"42 é a resposta para a questão da Vida, do Universo e do Todo".

Insertion Sort - Análise

Paulo Ricardo Lisboa de Almeida

DSBD

Alguns conteúdos e exemplos foram baseados nas aulas do Professor Renato Carmo - www.inf.ufpr.br/renato

UFPR
UNIVERSIDADE FEDERAL DO PARANÁ

Insertion Sort

```
função insertionSort (v,a,b)
entrada: vetor v, indexado por [a..b]
saída: o vetor v modificado de forma que v[i] ≤ v[j], ∀ i,j ∈ [a..b], e i < j.
se a ≥ b
     retorne v
insertionSort(v,a,b-1)
inserir(v,a,b)
retorne v
```

```
função inserir (v,a,b)
entrada: vetor v, indexado por [a..b], em que (v,a,b-1) é um vetor ordenado
saída: o vetor v modificado de forma que v[i] ≤ v[j], ∀ i,j ∈ [a..b], e i < j.
p ← buscar(v[b],v,a,b-1)
i ← b
enquanto i > p + 1
     trocar(v, i, i - 1)
     i ← i - 1
retorne v
```

```
função trocar (v,a,b)
entrada: vetor v, indexado por [a..b]
saída: o vetor v modificado de forma que v[a] é trocado com v[b]
x ← v[a]
v[a] ← v[b]
v[b] ← x
```

Análise

Considerando o número de comparações entre elementos do vetor, e *n* o tamanho do vetor

$$C(n) = \ldots$$

```
função insertionSort (v,a,b)
se a ≥ b
      retorne v
insertionSort(v,a,b-1)
inserir(v,a,b)
retorne v
```

Análise

Considerando o número de comparações entre elementos do vetor, e *n* o tamanho do vetor

$$C(n) = \begin{cases} 0, & \text{se } n \leq 1, \\ C(n-1) + C_i(n), & \text{se } n > 1 \end{cases}$$

```
função insertionSort (v,a,b)
se a ≥ b
      retorne v
insertionSort(v,a,b-1)
inserir(v,a,b)
retorne v
```

Pior Caso

Considerando o número de comparações entre elementos do vetor, e *n* o tamanho do vetor

$$C^+(n) = \begin{cases} 0, & \text{se } n \leq 1, \\ C^+(n-1) + C_i^+(n), & \text{se } n > 1 \end{cases}$$

```
função insertionSort (v,a,b)
se a ≥ b
      retorne v
insertionSort(v,a,b-1)
inserir(v,a,b)
retorne v
```

Resolvendo a recorrência

$$C^+(n) = \begin{cases} 0, & \text{se } n \leq 1, \\ C^+(n-1) + C_i^+(n), & \text{se } n > 1 \end{cases}$$

$$\begin{aligned} C^+(n) &= C^+(n-1) + C_i^+(n) = C^+(n-2) + C_i^+(n-1) + C_i^+(n) \\ &= C^+(n-3) + C_i^+(n-2) + C_i^+(n-1) + C_i^+(n) = \ldots \end{aligned}$$

Resolvendo a recorrência

$$C^+(n) = \begin{cases} 0, & \text{se } n \leq 1, \\ C^+(n-1) + C_i^+(n), & \text{se } n > 1 \end{cases}$$

$$C^+(n) = C^+(n-1) + C_i^+(n) = C^+(n-2) + C_i^+(n-1) + C_i^+(n)$$
$$= C^+(n-3) + C_i^+(n-2) + C_i^+(n-1) + C_i^+(n) = \ldots$$

Somamos todos os C^+_i, partindo de n - ????, e terminando em n.

$$= C^+(n-\mu) + \ldots$$

Resolvendo a recorrência

$$C^+(n) = \begin{cases} 0, & \text{se } n \leq 1, \\ C^+(n-1) + C_i^+(n), & \text{se } n > 1 \end{cases}$$

$$C^+(n) = C^+(n-1) + C_i^+(n) = C^+(n-2) + C_i^+(n-1) + C_i^+(n)$$
$$= C^+(n-3) + C_i^+(n-2) + C_i^+(n-1) + C_i^+(n) = \ldots$$

Somamos todos os C^+_i, partindo de $n - \mu + 1$, e terminando em n.

