Aula 12 – Algoritmos de Ordenação: Modelo Incremental

Norton Trevisan Roman
norton@usp.br

27 de setembro de 2018

Ordenação

- Trata do problema de ordenar um conjunto de $n \geq 1$ valores

Ordenação

- Trata do problema de ordenar um conjunto de $n \geq 1$ valores
  - Arranjo, lista ligada etc

Ordenação

- Trata do problema de ordenar um conjunto de $n \geq 1$ valores
  - Arranjo, lista ligada etc
- Podemos projetar por indução diversos algoritmos para o problema da ordenação

Ordenação

- Trata do problema de ordenar um conjunto de $n \geq 1$ valores
  - Arranjo, lista ligada etc
- Podemos projetar por indução diversos algoritmos para o problema da ordenação
- De fato, todos os algoritmos básicos de ordenação surgem de projetos por indução sutilmente diferentes

Ordenação

Modelo Incremental: Indução Fraca

Ordenação

Modelo Incremental: Indução Fraca

- Padrão seguido pelos algoritmos:

```
OrdenaçãoIncremental(A, n):
  Entrada: Arranjo A de n valores
  Saída: Arranjo A ordenado

  se n == 1 então retorne
  senão:
    <comandos iniciais>
    OrdenaçãoIncremental(A, n - 1)
    <comandos finais>
  retorne
```

Ordenação

Divisão e Conquista: Indução Forte

Ordenação

Divisão e Conquista: Indução Forte

- Padrão seguido pelos algoritmos:

```
OrdenaçãoD&C(A, ini, fim):
  Entrada: Arranjo A de n valores
  Saída: Arranjo A ordenado

  n = fim - ini + 1
  se n == 1 então retorne
  senão:
    <comandos iniciais: a divisão> (cálculo de q)
    OrdenaçãoD&C(A, ini, q)
    OrdenaçãoD&C(A, q+1, fim)
    <comandos finais: a conquista>
  retorne
```

Projeto por Indução Fraca

Primeira Alternativa

Projeto por Indução Fraca

Primeira Alternativa

- **Base**: $n = 1$. Um conjunto de um único elemento está ordenado

Projeto por Indução Fraca

Primeira Alternativa

- **Base**: $n = 1$. Um conjunto de um único elemento está ordenado
- **H.I.**: Sei ordenar um conjunto de $n - 1 \geq 1$ valores

Projeto por Indução Fraca

Primeira Alternativa

- **Base**: $n = 1$. Um conjunto de um único elemento está ordenado
- **H.I.**: Sei ordenar um conjunto de $n - 1 \geq 1$ valores
- **Passo**:

Projeto por Indução Fraca

Primeira Alternativa

- **Base**: $n = 1$. Um conjunto de um único elemento está ordenado
- **H.I.**: Sei ordenar um conjunto de $n - 1 \geq 1$ valores
- **Passo**:
  - Seja S um conjunto de $n \geq 2$ valores, e x um elemento qualquer de S

Projeto por Indução Fraca

Primeira Alternativa

- **Base**: $n = 1$. Um conjunto de um único elemento está ordenado
- **H.I.**: Sei ordenar um conjunto de $n - 1 \geq 1$ valores
- **Passo**:
  - Seja S um conjunto de $n \geq 2$ valores, e x um elemento qualquer de S
  - Pela H.I., sabemos ordenar o conjunto $S - x$. Basta então inserir x na posição correta para obtermos S ordenado

Projeto por Indução Fraca

Primeira Alternativa

- **Base**: $n = 1$. Um conjunto de um único elemento está ordenado
- **H.I.**: Sei ordenar um conjunto de $n - 1 \geq 1$ valores
- **Passo**:
  - Seja S um conjunto de $n \geq 2$ valores, e x um elemento qualquer de S
  - Pela H.I., sabemos ordenar o conjunto $S - x$. Basta então inserir x na posição correta para obtermos S ordenado
- Método da Inserção (*Insertion Sort*)

Projeto por Indução Fraca

Método da Inserção (recursivo)

```
Inserção(A, n):
  Entrada: Arranjo A de n valores
  Saída: Arranjo A ordenado

  se n == 1: retorna // está ordenado
  senão:
    Inserção(A, n - 1)
    v = A[n-1]
    j=n-1
    enquanto (j > 0) e (A[j-1] > v) faça:
      A[j] = A[j-1]
      j=j-1
    A[j] = v
```

Projeto por Indução Fraca

Método da Inserção (recursivo)

