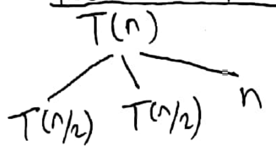
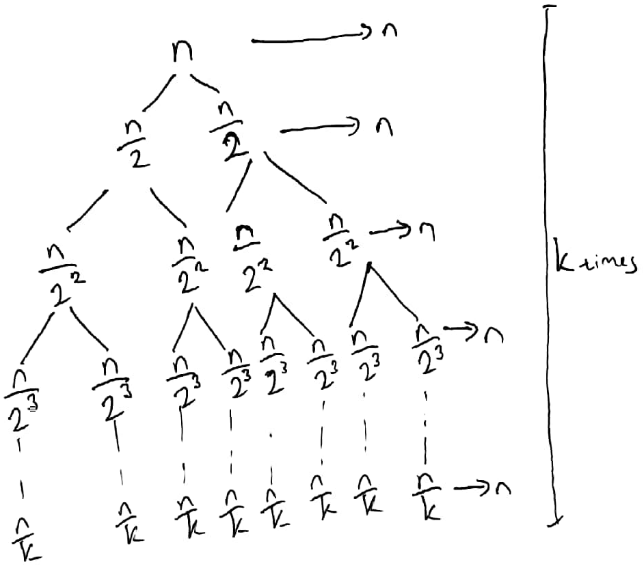
The worst-case running time $T(n)$ satisfies the recurrence

$$T(n) = \begin{cases} \Theta(1) & \text{if } n=1 \\ T(\lfloor n/2 \rfloor) + T(\lceil n/2 \rceil) + \Theta(n) & \text{if } n>1 \end{cases}$$

Recursion Tree method





nk times
$= n\log n$

Assume $\frac{n}{2^k} = 1$

$n = 2^k$

$k = \log n$

$\therefore$ Time Complexity
$= \Theta(n\log n)$

Substitution Method

$T(n) = 2T(n/2) + n$ ——— ①

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

$T(n) = 2^2 T(n/2^2) + n + n$ ——— ②

$= 2^2[2T(\frac{n}{2^3}) + \frac{n}{2^2}] + 2n$

$T(n) = 2^3 T(\frac{n}{2^3}) + 3n$ ——— ③

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

Assume

$T(\frac{n}{2^k}) = T(1)$

$\therefore \frac{n}{2^k} = 1 \quad n = 2^k$

$k = \log n$

$T(n) = 2^k T(1) + kn$

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

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

Scanned by CamScanner