Algoritmos e Estrutura de Dados

Aula 19 – Estrutura de Dados: Algoritmos de Ordenamento

Prof. Tiago A. E. Ferreira

Algoritmos de Ordenamento

- Os algoritmos de ordenamento são aplicados para a organização de listas de elementos quaisquer em ordem crescente ou decrescente
- Existem vários algoritmos para este fim:
  - Merge Sort
  - Quick Sort
  - Etc...

Dividir e Conquistar

- Problema de tamanho n
- Dividir em k sub-instâncias disjuntas (1<k<=n)
- Os problemas são resolvidos separadamente
- As soluções parciais são combinadas de modo a se obter uma solução para o problema original

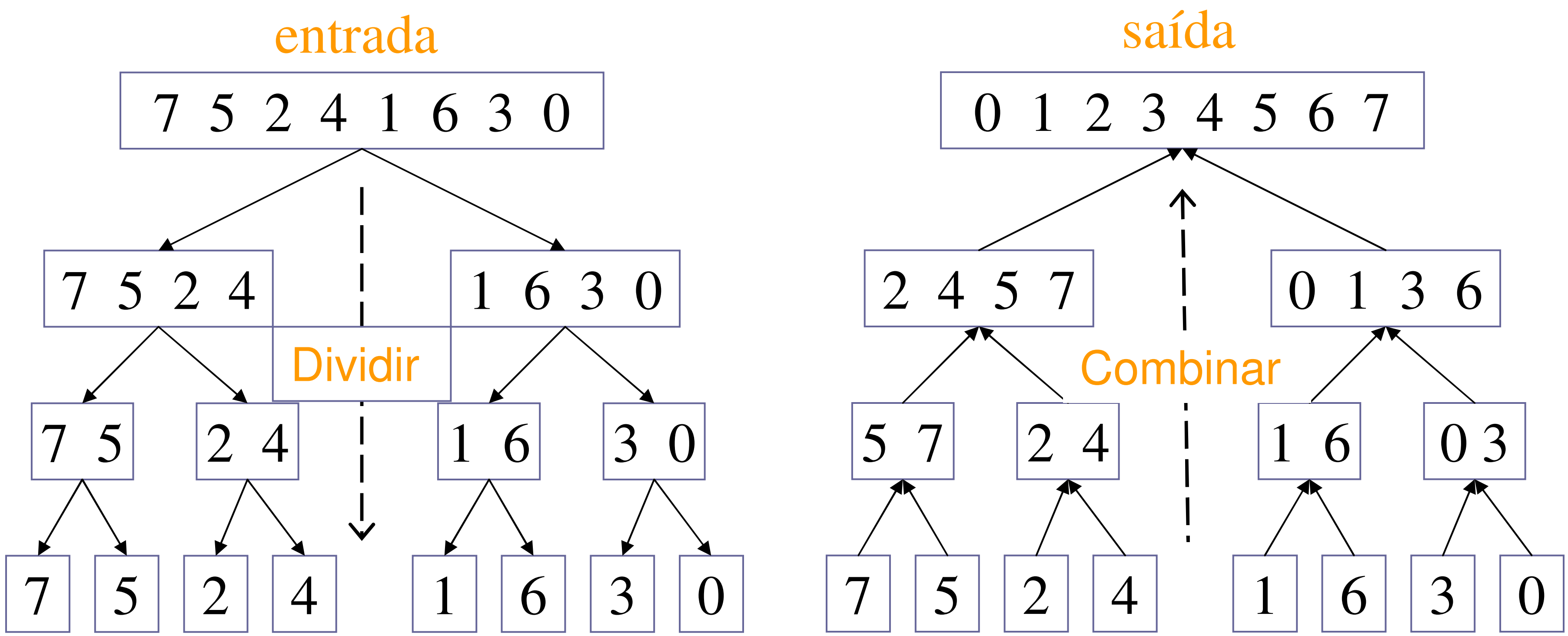
Dividir e Conquistar

- Aplicável a sub-isntâncias
- Subproblemas do mesmo tipo que o problema original
- Expressa por algoritmos recursivos
- 3 passos
  - Dividir
  - Conquistar
  - Combinar

