Exercice 6 – Policiers et voleurs

Soit un tableau T indicé de 1 à n, contenant dans chaque case soit le caractère $'P'$ (pour policier), soit $'V'$ (pour voleur). Le problème étudié est le suivant : un policier peut capturer au plus un voleur, un policier ne peut capturer un voleur que si celui-ci se trouve à une distance inférieure ou égale au paramètre k dans le tableau. Une *capture* est un couple (i,j) d'indices tels que $T[i]='P'$, $T[j]='V'$ et $|i-j|\leq k$ (autrement dit, un policier est en case i, un voleur en case j, et le policier en case i est à une distance qui lui permet de capturer le voleur en case j).

On appelle *appariement* un ensemble de captures où chaque policier et chaque voleur apparaît au plus une fois. Par exemple :

- pour $T=['P','V','V','P','V']$ et $k=1$, l'appariement $\{(1,2),(4,5)\}$ conduit à 2 captures.
- pour $T=['P','V','P','V','V','P']$ et $k=2$, l'appariement $\{(1,2),(3,5),(6,4)\}$ conduit à 3 captures.

On cherche un algorithme qui, pour T et k fixés, permet d'obtenir un appariement avec un nombre maximum c_{max} de captures. Par exemple, dans les deux exemples précédents, on a respectivement $c_{max}=2$ et $c_{max}=3$ car les deux appariements sont optimaux.

Q 6.1 Soit n_P le nombre de caractères $'P'$ dans T, et n_V le nombre de caractères $'V'$ dans T. Donnez un exemple de T et k pour lesquels on vérifie à la fois $c_{max}\neq n_P$ et $c_{max}\neq n_V$.

P	P	P	V	V	V

$k=1 \rightarrow c_{max}=1$

On considère maintenant l'algorithme suivant : on parcourt le tableau dans l'ordre des indices croissants, et, pour chaque policier rencontré, ce dernier capture le voleur le plus proche possible n'ayant pas déjà été capturé par un autre policier (si un tel voleur existe).
Précision : Si deux voleurs non encore capturés sont à même plus proche distance, à gauche et à droite du policier, alors le policier capture le voleur à sa gauche.

Q 6.3 À quelle famille d'algorithme cet algorithme appartient-il ?

Q 6.4 Quelle est sa complexité temporelle en fonction de n ? Justifiez brièvement.

Q 6.5 Montrez que cet algorithme n'est pas optimal.

3) C'est un algorithme glouton.

4) On parcourt une seule fois le tableau et, pour chaque policier rencontré, il y a au plus $2k$ cellules du tableau à examiner. Donc $O(nk)$. $O(n^2)$ est vrai également car $k \leq n-1$.

5)

1	2	3	4
V	V	P	P

$k=2$. Ici, l'algorithme retourne la seule capture $\{(3,2)\}$, alors qu'on aurait pu faire $\{(3,1),(4,2)\}$.

Q 6.6 On appelle Suivant_P(T,i) la fonction retournant le premier indice $x > i$ tel que $T[x] = 'P'$ s'il existe, et $n+1$ sinon. On appelle Suivant_V(T,j) la fonction retournant le premier indice $y > j$ tel que $T[y] = 'V'$ s'il existe, et $n+1$ sinon.

Complétez l'algorithme Capture(T,k) pour qu'il retourne une solution optimale au problème.

$i \leftarrow$ Suivant_P(T,0)
$j \leftarrow$ Suivant_V(T,0)
$c \leftarrow 0$
while $i \leq n$ *et* $j \leq n$ **do**
  if $|i - j| \leq k$ **then**
    $c \leftarrow c + 1$
    $i \leftarrow$ Suivant_P(T,i)
    $j \leftarrow$ Suivant_V(T,j)
  else
    if $i < j$ **then**
      $i \leftarrow$ Suivant_P(T,i)
    else
      $j \leftarrow$ Suivant_V(T,j)
retourner c → nb de captures

Algorithme 2 : Capture(T,k)

En examinant les policiers par indices croissants chaque police capture le voleur de plus petit indice parmi les voleurs en cavale qu'il peut attraper.

2) Complexité de l'algorithme naïf.

$$n_p = n_V = \frac{n}{2} \quad \text{et} \quad k = n-1$$

$$n_V \times (n_V - 1) \times \dots \times 1 = n_V! = \left(\frac{n}{2}\right)!$$

possibilités de capture pour le 1er policier

possibilités pour le 2e policier

L'algorithme naïf est de complexité exponentielle.