$$= C^+(n-\mu) + \ldots$$

Resolvendo a recorrência

$$C^+(n) = \begin{cases} 0, & \text{se } n \leq 1, \\ C^+(n-1) + C_i^+(n), & \text{se } n > 1 \end{cases}$$

$$\begin{aligned} C^+(n) &= C^+(n-1) + C_i^+(n) = C^+(n-2) + C_i^+(n-1) + C_i^+(n) \\ &= C^+(n-3) + C_i^+(n-2) + C_i^+(n-1) + C_i^+(n) = \dots \\ &= C^+(n-\mu) + \sum_{i=n-\mu+1}^{n} C_i^+(i) \end{aligned}$$

Resolvendo a recorrência

$$C^+(n) = \begin{cases} 0, & \text{se } n \leq 1, \\ C^+(n-1) + C_i^+(n), & \text{se } n > 1 \end{cases}$$

$$\begin{aligned} C^+(n) &= C^+(n-1) + C_i^+(n) = C^+(n-2) + C_i^+(n-1) + C_i^+(n) \\ &= C^+(n-3) + C_i^+(n-2) + C_i^+(n-1) + C_i^+(n) = \dots \\ &= C^+(n-\mu) + \sum_{i=n-\mu+1}^{n} C_i^+(i) \end{aligned}$$

$$n - \mu = 1 \Leftrightarrow \mu = n - 1$$

Resolvendo a recorrência

$$C^+(n) = \begin{cases} 0, & \text{se } n \leq 1, \\ C^+(n-1) + C_i^+(n), & \text{se } n > 1 \end{cases}$$

$$C^+(n) = C^+(n-\mu) + \sum_{i=n-\mu+1}^{n} C_i^+(i) \qquad n - \mu = 1 \Leftrightarrow \mu = n - 1$$

$$C^+(n) = C^+(n-n+1) + \sum_{i=n-(n-1)+1}^{n} C_i^+(i) = C^+(1) + \sum_{i=2}^{n} C_i^+(i)$$

$$C^+(n) = \sum_{i=2}^{n} C_i^+(i)$$

Melhor Caso

De maneira análoga

$$C^-(n) = \begin{cases} 0, & \text{se } n \leq 1, \\ C^-(n-1) + C_i^-(n), & \text{se } n > 1 \end{cases}$$

$$C^-(n) = \sum_{i=2}^{n} C_i^-(i)$$

Custo da inserção

Resta definir $C_I(n)$

```
função insertionSort (v,a,b)
se a ≥ b
      retorne v
insertionSort(v,a,b-1)
inserir(v,a,b)
retorne v
```

```
função inserir (v,a,b)
p ← buscar(v[b],v,a,b-1)
i ← b
enquanto i > p + 1
      trocar(v, i, i - 1)
      i ← i - 1
retorne v
```

Custo da inserção

Resta definir $C_I(n)$

Na inserção, as comparações são feitas em *buscar*

Agora depende de como a busca será realizada

```
função inserir (v,a,b)
p ← buscar(v[b],v,a,b-1)
i ← b
enquanto i > p + 1
      trocar(v, i, i - 1)
      i ← i - 1
retorne v
```

Inserção usando Busca Sequencial (Ingênua)

Das aulas passadas

$$C_B^+(n) = n$$
$$C_B^-(n) = 1$$

```
função inserir (v,a,b)
p ← buscar(v[b],v,a,b-1)
i ← b
enquanto i > p + 1
      trocar(v, i, i - 1)
      i ← i - 1
retorne v
```

Inserção usando Busca Sequencial (Ingênua)

Das aulas passadas

$$C_B^+(n) = n$$
$$C_B^-(n) = 1$$

Como na inserção apenas a busca faz comparações

$$C_I^+(n) = C_B^+(n) = n$$
$$C_I^-(n) = C_B^-(n) = 1$$

```
função inserir (v,a,b)
p ← buscar(v[b],v,a,b-1)
i ← b
enquanto i > p + 1
      trocar(v, i, i - 1)
      i ← i - 1
retorne v
```

Então

$$C^+(n) = \begin{cases} 0, & \text{se } n \leq 1, \\ C^+(n-1) + C_i^+(n), & \text{se } n > 1 \end{cases}$$

$$C^+(n) = \sum_{i=2}^{n} C_i^+(i) = \sum_{i=2}^{n} i$$

Resolvendo o somatório

Alguém lembra como resolver?

$$\sum_{i=2}^{n} i = ?$$

Resolvendo o somatório

Alguém lembra como resolver?

$$\sum_{i=2}^{n} i = ?$$

O somatório é uma Progressão Aritmética com uma razão 1

Progressão aritmética

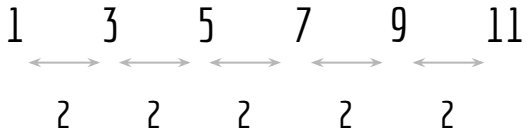
Considere o exemplo

1 3 5 7 9 11

Qual a razão?

