Aula 08: Algoritmos de ordenação em arranjos

Ordenação por inserção

David Déharbe
Programa de Pós-graduação em Sistemas e Computação
Universidade Federal do Rio Grande do Norte
Centro de Ciências Exatas e da Terra
Departamento de Informática e Matemática Aplicada

Download me from `http://DavidDeharbe.github.io`.

PPgSC

Algoritmos

Ordenação de arranjo

Entrada
- uma sequência linear $A = \langle A_1, \ldots A_n \rangle$,

Saída
- A é uma permutação de $A_1, \ldots, A_n$ tal que $A_i \leq A_{i+1}$ para $1 \leq i < n$.

Algoritmos
- ordenação por inserção (*insertion sort*)
- ordenação por fusão (*merge sort*)
- ordenação por *heap* (*heap sort*)
- ordenação *quick-sort*
- ordenação por contadores (*counting sort*)

PPgSC

Plano da aula

Introdução

Ordenação por inserção
- Algoritm
- Simulação
- Complexidade
- Correção
- Exercício

Ordenação por inserção *insertion sort*

O algoritmo

(Cormen et al. 1990)

INSERTION-SORT(A)

```
1  for j = 2 to length[A]
2      key = A[j]
       // Insert A[j] into the sorted sequence A[1..j − 1].
3      i = j − 1
4      while i > 0 and A[i] > key
5          A[i + 1] = A[i]
6          i = i − 1
7      A[i + 1] = key
```

Ordenação por inserção

Simulação

INSERTION-SORT(A)

```
1  for j = 2 to length[A]
2      key = A[j]
3      i = j − 1
4      while i > 0 and A[i] > key
5          A[i + 1] = A[i]
6          i = i − 1
7      A[i + 1] = key
```

j	A
	$\langle 5, 2, 4, 6, 1, 3 \rangle$

PPgSC

Ordenação por inserção

Simulação

INSERTION-SORT(A)

```
1  for j = 2 to length[A]
2      key = A[j]
3      i = j − 1
4      while i > 0 and A[i] > key
5          A[i + 1] = A[i]
6          i = i − 1
7      A[i + 1] = key
```

j	A
	$\langle 5, 2, 4, 6, 1, 3 \rangle$
2	$\langle \times 5, 2, 4, 6, 1, 3 \rangle$

Ordenação por inserção

Simulação

INSERTION-SORT(A)

```
1  for j = 2 to length[A]
2      key = A[j]
3      i = j − 1
4      while i > 0 and A[i] > key
5          A[i + 1] = A[i]
6          i = i − 1
7      A[i + 1] = key
```

j	A
	$\langle 5, 2, 4, 6, 1, 3 \rangle$
2	$\langle \times 5, 2, 4, 6, 1, 3 \rangle$
3	$\langle 2, \times 5, 4, 6, 1, 3 \rangle$

PPgSC

Ordenação por inserção

Simulação

INSERTION-SORT(A)

```
1  for j = 2 to length[A]
2      key = A[j]
3      i = j − 1
4      while i > 0 and A[i] > key
5          A[i + 1] = A[i]
6          i = i − 1
7      A[i + 1] = key
```

j	A
	$\langle 5, 2, 4, 6, 1, 3 \rangle$
2	$\langle \times 5, 2, 4, 6, 1, 3 \rangle$
3	$\langle 2, \times 5, 4, 6, 1, 3 \rangle$
4	$\langle 2, 4, 5, \times 6, 1, 3 \rangle$

PPgSC

Ordenação por inserção

Simulação

INSERTION-SORT(A)

```
1  for j = 2 to length[A]
2      key = A[j]
3      i = j − 1
4      while i > 0 and A[i] > key
5          A[i + 1] = A[i]
6          i = i − 1
7      A[i + 1] = key
```

j	A
	$\langle 5, 2, 4, 6, 1, 3 \rangle$
2	$\langle \times 5, 2, 4, 6, 1, 3 \rangle$
3	$\langle 2, \times 5, 4, 6, 1, 3 \rangle$
4	$\langle 2, 4, 5, \times 6, 1, 3 \rangle$
5	$\langle \times 2, 4, 5, 6, 1, 3 \rangle$

PPgSC

Ordenação por inserção

Simulação

INSERTION-SORT(A)

```
1  for j = 2 to length[A]
2      key = A[j]
3      i = j − 1
4      while i > 0 and A[i] > key
5          A[i + 1] = A[i]
6          i = i − 1
7      A[i + 1] = key
```

j	A
	$\langle 5, 2, 4, 6, 1, 3 \rangle$
2	$\langle \times 5, 2, 4, 6, 1, 3 \rangle$
3	$\langle 2, \times 5, 4, 6, 1, 3 \rangle$
4	$\langle 2, 4, 5, \times 6, 1, 3 \rangle$
5	$\langle \times 2, 4, 5, 6, 1, 3 \rangle$
6	$\langle 1, 2, \times 4, 5, 6, 3 \rangle$

