TP5_correction

January 8, 2018

CORRECTION DU TP 5
Activité 1-2

```
In [1]: def dichot(T,x):    #algorithme de recherche de l'élément x dans le tableau T : on déter
            #ce-dernier par des indices gauches (g) et droits (d) qui se rapprochent de plus en
            g=0
            d=len(T)-1   #correspond à l'indice du dernier élément du tableau, le premier indice
            while g<=d:

                m=(g+d)//2 # on regarde l'indice "milieu" du tableau (le sens de "milieu" dépend
                if T[m]==x:
                    return m
                elif T[m]<x :
                    g=m+1       #Le tableau étant trié, si T[m]<x, c'est que l'indice correspond
                else :
                    d=m-1

            return False
```

```
In [2]: T1=[1,3,5,7,9]
        T2=[1,3,4,5,9,11]
```

```
In [3]: dichot(T1,5),dichot(T2,5)
```

```
Out[3]: (2, 3)
```

```
In [4]: dichot(T1,10),dichot(T2,10)
```

```
Out[4]: (False, False)
```

```
In [5]: dichot(T1,9),dichot(T2,11)
```

```
Out[5]: (4, 5)
```

L'algorithme semble fonctionner, mais il nous faut prouver deux choses :

1. *Sa terminaison :* Ceci est problématique en raison de la boucle while. Pour justifier, rigoureusement ce point, il nous faut exhiber un variant de boucle, c'est à dire une suite d'entiers strictement décroissante et positive (ce qui ne peut arriver indéfiniment, d'où la sortie de boucle). Il s'agit ici de la variable : $\boxed{d-g.}$ En effet :

- Ces deux nombres sont entiers car $m = \lfloor \frac{g+d}{2} \rfloor$ est entier et par somme d'entiers.
- Tant qu'on est dans la boucle, par définition de cette dernière : $g \geq d \Leftrightarrow \boxed{d - g \geq 0}$, d'où la positivité.
- si l'on note d' et g' les valeurs de g et d à l'itération d'après :
  - premier cas : $d' - g' = d - \lfloor \frac{d+g}{2} \rfloor - 1 = \rfloor$. Or, $\lfloor \frac{d+g}{2} \rfloor \geq \frac{d+g}{2} - 1$ donc : $d' - g' \leq d - \frac{d+g}{2} + 1 - 1 \Leftrightarrow d' - g' \leq \frac{d-g}{2}$.
  - deuxième cas : $d' - g' = \lfloor \frac{d+g}{2} \rfloor - 1 - g = \lfloor \frac{d-g-2}{2} \rfloor$. Or, $\lfloor \frac{d+g}{2} \rfloor \leq \frac{d+g}{2}$ donc : $d' - g' \leq \frac{d+g}{2} - 1 - g \Leftrightarrow d' - g' \leq \frac{d-g}{2} - 1 < \frac{d-g}{2}$.

  Or, $\frac{d-g}{2} < d - g$ car $d > g$. En effet, sinon $d = g$ car $d - g \geq 0$ et dans ce cas : $m = d = g$ et donc : $d' = d - 1$ et $g' = d + 1$ donc : $g' > d'$ ce qui est impossible car on est encore dans la boucle. Ainsi : $d' - g' < d - g$, ce qui prouve la stricte décroissance.

2. *Sa validité :* Pour être sûr que l'algorithme retourne bien ce que l'on veut, il nous faut un invariant de boucle, c'est à dire une propriété qui est toujours vraie à chaque itération de la boucle. Ici, lorque x appartient au tableau l'invariant en question est : $T[g] \leq x \leq T[d]$. En effet :
   - L'initialisation est vérifiée puisque : x appartient bien au tableau.
   - Si $T[g] \leq x \leq T[d]$, alors à l'itération d'après, nous distinguons trois cas :
     - Si $T[m] < x$, alors $x \geq T[m+1]$ donc $T[g'] \leq x$ car $g' = m + 1$ et $x \leq T[d]$ par hypothèse de récurrence.
     - Si $T[m] > x$, alors $x \leq T[m-1]$ donc $T[d'] \geq x$ car $d = m - 1$ et $x \geq T[g]$ par hypothèse de récurrence.
     - Si $T[m] = x$, alors rien ne change donc le résultat est OK par hypothèse de récurrence.

