Master Method - Bound

Exact Answer

⇑

Memorize Certain case - 10sec

- Master Method
- "Cookbook"

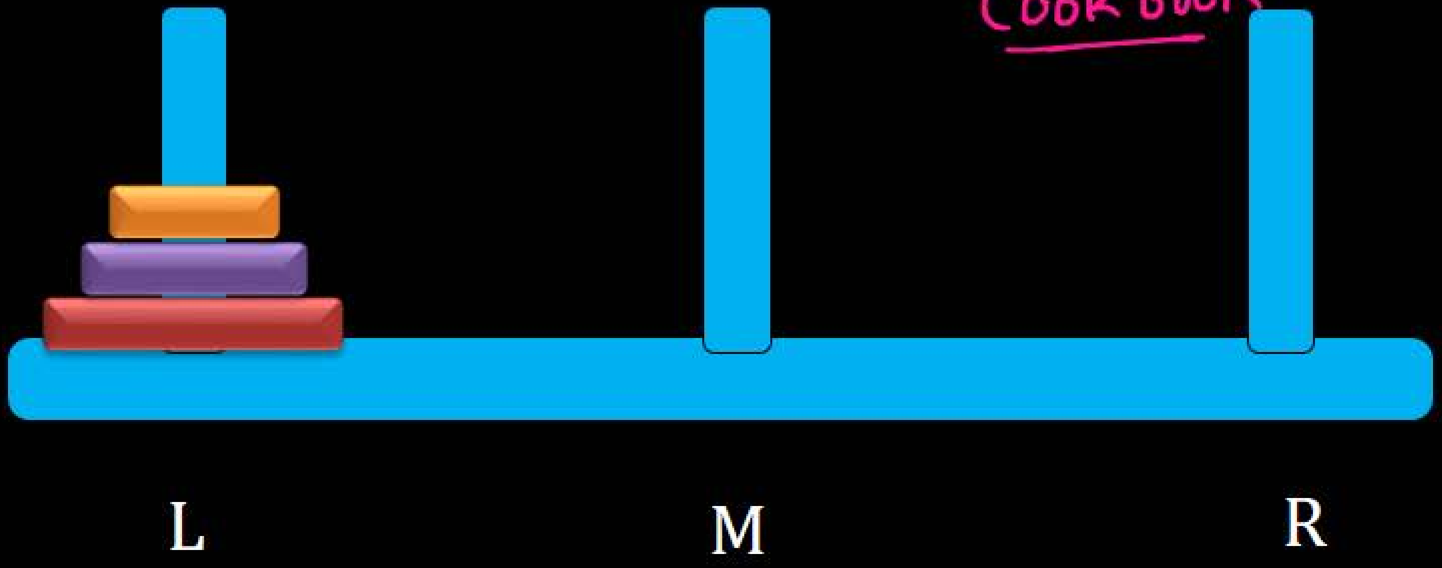
Recurrence Relation

Substitution

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

- Substitution Method
- Recurrence tree
- Master Method

a subproblem $a \geq 1$ each of size n/b $b > 1$



$$T(n) = 2T(n/2) + 1$$

2 subprolem. each of size $n/2$

Recurrence Relation

Question

Consider the following Recurrence Relation

$T(n) = T(n-1) + \log n$ — (I)

$T(1) = 1$

$T(n-1) = T(n-2) + \log(n-1)$ — (II)

put (II) in I

$T(n) = T(n-2) + \log(n-1) + \log n$

~~guest~~ guess the kth

$T(n) = \underline{T(n-k)} + T(n-(k-1) \dots + \log n$

$= \quad \cancel{n-1} \quad \underline{n-k=1} =$

$= 1 + \log 2 + \log 3 + \dots + \log n$

$1 + \log 1 \cdot 2 \cdot 3 \cdot \dots (n-1)$

$1 + \underline{\log n!}$

$\Theta(n \log n)$

Question

Best case Analysis - Quicksort

- Analysis of Mergesort

Consider the following Recurrence Relation

$T(n) = 2T(n/2) + n$ — (I)

$T(1) = 1$

$T(n/2) = 2T(n/2^2) + n/2$ — (II)

put (II) in (I)