- Quantas comparações no arranjo são feitas no pior caso?

```
Inserção(A, n):
  se n == 1: retorna
  senão:
    Inserção(A, n - 1)
    v = A[n-1]
    j=n-1
    enquanto (j > 0) e (A[j-1] > v) faça:
      A[j] = A[j-1]
      j=j-1
    A[j] = v
```

Projeto por Indução Fraca

Método da Inserção (recursivo)

- Quantas comparações no arranjo são feitas no pior caso?

```
Inserção(A, n):
  se n == 1: retorna
  senão:
    Inserção(A, n - 1)
    v = A[n-1]
    j=n-1
    enquanto (j > 0) e (A[j-1] > v) faça:
      A[j] = A[j-1]
      j=j-1
    A[j] = v
```

$$T(n) = \begin{cases} & \text{se } n = 1 \\ & \text{para } n \geq 2 \end{cases}$$

Projeto por Indução Fraca

Método da Inserção (recursivo)

- Quantas comparações no arranjo são feitas no pior caso?

```
Inserção(A, n):
  se n == 1: retorna
  senão:
    Inserção(A, n - 1)
    v = A[n-1]
    j=n-1
    enquanto (j > 0) e (A[j-1] > v) faça:
      A[j] = A[j-1]
      j=j-1
    A[j] = v
```

$$T(n) = \begin{cases} 0 & \text{se } n = 1 \\ & \text{para } n \geq 2 \end{cases}$$

Projeto por Indução Fraca

Método da Inserção (recursivo)

- Quantas comparações no arranjo são feitas no pior caso?

```
Inserção(A, n):
  se n == 1: retorna
  senão:
    Inserção(A, n - 1)
    v = A[n-1]
    j=n-1
    enquanto (j > 0) e (A[j-1] > v) faça:
      A[j] = A[j-1]
      j=j-1
    A[j] = v
```

$$T(n) = \begin{cases} 0 & \text{se } n = 1 \\ T(n-1) & \text{para } n \geq 2 \end{cases}$$

Projeto por Indução Fraca

Método da Inserção (recursivo)

- Quantas comparações no arranjo são feitas no pior caso?

```
Inserção(A, n):
  se n == 1: retorna
  senão:
    Inserção(A, n - 1)
    v = A[n-1]
    j=n-1
    enquanto (j > 0) e (A[j-1] > v) faça:
      A[j] = A[j-1]
      j=j-1
    A[j] = v
```

$$T(n) = \begin{cases} 0 & \text{se } n = 1 \\ T(n-1) + (n-1) & \text{para } n \geq 2 \end{cases}$$

Projeto por Indução Fraca

Método da Inserção (recursivo)

- Então

$$T(n) = T(n-1) + n - 1$$

Projeto por Indução Fraca

Método da Inserção (recursivo)

- Então

$$
\begin{aligned}
T(n) &= T(n-1) + n - 1 \\
&= (T(n-2) + (n-1) - 1) + n - 1
\end{aligned}
$$

Projeto por Indução Fraca

Método da Inserção (recursivo)

- Então

$$
\begin{aligned}
T(n) &= T(n-1)+n-1 \\
&= (T(n-2)+(n-1)-1)+n-1 \\
& \quad T(n-2)+2n-2-1
\end{aligned}
$$

Projeto por Indução Fraca

Método da Inserção (recursivo)

- Então

$$
\begin{aligned}
T(n) &= T(n-1) + n - 1 \\
&= (T(n-2) + (n-1) - 1) + n - 1 \\
& \quad T(n-2) + 2n - 2 - 1 \\
&= (T(n-3) + (n-2) - 1) + 2n - 2 - 1
\end{aligned}
$$

Projeto por Indução Fraca

Método da Inserção (recursivo)

- Então

$$
\begin{aligned}
T(n) &= T(n-1) + n - 1 \\
&= (T(n-2) + (n-1) - 1) + n - 1 \\
& \quad T(n-2) + 2n - 2 - 1 \\
&= (T(n-3) + (n-2) - 1) + 2n - 2 - 1 \\
& \quad T(n-3) + 3n - 3 - 2 - 1
\end{aligned}
$$

Projeto por Indução Fraca

Método da Inserção (recursivo)

- Então

$$
\begin{aligned}
T(n) &= T(n-1) + n - 1 \\
&= (T(n-2) + (n-1) - 1) + n - 1 \\
& \quad T(n-2) + 2n - 2 - 1 \\
&= (T(n-3) + (n-2) - 1) + 2n - 2 - 1 \\
& \quad T(n-3) + 3n - 3 - 2 - 1 \\
& \quad \ldots
\end{aligned}
$$

