TD d'algorithmique avancée
Corrigé du TD 4 : recherche de l'élément majoritaire

Jean-Michel Dischler et Frédéric Vivien

Nous nous intéressons à un tableau A de n éléments, n étant supposé être une puissance de deux. Nous supposons également que la seule opération à notre disposition nous permet de vérifier si deux éléments sont ou non égaux. Un élément x de A est dit majoritaire si et seulement si A contient strictement plus de $n/2$ occurrences de x. Nous nous intéresserons à la complexité au pire.

Algorithme naïf

1. Écrivez un algorithme qui calcule le nombre d'occurrences d'une valeur x présentes entre les indices i et j d'un tableau A.

   Occurrences(x, A, i, j)
     compteur $\leftarrow 0$
     pour $k \leftarrow i$ **à** j **faire**
       si $A[k] = x$ **alors** *compteur* $\leftarrow$ *compteur* $+ 1$
     renvoyer *compteur*

2. Quelle est la complexité de cet algorithme ?
   La boucle exécute $j - i + 1$ itérations. La complexité de cet algorithme est donc en $\Theta(j - i)$.

3. Au moyen de l'algorithme précédent, écrivez un algorithme Majoritaire qui vérifie si un tableau A contient un élément majoritaire.

   Majoritaire(A)
     pour $i \leftarrow 1$ **à** *longueur*$(A)/2$ **faire**
       si Occurrences($A[i]$, A, i, *longueur*(A)) $>$ *longueur*$(A)/2$ **alors renvoyer** Vrai
     renvoyer Faux

4. Quelle est la complexité de cet algorithme ?
   Dans le pire cas, la boucle effectue $n/2$ itérations, chacune de ces itérations effectuant un appel à Occurrences *sur un tableau de taille $n - i$ (i variant de 1 à n) donc de coût $\Theta(n - i)$. Le coût total de l'algorithme est donc en $\Theta(n^2)$.*

Premier algorithme « diviser pour régner »

1. Proposez un algorithme Majoritaire construit suivant le paradigme « diviser pour régner ». Cet algorithme divisera en deux le tableau A sur lequel il travaille. Il renverra le couple (Vrai, x) si le tableau A contient un élément majoritaire (x étant cet élément) et renverra le couple (Faux, 0) si le tableau A ne contient pas d'élément majoritaire.

   Majoritaire(A, i, j)
     si $i = j$ **alors renvoyer** (Vrai, $A[i]$)
     $(r_x, x) \leftarrow$ Majoritaire(A, i, $\frac{i+j-1}{2}$)
     $(r_y, y) \leftarrow$ Majoritaire(A, $\frac{i+j+1}{2}$, j)
     si $r_x =$ Faux et $r_y =$ Faux **alors renvoyer** (Faux, 0)

si $r_x = \textsc{Vrai}$ et $r_y = \textsc{Vrai}$
  alors si $x = y$
    alors renvoyer ($\textsc{Vrai}$, x)
    sinon
      $c_x \leftarrow \textsc{Occurrences}(x, A, i, j)$
      $c_y \leftarrow \textsc{Occurrences}(y, A, i, j)$
      si $c_x > \frac{j-i+1}{2}$
        alors renvoyer ($\textsc{Vrai}$, x)
        sinon si $c_y > \frac{j-i+1}{2}$
          alors renvoyer ($\textsc{Vrai}$, y)
          sinon renvoyer ($\textsc{Faux}$, 0)
  sinon si $r_x = \textsc{Vrai}$
    alors si $\textsc{Occurrences}(x, A, i, j) > \frac{j-i+1}{2}$
      alors renvoyer ($\textsc{Vrai}$, x)
      sinon renvoyer ($\textsc{Faux}$, 0)
    sinon si $\textsc{Occurrences}(y, A, i, j) > \frac{j-i+1}{2}$
      alors renvoyer ($\textsc{Vrai}$, y)
      sinon renvoyer ($\textsc{Faux}$, 0)

Justifications

Les deux seuls cas qui ne sont peut-être pas immédiats sont les suivants :

