CI 2 : Algorithmique & Programmation

Chapitre 4 – Introduction à la complexité

Savoir

Savoirs :
- s'interroger sur l'efficacité algorithmique temporelle.

1 Premier exemple

On introduit les algorithmes de tri suivant :

```python
#Tri par sélection
def tri_selection(tab):
    for i in range(0,len(tab)):
        indice = i
        for j in range(i+1,len(tab)):
            if tab[j]<tab[indice]:
                indice = j
        tab[i],tab[indice]=tab[indice],tab[i]
    return tab
```

```python
#Tri par insertion
def tri_insertion(tab):
    for i in range(1,len(tab)):
        a=tab[i]
        j=i-1
        while j>=0 and tab[j]>a:
            tab[j+1]=tab[j]
            j=j-1
        tab[j+1]=a
    return tab
```

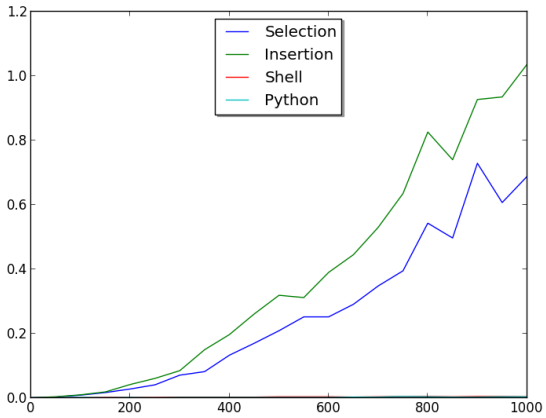
```python
def shellSort(array):
    "Shell sort using Shell's (original) gap sequence: n/2, n/4, ..., 1."
    "http://en.wikibooks.org/wiki/Algorithm_Implementation/Sorting/Shell_sort#Python"
    gap = len(array) // 2
    # loop over the gaps
    while gap > 0:
        # do the insertion sort
        for i in range(gap, len(array)):
            val = array[i]
            j = i
            while j >= gap and array[j - gap] > val:
                array[j] = array[j - gap]
```

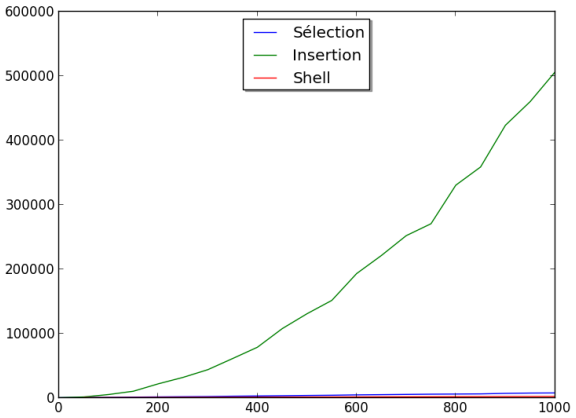
```python
            j -= gap
        array[j] = val
    gap //= 2
```

La figure ci-dessous montre le temps en secondes pour trier des tableaux de 1 à 1000 éléments en utilisant les méthodes de tri suivant :

- tri par sélection ;
- tri par insertion ;
- tri shell ;
- méthode de tri utilisée par Python.

Le premier graphe montre le temps de calcul et le second une estimation du nombre d'opérations.



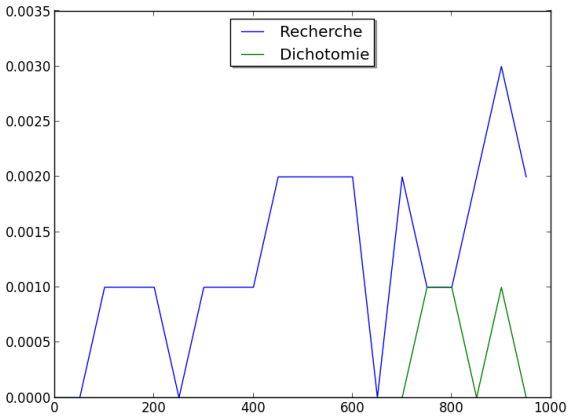


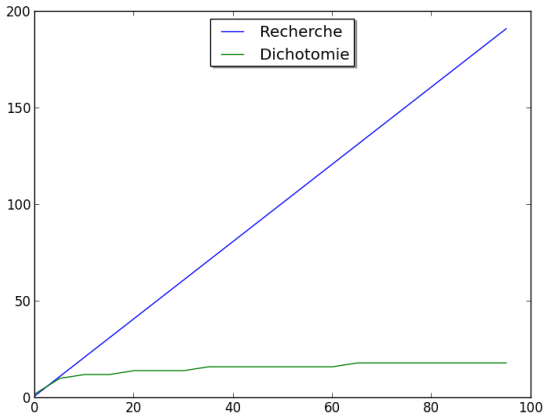
2 Deuxième exemple

On prend maintenant l'exemple de la recherche d'un élément dans une liste :

```python
def recherche(x,tab):
    for i in range(len(tab)):
        if tab[i]==x:
            return True
    return False
```

```python
def recherche_dichotomique(x, a):
    g, d = 0, len(a)-1
    while g <= d:
        m = (g + d) // 2
        if a[m] == x:
            return True
        if a[m] < x:
            g = m+1
        else:
            d = m-1
    return None
```


Recherche
Dichotomie
0.0035
0.0030
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
0
200
400
600
800
1000


Recherche
Dichotomie
200
150
100
50
0
0
20
40
60
80
100