Progressão aritmética

Considere o exemplo



Progressão aritmética

Considere o exemplo

1	1+1*2	1+2*2	1+3*2	1+4*2	1+5*2
1	3	5	7	9	11
11-5*2	11-4*2	11-3*2	11-2*2	11-1*2	11

Progressão aritmética

No caso geral, uma progressão

$a_1, a_2, \ldots, a_{n-1}, a_n$

Progressão aritmética

No caso geral, uma progressão

$a_1, a_2, \dots, a_{n-1}, a_n$

Pode ser descrita como

$a_1, \ a_1 + d, a_1 + 2d, \dots, a_1 + (n-2)d, a_1 + (n-1)d$

Progressão aritmética

No caso geral, uma progressão

$a_1, a_2, \dots, a_{n-1}, a_n$

Pode ser descrita como

$a_1, \ a_1 + d, a_1 + 2d, \dots, a_1 + (n-2)d, a_1 + (n-1)d$

Ou ainda, usando a_n

$a_n - (n-1)d, \ a_n - (n-2)d, \dots, a_n - 2d, a_n - d, a_n$

Somatório

O somatório da progressão é então

$S = a_1 + a_2 + \ldots + a_{n-1}, a_n$

Ou

$S = a_1 + (a_1 + d) + (a_1 + 2d) + \ldots + (a_1 + (n-2)d) + a_1 (+ (n-1)d)$

Somatório

O somatório da progressão é então

$S = a_1 + a_2 + \ldots + a_{n-1}, a_n$

Ou

$S = a_1 + (a_1 + d) + (a_1 + 2d) + \ldots + (a_1 + (n-2)d) + a_1 (+ (n-1)d)$

Ou usando a_n

$S = (a_n - (n-1)d) + (a_n - (n-2)d) + \ldots + (a_n - 2d) + (a_n - d) + a_n$

Somatório

Usando as duas equações para S, para computar S + S = 2S, temos

$S = a_1 + (a_1 + d) + (a_1 + 2d) + \ldots + (a_1 + (n-2)d) + a_1 (+ (n-1)d)$

$S = (a_n - (n-1)d) + (a_n - (n-2)d) + \ldots + (a_n - 2d) + (a_n - d) + a_n$

Somatório

Usando as duas equações para S, para computar S + S = 2S, temos

$S = a_1 + (a_1 + d) + (a_1 + 2d) + \ldots + (a_1 + (n-2)d) + a_1 (+ (n-1)d)$

$S = (a_n - (n-1)d) + (a_n - (n-2)d) + \ldots + (a_n - 2d) + (a_n - d) + a_n$

$2S = a_1 + (a_1 + d) + (a_1 + 2d) + \ldots + (a_1 + (n-2)d) + a_1 (+ (n-1)d) + (a_n - (n-1)d) + (a_n - (n-2)d) + \ldots + (a_n - 2d) + (a_n - d) + a_n$

Somatório

Usando as duas equações para S, para computar S + S = 2S, temos

$S = a_1 + (a_1 + d) + (a_1 + 2d) + \dots + (a_1 + (n-2)d) + a_1 (+ (n-1)d)$

$S = (a_n - (n-1)d) + (a_n - (n-2)d) + \dots + (a_n - 2d) + (a_n - d) + a_n$

$2S = a_1 + (a_1 + d) + (a_1 + 2d) + \dots + (a_1 + (n-2)d) + a_1 (+ (n-1)d) + (a_n - (n-1)d) + (a_n - (n-2)d) + \dots + (a_n - 2d) + (a_n - d) + a_n$

$2S = a_1 + (a_1 + \cancel{d}) + (a_1 + \cancel{2d}) + \dots + (a_1 + \cancel{(n-2)d}) + a_1 (+ \cancel{(n-1)d}) + (a_n - \cancel{(n-1)d}) + (a_n - \cancel{(n-2)d}) + \dots + (a_n - \cancel{2d}) + (a_n - \cancel{d}) + a_n$

$2S = na_1 + na_n$

$2S = n(a_1 + a_n)$

Somatório de Gaus

Dividindo-se os dois lados por 2, temos que

$$S = \frac{n(a_1 + a_n)}{2}$$

Que é a fórmula do somatório dos itens de uma P.A. qualquer.