Projeto por Indução Fraca

Método da Inserção (recursivo)

- Então

$$
\begin{aligned}
T(n) &= T(n-1) + n - 1 \\
&= (T(n-2) + (n-1) - 1) + n - 1 \\
& \quad T(n-2) + 2n - 2 - 1 \\
&= (T(n-3) + (n-2) - 1) + 2n - 2 - 1 \\
& \quad T(n-3) + 3n - 3 - 2 - 1 \\
& \quad \ldots \\
&= T(n-k) + kn - \sum_{i=1}^{k} i
\end{aligned}
$$

Projeto por Indução Fraca

Método da Inserção (recursivo)

- E

$$T(n) = T(n-k) + kn - \sum_{i=1}^{k} i$$

Projeto por Indução Fraca

Método da Inserção (recursivo)

- E

$$\begin{aligned} T(n) &= T(n-k) + kn - \sum_{i=1}^{k} i \\ &= T(1) + kn - \sum_{i=1}^{k} i, \text{ em } T(n-k) = T(1) \end{aligned}$$

Projeto por Indução Fraca

Método da Inserção (recursivo)

- E

$$
\begin{aligned}
T(n) &= T(n-k) + kn - \sum_{i=1}^{k} i \\
&= T(1) + kn - \sum_{i=1}^{k} i, \text{ em } T(n-k) = T(1) \\
&= T(1) + (n-1)n - \sum_{i=1}^{n-1} i, \text{ pois } n-k = 1
\end{aligned}
$$

Projeto por Indução Fraca

Método da Inserção (recursivo)

- E

$$
\begin{aligned}
T(n) &= T(n-k) + kn - \sum_{i=1}^{k} i \\
&= T(1) + kn - \sum_{i=1}^{k} i, \text{ em } T(n-k) = T(1) \\
&= T(1) + (n-1)n - \sum_{i=1}^{n-1} i, \text{ pois } n-k=1 \\
&= (n-1)n - \sum_{i=1}^{n-1} i, \text{ pois } T(1) = 0
\end{aligned}
$$

Projeto por Indução Fraca

Método da Inserção (recursivo)

- E

$$
\begin{aligned}
T(n) &= T(n-k) + kn - \sum_{i=1}^{k} i \\
&= T(1) + kn - \sum_{i=1}^{k} i, \text{ em } T(n-k) = T(1) \\
&= T(1) + (n-1)n - \sum_{i=1}^{n-1} i, \text{ pois } n-k = 1 \\
&= (n-1)n - \sum_{i=1}^{n-1} i, \text{ pois } T(1) = 0 \\
&= (n-1)n - \frac{(n-1)n}{2}
\end{aligned}
$$

Projeto por Indução Fraca

Método da Inserção (recursivo)

$$T(n) = (n-1)n - \frac{(n-1)n}{2}$$

Projeto por Indução Fraca

Método da Inserção (recursivo)

$$
\begin{aligned}
T(n) &= (n-1)n - \frac{(n-1)n}{2} \\
&= n^2 - n - \frac{n^2 - n}{2}
\end{aligned}
$$

Projeto por Indução Fraca

Método da Inserção (recursivo)

$$
\begin{aligned}
T(n) &= (n-1)n - \frac{(n-1)n}{2} \\
&= n^2 - n - \frac{n^2 - n}{2} \\
&= \frac{n^2 - n}{2}
\end{aligned}
$$

Projeto por Indução Fraca

Método da Inserção (recursivo)

$$
\begin{aligned}
T(n) &= (n-1)n - \frac{(n-1)n}{2} \\
&= n^2 - n - \frac{n^2 - n}{2} \\
&= \frac{n^2 - n}{2} \\
&= \Theta(n^2)
\end{aligned}
$$

Método da Inserção (recursivo)

- Usando a notação O, quantas comparações no arranjo são feitas no pior caso?

```
Inserção(A, n):
  se n == 1: retorna
  senão:
    Inserção(A, n - 1)
    v = A[n-1]
    j=n-1
    enquanto (j > 0) e (A[j-1] > v) faça:
      A[j] = A[j-1]
      j=j-1
    A[j] = v
```

Projeto por Indução Fraca

Método da Inserção (recursivo)

- Usando a notação O, quantas comparações no arranjo são feitas no pior caso?

```
Inserção(A, n):
  se n == 1: retorna
  senão:
    Inserção(A, n - 1)
    v = A[n-1]
    j=n-1
    enquanto (j > 0) e (A[j-1] > v) faça:
      A[j] = A[j-1]
      j=j-1
    A[j] = v
```

$$T(n) = \begin{cases} & \text{se } n = 1 \\ & \text{para } n \geq 2 \end{cases}$$

