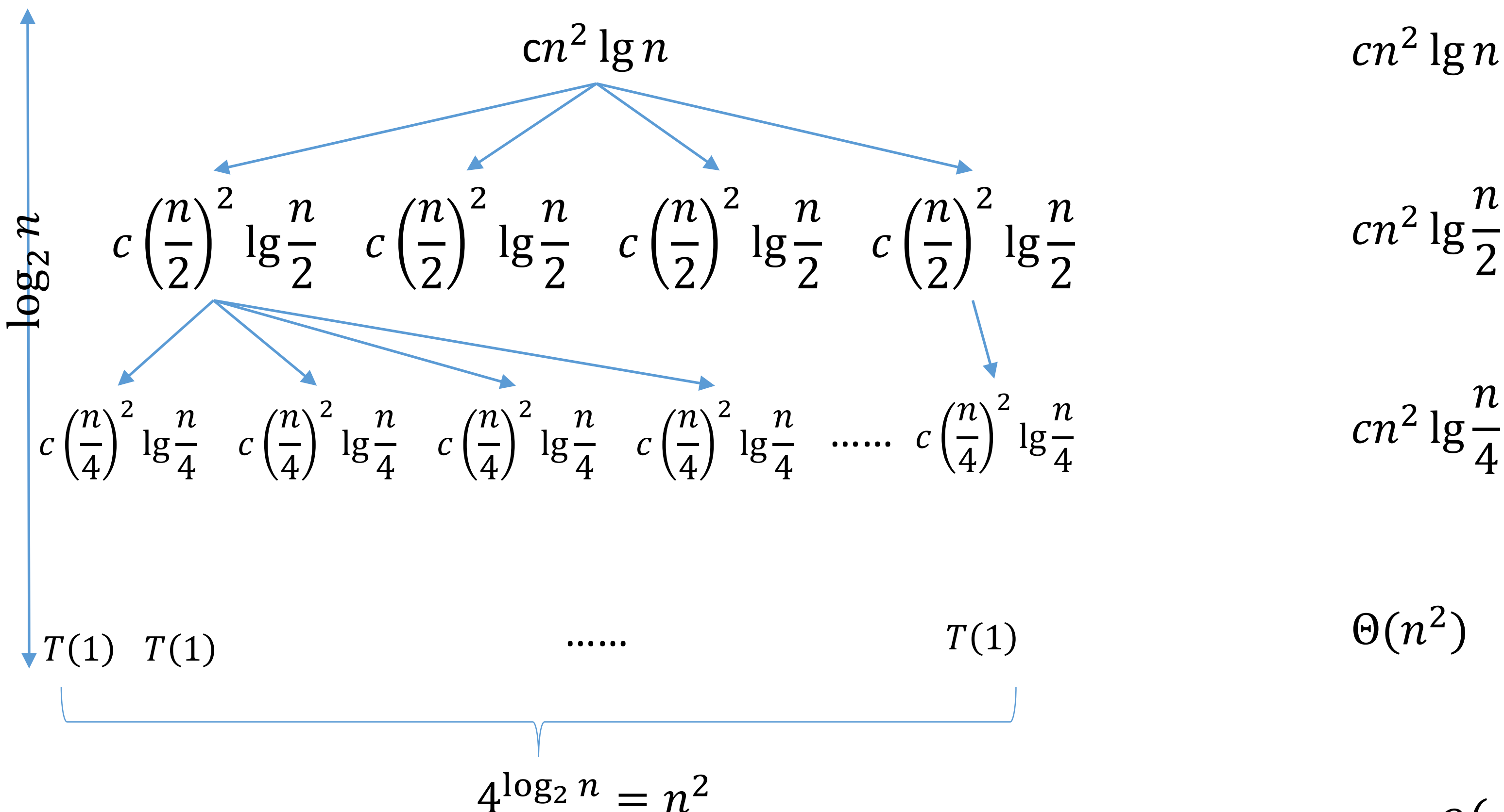
$$= cn\left(\frac{n}{a} - 1\right) - ca\frac{\left(\frac{n}{a} - 1\right)*\left(\frac{n}{a} - 2\right)}{2} + \frac{n}{a}T(a)$$

$T(a) = \Theta(1)$

$$= cn\left(\frac{n}{a} - 1\right) - ca\frac{\left(\frac{n}{a} - 1\right)*\left(\frac{n}{a} - 2\right)}{2} + \frac{n}{a}T(a) = \frac{cn^2}{2a} + \frac{(3a + 2T(a) - 2ca)}{2a}n - ca = \Theta(n^2)$$