Essa é uma generalização do **Somatório de Gauss**.

Substituindo

$$C^+(n) = \sum_{i=2}^{n} C_i^+(i) = \sum_{i=2}^{n} i$$

$$C^+(n) = \frac{(n-1)(2+n)}{2}$$

$$C^+(n) = \frac{n^2+n-2}{2}$$

$$C^+(n) \approx \frac{n^2}{2}$$

Melhor Caso

De maneira análoga

$$C^-(n) = \begin{cases} 0, & \text{se } n \leq 1, \\ C^-(n-1) + C_i^-(n), & \text{se } n > 1 \end{cases}$$

$$C_i^-(n) = C_B^-(n) = 1$$

$$C^-(n) = \sum_{i=2}^{n} C_i^-(i) = n - 1$$

$$C^-(n) \approx n$$

Teorema

Para toda instância *(v, a, b)* de tamanho *b - a + 1 = n > 0* do problema de Ordenação por Inserção, o número C(v, a, b) de comparações com elementos de *v* efetuadas utilizando o algoritmo de Busca Sequencial é

$$n - 1 \leq C(v, a, b) \leq \frac{n^2 + n - 2}{2}$$

Com busca binária

Podemos ainda usar a busca binária

O processo de resolução é o mesmo, mas com o complicador de que temos

$$C^{+}(n) = \sum_{i=2}^{n} \lfloor log_2 n \rfloor$$

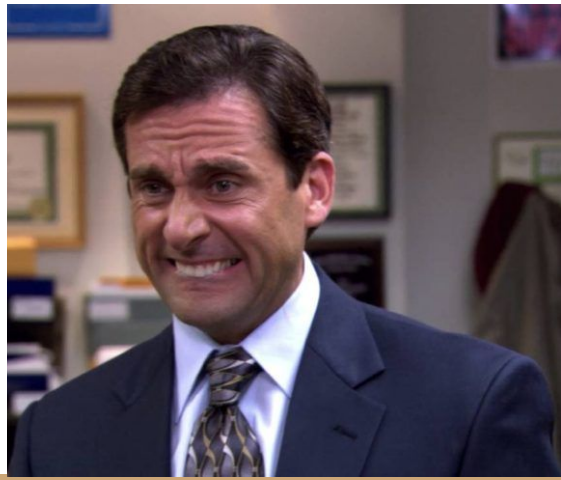
Com busca binária

Podemos ainda usar a busca binária

O processo de resolução é o mesmo, mas com o complicador de que temos

$$C^{+}(n) = \sum_{i=2}^{n} \lfloor log_2 n \rfloor$$

Somatório de pisos de logs de base 2



Com busca binária

Resolvendo (não vamos resolver o passo a passo na disciplina) temos:

Para toda instância *(v, a, b)* de tamanho *b - a + 1 = n > 0* do problema de Ordenação por Inserção, o número *C(v, a, b)* de comparações com elementos de *v* efetuadas utilizando o algoritmo de Busca Binária é

$$n\lfloor \log_2 n \rfloor - 2^{\lfloor \log_2 n \rfloor + 1} + \lfloor \log_2 n \rfloor - 1 \leq C(v, a, b) \leq n\lfloor \log_2 n \rfloor + n - 2^{\lfloor \log_2 n \rfloor + 1} + \lfloor \log_2 n \rfloor - 2$$

O que é aproximadamente

$$C(v, a, b,) \approx n\lfloor log_2 n \rfloor$$

Dica: É possível usar a fórmula de Stirling para aproximar o somatório.