Projeto por Indução Fraca

Método da Inserção (recursivo)

- Usando a notação O, quantas comparações no arranjo são feitas no pior caso?

```
Inserção(A, n):
  se n == 1: retorna
  senão:
    Inserção(A, n - 1)
    v = A[n-1]
    j=n-1
    enquanto (j > 0) e (A[j-1] > v) faça:
      A[j] = A[j-1]
      j=j-1
    A[j] = v
```

$$T(n) = \begin{cases} O(1) & \text{se } n = 1 \\ & \text{para } n \geq 2 \end{cases}$$

Projeto por Indução Fraca

Método da Inserção (recursivo)

- Usando a notação O, quantas comparações no arranjo são feitas no pior caso?

```
Inserção(A, n):
  se n == 1: retorna
  senão:
    Inserção(A, n - 1)
    v = A[n-1]
    j=n-1
    enquanto (j > 0) e (A[j-1] > v) faça:
      A[j] = A[j-1]
      j=j-1
    A[j] = v
```

$$T(n) = \begin{cases} O(1) & \text{se } n = 1 \\ T(n-1) & \text{para } n \geq 2 \end{cases}$$

Projeto por Indução Fraca

Método da Inserção (recursivo)

- Usando a notação O, quantas comparações no arranjo são feitas no pior caso?

```
Inserção(A, n):
  se n == 1: retorna
  senão:
    Inserção(A, n - 1)
    v = A[n-1]
    j=n-1
    enquanto (j > 0) e (A[j-1] > v) faça:
      A[j] = A[j-1]
      j=j-1
    A[j] = v
```

$$T(n) = \begin{cases} O(1) & \text{se } n = 1 \\ T(n-1) + O(n) & \text{para } n \geq 2 \end{cases}$$

Projeto por Indução Fraca

Método da Inserção (recursivo)

- E

$$T(n) = T(n-1) + O(n)$$

Projeto por Indução Fraca

Método da Inserção (recursivo)

- E

$$\begin{aligned} T(n) &= T(n-1) + O(n) \\ &= (T(n-2) + O(n)) + O(n) \end{aligned}$$

Projeto por Indução Fraca

Método da Inserção (recursivo)

- E

$$
\begin{aligned}
T(n) &= T(n-1) + O(n) \\
&= (T(n-2) + O(n)) + O(n) \\
&= (T(n-3) + O(n)) + O(n) + O(n)
\end{aligned}
$$

Projeto por Indução Fraca

Método da Inserção (recursivo)

- E

$$\begin{aligned} T(n) &= T(n-1) + O(n) \\ &= (T(n-2) + O(n)) + O(n) \\ &= (T(n-3) + O(n)) + O(n) + O(n) \\ &\ldots \end{aligned}$$

Projeto por Indução Fraca

Método da Inserção (recursivo)

- E

$$
\begin{aligned}
T(n) &= T(n-1) + O(n) \\
&= (T(n-2) + O(n)) + O(n) \\
&= (T(n-3) + O(n)) + O(n) + O(n) \\
&\ldots \\
&= T(n-k) + kO(n)
\end{aligned}
$$

Projeto por Indução Fraca

Método da Inserção (recursivo)

- E

$$
\begin{aligned}
T(n) &= T(n-1) + O(n) \\
&= (T(n-2) + O(n)) + O(n) \\
&= (T(n-3) + O(n)) + O(n) + O(n) \\
&\dots \\
&= T(n-k) + kO(n) \\
&= T(1) + (n-1)O(n), \text{ quando } T(n-k) = T(1)
\end{aligned}
$$

Projeto por Indução Fraca

Método da Inserção (recursivo)

- E

$$
\begin{aligned}
T(n) &= T(n-1) + O(n) \\
&= (T(n-2) + O(n)) + O(n) \\
&= (T(n-3) + O(n)) + O(n) + O(n) \\
&\ldots \\
&= T(n-k) + kO(n) \\
&= T(1) + (n-1)O(n), \text{ quando } T(n-k) = T(1) \\
&= O(1) + O(n^2) - O(n)
\end{aligned}
$$

Projeto por Indução Fraca

Método da Inserção (recursivo)

