[ALGO] Complexité des Algorithmes (2)

Par Rhaeven, relecture par Find3r

Algorithmes de tris

Selection sort

i : + petits éléments triés	**n - i : + grands éléments non triés**

Exemple :

2 6 8 4 1 7 5 3	**i**
1 6 8 4 2 7 5 3	i = 0
1 2 8 4 6 7 5 3	i = 1
1 2 3 4 6 7 5 8	i = 2
1 2 3 4 6 7 5 8	i = 3
1 2 3 4 5 7 6 8	i = 4
1 2 3 4 5 6 7 8	i = 5
1 2 3 4 5 6 7 8	i = 6

```
SelectionSort(A, n) :
    for i <- 0 to n - 2
        posmin <- i
        for j <- i + 1 to n - 1
            if A[j] < A[posmin]
                posmin <- j
        A[posmin] <-> A[i]
```

> Ou **A** est un tableau de **n** entiers

Calcul de la complexité

Ligne de code	Nb d'exécution	Coût
for i <- 0 to n - 2	n-1	c1
posmin <- i	n-1	c2
for j <- i + 1 to n - 1	n(n-1)/2 *	c3
if A[j] < A[posmin]	n(n-1)/2 *	c4
posmin <- j	<= n(n-1)/2	c5
A[posmin] <-> A[i]	n-1	c6

> (*) $\sum_{i=0}^{n-2} \sum_{j=i+1}^{n-1} 1 = \sum_{i=0}^{n-2} n - i - 1 = \frac{(n-1+1)(n-1)}{2} = \frac{n(n-1)}{2}$

$$(n-1)(c1+c2+c6) + \frac{n(n-1)}{2}(c3+c4) \leq T(n) \leq (n-1)(c1+c2+c6) + \frac{n(n-1)}{2}(c3+c4+c5$$

$$an^2 + bn + c \leq T(n) \leq a'n^2 + b'n + c'$$

> Ou $an^2 + bn + c$ est le meilleur des cas et $a'n^2 + b'n + c$ le pire cas

La compléxite est quadratique dans tous les cas et on peut faire des expériences si on veut trouver les coefficients

Insertion sort

trie (i premières valeurs du tableau d'origine)	non trié

Exemple :

6 | 2 8 4 1 7 5 3	**i**
6 2 | 8 4 1 7 5 3	i = 1
2 6 | 8 4 1 7 5 3	
2 6 8 | 4 1 7 5 3	i = 2
2 6 8 4 | 1 7 5 3	i = 3
2 4 6 8 | 1 7 5 3	Cas general
2 4 6 8 1 | 7 5 3	i = 4
1 2 4 6 8 | 7 5 3	
1 2 4 6 8 7 | 5 3	i = 5
1 2 4 6 7 8 | 5 3	
1 2 4 6 7 8 5 | 3	i = 6
1 2 4 5 6 7 8 | 3	
1 2 3 4 5 6 7 8 |	i = 7

```
InsertionSort(A, n):
    for i <- 1 to n - 1
        key <- A[i]
        j <- i - 1
        while j >= 0 and A[j] > key
            A[j+1] <- A[j]
            j <- j - 1
        A[j + 1] <- key
```

Calcul de la complexité

Ligne de code	Nb d'exécution
for i <- 1 to n - 1	n - 1
key <- A[i]	n - 1
j <- i - 1	n - 1
while j >= 0 and A[j] > key	$\sum_{i=1}^{n-1} t_i + 1$
A[j+1] <- A[j]	$\sum_{i=1}^{n-1} t_i$
j <- j - 1	$\sum_{i=1}^{n-1} t_i$
A[j + 1] <- key	n - 1

> Soit t_i le nombre d'exécutions du corps du while pour un i donne

Dans le meilleur cas :
$t_i = 0$ (tableau trié)

$$\sum t_i = 0 \text{ et } \sum_{i=1}^{n-1} t_i + 1 = n - 1$$

T(n) est de la forme $an + b$

Dans le pire cas :
$t_i = i$ (Tableau à l'envers)

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

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

T(n) est de la forme $a'n^2 + b'n + c'$

Dans le cas moyen :
$t_i = i/2$
$\sum_{i=0}^{n-1} t_i = \frac{n(n-1)}{4}$
$\sum t_i + 1 = n + \frac{n(n-1}{4}$

T(n) est de la forme $a''n^2 + b''n + c''$
Ou $a'' \approx (1/2)a'$