$T(n) = 2\left(2T(n/2^2) + n/2\right) + n$

$= 2^2T(n/2^2) + n + n \; (2n)$

. guess the k^{th} term

$T(n) = 2^k T(n/2^k) + \underline{kn}$

Reduce to base condition

$\underline{n/2^k = 1}$

$\underline{n = 2^k}$ $\quad k = \log_2 n$

$T(n) = \underline{2^k} T(1) + n\log n$

$= n + \underline{n\log n}$ $\quad \underline{\theta(n\log n)}$

Question

$T(n) = 2T(n/2) + n$

Consider the following Recurrence Relation

$T(n) = T(n/2) + n$ — I $\quad \underline{\Theta(n)}$

$T(1) = 1$

$T(n/2) = T(n/2^2) + n/2$ — II

put (II) in (I)

$T(n) = T(n/2^2) + n/2 + n$ — (IV)

$T(n/2^2) = T(n/2^3) + n/2^2$ — III

put (III) in (IV)

$T(n) = T(n/2^3) + n/2^2 + n/2 + n$

guess the kth term

$T(n) = \underline{T(n/2^k)} + n/2^{k-1} + \cdots + n/2 + n$

Reduce to base condition

$n = 2^k \quad = \quad k = \log_2 n$

$n + n/2 + n/2^2 + \cdots n/2^{k-1} + 1$

$n\left(\underline{\frac{1}{2^0} + 1/2} + \underline{1/2^2 +} \cdots \underline{1/2^{k-1}}\right) + 1 = 2n-1$

$\Theta(n)$

Mute Stop Video Participants Chat Q&A Polls New Share Pause Share Annotate More

You are screen sharing Stop Share

ACE Live Class 1

$$n \times \frac{\frac{1}{2}\left(1 - \frac{1}{2^k}\right)}{1 - \frac{1}{2}} + 1$$

$$2\,n\left(1 - \frac{1}{n}\right) + 1$$

$$2n - 2 + 1 = 2n-1$$

Question

Consider the following Recurrence Relation

$T(n) = T(\sqrt{n}) + \underline{c}$ —(I) C is constant

$T(\underline{2}) = 1$

$T(\sqrt{n}) = T(n^{1/2})$

$T(n^{1/2}) = T(n^{1/2^2}) + C$ — $\underline{\text{II}}$

put (II) in $\underline{\text{I}}$

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

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

lets guess the kth term

$T(n) = T(n^{1/2^k}) + \underline{KC}$

Reducing to base condition

$n^{1/2^k} = 2$

$\Rightarrow n = 2^{2^k}$

$\log_2$ both side

$\log_2 n = \log_2 \underline{2}^{2^k}$

$2^k = \log_2 n$

$\log_2$ both side

$\underline{\log_2 2^k} = \log\log n$

$K = \log\log n$

$T(n) = \underline{T(2)} +$

$\log\log_2 n \cdot C$

$1 + c\log\log_2 n$

$\theta(\log\log_2 n)$

Answer

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

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

$= \left(T\left(n^{1/2^2}\right) + c\right) + c$

$= T\left(n^{1/2^2}\right) + c + c$

$= T\left(n^{1/2^2}\right) + c + c + c$

$$= T\left(n^{1/2^k}\right) + \underbrace{c + - - - - + c + c + c}_{k\ -1\ \text{times}}$$

Assume $n^{1/2^k} = 2 \Rightarrow$ n = 2 $n^{2^k} \Rightarrow$ 2k = logn $\Rightarrow$k

= log log n

= T(2) + c + ….. + c + c +c, t(2) =1

$$= \underbrace{c + c + c \ldots\ldots + c + c + c}_{\text{k times}}$$

= k. c

Question

Consider the following Recurrence Relation

$$T(n) = 2T(\sqrt{n}) + 1 \quad \text{— (I)}$$

$$T(2) = 1$$

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

put II in (I)