- E

$$
\begin{aligned}
T(n) &= T(n-1) + O(n) \\
&= (T(n-2) + O(n)) + O(n) \\
&= (T(n-3) + O(n)) + O(n) + O(n) \\
&\quad \ldots \\
&= T(n-k) + kO(n) \\
&= T(1) + (n-1)O(n), \text{ quando } T(n-k) = T(1) \\
&= O(1) + O(n^2) - O(n) \\
&= O(n^2)
\end{aligned}
$$

Projeto por Indução Fraca

Método da Inserção (iterativo)

```
Inserção(A, n):
  Entrada: Arranjo A de n valores
  Saída: Arranjo A ordenado

  para i = 1 até n - 1 faça:
    v = A[i]
    j=i
    enquanto (j > 0) e (A[j-1] > v) faça:
      A[j] = A[j-1]
      j=j-1
    A[j] = v
```

Projeto por Indução Fraca

Método da Inserção (iterativo)

- Usando a notação O, quantas comparações no arranjo são feitas no pior caso?

```
Inserção(A, n):
  Entrada: Arranjo A de n valores
  Saída: Arranjo A ordenado

  para i = 1 até n - 1 faça:
    v = A[i]
    j=i
    enquanto (j > 0) e (A[j-1] > v) faça:
      A[j] = A[j-1]
      j=j-1
    A[j] = v
```

Projeto por Indução Fraca

Método da Inserção (iterativo)

- Usando a notação O, quantas comparações no arranjo são feitas no pior caso?

```
Inserção(A, n):
  Entrada: Arranjo A de n valores
  Saída: Arranjo A ordenado

  para i = 1 até n - 1 faça:
    v = A[i]
    j=i
    enquanto (j > 0) e (A[j-1] > v) faça:
      A[j] = A[j-1]
      j=j-1
    A[j] = v
```

- $O(n)$

Projeto por Indução Fraca

Método da Inserção (iterativo)

- Usando a notação O, quantas comparações no arranjo são feitas no pior caso?
- $O(n) \times O(n)$

```
Inserção(A, n):
  Entrada: Arranjo A de n valores
  Saída: Arranjo A ordenado

  para i = 1 até n - 1 faça:
    v = A[i]
    j=i
    enquanto (j > 0) e (A[j-1] > v) faça:
      A[j] = A[j-1]
      j=j-1
    A[j] = v
```

Projeto por Indução Fraca

Método da Inserção (iterativo)

- Usando a notação O, quantas comparações no arranjo são feitas no pior caso?

```
Inserção(A, n):
  Entrada: Arranjo A de n valores
  Saída: Arranjo A ordenado

  para i = 1 até n - 1 faça:
    v = A[i]
    j=i
    enquanto (j > 0) e (A[j-1] > v) faça:
      A[j] = A[j-1]
      j=j-1
    A[j] = v
```

- $O(n) \times O(n) = O(n^2)$

Projeto por Indução Fraca

Método da Inserção (iterativo)

- Usando a notação O, quantas comparações no arranjo são feitas no pior caso?

```
Inserção(A, n):
  Entrada: Arranjo A de n valores
  Saída: Arranjo A ordenado

  para i = 1 até n - 1 faça:
    v = A[i]
    j=i
    enquanto (j > 0) e (A[j-1] > v) faça:
      A[j] = A[j-1]
      j=j-1
    A[j] = v
```

- $O(n) \times O(n) = O(n^2)$ Bem mais fácil!!!

Projeto por Indução Fraca

Método da Inserção (iterativo)

- E quantas <u>trocas de elementos</u> são feitas no arranjo no pior caso?

```
Inserção(A, n):
  Entrada: Arranjo A de n valores
  Saída: Arranjo A ordenado

  para i = 1 até n - 1 faça:
    v = A[i]
    j=i
    enquanto (j > 0) e (A[j-1] > v) faça:
      A[j] = A[j-1]
      j=j-1
    A[j] = v
```

Projeto por Indução Fraca

Método da Inserção (iterativo)

- E quantas trocas de elementos são feitas no arranjo no pior caso?

```
Inserção(A, n):
  Entrada: Arranjo A de n valores
  Saída: Arranjo A ordenado

  para i = 1 até n - 1 faça:
    v = A[i]
    j=i
    enquanto (j > 0) e (A[j-1] > v) faça:
      A[j] = A[j-1]
      j=j-1
    A[j] = v
```

- $O(n)$

Projeto por Indução Fraca

Método da Inserção (iterativo)

- E quantas <u>trocas de elementos</u> são feitas no arranjo no pior caso?

```
Inserção(A, n):
  Entrada: Arranjo A de n valores
  Saída: Arranjo A ordenado

  para i = 1 até n - 1 faça:
    v = A[i]
    j=i
    enquanto (j > 0) e (A[j-1] > v) faça:
      A[j] = A[j-1]
      j=j-1
    A[j] = v
```

