Austin Stejoh
192311455

1) If $f_1(n) \in O(g_1(n))$ and $f_2(n) \in O(g_2(n))$, then $f_1(n) + f_2(n) \in O(\max\{g_1(n), g_2(n)\})$. Prove that assertions.

We need to show that $F_1(n) + F_2(n) \in O(\max\{g_1(n); g_2(n)\}$. This means there exists a positive constant c and no such that $F_1(n) + F_2(n) \le c$.

$F_1(n) \le c_1 g_1(n)$ for all $n \ge n_1$

$F_2(n) \le c_2 g_2(n)$ for all $n \ge n_0$

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

Consider $F_1(n) + F_2(n)$ for all $n \ge n_0$

$F_1(n) + F_2(n) \le c_1 g_1(n) + c_2 g_2(n)$

we need to relate $g_1(n)$ and $g_2(n)$ to $\max\{g_1(n), g_2(n)\}$.

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

$g_2(n) \le \max\{g_1(n), g_2(n)\}$.

Thus:

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

$c_2 g_2(n) \le c_2 \max\{g_2(n), g_2(n)\} +$

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

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

$f_1(n) + f_2(n) \le (c_1 + c_2) \max\{g_1(n) g_2(n)\}$.

for all $n \ge n_0$

By the definition of Big-O Notation

$f_1(n) + f_2(n) \in O(\max\{\max\{g_1(n), g_2(n)\}$,

$C = c_1 + c_2$

$t_1(n) \in O(g_1(n)$ and $t_2(n) \in O(g_2(n))$, then

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

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 $a \geq 1$, $b > 1$ and $F(n)$ is positive functions.

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

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

By comparing of $F(n)$ with $n^{\log_b a}$.

$F(n) = n$.

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 $T(n) = O(n^{\log_b a}) = O(n^1) = O(n)$

Time complexity of recurrence relation.

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

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

Here, where $n = 0$

$T(0) = 1$

Here, where $n = 0$

$T(0) = 1$.

Recurrence Relation Analysis.

For $n > 0$:

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

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

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

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

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

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

$F(n) = O(n^{\log_b a})$, 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 $O(n \log n)$.

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

By Applying of master theorem.

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

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

Here $a = 2$, $b = 2$, $F(n) = 1$

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

From this pattern.

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

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 $O(n^2)$.

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

$F(n) \leq g(n)$ 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 $c = 2$.

$F(n) \leq 2, n^2$.

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

$= 9n^2$.

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

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