$$T(n) = 2\left(2T(n^{1/2^2}) + 1\right) + 1$$

$$= \underline{2^2} + (n^{1/2^2}) + 2 + 1$$

guess the kth team

$$T(n) = 2^k T(n^{1/2^k}) + 2^{k-1} + \dots + 2 + 1$$

Reduce to base Condition

$$n^{1/2^k} = 2$$

$$\therefore n = 2^{2^k} \qquad \therefore 2^k = \underline{\log_2 n}$$

$$k = \log\log_2 n$$

$$T(n) = 2^k \underline{T(2)} + 2^{k-1} + \dots + 2 + 1$$

$$= 1 + 2 + 2^2 + \dots 2^k$$

$$= \frac{1(2^{k+1} - 1)}{2 - 1} = 2^{k+1} - 1$$

$$= 2 \cdot \underline{2^k} - 1$$

$$= \underline{2\log_2 n - 1} \quad \Theta(\log_2 n)$$

Answer

$T(n) = 2T(\sqrt{n}) + 1$

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

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

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

$= 2^3T(n^{1/2^3}) + 2^2 + 2 + 1$

………

$= 2^kT(n^{1/2^k}) + 2^{k-1} + \dots\dots\dots 2^2 + 2 + 1$

$= 2^k + 2^{k-1} + \dots\dots\dots 2^2 + 2 + 1$

$$= \frac{1\left(2^{k+1}-1\right)}{2-1}$$

Assume $n^{1/2^{k}} = 2 \Rightarrow \mathrm{n} = 2^{2^{k}} \Rightarrow 2^{\mathrm{k}} = \log n$

= 2log n -1

= logn

Question

I Cant Solve by Master Method

Consider the following Recurrence Relation

5 min $T(n) = \sqrt{n}T(\sqrt{n}) + n$, n > 2 — (I)
$T(2) = 2$

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

put equation (II) in (I)

$T(n) = n^{1/2}\left[n^{1/4} T(n^{1/2^2}) + n^{1/2}\right] + n$

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

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

$\frac{1/2(1 - 1/2^k)}{1 - 1/2}$

$n^{1/2+1/2^2+1/2^2} = n^{1/2+1/2} = n$

$T(n^{1/2^2}) = n^{1/2^3} T(n^{1/2^3}) + n^{1/2^2}$

guess the kth term

$n^{1/2+1/2^2+\cdots+1/2^k} T(n^{1/2^k}) + kn$

Sum

$n^{1-1/2^k} T(n^{1/2^k}) + kn$

Reduce to base

$n^{1/2^k} = 2 \Rightarrow n = 2^{2^k}$

$2^k = \log_2 n$ $\quad k = \log\log n$

$n^{1-1/2^k} T(2) + kn$

$+ n \cdot \log\log n$

$\frac{n}{n^{1/2^k}}$

$= n/2 * 2 + n\log\log n$

$\Theta(n\log\log n)$

"Algebraic transformation"

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

$\Rightarrow$ form Master Method

$2^k = n$ $\qquad$ $n^{1/2^k} = 2$

$n^{1 - 1/2^k}$

$n \cdot n^{-1/2^k}$

$\dfrac{n}{n^{1/2^k}}$

$= \dfrac{n}{2} \times T(2)$

$= n/2 * 2 = n$

$n^{1 - 1/2^k} * T(2)$

$= n * n^{-1/2^k} * T(2)$

$= \dfrac{n}{n^{1/2^k}} \times T(2)$

$= \dfrac{n}{\cancel{2}} \times \cancel{2}$

$= n$

Question

Homework problem

Consider the following Recurrence Relation

$$T(n) = T(\sqrt{n}) + \log n, \ n > 1$$
$$T(1) = 1$$

Summary

Recurrence Relation	Solution (Bound)
$T(n) = T(n-1) + 1$	$1+1+1+\cdots+1$ $\Theta(n)$
$T(n) = T(n-1) + n$	$n+n-1+n-2+\cdots+1 = \frac{n(n+1)}{2}$ $\Theta(n^2)$
$T(n) = T(n-1) + \frac{1}{n} + a$	$\log_2 n + an$ $\Theta(n)$
$T(n) = T(n-1) + \frac{1}{n}$	$1/n + 1/(n-1) + \cdots + 1/1 =$ $\Theta(\log_2 n)$
$T(n) = 2T(n-1) + 1$	$\Theta(2^n) - 2^n - 1$
$T(n) = T(\sqrt{n}) + c$	$T(n) = \Theta(\log\log_2 n)$
$T(n) = 2T(\sqrt{n}) + c$	$T(n) = \Theta(\log_2 n)$

H.W

$T(n) = T(n-1) + \log n$

$\Theta = (n\log n)$

Tower of Hanoi

Summary

Recurrence Relation	Solution
$T(n) = 2T(n/2) + c$	$\Theta(n)$
$T(n) = 2T(n/2) + n$	$\Theta(n\log n)$
$T(n) = T(n/2) + c$	$\Theta(\log_2 n)$
$T(n) = \sqrt{n}T(\sqrt{n}) + n$, $n > 2$ $T(2) = 2$	$\Theta(n\log\log n)$
$T(n) = T(\sqrt{n}) + \log n$,	

