CS5110: Complexity Theory

Shreyas Havaldar
CS18BTECH11042

November 10, 2020

1 Exercise Set 13, Lecture 18

1.1 Solving Recurrence for Size

$$f(n) = 2f\left(\sqrt{n}\right) + 2\sqrt{n} + 1$$

$$\text{Replacing n by } 2^k \text{ throughout}$$

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

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

$$\text{To simplify our analysis. now let: } T\left(k\right) = f\left(2^k\right)$$

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

Now looking at the Master Theorem stated below.

The Master Theorem

$$T(n) = \begin{cases} c & \text{if } n < d, \\ aT\left(\frac{n}{b}\right) + f(n) & \text{if } n \geq d, \end{cases}$$

where $a \geq 1, b > 1$, and d are integers and c is a positive constant.

- Case (i) $f(n) = O(n^{\log_b a - \epsilon})$ for some $\epsilon > 0$: $T(n) = \Theta(n^{\log_b a})$
- Case (ii) $f(n) = \Theta(n^{\log_b a})$: $T(n) = \Theta\left(\left(n^{\log_b a}\right) \cdot \log n\right)$
- Case (iii) $f(n) = \Omega(n^{\log_b a + \epsilon})$ for some $\epsilon > 0$ and $af\left(\frac{n}{b}\right) \leq cf(n)$ for some $c < 1$ for large enough n: $T(n) = \Theta\left(f(n)\right)$

So in our recurrence relation for T(n), our $a = 2, b = 2$, thus $n^{\log_b a} = n$, and f(n) is clearly $O(n^{\log_b a + \epsilon})$ for some ϵ, as our $f(n) = 2^{\frac{n}{2}+1} + 1$, which is exponential and thus Case (iii) of The Master Theorem can be applied and we get, on re-substituting k as $\log n$, throughout the expression.

$$T(n) = \Theta\left(f(n)\right)$$

$$\implies \boxed{T(n) = \Theta\left(\sqrt{n}\right)}$$

Note for number of wires: We would solve the recursion relation for number of wires in the same manner. Only difference would be in $f(n) = 2^{n+1} + 2$ as shown below. And we again would apply Master Theorem and as Case 3 condition is satisfied again as a and b values are same just the f(n) differs, $T(n) = \Theta(n)$ on re-substituting k as $\log n$ in f(k). The bound remains the same on solving recursively so no advantage over the result we found in class by doing brute-force.

$$f(n) = 2f\left(\sqrt{n}\right) + 2n + 2$$
$$\text{Replacing n by } 2^k \text{ throughout}$$
$$f\left(2^k\right) = 2f\left(2^{\frac{k}{2}}\right) + 2 \cdot 2^k + 2$$
$$\implies f\left(2^k\right) = 2f\left(2^{\frac{k}{2}}\right) + 2^{k+1} + 2$$
$$\text{To simplify our analysis. now let: } T(k) = f\left(2^k\right)$$
$$\implies T(k) = 2T\left(\frac{k}{2}\right) + 2^{k+1} + 2$$

So in our recurrence relation for T(n), our $a = 2, b = 2$, thus $n^{\log_b a} = n$, and f(n) is clearly $O(n^{\log_b a + \epsilon})$ for some ϵ, as our $f(n) = 2^{n+1} + 2$, which is exponential and thus Case (iii) of The Master Theorem can be applied and we get, on re-substituting k as $\log n$, throughout the expression.

$$T(n) = \Theta(f(n))$$
$$\implies \boxed{T(n) = \Theta(n)}$$

1.2

Please look at the circuit on the next page.

Size: (2 OR gates for each $\log n$ divisions into two buckets, for $\log n$ bit positions, one each for each bucket) + ($\log n$ AND gates corresponding to each F, to combine the outputs of the 2 OR gates corresponding to each bucket a level below) + (One topmost OR gate to combine the results of all $\log n$ AND gates corresponding to each F)

$$\begin{aligned} Size &= 2\log n + \log n + 1 \\ &= 3\log n + 1 \\ &= \boxed{O(\log n)} \end{aligned}$$

Number of Wires: (2 * (n/2) wires, one each for each variable as input to the OR gate corresponding to each bucket, both of size (n/2) as with equal number of numbers with 1 at a position k and 0 at the position k)*(Total lowermost OR gates which are $\log n$) + (2 inputs per $\log n$ AND gates corresponding