Ordenação por inserção

Simulação

INSERTION-SORT(A)

```
1  for j = 2 to length[A]
2      key = A[j]
3      i = j − 1
4      while i > 0 and A[i] > key
5          A[i + 1] = A[i]
6          i = i − 1
7      A[i + 1] = key
```

j	A
	$\langle 5, 2, 4, 6, 1, 3 \rangle$
2	$\langle \times 5, 2, 4, 6, 1, 3 \rangle$
3	$\langle 2, \times 5, 4, 6, 1, 3 \rangle$
4	$\langle 2, 4, 5, \times 6, 1, 3 \rangle$
5	$\langle \times 2, 4, 5, 6, 1, 3 \rangle$
6	$\langle 1, 2, \times 4, 5, 6, 3 \rangle$
7	$\langle 1, 2, 3, 4, 5, 6 \rangle$

PPgSC

Ordenação por inserção

Complexidade

- Pior caso
- Melhor caso

Ordenação por inserção

Complexidade

- Pior caso
  - Quando A está inicialmente em ordem inversa
  - $\Theta(n^2)$
- Melhor caso

Ordenação por inserção

Complexidade

- Pior caso
  - Quando A está inicialmente em ordem inversa
  - $\Theta(n^2)$
- Melhor caso
  - Quando A está inicialmente ordenado
  - $\Theta(n)$

PPgSC

Ordenação por inserção

Correção

contrato o que é ser correto?

- **pré-condição** descrição dos valores legais para os parâmetros
- **pós-condição** descrição do resultado e efeitos colaterais da sub-rotina

correção (parcial) invariantes de laço: propriedades sobre as variáveis mantidas durante a execução dos laços.

término variantes de laço: número natural que decresce a cada execução do laço

Ordenação por inserção

Correção

INSERTION-SORT(A)

```
1  for j = 2 to length[A]
2      key = A[j]
3      i = j − 1
4      while i > 0 and A[i] > key
5          A[i + 1] = A[i]
6          i = i − 1
7      A[i + 1] = key
```

Ordenação por inserção

Correção

$\{A = \langle a_1, \ldots, a_n \rangle\}$

```
1  for j = 2 to length[A]
2      key = A[j]
3      i = j − 1
4      while i > 0 and A[i] > key
5          A[i + 1] = A[i]
6          i = i − 1
7      A[i + 1] = key
```

Ordenação por inserção

Correção

$\{A = \langle a_1, \ldots, a_n \rangle\}$

```
1  for j = 2 to length[A]
2      key = A[j]
3      i = j − 1
4      while i > 0 and A[i] > key
5          A[i + 1] = A[i]
6          i = i − 1
7      A[i + 1] = key
```

$\{A \in permutation(\langle a_1, \ldots, a_n \rangle) \wedge \forall k \mid 1 \leq k < n \cdot A[k] \leq A[k+1]\}$

Ordenação por inserção

Correção

$\{A = \langle a_1, \ldots, a_n \rangle\}$

1 **for** $j = 2$ **to** $length[A]$

$\{\ 2 \leq j \leq n+1 \wedge$
$A[1\,..\,j-1] \in permutation(\langle a_1, \ldots, a_{j-1} \rangle) \wedge$
$\forall k \mid 1 \leq k < j-1 \cdot A[k] \leq A[k+1] \wedge$
$\forall k \mid j \leq k \leq n \cdot A[k] = a_k\}$

2 $key = A[j]$

3 $i = j - 1$

4 **while** $i > 0$ and $A[i] > key$

5 $A[i+1] = A[i]$

6 $i = i - 1$

7 $A[i+1] = key$

$\{A \in permutation(\langle a_1, \ldots, a_n \rangle) \wedge \forall k \mid 1 \leq k < n \cdot A[k] \leq A[k+1]\}$

Ordenação por inserção

Correção

$\{A = \langle a_1, \ldots, a_n \rangle\}$

1 **for** $j = 2$ **to** $length[A]$

$\{\ 2 \leq j \leq n+1 \wedge$

$A[1\,..\,j-1] \in permutation(\langle a_1, \ldots, a_{j-1} \rangle) \wedge$

$\forall k \mid 1 \leq k < j-1 \cdot A[k] \leq A[k+1] \wedge$

$\forall k \mid j \leq k \leq n \cdot A[k] = a_k\}$

2 $key = A[j]$

3 $i = j - 1$

4 **while** $i > 0$ and $A[i] > key$

5 $A[i+1] = A[i]$

6 $i = i - 1$

7 $A[i+1] = key$

$\{A \in permutation(\langle a_1, \ldots, a_n \rangle) \wedge \forall k \mid 1 \leq k < n \cdot A[k] \leq A[k+1]\}$

Considere $j = n + 1$ (término da execução do laço).

