QuickSort

Prof. Andrey Masiero

28 de setembro de 2017

Agenda

QuickSort

- Assim como o MergeSort, é um algoritmo que tem base no paradigma de divisão e conquista;

QuickSort

- Assim como o MergeSort, é um algoritmo que tem base no paradigma de divisão e conquista;
- Só que utiliza a técnica de maneira contrária. Trabalho pesado é feito antes das chamadas recursivas;

QuickSort

- Assim como o MergeSort, é um algoritmo que tem base no paradigma de divisão e conquista;
- Só que utiliza a técnica de maneira contrária. Trabalho pesado é feito antes das chamadas recursivas;
- Dado uma sequência S:

QuickSort

- Assim como o MergeSort, é um algoritmo que tem base no paradigma de divisão e conquista;
- Só que utiliza a técnica de maneira contrária. Trabalho pesado é feito antes das chamadas recursivas;
- Dado uma sequência S:
  - Aplica a técnica de divisão e conquista, fazendo subsequências de S;

QuickSort

- Assim como o MergeSort, é um algoritmo que tem base no paradigma de divisão e conquista;
- Só que utiliza a técnica de maneira contrária. Trabalho pesado é feito antes das chamadas recursivas;
- Dado uma sequência S:
  - Aplica a técnica de divisão e conquista, fazendo subsequências de S;
  - Aplica a recursão ordenando cada subsequência;

QuickSort

- Assim como o MergeSort, é um algoritmo que tem base no paradigma de divisão e conquista;
- Só que utiliza a técnica de maneira contrária. Trabalho pesado é feito antes das chamadas recursivas;
- Dado uma sequência S:
  - Aplica a técnica de divisão e conquista, fazendo subsequências de S;
  - Aplica a recursão ordenando cada subsequência;
  - Por fim, combina as subsqûencias ordenadas através de uma concatenação simples.

QuickSort

Os três passos do algoritmo:

- Dividir: se S tiver pelo menos dois elementos, escolhe-se um elemento x de S, chamado de *pivô*. Uma opção é escolher x como o último elemento de S. Os demais elementos de S são removidos, e colocados em três sequências:

QuickSort

Os três passos do algoritmo:

- Dividir: se S tiver pelo menos dois elementos, escolhe-se um elemento x de S, chamado de *pivô*. Uma opção é escolher x como o último elemento de S. Os demais elementos de S são removidos, e colocados em três sequências:
  - L: elementos de S menores que x;

QuickSort

Os três passos do algoritmo:

- Dividir: se S tiver pelo menos dois elementos, escolhe-se um elemento x de S, chamado de *pivô*. Uma opção é escolher x como o último elemento de S. Os demais elementos de S são removidos, e colocados em três sequências:
  - L: elementos de S menores que x;
  - E: elementos de S iguais a x;

QuickSort

Os três passos do algoritmo:

- Dividir: se S tiver pelo menos dois elementos, escolhe-se um elemento x de S, chamado de *pivô*. Uma opção é escolher x como o último elemento de S. Os demais elementos de S são removidos, e colocados em três sequências:
  - L: elementos de S menores que x;
  - E: elementos de S iguais a x;
  - G: elementos de S maiores que x.

QuickSort

Os três passos do algoritmo:

- Dividir: se S tiver pelo menos dois elementos, escolhe-se um elemento x de S, chamado de *pivô*. Uma opção é escolher x como o último elemento de S. Os demais elementos de S são removidos, e colocados em três sequências:
  - L: elementos de S menores que x;
  - E: elementos de S iguais a x;
  - G: elementos de S maiores que x.
- Recursão: Ordena as sequências de L e G, recursivamente;

QuickSort

Os três passos do algoritmo:

- Dividir: se S tiver pelo menos dois elementos, escolhe-se um elemento x de S, chamado de *pivô*. Uma opção é escolher x como o último elemento de S. Os demais elementos de S são removidos, e colocados em três sequências:
  - L: elementos de S menores que x;
  - E: elementos de S iguais a x;
  - G: elementos de S maiores que x.
- Recursão: Ordena as sequências de L e G, recursivamente;
- Conquista: Uni os elementos de S em ordem, a partir de L, passando por E e finalizando com G.