   Ainsi, lorsque la boucle se termine, nous avons $g \geq d$, donc :
     - cas 1 : $g > d$. Si x est initialement dans la liste, alors : $T[g] \leq x \leq T[d]$ et donc $g \leq d$. Ceci est impossible. On en déduit que x n'est pas dans la liste.
     - cas 2 : $g = d$, alors $T[g] = T[d] = x$ car $T[g] \leq x \leq T[d]$. Nous avons donc bien trouvé x, ce qui prouve que l'algorithme est correct.

Notons $d[k]$ (resp. $g[k]$)la valeur de d (resp. g) à la k ième itération de la boucle. Alors, à l'itération d'après de la boucle while, nous avons :

- cas 1 :$d[k+1] - g[k+1] = d[k] - \lfloor \frac{d[k]+g[k]}{2} \rfloor - 1$. D'après les propriétés de la partie entière, $\lfloor \frac{d[k]+g[k]}{2} \rfloor > \frac{d[k]+g[k]}{2} - 1$. Par conséquent : $d[k+1] - g[k+1] < \frac{d[k]-g[k]}{2}$.
- Cas 2 : $d[k+1] - g[k+1] = \lfloor \frac{d[k]+g[k]}{2} \rfloor - 1 - g[k]$. D'après les propriétés de la partie entière, $\lfloor \frac{d[k]+g[k]}{2} \rfloor \leq \frac{d[k]+g[k]}{2}$. Par conséquent : $d[k+1] - g[k+1] \leq \frac{d[k]-g[k]}{2} - 1 < \frac{d[k]-g[k]}{2}$.

Par récurrence, on en déduit au bout de k itérations de la boucle while : $d[k] - g[k] \leq \frac{d_0 - g_0}{2^k} = \frac{N}{2^k}$.

La complexité est estimée par le nombre d itérations de la boucle while. Notons k_0 son nombre d itérations. Sachant que pour cette valeur de k_0, le nombre recherché n a pas encore été trouvé, on sait que $d[k_0] \neq g[k_0]$ d'où : $d[k_0] - g[k_0]$ est un entier strictement positif donc supérieur ou égal à 1.

On en déduit :

$1 < \frac{N}{2^{k_0}}$ soit : $2^{k_0} < N$, et donc $k_0 \ln(2) < \ln(N)$. Ainsi, $k_0 < \frac{\ln(N)}{\ln(2)}$. Le nombre d itérations maximal est donc au pire égal eu logarithme décimal de la taille du tableau initial, ce qui prouve que la complexité au pire est de cet ordre de grandeur.

Activité 1-3

Q1 et Q4

```
In [6]: def dicho(f,a,b,precision):
            u=a
            v=b
            if f(a)==0:
                return a
            elif f(b)==0:
                return b
            else :
                while v-u>precision:
                    if f(u)*f((u+v)/2)==0:
                        return (u+v)/2
                    elif f(u)*f((u+v)/2)<0:
                       v=(u+v)/2
                    else :
                        u=(u+v)/2
                return u
```

Q2 Ici, on sait (cf cours de maths) que $v[n] - u[n] = \frac{b-a}{2^n}$. Ainsi, cette suite tend vers 0 donc sera strictement plus petite que "précision" à partir d'un certain rang, ce qui assure que l'algorithme se termine. Q3 A la sortie de la boucle while, nous avons donc $v - u \leq$ precision. Un invariant de boucle étant " $f(u_n) \leq f(v_n)$ ", nous savons que le zéro l de f est compris entre u et v. Ainsi, u est bien une valeur approché de l à la précison demandée puisque : $0 \leq l - u \leq v - u \leq$ precision.

Q5

```
In [7]: from math import sin,pi

        dicho(sin,4,10,10**(-3))

Out[7]: 6.282958984375

In [8]: print(2*pi)
```

```
6.283185307179586
```

Q6

```
In [9]: f=lambda x:x**(2)-2
        dicho(f,1,2,10**(-3))

Out[9]: 1.4140625
```

Activité 2

```
In [10]: def recherche_mot(m,T):
             for i in range(len(T)-len(m)+1):
                 j=0
                 while j<len(m) and m[j]==T[i+j]:
                     j+=1
                 if j==len(m):
                     return i
             return None

In [11]: T=[2,1,4,1,2,6,1,2,3,7]
         m=[1,2,3]
         recherche_mot(m,T)

Out[11]: 6
```