(a) $r_x = \textsc{Faux}$ *et* $r_y = \textsc{Faux}$ *: dans ce cas il n'y a pas d'élément qui soit majoritaire dans la première moitié du tableau, ni d'élément qui soit majoritaire dans la deuxième moitié du tableau. Si le table contient n éléments, le nombre d'occurrences d'un élément quelconque dans la première moitié du tableau est donc inférieur ou égal à* $\frac{\frac{n}{2}}{2}$ *—la première moitié ayant* $\frac{n}{2}$ *éléments— et il en va de même pour le deuxième moitié. Donc le nombre d'occurences d'un élément quelconque dans le tableau est inférieur à* $\frac{n}{2}$ *et le tableau ne contient pas d'élément majoritaire.*

(b) $r_x = \textsc{Vrai}$ *et* $r_y = \textsc{Vrai}$ *avec* $x = y$ *: dans ce cas* x *est présent au moins* $1 + \frac{n}{4}$ *fois dans chacune des deux parties —qui sont de taille* $\frac{n}{2}$*— et donc au moins* $2 + \frac{n}{2}$ *fois dans le tableau.*

2. Quelle est la complexité de cet algorithme ?
   La complexité de cet algorithme est définie par la relation de récurrence :

$$T(n) = 2T\left(\frac{n}{2}\right) + \Theta(n).$$

En effet, la phase de combinaison nécessite, dans le pire des cas, la recherche du nombre d'occurences de deux éléments dans le tableau, ce qui a un coût de n, toutes les autres opérations étant de coût constant ($\Theta(1)$*).*
Nous avons donc ici : $a = 2$, $b = 2$ *et* $f(n) = \Theta(n) == \Theta(n^{\log_2 2})$*. Nous sommes donc dans le cas 2 du théorème et donc :*

$$T(n) = \Theta(n \log n).$$

Deuxième algorithme « diviser pour régner »

1. Écrivez un algorithme construit suivant le paradigme « diviser pour régner », prenant en entrée un tableau A —qu'il divisera en deux— et possédant la propriété suivante :
   – soit cet algorithme nous garantit que le tableau A ne contient pas d'élément majoritaire ;
   – soit cet algorithme nous renvoie un élément x et un entier $c_x > n/2$ tels que x apparaisse *au plus* c_x fois dans A et que tout autre élément de A apparaisse *au plus* $n - c_x$ fois dans A.

$\textsc{PseudoMajoritaire}(A, i, j)$
  si $i = j$ **alors renvoyer** ($\textsc{Vrai}$, $A[i]$, 1)

$(r_x, x, c_x) \leftarrow$ MAJORITAIRE$(A, i, \frac{i+j-1}{2})$
$(r_y, y, c_y) \leftarrow$ MAJORITAIRE$(A, \frac{i+j+1}{2}, j)$
si $r_x =$ FAUX et $r_y =$ FAUX **alors renvoyer** (FAUX, 0, 0)
si $r_x =$ VRAI et $r_y =$ FAUX **alors renvoyer** (VRAI, x, $c_x + \frac{j-i+1}{4}$)
si $r_x =$ FAUX et $r_y =$ VRAI **alors renvoyer** (VRAI, y, $c_y + \frac{j-i+1}{4}$)
si $r_x =$ VRAI et $r_y =$ VRAI
  alors si $x = y$
    alors renvoyer (VRAI, x, $c_x + c_y$)
    sinon si $c_x = c_y$
      alors renvoyer (FAUX, 0, 0)
      sinon si $c_x > c_y$
        alors renvoyer (VRAI, x, $\frac{j-i+1}{2} + c_x - c_y$)
        sinon renvoyer (VRAI, y, $\frac{j-i+1}{2} + c_y - c_x$)

Justifications

Nous considérons un par un les différents cas de figure :

