

1) If  $t_1(n) \in o(g_1(n))$  and  $t_2(n) \in o(g_2(n))$  then  $t_1(n) + t_2(n) \in o(\max\{g_1(n), g_2(n)\})$ . Prove the assertions.

Sol: We need to show that  $t_1(n) + t_2(n) \in o(\max\{g_1(n), g_2(n)\})$ . This means there exists a positive constant  $c$  and  $n_0$  such that  $t_1(n) + t_2(n) \leq c \max\{g_1(n), g_2(n)\}$

$$t_1(n) \leq c_1 g_1(n) \text{ for all } n \geq n_1$$

$$t_2(n) \leq c_2 g_2(n) \text{ for all } n \geq n_2$$

Let  $n_0 = \max\{n_1, n_2\}$  for all  $n \geq n_0$

consider  $t_1(n) + t_2(n)$  for all  $n \geq n_0$

$$t_1(n) + t_2(n) \leq c_1 g_1(n) + c_2 g_2(n).$$

We need to relate  $g_1(n)$  and  $g_2(n)$  to  $\max\{g_1(n), g_2(n)\}$ :

$$g_1(n) \leq \max\{g_1(n), g_2(n)\} \text{ and } g_2(n) \leq \max\{g_1(n), g_2(n)\}$$

thus,

$$c_1 g_1(n) \leq c_1 \max\{g_1(n), g_2(n)\}$$

$$c_2 g_2(n) \leq c_2 \max\{g_1(n), g_2(n)\}$$

$$c_1 g_1(n) + c_2 g_2(n) \leq c_1 \max\{g_1(n), g_2(n)\} + c_2 \max\{g_1(n), g_2(n)\}$$

$$c_1 g_1(n) + c_2 g_2(n) \leq (c_1 + c_2) \max\{g_1(n), g_2(n)\}.$$

$$t_1(n) + t_2(n) \leq (c_1 + c_2) \max\{g_1(n), g_2(n)\} \text{ for all } n \geq n_0$$

By the definition of Big O notation.

$$t_1(n) + t_2(n) \in o(\max\{g_1(n), g_2(n)\})$$

$$t_1(n) + t_2(n) \in o(\max\{g_1(n), g_2(n)\})$$

thus, the assertion is proved.

2)

Find the time complexity of the recurrence equation:

Let us consider such that recurrence for merge sort

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

By using master theorem

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

where  $a \geq 1$ ,  $b \geq 1$  and  $f(n)$  is positive function.

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

$$a = 2, b = 2, f(n) = n$$

By comparing of  $f(n)$  with  $\log_b^a$

$$\log_b^a = \log_2^2 = 1$$

compare  $f(n)$  with  $n \log_b^a$ :

$$f(n) = n$$

$$n \log_b^a = n^1 = n$$

$f(n) = O(n \log_b^a)$ ; then  $T(n) = O(n \log_b^a \log n)$ .

In our case:

$$\log_b^a = 1$$

$$T(n) = O(n^1 \log n) = O(n \log n)$$

Then time complexity of recurrence relation is  $T(n) = 2T(n/2) + n$   
is  $O(n \log n)$ .

$$3) T(n) = \begin{cases} 2T(n/2) + 1 & \text{if } n > 1 \\ , & \text{otherwise.} \end{cases}$$

Sol<sub>2</sub> By applying of master theorem.

$$T(n) = aT(n/b) + f(n) \text{ where } a \geq 1, b \geq 1$$

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

$$\text{Here } a=2, b=2, f(n)=1$$

By comparison of  $f(n)$  and  $n \log_b^a$ .

If  $f(n) = O(n^c)$  where  $c < \log_b^a$ , then  $T(n) = O(n \log_b^a)$

If  $f(n) = O(n \log_b^a)$ , then  $T(n) = O(n \log_b^a \log n)$

If  $f(n) = \Omega(n^d)$  where  $d > \log_b^a$  then  $T(n) = O(f(n))$

Let's calculate  $\log_b^a$ :

$$\log_b^a = \log_2^2 = 1$$

$$f(n) = 1$$

$$n \log_b^a = n^1 = n$$

$f(n) = O(n^c)$  with  $c < \log_b^a$  (case 1)

In this case  $c=0$  and  $\log_b^a = 1$

$$c < 1, \text{ so } T(n) = O(n \log_b^a) = O(n^1) = O(n)$$

Time complexity of Recurrence Relation.

$$T(n) = 2T(n/2) + 1 \text{ is } O(n)$$

$$4) T(n) = \begin{cases} 2T(n-1) & \text{if } n > 0 \\ 1 & \text{otherwise} \end{cases}$$

Sol. Here, where  $n=0$

$$T(0)=1$$

### Recurrence Relation Analysis.

for  $n > 0$ :

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

$$T(n)=2^2T(n-2)$$

$$T(n)=2^3T(n-3)$$

$$T(n)=2^nT(0)$$

from this pattern

$$T(n)=2 \cdot 2 \cdot 2 \cdots 2 \cdot T(0)=2^nT(0)$$

since  $T(0)=1$ , we have

the Recurrence Relation is

$$T(n)=2T(n-1) \text{ for } n > 0 \text{ and } T(0)=1 \text{ is } T(n)=2^n$$

$$T(n)=2^n \text{ for all } n \geq 0$$

5) Big O Notation show that  $f(n)=n^2+3n+5$  is  $O(n^2)$

$f(n)=O(g(n))$  means  $c > 0$  and  $n_0 \geq 0$

$f(n) \leq c g(n)$  for all  $n \geq n_0$

$$\text{given is } f(n)=n^2+3n+5$$

$c > 0$ ,  $n_0 \geq 0$  such that  $f(n) \leq c n^2$

$$f(n)=n^2+3n+5$$

lets choose  $c=2$

$$f(n) \leq 2 \cdot n^2$$

$$f(n)=n^2+3n+5 \leq n^2+3n^2+5n^2=9n^2$$

$$\text{so, } c=9, n_0=1 \quad f(n) \leq 9n^2 \text{ for all } n \geq 1$$

$$f(n)=n^2+3n+5 \text{ is } O(n^2).$$