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.



$$O\left(n^2 \lg^2 n\right)$$

$$T(n) = \Theta(n^2) + \sum_{i=0}^{\log_2 n-1} cn^2 \lg\left(\frac{n}{2^i}\right) \le \Theta(n^2) + \sum_{i=0}^{\log_2 n-1} cn^2 \lg(n) = \Theta(n^2) + cn^2 \lg(n) * \log_2 n$$

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

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

5. $\Theta(n \lg\lg n)$

$$T(n) = 2T(n/2) + \frac{n}{\lg n} = 4(n/4) + 2\frac{n/2}{\lg(n/2)} + \frac{n}{\lg n} = 4T(n/4) + \frac{n}{\lg n - 1} + \frac{n}{\lg n}$$

$$= nT(1) + \sum_{i=0}^{\lg n - 1} \frac{n}{\lg n - i} = \underline{nT(1) + n\sum_{i=1}^{\lg n} \frac{1}{\lg n}} \quad ?$$

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

$$nT(1) + n\sum_{i=1}^{\lg n} \frac{1}{i} = \Theta(n \lg\lg n)$$

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

$$\begin{aligned} T(n) &= T(n-1) + 1/n = \frac{1}{n} + \frac{1}{n-1} + T(n-2) \\ &= \frac{1}{n} + \frac{1}{n-1} + \frac{1}{n-2} + T(n-3) \\ &= \sum_{i=0}^{n-1} \frac{1}{n-i} = \sum_{i=1}^{n} \frac{1}{i} \\ &= \Theta(\lg n) \end{aligned}$$

Harmonic series

For positive integers n, the nth ***harmonic number*** is

$$\begin{aligned} H_n &= 1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \cdots + \frac{1}{n} \\ &= \sum_{k=1}^{n} \frac{1}{k} \\ &= \ln n + O(1) . \end{aligned}$$

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

$$\begin{aligned} T(n) &= \lg n + T(n-1) = \lg n + \lg n - 1 + T(n-2) \\ &= \sum_{i=0}^{n-1} \lg(n-i) = \sum_{i=1}^{n} \lg i = \lg(n!) \\ &= \Theta(n \lg n) \end{aligned}$$

terms in the factorial product is at most n. ***Stirling's approximation***,

$$n! = \sqrt{2\pi n}\left(\frac{n}{e}\right)^n \left(1 + \Theta\left(\frac{1}{n}\right)\right),$$

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

$$T(n) = \frac{1}{\lg n} + \frac{1}{\lg n - 2} + \dots$$

$$= \sum_{i=1}^{n/2} \frac{1}{\lg(2i)}$$

Guess O(n)

$T(n) \leq \mathrm{cn}$

$$T(n) = T(n-2) + \frac{1}{\lg n} \leq c(n-2) + \frac{1}{\lg n} = cn - 2c + \frac{1}{\lg n} \leq cn$$

仅需$2c - \frac{1}{\lg n} \geq 0 \Rightarrow c \geq \frac{1}{2\lg n}$,取c=1/2, $n_0 = 2$即可

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

$$T(n) = \frac{1}{\lg n} + \frac{1}{\lg n - 2} + \dots$$

$$= \sum_{i=1}^{n/2} \frac{1}{\lg(2i)}$$

Guess $\Omega(\lg \lg n)$?

Guess $\Omega\left(\frac{n}{\lg n}\right)$? $\quad \mathrm{T}(n) \geq c\frac{n}{\lg n}$

$$T(n) = T(n-2) + \frac{1}{\lg n} \geq \frac{c(n-2)}{\lg(n-2)} + \frac{1}{\lg n} = \frac{cn}{\lg(n-2)} - \frac{2c}{\lg(n-2)} + \frac{1}{\lg n}$$

$$\geq \frac{cn}{\lg n} - \frac{2c}{\lg(n-2)} + \frac{1}{\lg n}$$

$$\geq \frac{cn}{\lg n}$$

$$-\frac{2c}{\lg(n-2)} + \frac{1}{\lg n} \geq 0$$

$$c \leq \frac{\lg(n-2)}{2\lg n} = \frac{\lg_n(n-2)}{2}$$

Let $c = \frac{1}{4}$, $n_0 = 4$, the condition can be satisfied