$T(2) = 1$

12 set of problem –

Remember them

← Home work problem

Summary

Recurrence Relation	Solution
$T(n) = T(n-1) + c$	$\Theta(n)$
$T(n) = T(n-1) + n$	$\Theta(n^2)$
$T(n) = T(n-1) + \frac{1}{n} + a$	$\Theta(n)$
$T(n) = T(n-1) + \frac{1}{n}$	$\Theta(\log n)$
$T(n) = 2T(n-1) + c$	$\Theta(2^n)$
$T(n) = T(\sqrt{n}) + c$	$\Theta(\log\log n)$
$T(n) = 2T \langle\sqrt{n}) + c$	$\Theta(\log n)$

Summary

Recurrence Relation	Solution
T(n) = 2T (n/2) + c	Θ(n)
T(n) = 2T (n/2) + n	Θ(nlogn)
T(n) = T (n/2) +c	Θ(logn)
$T(n) = \sqrt{n}T(\sqrt{n}) + n$, n > 2 $T(2) = 2$	Θ(nloglogn)
$T(n) = T(\sqrt{n}) + \log n$,	Θ(logn)

Home Work

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

$$T(n) = 3T\left(n/2\right) + n$$

$$T(n) = 8.T\left(n/2\right) + n^2$$

$$T(n) = 7.T\left(n/2\right) + n^2$$

$$T(n) = 2T\left(n/2\right) + n\log n$$

$$T(n) = 9T\left(n/3\right) + n^2$$

$$T(n) = 2T(\sqrt{n}) + \log n$$

$$T(n) = 3T\left(n/4\right) + n^2$$

Work Book Level-1 42

42. If $T_1 = O(1)$, give the order for the following, Regarding the Complexities

a. $T_n = T_{n-1} + n$ – $n + n-1 + \cdots + 1$ – $\Theta(n^2)$

b. $T_n = T_{n/2} + n$ – $n + n/2 + n/2^2 + \cdots n/2^k$ $\Theta(n)$

c. $T_n = T_{n-1} + \log n$ – $\log n + \log(n-1) \cdots + \log 1$

(a) a > b > c

(b) b > c > a

(c) a > c > b

(d) c > b > a

[C]

$\boxed{n = 2^k}$

trick

Complexity $\lfloor n/2 \rfloor$

$n\left(1 + 1/2 + 1/2^2 + \cdots + 1/2^k\right)$

$\frac{1}{1-1/2} = \frac{2}{1}$

constant

$\Theta(\log n!)$

$\Theta = (n\log n)$

1 a > c > b

Work Book Level-1 43

43. Match the following:

P. $T(n) = T(\lceil n/2 \rceil) + 1$ — Binary Search

Q. $T(n) = 2T(\lfloor n/2 \rfloor) + n$

R. $T(n) = 2T(\lfloor n/2 \rfloor) + 17$

1. $O(n)$
2. $O(\log n)$
3. $O(n \log n)$

Codes:

(a) P–2, Q–2, R–3
(b) P–2, Q–3, R–1
(c) P–2, Q–2, R–1
(d) P–2, Q–3, R–3

$$2T(n/2) + 17$$

$$T(n/2) = 2T(n/2^2) + 17$$

$$\lceil n/2 \rceil \qquad T(n) = 2^2 T(n/2^2) + 2 \times 17$$

$$\lfloor n/2 \rfloor \qquad 2^k T(n/2^k) + k\,17$$

$$n + \log_2 n\, 17$$

$$T(n) = T(n/2) + 1$$

Merge Sort

Quick Sort

$$T(n) = 2T(n/2) + n$$

$$T(n) = 2T(n/2) + 17$$

constant

$$T(n) = T(n/2) + 1$$

$$= T(n/2^2) + 1 + 1$$

$$= T(n/2^k) + k$$

$$n/2^k = 1$$

$$n = 2^k$$

$$k =$$

Work Book Level-1 41

41. Solution for the Recurrence equation

$$T(n) = \begin{cases} 2.T(n/2) + (n-1) & ; n > 1 \\ = 1 & ; n = 1 \end{cases}$$

$T(n) = ?$

(a) $n + 1$ (b) $2n - 1$

(c) $1 + n\log n$ (d) None

Work Book Level-1 37

37. Consider the recurrence given below:

$$T(n) = 2T(n/2) + \log n,$$

What is the order of T(n) ?