- $O(n) + O(n)$

Projeto por Indução Fraca

Método da Inserção (iterativo)

- E quantas trocas de elementos são feitas no arranjo no pior caso?

```
Inserção(A, n):
  Entrada: Arranjo A de n valores
  Saída: Arranjo A ordenado

  para i = 1 até n - 1 faça:
    v = A[i]
    j=i
    enquanto (j > 0) e (A[j-1] > v) faça:
      A[j] = A[j-1]
      j=j-1
    A[j] = v
```

- $O(n) + O(n) \times O(n)$

Projeto por Indução Fraca

Método da Inserção (iterativo)

- E quantas trocas de elementos são feitas no arranjo no pior caso?

```
Inserção(A, n):
  Entrada: Arranjo A de n valores
  Saída: Arranjo A ordenado

  para i = 1 até n - 1 faça:
    v = A[i]
    j=i
    enquanto (j > 0) e (A[j-1] > v) faça:
      A[j] = A[j-1]
      j=j-1
    A[j] = v
```

- $O(n) + O(n) \times O(n) = O(n^2)$

Projeto por Indução Fraca

Método da Inserção (iterativo)

- E quantas trocas de elementos são feitas no arranjo no pior caso?

```
Inserção(A, n):
  Entrada: Arranjo A de n valores
  Saída: Arranjo A ordenado

  para i = 1 até n - 1 faça:
    v = A[i]
    j=i
    enquanto (j > 0) e (A[j-1] > v) faça:
      A[j] = A[j-1]
      j=j-1
    A[j] = v
```

- $O(n) + O(n) \times O(n) = O(n^2)$ O mesmo valor.

Projeto por Indução Fraca

Método da Inserção: recursivo × iterativo

```
Inserção(A, n):
  se n == 1: retorna
  senão:
    Inserção(A, n - 1)
    v = A[n-1]
    j=n-1
    enquanto (j > 0) e
            (A[j-1] > v) faça:
      A[j] = A[j-1]
      j=j-1
    A[j] = v
```

```
Inserção(A, n):
  para i = 1 até n - 1 faça:
    v = A[i]
    j=i
    enquanto (j > 0) e
              (A[j-1] > v) faça:
      A[j] = A[j-1]
      j=j-1
    A[j] = v
```

Projeto por Indução Fraca

Segunda Alternativa

Projeto por Indução Fraca

Segunda Alternativa

- **Base**: $n = 1$. Um único elemento está ordenado

Projeto por Indução Fraca

Segunda Alternativa

- **Base**: $n = 1$. Um único elemento está ordenado
- **H.I.**: Sei ordenar um conjunto de $n - 1 \geq 1$ valores

Projeto por Indução Fraca

Segunda Alternativa

- **Base**: $n = 1$. Um único elemento está ordenado
- **H.I.**: Sei ordenar um conjunto de $n - 1 \geq 1$ valores
- **Passo**:

Projeto por Indução Fraca

Segunda Alternativa

- **Base**: $n = 1$. Um único elemento está ordenado
- **H.I.**: Sei ordenar um conjunto de $n - 1 \geq 1$ valores
- **Passo**:
  - Seja S um conjunto de $n \geq 2$ valores, e x o menor elemento de S

Projeto por Indução Fraca

Segunda Alternativa

- **Base**: $n = 1$. Um único elemento está ordenado
- **H.I.**: Sei ordenar um conjunto de $n - 1 \geq 1$ valores
- **Passo**:
  - Seja S um conjunto de $n \geq 2$ valores, e x o menor elemento de S
  - Então x certamente é o primeiro elemento da sequência ordenada de S. Por hipótese de indução, sabemos ordenar os demais $S - x$ elementos, e assim obtemos S ordenado

Projeto por Indução Fraca

Segunda Alternativa

- **Base**: $n = 1$. Um único elemento está ordenado
- **H.I.**: Sei ordenar um conjunto de $n - 1 \geq 1$ valores
- **Passo**:
  - Seja S um conjunto de $n \geq 2$ valores, e x o menor elemento de S
  - Então x certamente é o primeiro elemento da sequência ordenada de S. Por hipótese de indução, sabemos ordenar os demais $S - x$ elementos, e assim obtemos S ordenado
- Método da Seleção (*Selection Sort*)

