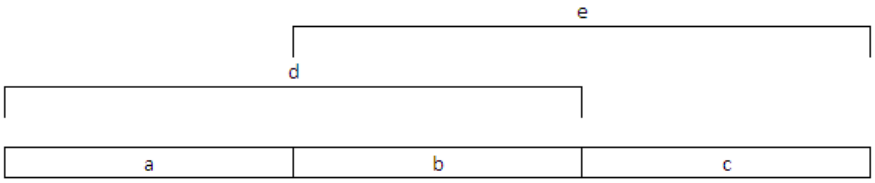
MO417 – Ata de Exercício

Redator: Luiz Augusto Muniz de Paula

Problema 7-3

a – Considere as divisões do array ilustradas abaixo. O algoritmo basicamente faz o seguinte:
- Na primeira chamada recursiva, ele coloca os menores de d em a e os maiores de d em b.
- Na segunda chamada recursiva, ele coloca os menores de e em b e os maiores de e em c.
- Como a segunda chamada recursiva pode alterar b, na terceira chamada recursiva ele refaz o que é feito na primeira.

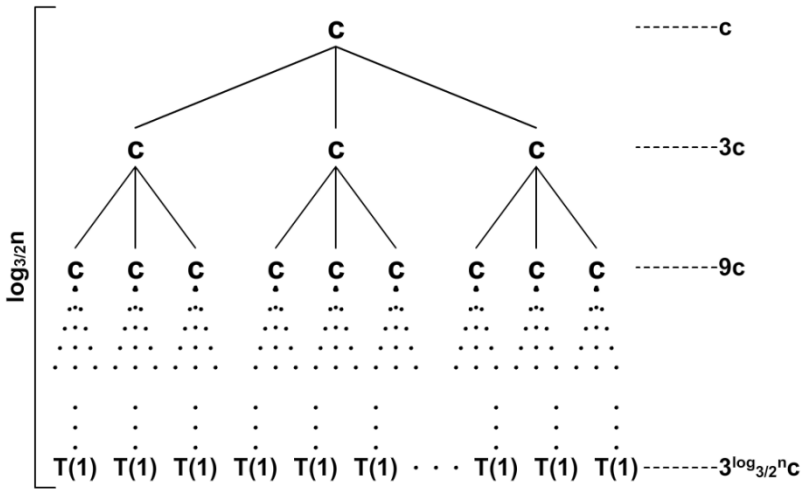


Ao final das 3 chamadas recursivas, em a estarão o menores elementos do array, em c estarão os maiores elementos do array, e em b estarão os elementos "intermediários".
É fácil notar que o algoritmo ordena arrays de tamanho 3 ou menores.
Devido à isso e ao fato de as chamadas recursivas se estenderem até arrays com tamanho menores ou iguais a 2, cada um desses grupos estará ordenados, conseqüentemente, todo o array estará.

b – A formula de recorrência para esse algoritmo é dada por:

$$T(n) = 3T\left(\frac{2n}{3}\right) + c$$

Logo, a árvore de recorrência gerada é:



Para um nível i, o tamanho do subproblema é $\left(\frac{2}{3}\right)^i n$.
No último nível, o tamanho do subproblema será 1. Logo:

$$\left(\frac{2}{3}\right)^i n = 1$$
$$\Rightarrow \left(\frac{2}{3}\right)^i = \frac{1}{n}$$
$$\Rightarrow \left(\frac{3}{2}\right)^i = n$$
$$\Rightarrow i = \log_{3/2} n$$

Portanto, a árvore tem $\log_{3/2} n + 1$ níveis.
Cada nodo no nível i tem custo c. Como o nível i tem 3^i nodos, ele tem um custo total de $c3^i$.
Assim, o custo total da árvore é:

$$\sum_{i=0}^{\log_{3/2} n - 1} c3^i + c3^{\log_{3/2} n} = c\left(\frac{3^{\log_{3/2} n} - 1}{3-1} + 3^{\log_{3/2} n}\right) = \frac{c}{2}\left(n^{\log_{3/2} 3} + 2n^{\log_{3/2} 3} - 1\right)$$

O que leva a supor um custo $O\left(n^{\log_{3/2} 3}\right)$.

Usando Teorema Mestre apresentado no livro, $a = 3$, $b = 3/2$ e $f(n) = c$.
Como $f(n) = O(n^{\log_b a - \epsilon})$ para qualquer $0 < \epsilon \leq \log_b a$, como por exemplo $\epsilon = 1$, o caso 1 do teorema garante que $T(n) = O(n^{\log_b a})$, ou seja, $T(n) = O\left(n^{\log_{3/2} 3}\right)$.

Obs.: $\log_{3/2} 3 \cong 2{,}7$.

c – O tempo de execução do algoritmo apresentado – $O(n^{2,7})$ – é sempre pior do que os dos algoritmos:
- Insertion sort – $O(n^2)$ no pior caso;
- merge sort – $O(nlgn)$ no pior caso;
- heapsort – $O(nlgn)$ no pior caso;
- quicksort – $O(n^2)$ no pior caso.