Ordenamento por Intercalação MERGESORT

- Seja uma lista *A* de *n* elementos:
  - **Dividir** *A* em 2 sub-listas de tamanho ≈ *n*/2
  - **Conquistar**: ordenar cada sub-lista chamando *MergeSort* recursivamente
  - **Combinar** as sub-listas ordenadas formando uma única lista ordenada
- caso base: lista com um elemento

MERGESORT



entrada
7 5 2 4 1 6 3 0
7 5 2 4
1 6 3 0
Dividir
7 5
2 4
1 6
3 0
7
5
2
4
1
6
3
0
saída
0 1 2 3 4 5 6 7
2 4 5 7
0 1 3 6
Combinar
5 7
2 4
1 6
0 3
7
5
2
4
1
6
3
0

Algoritmo de Ordenação por Intercalação

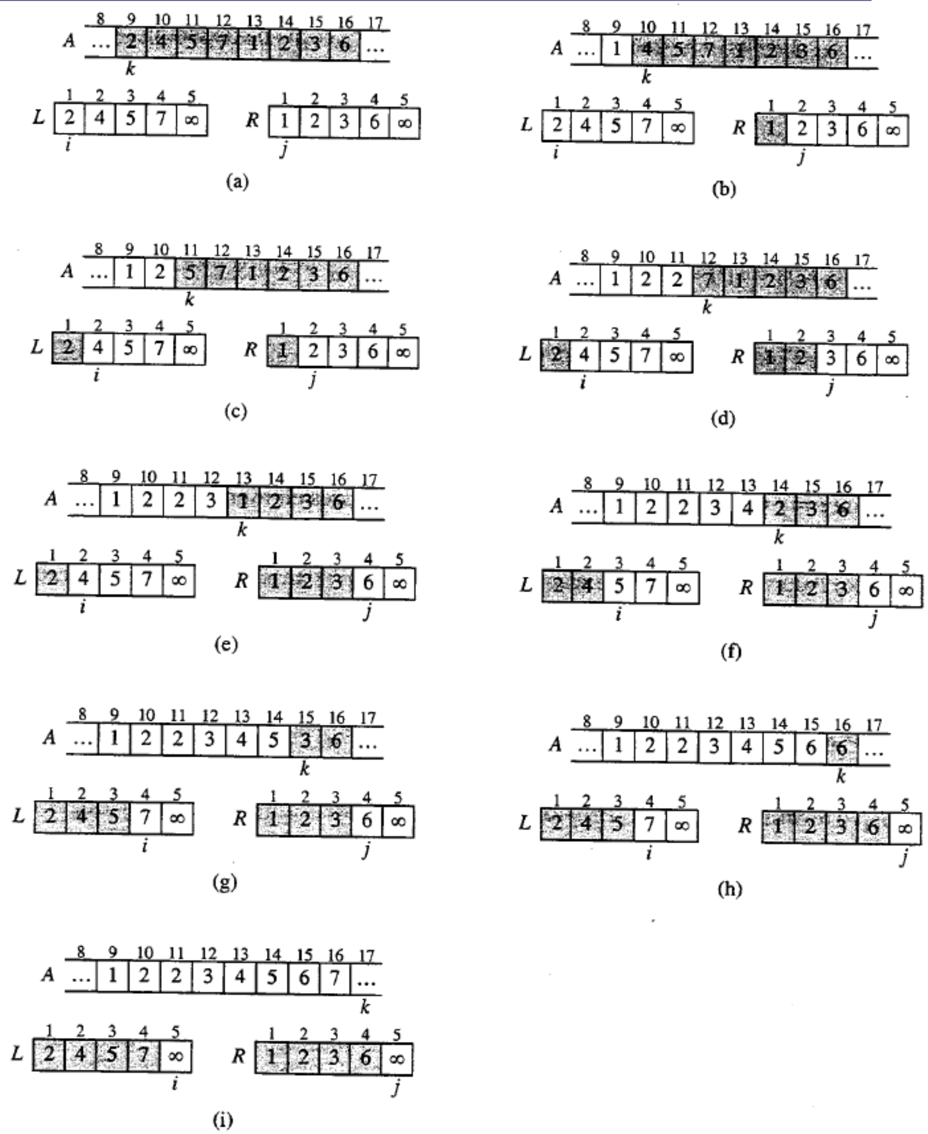
- A operação chave está no passo de combinação, onde são intercaladas (merge) duas subseqüências já ordenadas
  - Será utilizado o procedimento MERGE(A,p,q,r)
    - Onde A é um arranjo, p,q e r são índices de enumeração dos elementos do arranjo, tais que $p \leq q < r$
  - É pressuposto que os sub-arranjos A[p..q] e A[q+1..r] estejam em seqüência ordenada

