CI165 — QuickSort: Intuição do Caso Médio e Versão Aleatorizada

André Vignatti

Intuição do Caso médio

Apesar da complexidade de tempo do QUICKSORT no pior caso ser $\Theta(n^2)$, na prática ele é o algoritmo mais eficiente.

Mais precisamente, a complexidade de tempo do QUICKSORT no caso médio é mais próximo do melhor caso do que do pior caso.

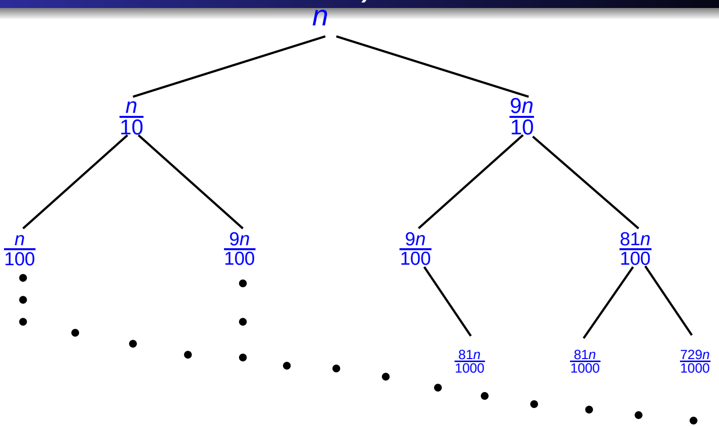
Por quê??

Suponha que (por sorte) o algoritmo PARTICIONE sempre divide o vetor na proporção $\frac{1}{9}$ para $\frac{9}{10}$. Então

$$T(n) = T(\left\lfloor \frac{n-1}{9} \right\rfloor) + T(\left\lceil \frac{9(n-1)}{10} \right\rceil) + \Theta(n)$$

Solução: $T(n)$ é $\Theta(n \lg n)$.

Intuição do Caso Médio: Recorrência de Particionamento Constante)



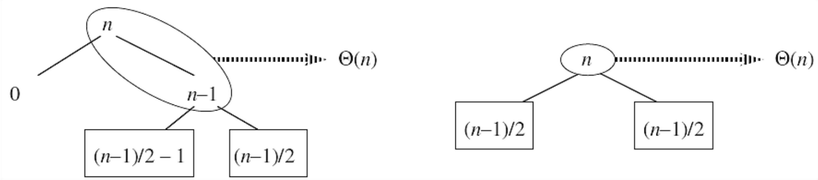
Número de níveis $\leq \log_{10/9} n$.

Em cada nível o custo é $\leq n$.

Custo total é $O(n \log n)$.

Intuição do Caso Médio: Particionamento Não-Constante

- A **divisão** da árvore de recursão não será sempre constante.
- Há boas e más divisões através da recursão.
- Para ver que isso não afeta o tempo de execução assintótico, assuma que os níveis se alternam entre o melhor-caso e o pior-caso.



- O nível extra na figura da esquerda somente adiciona à constante escondida na notação Θ.
- Somente será feito duas vezes o mesmo trabalho.
- Ambas figuras resultam em $O(n \log n)$, mas a figura da esquerda tem uma constante maior.

QuickSort Aleatorizado

Tudo é igualmente provável?

- A análise de **caso médio** assume que todas permutações da entrada são igualmente prováveis.
- Isso não é sempre verdade! [Exercício CLRS]

Para corrigir isso, usaremos **aleatorização** no QUICKSORT

- Poderíamos permutar aleatoriamente a entrada
  - é igual à análise **difícil** do caso médio
- Ao invés disso, usaremos amostragem aleatória, ou seja, pegar um elemento aleatoriamente
  - análise mais **fácil**

QuickSort Aleatorizado

Ideia

Escolher o pivô aleatoriamente

$\text{PARTICIONE-ALEATÓRIO}(A, p, r)$

1 $i \leftarrow \text{RANDOM}(p, r)$
2 $A[i] \leftrightarrow A[r]$
3 **devolva** $\text{PARTICIONE}(A, p, r)$

Ao selecionar aleatoriamente o pivô, na média, iremos dividir o vetor de maneira **bem balanceada**.

$\text{QUICKSORT-ALEATÓRIO}(A, p, r)$

1 **se** $p < r$
2 **então** $q \leftarrow \text{PARTICIONE-ALEATÓRIO}(A, p, r)$
3 $\text{QUICKSORT-ALEATÓRIO}(A, p, q-1)$
4 $\text{QUICKSORT-ALEATÓRIO}(A, q+1, r)$

QuickSort Aleatorizado

Observação

- Aleatorizar impede, na média (valor esperado), o comportamento de pior caso do algoritmo
- Por exemplo, um vetor **já ordenado** provoca o pior caso no QUICKSORT, mas não no QUICKSORT-ALEATÓRIO.