QuickSort

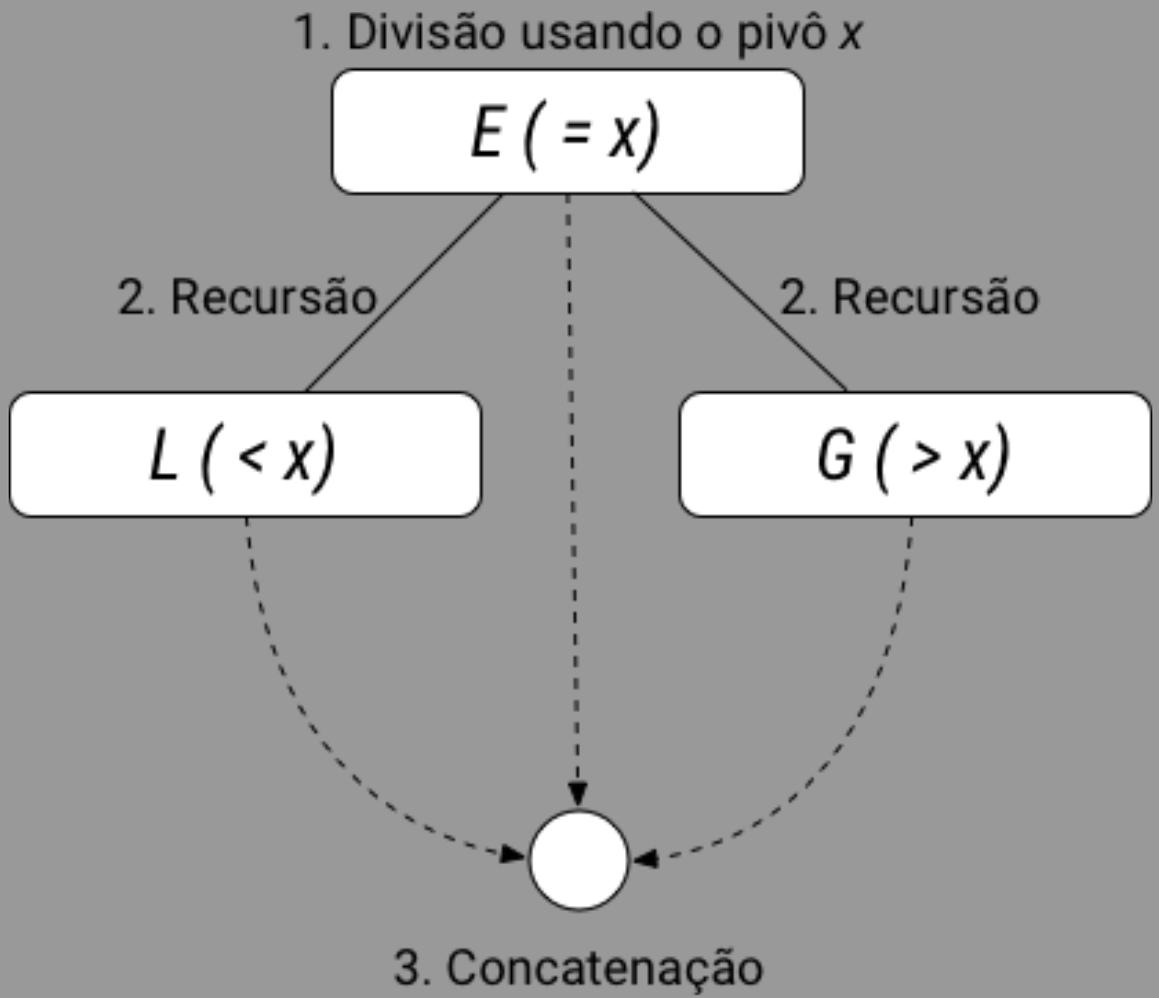

Figura: Goodrich e Tamassia, 2013

QuickSort

- Assim como o MergeSort, o QuickSort pode ser visualizado com uma árvore binária recursiva;

QuickSort

- Assim como o MergeSort, o QuickSort pode ser visualizado com uma árvore binária recursiva;
- A altura da árvore do QuickSort é no pior caso linear;

QuickSort

- Assim como o MergeSort, o QuickSort pode ser visualizado com uma árvore binária recursiva;
- A altura da árvore do QuickSort é no pior caso linear;
- O pior caso ocorre quando a sequência consiste em n elementos distintos e já ordenada.