Combinação

- Suponha a existência de duas pilha de cartas com as numerações para cima
  - Será gerada uma pilha de saída, onde será depositada a carta de menor valor dentre as que estão expostas nas duas pilhas iniciais. Esta pilha de saída é formada com as cartas viradas para baixo.
  - Ao término, será gerada uma única pilha ordenada com todas as cartas das duas pilhas iniciais
  - Sendo n o número total de cartas (duas pilhas iniciais), o custo em tempo será $\Theta(n)$

Algoritmo MERGE(A,p,q,r)

```
MERGE(A, P, q, r)
  n1 ← q − p + 1
  n2 ← r − q
  criar arranjos L[1..n1 + 1] e R[1..n2 + 1]
  for i ← 1 to n1
       do L[i] ← A[p + i − 1]
  for j ← 1 to n2
       do R[j] ← A[q + j]
  L[n1 + 1] ← ∞
  R[n2 + 1] ← ∞
  i ← 1
  j ← 1
  for k ← p to r
       do if L[i] ≤ R[j]
             then A[k] ← L[i]
                  i ← i + 1
             else A[k] ← R[j]
                  j ← j + 1
```



Algoritmo de ordenação por intercalação

- O Algoritmo MERGE-SORT(A,p,r) ordena os elementos do sub-arranjo A[p..r]
  - Se $p \geq r$, então o arranjo tem 1 elemento
  - Caso contrario, é calculado o índice q que particiona o arranjo A[p..r] em dois sub-arranjos: A[p..q], com $\lceil n/2 \rceil$ elementos, e A[q+1,r] com $\lfloor n/2 \rfloor$ elementos

```
MERGE-SORT(A, p, r)
if p < r
   then q ← ⌊(p + r)/2⌋
        MERGE-SORT(A, p, q)
        MERGE-SORT(A, q + 1, r)
        MERGE(A, p, q, r)
```

Complexidade do MERGESORT

- O Algoritmos MERGESORT é uma recursividade, assim o custo T(n) é da forma: $T(n)=aT(n/b)+f(n)$
- Onde:
  - a = 2
  - b =2
  - f(n) = c n + d (c = cte, d = cte)

Complexidade do MERGERSORT

- ***Recorrência***
  - $T_{MS}(n) = 2 \bullet T_{MS}(n/2) + c \bullet n + d$

    $= 2 \bullet [\ 2 \bullet T_{MS}(n/4) + c \bullet n/2 + d\ ] + c \bullet n + d$

    $= 4 \bullet T_{MS}(n/4) + 2c \bullet n + 3d$

    $= 4 \bullet [\ 2 \bullet T_{MS}(n/8) + c \bullet n/4 + d\ ] + 2c \bullet n + 3d$

    $= 8 \bullet T_{MS}(n/8) + 3c \bullet n + 7d$

    $= \ldots$
  - $T_{MS}(n) = 2^i \bullet T_{MS}(n/2^i) + i \bullet c \bullet n + (2^i - 1) \bullet d$

Removendo Elemento

- Note que sabemos o valor de TMS(1).
- Para obter um valor fechado, queremos que $(n/2^{i}) = 1$.
  - Isso ocorre quando $i = \log_2 n$.
- $T_{MS}(n) = 2^{\log_2 n} \bullet cte + (\log_2 n) \bullet c \bullet n + (2^{\log_2 n} - 1) \bullet d$

  $= n \bullet cte + c \bullet n \bullet \log_2 n + d \bullet (n-1),$

  $= c \bullet n\log_2 n + (cte+d) \bullet n - d$
- **$TMS(n) = O(n \bullet \log_2 n)$**

Exercício:

- Implemente o MERGESORT em Phyton.