(a) $T(n) = \Theta(\sqrt{n})$

(b) $T(n) = O(n \log n)$

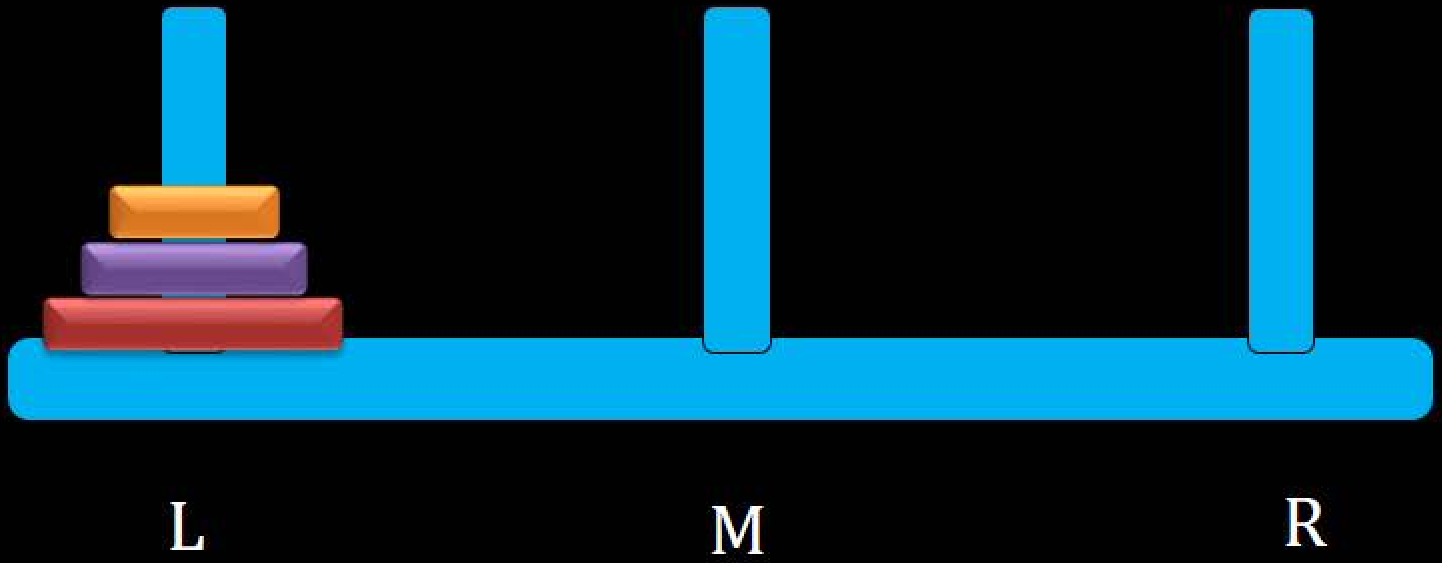
(c) $T(n) = \Theta(n)$

(d) $T(n) = \Omega(n)$

$\boxed{n = 2^k}$ $\quad \lfloor n/2 \rfloor$

$= n/2$

$\lceil n/2 \rceil = \underline{n/2}$



Program to Recurrence Relation

Recursive Function

```
long power(long x, long n){
   if (n == 0)
      return 1;
   else
      return x * power(x, n-1);
}
```

Cost of multiplication

$$\left.\begin{aligned} T(n) &= T(n-1) + 1 \\ T(0) &= 2 \end{aligned}\right\}$$

Solve $\Theta(n)$

GATE 2012

The recurrence relation capturing the optimal execution time of the Towers of Hanoi problem with n discs is

(a) $T(n) = 2T(n - 2) + 2$

(b) $T(n) = 2T(n - 1) + n$

(c) $T(n) = 2T(n/2) + 1$

(d) $T(n) = 2T(n - 1) + 1$

$[n]$

Tower of Hanoi

```
Algorithm Towers Of Hanoi (n, x, y, z){
// Move the top n disks from tower x to tower y.
  if (n ≥ 1) then {
  Towers Of Hanoi (n - 1, x, z, y);
  write ("move top disk from tower", x, "to top of tower", y);
  Towers Of Hanoi (n - 1, z, y, x);
  }
}
```

GATE 2004| 1 Mark Question

Substitution

Step count

The time complexity of the following C function is (assume n > 0)

```
int recursive (int n) {
    if (n == 1)
    return (1);
    else
    return (recursive (n-1) + recursive (n-1);
}
```