QuickSort

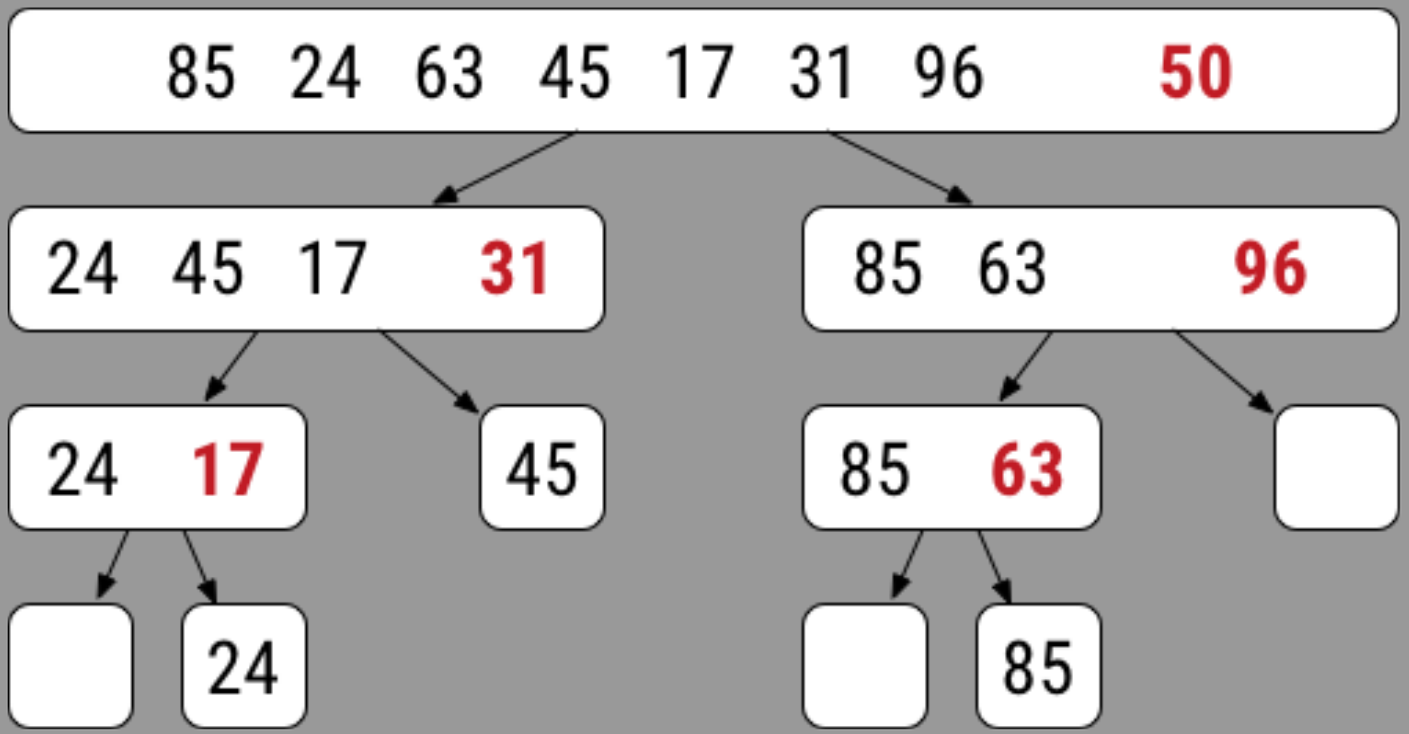

Figura: Goodrich e Tamassia, 2013

QuickSort

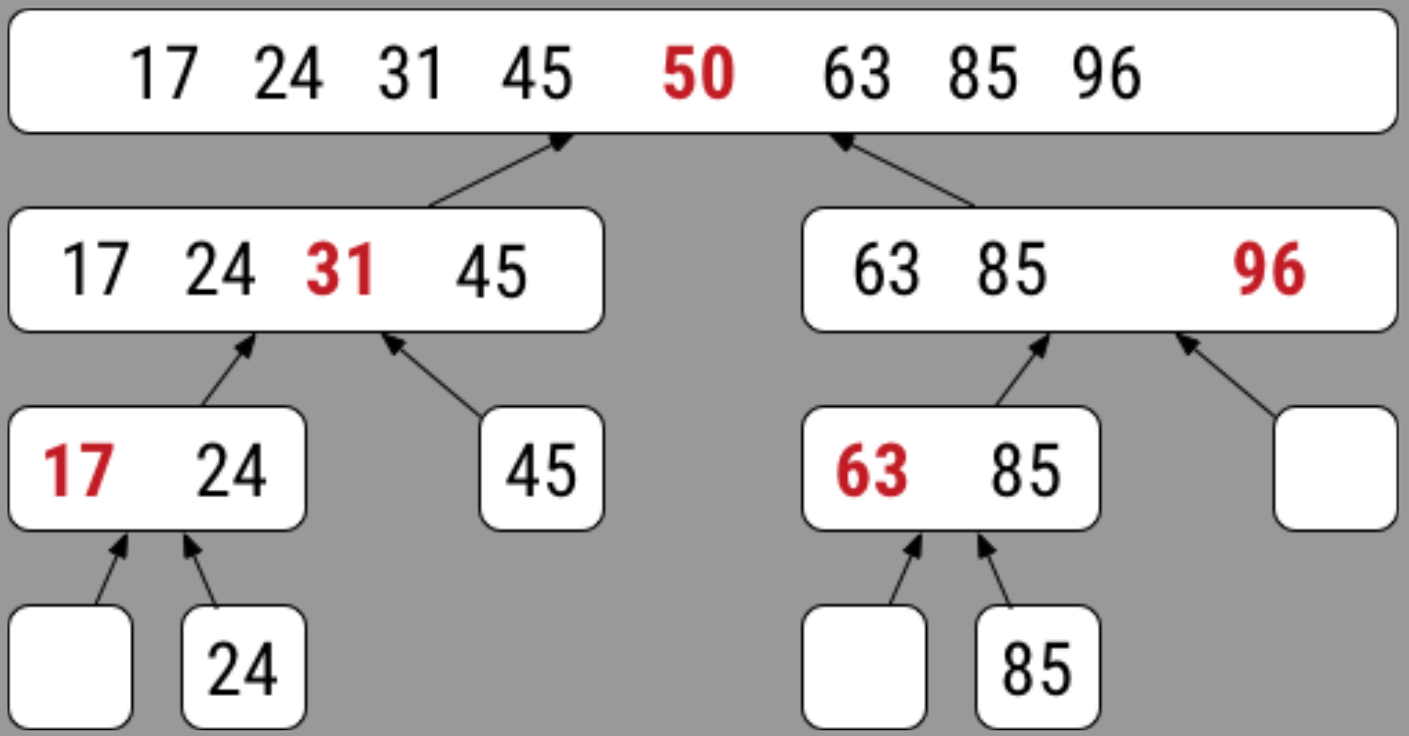

Figura: Goodrich e Tamassia, 2013

Algoritmo de Ordenação

```
public void sort(int X[], int inicio, int fim) {
    if (inicio < fim) {
        int pivot = this.divide(X, inicio, fim);
        this.sort(X, inicio, pivot - 1);
        this.sort(X, pivot + 1, fim);
    }
}
```

Método Divide

```
public int divide(int X[], int inicio, int fim) {
    int pivot = X[inicio];
    int postPivot = inicio;
    for (int i = inicio + 1; i <= fim; i++) {
        if(X[i] < pivot) {
            X[postPivot] = X[i];
            X[i] = X[postPivot + 1];
            postPivot++;
        }
    }
    X[postPivot] = pivot;
    return postPivot;
}
```

Exercícios

1. Implemente o QuickSort.
2. Teste os algoritmos em um programa principal, com o seguintes vetores:
    1. 42, 21, 37, 75, 98, 11, 50, 63
    2. 88, 15, 27, 55, 44, 38
    3. 12, 81, 75, 37, 47, 25, 34
    4. 48, 11, 88, 33, 57, 12, 18, 87, 54, 8

Referências Bibliográficas

1. Cormen, Thomas H., Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. "Introduction to algorithms second edition." (2001).
2. Goodrich, Michael T. and Tamassia, Roberto. "Estrutura de Dados e Algoritmos em Java." Porto Alegre, Ed. Bookman 5 (2013).
3. Ascencio, Ana Fernanda Gomes, and Graziela Santos de Araújo. "Estruturas de Dados: algoritmos, análise da complexidade e implementações em JAVA e C/C++." São Paulo: Perarson Prentice Halt 3 (2010).