Ordenação por inserção

Subsídios para a correção

$\{\ 2 \leq j \leq n+1 \wedge$
$A[1\,..\,j-1] \in \mathit{permutation}(\langle a_1, \ldots, a_{j-1} \rangle) \wedge$
$\forall k \mid 1 \leq k < j-1 \cdot A[k] \leq A[k+1] \wedge$
$\forall k \mid j \leq k \leq n \cdot A[k] = a_k \}$

```
1       key = A[j]
2       i = j − 1
3       while i > 0 and A[i] > key
4           A[i + 1] = A[i]
5           i = i − 1
6       A[i + 1] = key
```

Ordenação por inserção

Subsídios para a correção

$\{\ 2 \leq j \leq n+1 \wedge$
$A[1\,..\,j-1] \in \mathit{permutation}(\langle a_1, \ldots, a_{j-1} \rangle) \wedge$
$\forall k \mid 1 \leq k < j-1 \cdot A[k] \leq A[k+1] \wedge$
$\forall k \mid j \leq k \leq n \cdot A[k] = a_k\}$

1 $\quad key = A[j]$
2 $\quad i = j - 1$
3 $\quad$ **while** $i > 0$ and $A[i] > key$
$\quad \{\ 2 \leq j \leq n+1 \wedge 0 \leq i < j \wedge key = a_j \wedge$
$\quad\quad \forall k \mid j+1 \leq k \leq n \cdot A[k] = a_k\}$
4 $\quad\quad A[i+1] = A[i]$
5 $\quad\quad i = i - 1$
6 $\quad A[i+1] = key$

Ordenação por inserção

Subsídios para a correção

$\{\ 2 \leq j \leq n+1 \wedge$
$A[1\,..\,j-1] \in permutation(\langle a_1, \ldots, a_{j-1} \rangle) \wedge$
$\forall k \mid 1 \leq k < j-1 \cdot A[k] \leq A[k+1] \wedge$
$\forall k \mid j \leq k \leq n \cdot A[k] = a_k \}$

1 $key = A[j]$
2 $i = j - 1$
3 **while** $i > 0$ and $A[i] > key$
$\{\ 2 \leq j \leq n+1 \wedge 0 \leq i < j \wedge key = a_j \wedge$
$A[1..j] - \{A[i+1]\} \cup \{key\} \in permutation(\langle a_1, \ldots, a_j \rangle) \wedge$
$\forall k \mid j+1 \leq k \leq n \cdot A[k] = a_k \}$
4 $A[i+1] = A[i]$
5 $i = i - 1$
6 $A[i+1] = key$

Ordenação por inserção

Subsídios para a correção

$\{\ 2 \leq j \leq n+1 \wedge$
$A[1\,..\,j-1] \in permutation(\langle a_1, \ldots, a_{j-1} \rangle) \wedge$
$\forall k \mid 1 \leq k < j-1 \cdot A[k] \leq A[k+1] \wedge$
$\forall k \mid j \leq k \leq n \cdot A[k] = a_k \}$

1 $key = A[j]$
2 $i = j - 1$
3 **while** $i > 0$ and $A[i] > key$
$\{\ 2 \leq j \leq n+1 \wedge 0 \leq i < j \wedge key = a_j \wedge$
$A[1..j] - \{A[i+1]\} \cup \{key\} \in permutation(\langle a_1, \ldots, a_j \rangle) \wedge$
$\forall k \mid 1 \leq k < j \cdot A[k] \leq A[k+1] \wedge$
$\forall k \mid j+1 \leq k \leq n \cdot A[k] = a_k \}$
4 $A[i+1] = A[i]$
5 $i = i - 1$
6 $A[i+1] = key$

PPgSC

Ordenação por inserção

Subsídios para a correção

$\{\ 2 \leq j \leq n+1 \wedge$
$A[1\,..\,j-1] \in permutation(\langle a_1, \ldots, a_{j-1} \rangle) \wedge$
$\forall k \mid 1 \leq k < j-1 \cdot A[k] \leq A[k+1] \wedge$
$\forall k \mid j \leq k \leq n \cdot A[k] = a_k\}$

1 $key = A[j]$
2 $i = j - 1$
3 **while** $i > 0$ and $A[i] > key$
$\{\ 2 \leq j \leq n+1 \wedge 0 \leq i < j \wedge key = a_j \wedge$
$A[1..j] - \{A[i+1]\} \cup \{key\} \in permutation(\langle a_1, \ldots, a_j \rangle) \wedge$
$\forall k \mid 1 \leq k < j \cdot A[k] \leq A[k+1] \wedge$
$\forall k \mid i+1 \leq k \leq j \cdot key \leq A[k] \wedge$
$\forall k \mid j+1 \leq k \leq n \cdot A[k] = a_k\}$
4 $A[i+1] = A[i]$
5 $i = i - 1$
6 $A[i+1] = key$

PPgSC

Exercício

A ordenação por inserção pode ser realizada de forma recursiva: dado um arranjo $A[1\,..\,n]$, 1) ordenar $A[1\,..\,n-1]$ e 2) inserir $A[n]$ no arranjo resultante.

1. Escreva o algoritmo recursivo;
2. Determine a complexidade deste algoritmo;
3. Escreve a pré-condição e a pós-condição deste algoritmo.