Projeto por Indução Fraca

Método da Seleção (recursivo)

```
Seleção(A, ini, fim):
  Entrada: Arranjo A de n valores e os índices
        de início e término da sequência a ser ordenada
  Saída: Arranjo A ordenado

  se ini < fim então:
    min = ini
    para j = ini+1 até fim faça:
      se A[j] < A[min] então: min = j
    t = A[min]
    A[min] = A[ini]
    A[ini] = t
    Seleção(A, ini+1, fim)
```

Projeto por Indução Fraca

Método da Seleção (recursivo)

- Quantas comparações no arranjo são feitas no pior caso?

```
Seleção(A, ini, fim):
  se ini < fim então:
    min = ini
    para j = ini+1 até fim faça:
      se A[j] < A[min] então: min = j
    t = A[min]
    A[min] = A[ini]
    A[ini] = t
    Seleção(A, ini+1, fim)
```

Projeto por Indução Fraca

Método da Seleção (recursivo)

- Quantas comparações no arranjo são feitas no pior caso?

```
Seleção(A, ini, fim):
  se ini < fim então:
    min = ini
    para j = ini+1 até fim faça:
      se A[j] < A[min] então: min = j
    t = A[min]
    A[min] = A[ini]
    A[ini] = t
    Seleção(A, ini+1, fim)
```

$$T(n) = \begin{cases} & \text{se } n = 1 \\ & \text{para } n \geq 2 \end{cases}$$

Método da Seleção (recursivo)

- Quantas comparações no arranjo são feitas no pior caso?

```
Seleção(A, ini, fim):
  se ini < fim então:
    min = ini
    para j = ini+1 até fim faça:
      se A[j] < A[min] então: min = j
    t = A[min]
    A[min] = A[ini]
    A[ini] = t
    Seleção(A, ini+1, fim)
```

$$T(n) = \begin{cases} O(1) & \text{se } n = 1 \\ & \text{para } n \geq 2 \end{cases}$$

Projeto por Indução Fraca

Método da Seleção (recursivo)

- Quantas comparações no arranjo são feitas no pior caso?

```
Seleção(A, ini, fim):
  se ini < fim então:
    min = ini
    para j = ini+1 até fim faça:
      se A[j] < A[min] então: min = j
    t = A[min]
    A[min] = A[ini]
    A[ini] = t
    Seleção(A, ini+1, fim)
```

$$T(n) = \begin{cases} O(1) & \text{se } n = 1 \\ O(n) & \text{para } n \geq 2 \end{cases}$$

Projeto por Indução Fraca

Método da Seleção (recursivo)

- Quantas comparações no arranjo são feitas no pior caso?

```
Seleção(A, ini, fim):
  se ini < fim então:
    min = ini
    para j = ini+1 até fim faça:
      se A[j] < A[min] então: min = j
    t = A[min]
    A[min] = A[ini]
    A[ini] = t
    Seleção(A, ini+1, fim)
```

$$T(n) = \begin{cases} O(1) & \text{se } n = 1 \\ O(n) + T(n-1) & \text{para } n \geq 2 \end{cases}$$

Projeto por Indução Fraca

Método da Seleção (recursivo)

- Quantas comparações no arranjo são feitas no pior caso?

```
Seleção(A, ini, fim):
  se ini < fim então:
    min = ini
    para j = ini+1 até fim faça:
      se A[j] < A[min] então: min = j
    t = A[min]
    A[min] = A[ini]
    A[ini] = t
    Seleção(A, ini+1, fim)
```

$$T(n) = \begin{cases} O(1) & \text{se } n = 1 \\ O(n) + T(n-1) & \text{para } n \geq 2 \end{cases} = O(n^2)$$

Projeto por Indução Fraca

Método da Seleção (iterativo)

```
Seleção(A, n):
  Entrada: Arranjo A de n valores
  Saída: Arranjo A ordenado

  para i = 0 até n-2 faça:
    min = i
    para j = i+1 até n-1 faça:
      se A[j] < A[min] então: min = j
    t = A[min]
    A[min] = A[i]
    A[i] = t
```

Projeto por Indução Fraca

Método da Seleção (iterativo)

- Usando a notação O, quantas comparações no arranjo são feitas no pior caso?

```
Seleção(A, n):
  Entrada: Arranjo A de n valores
  Saída: Arranjo A ordenado

  para i = 0 até n-2 faça:
    min = i
    para j = i+1 até n-1 faça:
      se A[j] < A[min] então: min = j
    t = A[min]
    A[min] = A[i]
    A[i] = t
```

Projeto por Indução Fraca

Método da Seleção (iterativo)

