Chapitre 6. Diviser pour régner.

I Élément majoritaire

Le but de cet exercice est d'illustrer le soin avec lequel il faut parfois mener la réflexion pour l'étape de fusion dans la méthode diviser pour régner.

Dans un tableau de longueur n, un élément est dit majoritaire si et seulement si son occurrence (*i.e.* son nombre d'apparitions) est strictement supérieur à $n/2$. On se propose d'écrire une algorithme récursif utilisant la méthode « diviser pour régner » qui détermine l'élément majoritaire d'un tableau ainsi que son occurrence, s'il existe.

1.
   1-a Justifier brièvement l'unicité d'un élément majoritaire.

   1-b Donner le principe (pas de code) d'un algorithme simple qui répond à la question et préciser sa complexité en fonction de n.
2. Écrire une fonction `occurrence t x` de type 'a array -> 'a -> int qui donne l'occurrence dans un tableau `t` d'un objet `x`.
3. Étant donné un tableau t de longueur n, on le coupe en deux sous-tableaux t_1, t_2 de longueurs à peu près égales à $n/2$ et on suppose connaître dans chacun d'eux un élément majoritaire a_1, a_2 et son occurrence oc_1, oc_2 respectivement.

   3-a *Résultat préliminaire qui facilitera les démonstrations des questions suivantes.* Montrer que pour des entiers naturels a et b, et un entier strictement positif q on a l'encadrement
   $$a/q + b/q \leqslant (a + b)/q \leqslant a/q + b/q + 1,$$
   expression dans laquelle / désigne la division entière.

   3-b Montrer que si t a un élément majoritaire, alors ce ne peut être que a_1 ou a_2. On détaillera complètement le raisonnement...

   3-c Que peut-on conclure si $a_1 = a_2$?

   3-d Dans le cas contraire, comment savoir si l'un des deux est majoritaire dans t ?
4.
   4-a Et que dire si l'un des deux tableaux t_1, t_2 a un élément majoritaire et l'autre non ?

   4-b Et si aucun des deux n'a d'élément majoritaire ?
5. Écrire une fonction récursive `majoritaire t` de type 'a array -> 'a * int , qui donne comme résultat le couple `(x,0)` (où `x` est un élément quelconque du tableau) si le tableau `t` n'a pas d'élément majoritaire (ce qui est codé par le 0 du second élément du couple) ou le couple `(x,oc)` où `x` est l'élément majoritaire et `oc` son occurrence. On supposera (sans le vérifier) que le tableau `t` est non vide.

   Pour l'étape de division on pourra utiliser la fonction `Array.sub t i k` qui rend un nouveau tableau contenant dans cet ordre les k éléments à partir de celui d'indice i (inclus).
6. Quelle est la complexité d'un tel algorithme ?

Remarque finale : On peut faire encore mieux. Robert S. Boyer and J Strother Moore ont présenté en 1981 un algorithme permettant de déterminer s'il y a un élément majoritaire et lequel, le cas échéant, en $O(n)$.