Problème : un algorithme « diviser pour régner »

Problème : Recherche de pic

Dans un tableau, on dit qu'un élément est un pic s'il est supérieur ou égal à ses deux voisins (ou de son unique voisin si il est à une extrémité). Le maximum d'une liste est nécessairement un pic, mais il peut y en avoir d'autres : par exemple pour la liste `[|2; 1; 3; 3; 5; 7; 0|]`, les indices des pics sont 0, 2 et 5. Cette notion s'étend à des tableaux à deux dimensions : il faut que l'élément soit supérieur ou égal à ses 4 plus proches voisins (ou 3 s'il est sur le bord, 2 s'il est dans un coin).

1. Recherche des pics

Question 1. Donner une fonction `indices_pics t` prenant en entrée un tableau `t` (qu'on peut supposer posséder au moins deux éléments) à une dimension et renvoyant la liste des indices de ses pics.

On cherche maintenant un algorithme cherchant l'indice *d'un seul pic* avec une complexité plus efficace que $O(N)$ (en dimension 1) ou $O(N \times M)$ (en dimension 2, avec N et M les dimensions de la matrice).

2. Un seul pic en dimension 1

Supposons que l'indice i du tableau ne soit pas l'indice d'un pic. Alors, s'ils sont bien définis, on a soit `t.(i)` < `t.(i-1)`, soit `t.(i)` < `t.(i+1)` (soit les deux).

Question 2. Supposons que `t.(i)` < `t.(i-1)`. Montrer qu'il existe un pic d'indice $j \leq i - 1$.

Question 3. En déduire un algorithme `cherche_pic t` s'inspirant de la recherche dichotomique donnant l'indice d'un pic, avec une complexité $O(\log n)$ où n est la taille du tableau. *Attention à ne pas faire de dépassement d'indice !*

3. Un seul pic en dimension supérieure

On se donne un tableau à deux dimensions (N lignes, M colonnes). On propose l'algorithme suivant (récursif) pour trouver un pic en dimension 2 :

— Chercher le maximum global dans la ligne centrale ;
— Si c'est un pic, le renvoyer ;
— Sinon appeler l'algorithme récursivement sur les lignes au dessus (resp. dessous) si le voisin du dessus (resp. du dessous) est plus grand.

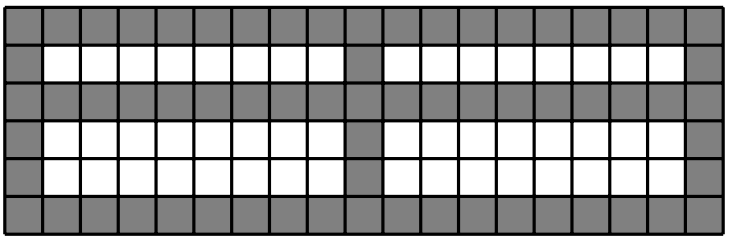
Question 4. Justifier que l'algorithme termine et est correct.

Question 5. Quelle est la complexité de l'algorithme ?

Question 6. Écrire une fonction `cherche_pic_2 t` prenant en entrée une matrice `t` (tableau de dimension 2, $N > 0$ lignes, $M > 0$ colonnes) et renvoyant le couple d'indices d'un pic. On pourra faire usage d'une fonction auxiliaire récursive. *Attention à ne pas faire de dépassement d'indice sur les lignes !* Remarque : `Array.sub t i k` extrait du tableau `t` les k éléments indexés à partir de i.

4. Un algorithme optimal en dimension 2

Question 7. On reprend la même idée que dans la section précédente pour le calcul d'un pic en dimension 2, mais au lieu de chercher le maximum sur la ligne centrale, on peut chercher le maximum sur un grand pourtour constitué d'une « croix centrale », et des bords comme dans la figure ci-dessous :



Expliciter comment obtenir un algorithme permettant de rechercher un pic, justifier sa correction, et estimer sa complexité (on ne demande pas de le coder) en fonction de N et M. On supposera que l'on ne fait pas du tout de recopie de portion de tableau : c'est possible en utilisant une fonction prenant en entrée des bornes délimitant les lignes et colonnes sur lesquelles on travaille.