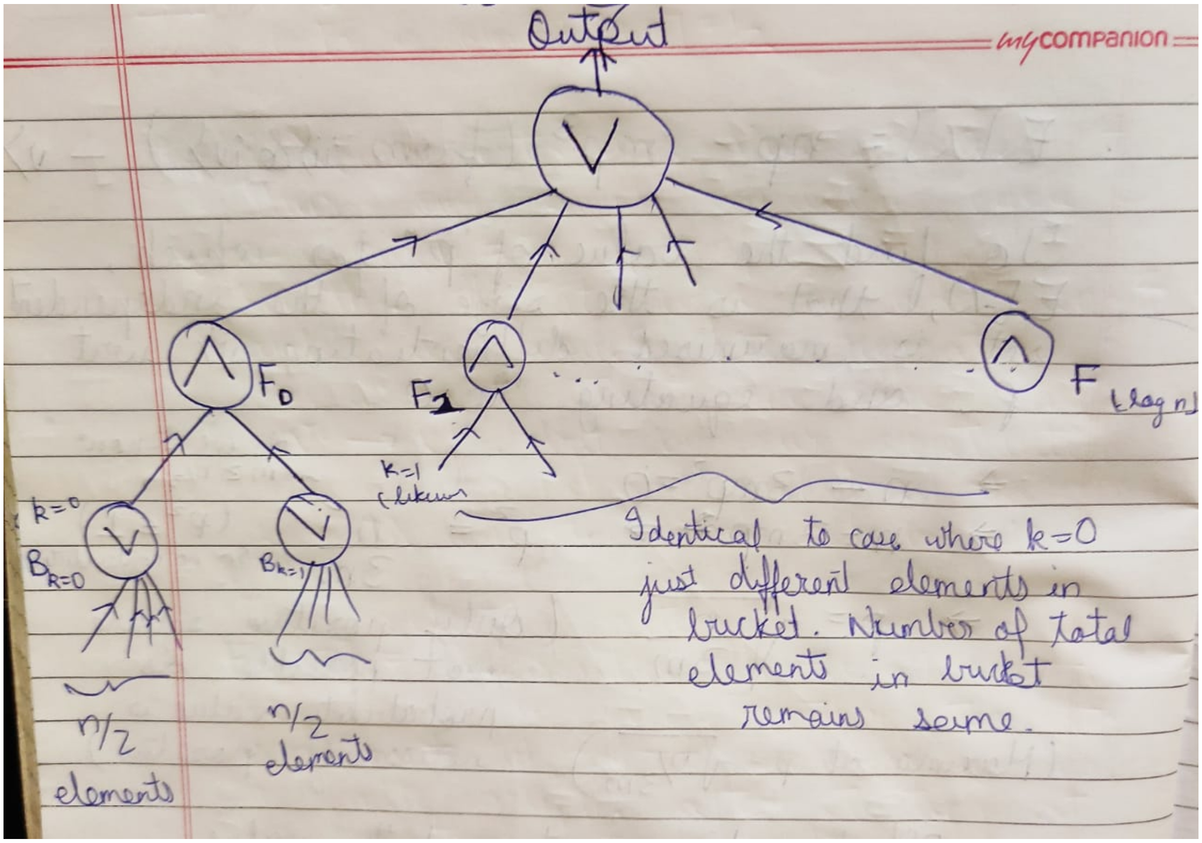

Figure 1: The circuit. Note: For $n = 2^k$ we would have exact n/2 elements in each bucket. Otherwise it would be at-most n/2, some buckets might have fewer, but as we are concerned about the asymptotic complexity finally, I'll proceed with the equality.

to each F, to take the outputs of the 2 OR gates corresponding to each bucket a level below as input to the AND) + ($\log n$ wires to the one topmost OR gate to combine the results of all $\log n$ AND gates corresponding to each F)

$$
\begin{aligned}
Wires &= 2 \times \frac{n}{2} \times \log n + 2 \log n + \log n \\
&= n \log n + 3 \log n \\
&= \boxed{O(n \cdot \log n)}
\end{aligned}
$$

Fan-In: **Unbounded**. As increase in the number of inputs for lowermost OR gates. As for increase in n, the number of elements in each bucket increases thus the number of inputs increase. Thus there is a gate with non-constant number of inputs.