(a) O(n)

(b) O(nlog n)

(c) O(n^2)

(d) O(2^n)

(I) Establish Recurrence Relation

$$T(n) = \begin{cases} n == 1 & T(1) = 2 \\ n > 1 & \boxed{2T(n-1) + 1} \end{cases}$$

Same as tower of Hanoi

$T(1) = 1$

$T(n-1) = 2T(n-2) + 1$

$2^2 T(n-2) + 2 + 1$

GATE 2002| 1 Mark Question

The running time of the following algorithm

```
Procedure A(n)

If n<=2 return(1) else return (A(⌈√n⌉));
```

Is best described by

(a) O(n)

(b) O(log n)

(c) O(log log n) ✓ 1

(d) O(1)

Establish Recurrence Relation – done

$$T(n) = \begin{cases} n=2 & T(2) = 2\,(1+1) \\ T(\sqrt{n}) + 1 & n>2 \end{cases}$$

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

GATE 2007| 2 Marks Question

Silly mistake

What is the time complexity of the following recursive function:

```
int DoSomething (int n) {
  if (n <= 2)
      return 1;
   else
      return (DoSomething(floor(sqrt(n))) + n);
}
```

Similar

$$T(n) = \begin{cases} T(2) = 2 & n = 2 \\ T(\sqrt{n}) + 1 & n > 2 \end{cases}$$

cons

cost of Addition

Addition of n requires constant time

$\Theta(n \log\log n)$

Question

Consider the following algorithms. Assume, procedure A and procedure B take O (1) time. Derive the time complexity of the algorithm in O-notation.

```
Algorithm what (n)
  begin
       if n = 1 then call A
       else
       begin
            what (n-1);
            call B(n)
       end
end
```

Question

```
Algorithm what (n)
  begin
        if n = 1 then call A
        else
        begin
              what (n-1);
              call B(n)
        end
end
```

Question

Consider the following algorithms. Assume, procedure A takes O(1) and procedure B take O(n) unit of time. Derive the time complexity of the algorithm in O-notation.

```
Algorithm what (n)
  begin
       if n = 1 then call A
       else
       begin
            what (n-1);
            call B(n)
       end
end
```

Question

```
Algorithm what (n)
  begin
        if n = 1 then call A
        else
        begin
              what (n-1);
              call B(n)
        end
end
```

GATE 1999| Question Number 11| 5 Mark Question

Recursive program complexity

Consider the following algorithms. Assume, procedure A takes O(1) and procedure B take $O\left(\frac{1}{n}\right)$ unit of time. Derive the time complexity of the algorithm in O-notation.

```
Algorithm what (n)
  begin
      if n = 1 then call A   ← constant time
      else
      begin
          what (n-1);
          call B(n)   ← 1/n time
      end
end
```

Linear time Algorithm

$O(n)$

Recurrence Relation

$T(n) =$

$n = 1 \quad T(1) = 1 + 1 = 2$

$n > 1 \quad = 1 + T(n-1) + \frac{1}{n}$

$$T(n) = \begin{cases} 2 & n = 1 \\ T(n-1) + 1 + \frac{1}{n} & \end{cases}$$

comparison

B(n)

yesterday

Ans

GATE 2020 Substitution Method

For parameter a and b, both of which are $\omega(1)$, $T(n) = T(n^{1/a}) + 1$, and $T(b) = 1$ Constant value

Then T(n) is

(a) $\Theta(\log_{ab} n)$ $\boxed{[B]}$

(b) $\Theta(\log_a \log_b n)$ ✓

(c) $\Theta(\log_b \log_a n)$

(d) $\Theta(\log_2 \log_2 n)$

$$T(n) = \begin{cases} T(n^{1/a}) + 1 \\ T(b) = 1 \end{cases}$$

$T(n) = T(n^{1/a}) + 1$ — (I)

$T(n^{1/a}) = T(n^{1/a^2}) + 1$ — (II)

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

$T(n) = T(n^{1/a^k}) + k$

$1 + \log_a \log_b n$

Reduce to base condition

$n^{1/a^k} = b$

$n = b^{a^k}$

log_b both side

$a^k = \log_b n$

log_a both s

$\log_a a^k = \cancel{\log_a b}$

$\log_a \log_b n$

$k = \log_a \log_b n$