MergeSort

Prof. Andrey Masiero

28 de setembro de 2017

Agenda

MergeSort

- Dividir: transformar os dados em pequenos subproblemas;

MergeSort

- Dividir: transformar os dados em pequenos subproblemas;
- Conquistar: Classificar os pequenos subproblemas resolvendo-os;

MergeSort

- Dividir: transformar os dados em pequenos subproblemas;
- Conquistar: Classificar os pequenos subproblemas resolvendo-os;
- Combinar: Juntar os subproblemas em um único conjunto classificado.

MergeSort

- O vetor inicial é dividido ao meio até que existam apenas pares de elementos;

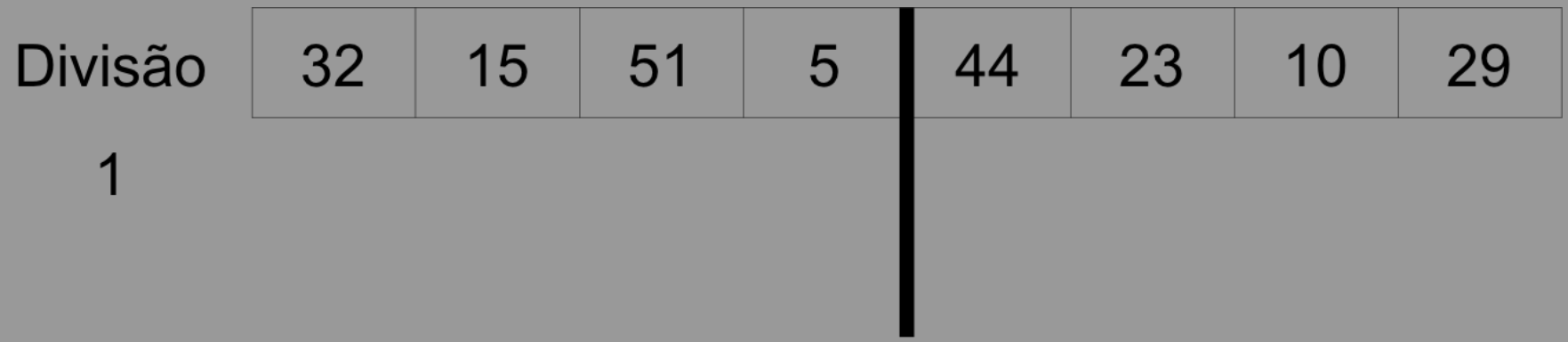
MergeSort

- O vetor inicial é dividido ao meio até que existam apenas pares de elementos;
- Cada par é ordenado e depois são agrupados com outro par, em um vetor ordenado de quatro elementos;

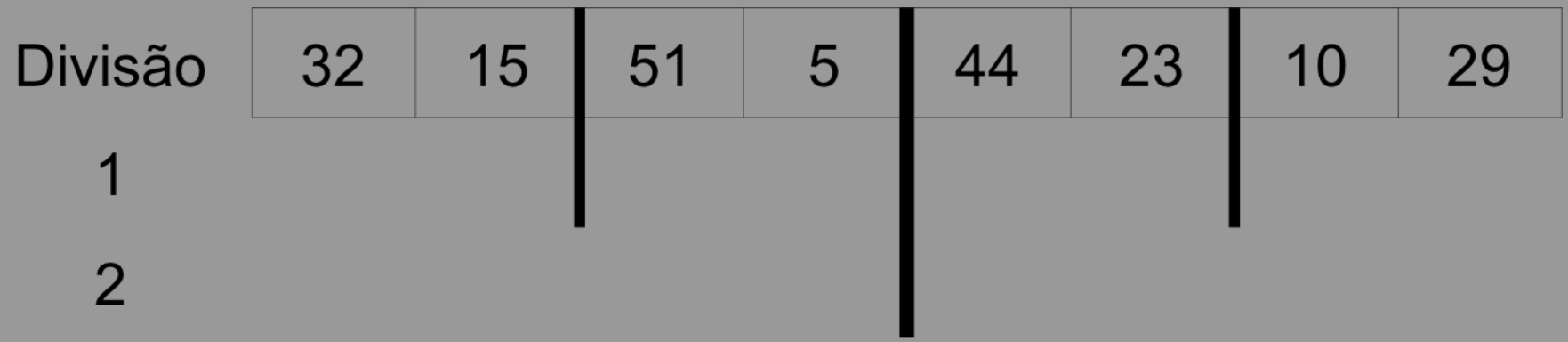
MergeSort

- O vetor inicial é dividido ao meio até que existam apenas pares de elementos;
- Cada par é ordenado e depois são agrupados com outro par, em um vetor ordenado de quatro elementos;
- E assim é realizado o processo .

MergeSort



MergeSort



MergeSort

Passo								
1	15	32	5	51	23	44	10	29

MergeSort

Passo								
1	15	32	5	51	23	44	10	29
2	5	15	32	51	10	23	29	44

MergeSort

Passo								
	32	15	51	5	44	23	10	29
1	15	32	5	51	23	44	10	29
2	5	15	32	51	10	23	29	44
3	5	10	15	23	29	32	44	51

Algoritmo de Ordenação

```
public void sort(int X[], int inicio, int fim) {
    if (inicio < fim) {
        int meio = (inicio + fim) / 2;
        this.sort(X, inicio, meio);
        this.sort(X, meio + 1, fim);
        this.merge(X, inicio, meio, fim);
    }
}
```

Método Merge

```
public void merge(int X[], int inicio, int meio,
    int fim) {

    int i, esquerda, direita;
    int aux[] = new int[X.length];

    for (i = inicio; i <= fim; i++) aux[i] = X[i];

    esquerda = inicio;
    direita = meio + 1;
    i = inicio;
```

Método Merge

```
    while (esquerda <= meio && direita <= fim) {
        if (aux[esquerda] <= aux[direita]) X[i++] =
            aux[esquerda++];
        else X[i++] = aux[direita++];
    }

    while (esquerda <= meio) X[i++] =
        aux[esquerda++];
}
```

Exercícios

1. Implemente o MergeSort.
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. Tamassia, Roberto, and Michael T. Goodrich. "Estrutura de Dados e Algoritmos em Java." Porto Alegre, Ed. Bookman 4 (2007).
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).