- Usando a notação O, quantas comparações no arranjo são feitas no pior caso?

```
Seleção(A, n):
  Entrada: Arranjo A de n valores
  Saída: Arranjo A ordenado

  para i = 0 até n-2 faça:
    min = i
    para j = i+1 até n-1 faça:
      se A[j] < A[min] então: min = j
    t = A[min]
    A[min] = A[i]
    A[i] = t
```

- $O(n)$

Projeto por Indução Fraca

Método da Seleção (iterativo)

- Usando a notação O, quantas comparações no arranjo são feitas no pior caso?

```
Seleção(A, n):
  Entrada: Arranjo A de n valores
  Saída: Arranjo A ordenado

  para i = 0 até n-2 faça:
    min = i
    para j = i+1 até n-1 faça:
      se A[j] < A[min] então: min = j
    t = A[min]
    A[min] = A[i]
    A[i] = t
```

- $O(n) \times O(n)$

Projeto por Indução Fraca

Método da Seleção (iterativo)

- Usando a notação O, quantas comparações no arranjo são feitas no pior caso?

```
Seleção(A, n):
  Entrada: Arranjo A de n valores
  Saída: Arranjo A ordenado

  para i = 0 até n-2 faça:
    min = i
    para j = i+1 até n-1 faça:
      se A[j] < A[min] então: min = j
    t = A[min]
    A[min] = A[i]
    A[i] = t
```

- $O(n) \times O(n) = O(n^2)$

Projeto por Indução Fraca

Método da Seleção (iterativo)

- E quantas trocas de elementos são feitas no arranjo no pior caso?

```
Seleção(A, n):
  Entrada: Arranjo A de n valores
  Saída: Arranjo A ordenado

  para i = 0 até n-2 faça:
    min = i
    para j = i+1 até n-1 faça:
      se A[j] < A[min] então: min = j
    t = A[min]
    A[min] = A[i]
    A[i] = t
```

Projeto por Indução Fraca

Método da Seleção (iterativo)

- E quantas trocas de elementos são feitas no arranjo no pior caso?

```
Seleção(A, n):
  Entrada: Arranjo A de n valores
  Saída: Arranjo A ordenado

  para i = 0 até n-2 faça:
    min = i
    para j = i+1 até n-1 faça:
      se A[j] < A[min] então: min = j
    t = A[min]
    A[min] = A[i]
    A[i] = t
```

- $O(n)$

Projeto por Indução Fraca

Método da Seleção: recursivo × iterativo

```
Seleção(A, ini, fim):
  se ini < fim então:
    min = ini
    para j=ini+1 até fim faça:
      se A[j] < A[min] então:
        min = j
    t = A[min]
    A[min] = A[ini]
    A[ini] = t
    Seleção(A, ini+1, fim)
```

```
Seleção(A, n):
  para i = 0 até n-2 faça:
    min = i
    para j = i+1 até n-1 faça:
      se A[j] < A[min] então:
        min = j
    t = A[min]
    A[min] = A[i]
    A[i] = t
```

Projeto por Indução Fraca

Inserção × Seleção

- Inserção:
- Seleção:

Projeto por Indução Fraca

Inserção × Seleção

- Inserção:
  - Comparações: $O(n^2)$
- Seleção:

Projeto por Indução Fraca

Inserção × Seleção

- Inserção:
  - Comparações: $O(n^2)$
- Seleção:
  - Comparações: $O(n^2)$

Projeto por Indução Fraca

Inserção × Seleção

- Inserção:
  - Comparações: $O(n^2)$
  - Trocas: $O(n^2)$
- Seleção:
  - Comparações: $O(n^2)$

Projeto por Indução Fraca

Inserção × Seleção

- Inserção:
  - Comparações: $O(n^2)$
  - Trocas: $O(n^2)$
- Seleção:
  - Comparações: $O(n^2)$
  - Trocas: $O(n)$

Projeto por Indução Fraca

Inserção × Seleção

- Inserção:
  - Comparações: $O(n^2)$
  - Trocas: $O(n^2)$
- Seleção:
  - Comparações: $O(n^2)$
  - Trocas: $O(n)$

- Apesar dos algoritmos de ordenação por Inserção e Seleção terem a mesma complexidade assintótica em comparações, em situações onde a operação de troca é muito custosa é preferível utilizar a Seleção

Referências

- Material baseado em slides dos professores Cid de Souza, Cândida da Silva e Delano Beder
- Cormen, Thomas H., Leiserson, Charles E., Rivest, Ronald L., Stein, Clifford. Introduction to Algorithms. 2a ed. MIT Press, 2001.