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.

A. we need to show that $t_1(n) \in O \max\{g_1(n), g_2(n)\}$. This means there exist a positive constant c and n_0 such that

$t_1(n)+t_2(n) \leq C$

$t_1(n) \leq C_1 g_1(n)$ For all $n \geq n_1$

$t_2(n) \leq C_2 g(n)$ 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 $\max\{g_1(n), g_2(n)\}$

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

Thus,
$C_1 g_1(n) \leq C_1 \max\{g_1(n), g_2(n)\}$

$C_2 g_2(n) \leq C_2 \max\{g_1(n), g_2(n)\}$

$C_1 g_1(n) + C_2 g_2(n) \leq C_1 \max\{g_1(n), g_2(n)\} + C_2 \max\{g_1(n), g_2(n)\}$

$C_1 g_1(n) + C_2 g_2(n) \leq (C_1 + C_2) \max\{g_1(n), g_2(n)\}$

$t_1(n) + t_2(n) \leq (C_1 + C_2) \max\{g_1(n), g_2(n)\}$ For all $n \geq n_0$

By the defination of Big o notation

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

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

thus the assertion is proved

2. Find the time complexity of the Recurrence equation

A. let us Consider such that Recurrence for merge sort

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

By using master therom

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

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 $n^{\log_b a}$

$\log_b a = \log_2 2 = 1$

Compare $F(n)$ with $n^{\log_b a}$:

$F(n) = n$

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

If $F(n) = O(n^{\log_b a})$, then $T(n) = O(n^{\log_b a} \log n)$

In our case:

$\log_b a = 1$

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

The time complexity of recurrence Relation is $T(n) = 2T(n/2) + n$

is $O(n \log n)$

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

A. By Apply of master theorem

$T(n) = aT(n/b) + F(n)$ where $a \geq 1$, $b > 1$

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

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 $T(n) = O(n^{\log_b a} \cdot \log n)$

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

Lets 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 $\log_b a = 1$

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

Time Complexity of Recurrence relation

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

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

A. Here where $n = 0$

$T(0) = 1$

Recurrence relation Analysis

For $n > 0$

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

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

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

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

$T(1) = 2T(0)$

From this pattern

$T(n) = 2 \cdot 2 \cdot 2 \ldots 2\, T(0) = 2^n T(0)$

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

$T(n) = 2^n$

The recurrence Relation is $T(n) = 2T(n-1)$ For $n > 0$ and

$T(0) = 1$ is $T(n) = 2^n$

5. Big O notation show that $F(n) = n^2 + 3n + 5$ is $O(n^2)$

A. $F(n) = O(g(n))$ means $C > 0$ and $n_0 \geq 0$

$F(n) \leq C \cdot g(n)$ for all $n \geq n_0$

Given is $F(n) = n^2 + 3n + 5$

$C > 0,\ n_0 \geq 0$ such that $F(n) \leq C \cdot n^2$

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

let choose $C = 2$

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

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

so, $C = 9,\ n_0 = 1$ $F(n) \leq 9n^2$ for all $n \geq 1$

$F(n)$ $n^2 + 3n + 5$ is $O(n^2)$