Técnicas de Projeto de Algoritmos - Divisão e Conquista

Kleber Jacques F. de Souza

Dividir e conquistar.

Júlio César

Divisão e Conquista

- A técnica de Divisão e Conquista consiste em **dividir** o problema em **partes menores**, encontrar soluções para essas partes e **combina-las** em uma **solução global**.
- Geralmente leva a **soluções eficientes e elegantes**, principalmente se forem recursivas.

Divisão e Conquista

- **Divisão**: Dividir o problema original em subproblemas menores.
- **Conquista**: Resolver cada subproblema recursivamente.
- **Combinação**: Combinar as soluções encontradas, compondo uma solução para o problema original.

Divisão e Conquista - Vantagens

- Resolução de **problemas difíceis**;
- Pode gerar algoritmos eficientes, com forte tendência a **complexidade logarítmica**;
- Requerem um número **menor de acessos à memória.**
- São altamente **paralelizáveis**.

Divisão e Conquista - Desvantagens

- **Recursão ou Pilha explícita**.
  - Número de chamadas recursivas e/ou armazenadas na pilha pode ser um inconveniente.
- Dificuldade na **seleção dos casos bases**
- **Repetição de sub-problemas**

Divisão e Conquista – Quando usar?

1. Deve ser possível decompor uma instância em sub-instâncias
2. A combinação dos resultados dever ser eficiente (trivial se possível)
3. As sub-instâncias devem ser mais ou menos do mesmo tamanho

Divisão e Conquista - Algoritmo Geral

```
Divisao_E_Conquista(x):
    if (x é pequeno)  return Resolve(x);
    else:
        decompor x em n conjuntos menores: x_1, x_2,..., x_n
            foreach(i in [1..N])
                y_i = Divisao_E_Conquista(x_i);
        combinar y_0, y_1, y_2 . . . y_1 em Y
        return y
```

Divisão e Conquista – Ordenação

- Problema
  - Ordenar um vetor de N inteiros
- Solução por Força Bruta
  - Realiza <u>todas as comparações e trocas possíveis</u> - Ordenação pelo método Bolha **$O(n^2)$**

Divisão e Conquista – MergeSort O(n log n)

Divisão e Conquista – Exponenciação

- Problema
  - Calcular a^n, para todo real a e inteiro $n \geq 0$

- ❑ $n \geq 0$; $a^0 = 1$; $a^1 = a$.
- ❑ $a^3 = a*a*a$
- ❑ $a^3 = a*a^2$

Divisão e Conquista - Exponenciação

```
double expo(double x, int n)
{
    if (n == 0) return 1;
    if (n == 1) return x;

    return x * expo(x, n-1);
}
```

Divisão e Conquista - Exponenciação

expo(2,4) = ? → **16**

expo(2,4) = 2 * expo(2, 3) → **16**

8

expo(2,3) = 2 * expo(2, 2) → **8**

4

expo(2,2) = 2 * expo(2, 1) → **4**

2

expo(2,1) = 2

Divisão e Conquista - Exponenciação

```
double expo2(double x, int n){
    if (n == 0) return 1;
    if (n == 1) return x;
    double p = expo2(x, n / 2);
    if (n % 2 == 0) return p * p;
    else return x * p * p;
}
```

Divisão e Conquista - Exponenciação

expo(2,4) = ? ➔ **16**

expo(2,4) ➔ p = expo(2, 2); ➔ **p*p = 16**

4

expo(2,2) ➔ p = expo(2, 1); ➔ **p*p = 4**

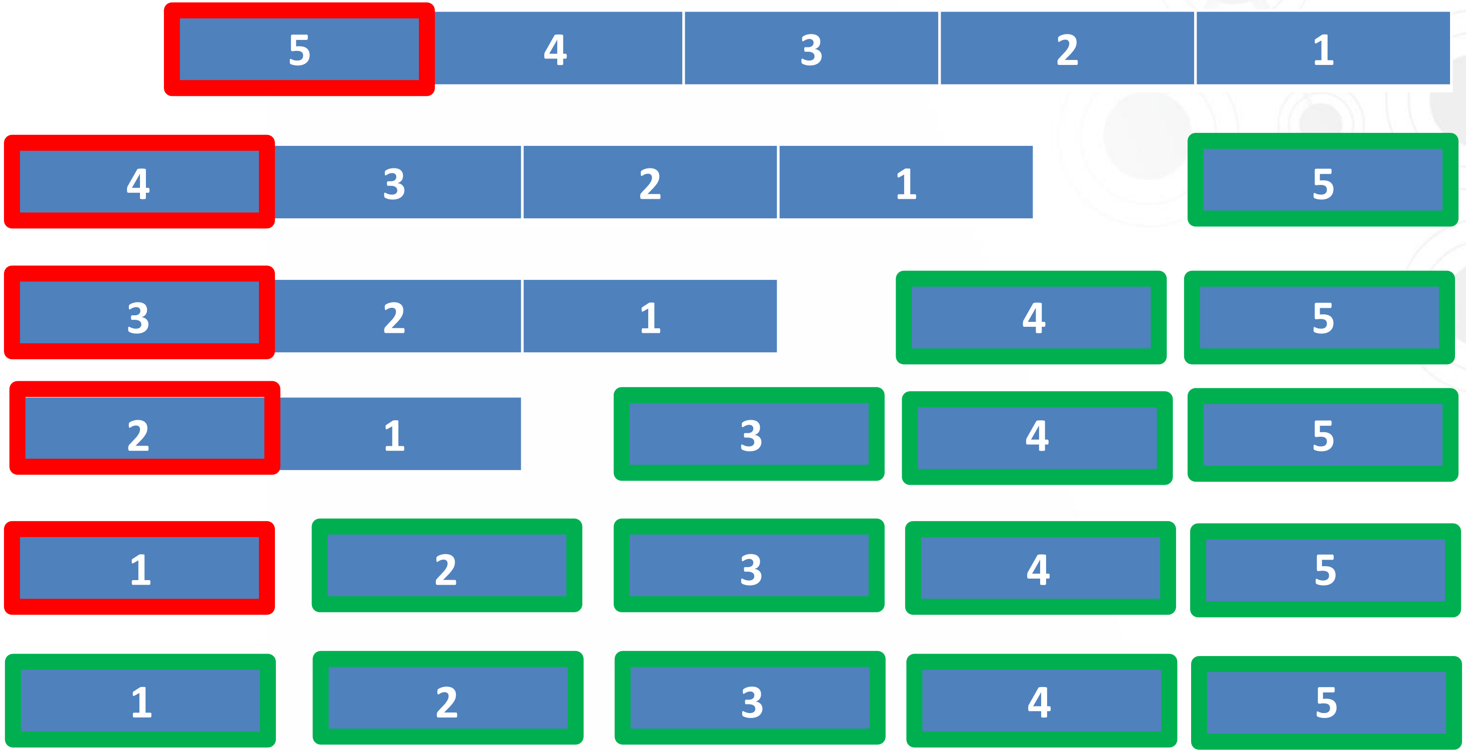
2

expo(2,1) = 2

Divisão e Conquista

- Na técnica de Divisão e Conquista, é importante procurar sempre **manter o balanceamento na sub-divisão** de um problema em partes menores.
- Exemplo: Pior caso do QuickSort.

Divisão e Conquista – QuickSort $O(n^2)$



Referências Bibliográficas

Ziviani, Nivio. **Projeto de Algoritmos:** com implementações em JAVA e C++. CENGACE Learning, 2012. (Livro Eletrônico)

CORMEN, Thomas H. et al. **Algoritmos**: teoria e prática. Elsevier, RJ, 2012.