Introdução à Ciência da Computação II

Merge-Sort Pt. II:
Implementação Seqüencial

Prof. Ricardo J. G. B. Campello

1

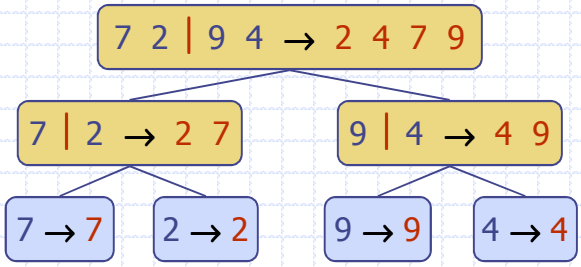
Aula de Hoje

- Merge-Sort Seqüencial
- Análise de Desempenho

2

Árvore Merge-Sort (Revisão)

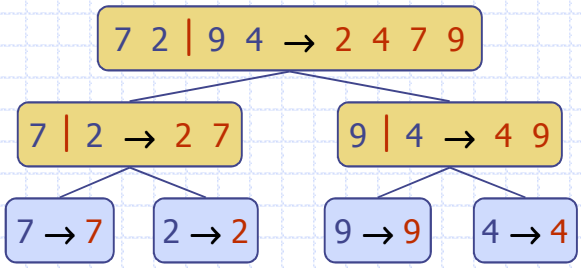
- Sabe-se que uma execução da versão recursiva de Merge-Sort pode ser representada por uma árvore de recursão binária



3

Árvore Merge-Sort (Revisão)

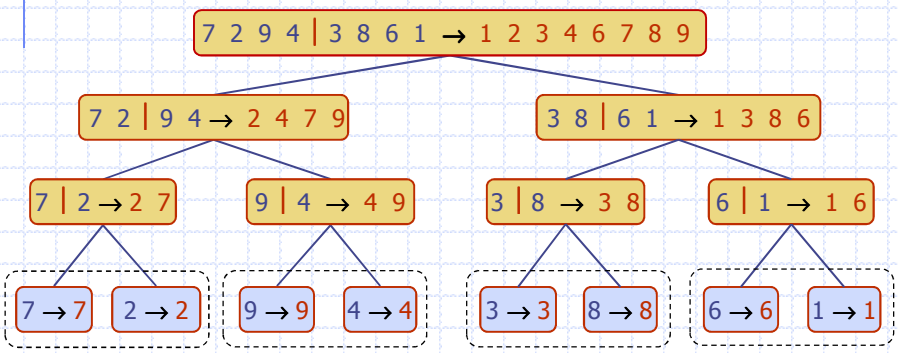
- Seria possível realizar as mesmas operações de forma não recursiva, das folhas para a raiz (**bottom-up**) ?



4

Árvore Merge-Sort (Revisão)

- A resposta é sim, observando que cada par de folhas é um par de valores consecutivos na seqüência original...



Merge-Sort Seqüencial

- Pode-se então iniciar a execução bottom-up intercalando os valores da seqüência original S em duas seqüências auxiliares:

  S_1: 7 9 3 6

  S_2: 2 4 8 1

  - Essas seqüências são vistas como "ordenadas" em **rodadas** de tamanho unitário, que correspondem aos nós folha da árvore
    - Rodadas de tamanho r são subconjuntos de r elementos consecutivos que estão internamente ordenados
  - Note que se fizermos a fusão ordenada das rodadas de S_1 com as respectivas rodadas de S_2, obteremos o penúltimo nível da árvore !

6

Merge-Sort Seqüencial

- Os resultados das fusões ordenadas das rodadas de S_1 com as respectivas rodadas de S_2 são intercalados em duas seqüências auxiliares adicionais:

  S_3: 2 7 | 3 8

  S_4: 4 9 | 1 6

  - Essas seqüências estão "ordenadas" em rodadas de tamanho 2
  - Cada rodada corresponde a um nó do penúltimo nível da árvore
  - Note que se fizermos a fusão ordenada das rodadas de S_3 com as respectivas rodadas de S_4, obteremos o nível acima na árvore

7

Merge-Sort Seqüencial

- Os resultados das fusões ordenadas das rodadas de S_3 com as respectivas rodadas de S_4 são intercalados agora nas duas seqüências auxiliares originais:

  S_1: 2 4 7 9

  S_2: 1 3 6 8

  - Essas seqüências estão "ordenadas" em rodadas de tamanho 4
  - Cada rodada corresponde a um nó do segundo nível da árvore
  - Note que se fizermos a fusão ordenada das rodadas de S_1 com as respectivas rodadas de S_2, obteremos o nível acima da árvore

8

Merge-Sort Seqüencial

- Os resultados das fusões ordenadas das rodadas de S_1 com as respectivas rodadas de S_2 são intercalados novamente nas duas seqüências auxiliares adicionais:

  S_3: 1 2 3 4 6 7 8 9

  S_4: ∅

  - Essas seqüências estão "ordenadas" em rodadas de tamanho 8
  - Como uma seqüência ficou vazia e todos os elementos estão contidos na outra (totalmente ordenada), o algoritmo encerra
  - A seqüência ordenada final corresponde à raiz da árvore de recursão

9

Merge-Sort Seqüencial (Exemplo)

- Seja a seguinte seqüência de 23 valores a serem ordenados:

S = [28 31 3 5 93 96 10 40 54 85 65 9 30 39 90 13 10 8 69 77 8 10 22]

- O primeiro passo é subdividir essa seqüência de forma intercalada:

Início:

S_1:	28	03	93	10	54	65	30	90	10	69	08	22
S_2:	31	05	96	40	85	09	39	13	08	77	10	

- Segue-se então uma sucessão de fusões intercaladas de duas seqüências parcialmente ordenadas em rodadas que dobram de tamanho a cada passagem

10

Merge-Sort Seqüencial (Exemplo)

1a Passagem:

S_3:	28	31	93	96	54	85	30	39	08	10	08	10
S_4:	03	05	10	40	09	65	13	90	69	77	22	

2a Passagem:

S_1:	03	05	28	31	09	54	65	85	08	10	69	77
S_2:	10	40	93	96	13	30	39	90	08	10	22	

3a Passagem

03	05	10	28	31	40	93	96	08	08	10	10	22	69	77	
09	13	30	39	54	65	85	90								

11

Merge-Sort Seqüencial (Exemplo)

4a Passagem

03	05	09	10	13	28	30	31	39	40	54	65	85	90	93	96
08	08	10	10	22	69	77									

- Como o tamanho das rodadas dobra a cada passagem, tem-se que após i passagens o tamanho é $r = 2^i$ e que quando $r \geq n$ (onde n é a quantidade total de elementos a serem ordenados) tem-se:

5a Passagem

03	05	08	08	09	10	10	10	13	22	28	30	31	39	40	54	65	69	77	85	90	93	96
∅																						

12

Desempenho

- O número de passagens necessárias é portanto tal que $2^i \geq n$
- Logo, qualquer $i \geq \log n$ número de passagens são suficientes:
  - ou seja, não mais que Teto[$\log n$] passagens bastam
- Como são n elementos a cada passagem e a fusão ordenada desses n elementos pode ser feita em tempo $O(n)$, tem-se que a complexidade do algoritmo Merge-Sort Seqüencial é $O(n \log n)$
  - a mesma que o Merge-Sort recursivo tradicional
- Mas então qual a vantagem desse algoritmo ???

13

Comparação

- Ambas as versões vistas de Merge Sort **não são in-place**:
  - Versão recursiva demanda pilha de recursão
  - Versão seqüencial demanda seqüências auxiliares
- Vantagem da versão seqüencial:
  - Permite acesso seqüencial aos dados
  - É a base para a construção de algoritmos de **ordenação externa**
    - Fundamentais em aplicações de grande porte, quando a seqüência a ser ordenada é grande demais para caber toda na memória principal

14

Implementação em C

- Exercício...

15

Exercícios

- Escolha uma seqüência de 31 números não ordenados e ilustre passo-a-passo, em detalhes, a ordenação dessa seqüência de números através de Merge-Sort Seqüencial
- Analise se o algoritmo Merge-Sort Seqüencial é estável ou não e discuta o porquê, apresentando exemplos
- Dada uma seqüência de 4386 elementos a serem ordenados, é possível prever a priori quantas passagens do algoritmo Merge-Sort Seqüencial serão necessárias para concluir a ordenação desses elementos ? Apresente o valor e justifique
- Implemente em C o algoritmo Merge-Sort Seqüencial

Bibliografia

- A. V. Aho, J. E. Hopcroft & J. Ullman, *Data Structures and Algorithms*, Addison Wesley, 1983
- N. Ziviani, *Projeto de Algoritmos*, Thomson, 2a. Edição, 2004

17