Considerações Finais

O algoritmo insertion sort é relativamente simples, e é um bom algoritmo especialmente se o vetor está “quase ordenado”

Em um vetor onde cada elemento está a uma distância média de $k \in \mathbb{N}$ elementos da sua posição correta, o número de comparações é aproximadamente *k.n*

Veja uma discussão sobre isso em

T. Cormen. Desmistificando algoritmos. 2017.

T. Cormen, C. Leiserson, R. Rivest, C. Stein. Algoritmos: Teoria e Prática. 3a ed. 2012

Exercícios

1. Faça a análise para o algoritmo de Ordenação por Inserção com busca sequencial novamente, mas agora considerando o número de trocas de elementos do vetor.
    a. Mostre o número de trocas para o pior caso
    b. Mostre o número de trocas para o melhor caso

    Obs.: Resposta nos próximos slides

2. Quando considerando o número de trocas, o melhor caso muda quando usamos uma busca linear ou binária?
3. Quando considerando o número de trocas, o pior caso muda quando usamos uma busca linear ou binária?

Resposta exercício 1

Considere o número de trocas T(n), onde *n* é o tamanho do vetor

De maneira análoga a análise para o número de comparações, temos

$$T(n) = \begin{cases} 0, & \text{se } n \leq 1, \\ T(n-1) + T_i(n), & \text{se } n > 1 \end{cases}$$

```
função insertionSort (v,a,b)
se a ≥ b
      retorne v
insertionSort(v,a,b-1)
inserir(v,a,b)
retorne v
```

```
função inserir (v,a,b)
p ← buscar(v[b],v,a,b-1)
i ← b
enquanto i > p + 1
      trocar(v, i, i - 1)
      i ← i - 1
retorne v
```

Resposta exercício 1

$$T(n) = \begin{cases} 0, & \text{se } n \leq 1, \\ T(n-1) + T_i(n), & \text{se } n > 1 \end{cases}$$

De maneira similar a feita para a análise do número de comparações, temos que

$$T^+(n) = \sum_{i=2}^{n} T_i^+(i)$$

$$T^-(n) = \sum_{i=2}^{n} T_i^-(i)$$

```
função insertionSort (v,a,b)
se a ≥ b
      retorne v
insertionSort(v,a,b-1)
inserir(v,a,b)
retorne v
```

```
função inserir (v,a,b)
p ← buscar(v[b],v,a,b-1)
i ← b
enquanto i > p + 1
      trocar(v, i, i - 1)
      i ← i - 1
retorne v
```

Resposta exercício 1

Resta definir custo da inserção

As trocas são feitas dentro do loop

No melhor caso, nunca entramos no loop

Então $T^-_i(i) = 0$

$$T^-(n) = \sum_{i=2}^{n} T_i^-(i) = \sum_{i=2}^{n} 0 = 0$$

```
função inserir (v,a,b)
p ← buscar(v[b],v,a,b-1)
i ← b
enquanto i > p + 1
    trocar(v, i, i - 1)
    i ← i - 1
retorne v
```

Resposta exercício 1

No pior caso, o loop passa por *n-1* elementos

Então $T^+(i) = i-1$

$$T^+(n) = \sum_{i=2}^{n} T_i^+(i) = \sum_{i=2}^{n} (i-1)$$

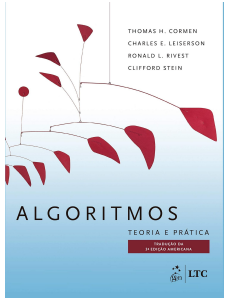
```
função inserir (v,a,b)
p ← buscar(v[b],v,a,b-1)
i ← b
enquanto i > p + 1
    trocar(v, i, i - 1)
    i ← i - 1
retorne v
```

O somatório pode ser visto como uma P.A. que começa em 1, e termina em n-1. Logo:

$$T^+(n) = \sum_{i=2}^{n} (i-1) = \frac{(n-1)(1+n-1)}{2} = \frac{(n-1)n}{2}$$

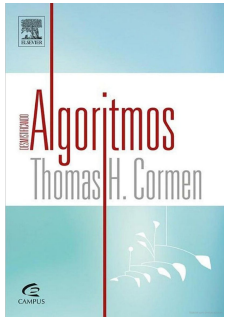
Referências

T. Cormen, C. Leiserson, R. Rivest, C. Stein. Algoritmos: Teoria e Prática. 3a ed. 2012

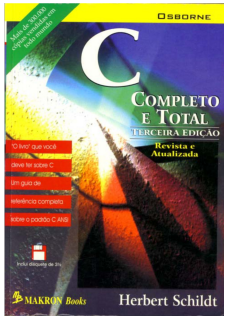


R. Sedgewick, K. Wayne. Algorithms Part I. 4a ed. 2014

T. Cormen. Desmistificando algoritmos. 2017.



H. Schildt. C completo e total. 1996



Licença