- $r_x =$ FAUX *et* $r_y =$ FAUX. *Aucun élément n'apparaît strictement plus de $\frac{n}{4}$ fois dans la première (resp. la deuxième) moitié du tableau A. Donc un élément quelconque de A apparaît au plus $\frac{n}{4}$ fois dans chacune des deux moitiés, et donc $\frac{n}{2}$ fois en tout dans A. Donc A ne contient pas d'élément majoritaire.*
- $r_x =$ VRAI *et* $r_y =$ FAUX. *Un élément quelconque de A apparaît donc* au plus $\frac{n}{4}$ *fois dans la deuxième moitié de A. Nous avons deux cas à considérer :*
  - *x apparaît donc au plus $c_x + \frac{n}{4}$ fois dans A.*
  - *Un élément autre que x apparaît* au plus $\frac{n}{2} - c_x$ *fois dans la première moitié de A. Par conséquent un tel élément apparaît au plus $\left(\frac{n}{2} - c_x\right) + \frac{n}{4} = \frac{3n}{4} - c_x = n - \left(c_x + \frac{n}{4}\right)$ fois dans A.*

  D'où le résultat.
- $r_y =$ VRAI *et* $r_x =$ FAUX *: ce cas est symétrique du précédent.*
- $r_x =$ VRAI *et* $r_y =$ VRAI *:*
  - $x = y$. *x est présent* au plus $c_x + c_y$ *fois dans A. De plus, tout autre élément est présent* au plus $\frac{n}{2} - c_x$ *fois dans la première moitié de A et* $\frac{n}{2} - c_y$ *fois dans la deuxième moitié, soit en tout* au plus $n - (c_x + c_y)$ *fois dans A.*
  - $x \neq y$ *et* $c_x = c_y$. *x est présent* au plus c_x *fois dans la première moitié et* $\frac{n}{2} - c_y = \frac{n}{2} - c_x$ *fois dans la deuxième moitié, soit $\frac{n}{2}$ fois en tout et x n'est pas un élément majoritaire de A. Symétriquement, il en va de même de y. Tout autre élément ne peut être un élément majoritaire (voir le tout premier cas).*
  - $x \neq y$ *et* $c_x > c_y$. *Alors x est présent* au plus c_x *fois dans la première moitié de A et* $\frac{n}{2} - c_y$ *fois dans la deuxième moitié, soit* au plus $\frac{n}{2} + c_x - c_y$ *fois dans A, et ce nombre est strictement supérieur à $\frac{n}{2}$ car $c_x > c_y$. y est présent au plus $\frac{n}{2} + c_y - c_x = n - (\frac{n}{2} + c_x - c_y)$ fois dans A. Tout autre élément est présent au plus $\left(\frac{n}{2} - c_x\right) + \left(\frac{n}{2} - c_y\right) = n - c_x - c_y = \frac{n}{2} - c_x + \frac{n}{2} - c_y \leq \frac{n}{2} - c_x + c_y$ (car $c_y > \frac{n}{4}$) $= n - \left(\frac{n}{2} + c_x - c_y\right)$.*

2. Quelle est la complexité de cet algorithme ?

   En dehors des appels récursifs, tous les traitements ont un coût constant : $\Theta(1)$. La complexité de l'algorithme est donc donnée par la relation de récurrence :

$$T(n) = 2T\left(\frac{n}{2}\right) + \Theta(1).$$

   Nous nous trouvons donc ici dans le cas 1) du théorème (avec $\epsilon = 1$) et la complexité de l'algorithme est donc :

$$T(n) = \Theta(n).$$

3. À partir de l'algorithme précédent, écrivez un algorithme Majoritaire qui vérifie si un tableau A contient un élément majoritaire.

   Majoritaire(A)
   ($\textit{réponse}$, x, c_x) $\leftarrow$ PseudoMajoritaire(A, 1, $\textit{longueur}(A)$)
   si *réponse* = Faux
   alors renvoyer Faux
   sinon si Occurrences(x, A, 1, $\textit{longueur}(A)$) $> \frac{\textit{longueur}(A)}{2}$
   alors renvoyer Vrai
   sinon renvoyer Faux

4. Quelle est la complexité de cet algorithme ?
   La complexité de cet algorithme est en $\Theta(n)$ car c'est la complexité de l'appel à l'algorithme Pseudo-Majoritaire *et celle de l'appel à l'algorithme* Occurrences.