QuickSort Aleatorizado - Ideia da Análise

- O custo dominante do algoritmo é no PARTICIONE
- A quantidade de trabalho em cada chamada de PARTICIONE (veja o pseudo-código) é uma constante mais o número de **for** executados

Ideia para a análise

- Em cada **for**, existe uma comparação.
- Se contarmos o número de comparações, então contamos o número de **for**

Então, se contarmos todas as comparações feitas pelo algoritmo, conseguiremos analisar o algoritmo.

- Seja X = o número total de **comparações** em todas chamadas de PARTICIONE
- Portanto, o trabalho total é $O(X)$.

Observação Importante:

Após o PARTICIONE, o pivô não é levado mais em consideração até o fim do algoritmo

Vamos agora contar o número total de comparações feitas pelo algoritmo.

Tempo Esperado do QuickSort

Para facilitar, supomos que os elementos são distintos

Seja $z_1 < z_2 < \ldots < z_n$ os elementos ordenados

Seja X_{ij} v.a. que indica se z_i foi comparado com z_j:

$$X_{ij} = \begin{cases} 1 & \text{se } z_i \text{ é comparado com } z_j \\ 0 & \text{caso contrário.} \end{cases}$$

Observação

Um par de elementos é comparado no **máximo** uma vez!

- Isso porque comparações são feitas somente com o pivô, mas o pivô nunca está em uma chamada futura a PARTICIONE
- Assim, $X_{ij} = X_{ji}$. Para evitar contar duas vezes, vamos somente contar X_{ij} se $i < j$.

Tempo Esperado do QuickSort

Como cada par é comparado no máximo uma vez, o número total de comparações X é

$$X = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij}.$$

e o número esperado de comparações é $E[X]$. Pela

linearidade da esperança:

$$E[X] = E\left[\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij}\right] = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} E[X_{ij}]$$

Tempo Esperado do QuickSort

Como X_{ij} é uma variável aleatória binária,

$$\begin{aligned} E[X_{ij}] &= 0 \cdot \Pr\{z_i \text{ não ser comparado com } z_j\} + \\ & \quad 1 \cdot \Pr\{z_i \text{ ser comparado com } z_j\} \\ &= \Pr\{z_i \text{ ser comparado com } z_j\} \end{aligned}$$

Então,

$$E[X] = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \Pr\{z_i \text{ ser comparado com } z_j\}$$

Agora basta encontrar a probabilidade de dois elementos serem comparados

Tempo Esperado do QuickSort

Considere a escolha do pivô e a comparação entre $\mathbf{z_i}$ e $\mathbf{z_j}$:

$$\underbrace{z_1, z_2, \ldots, z_{i-1}}_{\text{posterga}}, \mathbf{z_i}, \underbrace{z_{i+1}, \ldots, z_{j-1}}_{\text{não comp.}}, \mathbf{z_j}, \underbrace{z_{j+1}, \ldots, z_n}_{\text{posterga}}$$

- Se algum azul é escolhido como pivô, z_i e z_j **nunca mais** serão comparados (pois ficarão em partições diferentes)
- Para serem comparados, ou z_i ou z_j devem ser escolhidos como pivô **ANTES** de algum azul.

Tempo Esperado do QuickSort

Seja $Z_{ij} = \{z_i, z_{i+1}, \ldots, z_j\}$. Assim,

$$
\begin{aligned}
&\Pr\{z_i \text{ ser comparado com } z_j\} \\
&\quad = \Pr\{z_i \text{ ou } z_j \text{ é escolhido como pivô primeiro em } Z_{ij}\} \\
&\quad = \Pr\{z_i \text{ é escolhido como pivô primeiro em } Z_{ij}\} + \\
&\quad \Pr\{z_j \text{ é escolhido como pivô primeiro em } Z_{ij}\} \\
&\quad = \frac{1}{j-i+1} + \frac{1}{j-i+1} \\
&\quad = \frac{2}{j-i+1}
\end{aligned}
$$

(A segunda linha é verdade pois os dois eventos são mutuamente exclusivos)

Substituindo na equação para $E[X]$,

$$
\begin{aligned}
E[X] &= \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} E[X_{ij}] \\
&= \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{2}{j-i+1} \\
&= \sum_{i=1}^{n-1} \sum_{k=1}^{n-i} \frac{2}{k+1} \\
&< \sum_{i=1}^{n-1} \sum_{k=1}^{n} \frac{2}{k} \\
&= \text{(num. harmônico, eq. A7 do CLRS)} \sum_{i=1}^{n-1} O(\lg n) \\
&= O(n \lg n)
\end{aligned}
$$

Conclusão

O consumo de tempo esperado pelo QUICKSORT-ALEATÓRIO para itens distintos é $O(n \lg n)$.

O limitante de melhor caso visto anteriormente (dividir "exatamente" no meio) era $T(n) = \Omega(n \lg n)$.

Conclusão:

O consumo de tempo esperado pelo QUICKSORT-ALEATÓRIO para itens distintos é $\Theta(n \lg n)$.