Concours commun des Écoles Normales Supérieures
Épreuve pratique d'algorithmique et de programmation

Jeu d'alignements

Durée de l'épreuve : 4 heures — Coefficient : 4

Juillet 2003

Note. Lorsque la description d'un algorithme est demandée, celle-ci doit être courte et précise. Quand on demande la complexité d'un algorithme en fonction d'un paramètre n, on demande un ordre de grandeur du temps de calcul dans le cas le pire, par exemple $O(n^2)$ ou $O(n \log n)$.

1 Préliminaires

On considère la suite d'entiers positifs (u_n) ci-dessous :
– u_0 est donné sur votre table. **(À reporter en haut de votre copie !)**
– $u_n = (16383 \times u_{n-1}) \bmod 59047$, pour $n > 0$.

Question 1 *Que valent u_{996} et u_{9996} ?*

On définit $v_i = 1$ si $u_i \equiv 0 (\bmod 7)$ et $v_i = 0$ sinon.

Question 2 *Combien y a-t-il d'indices i tels que $v_i = 0$, avec $0 \leq i < 10000$?*

Dans la suite de ce problème, on considérera un plateau de jeu carré de dimensions $k \times k$ et la configuration de départ est définie ainsi : pour $0 \leq i, j < k$, la case de coordonnées (i, j) est vide si $v_{i+j\times k} = 0$, et contient un pion sinon.

2 Alignements

Pour chaque question de cette partie, on donnera la réponse pour les cas $k = 4$ à $k = 12$, $k = 100$ et $k = 1000$. Si votre programme s'avère trop lent, il est déconseillé d'attendre passivement le résultat pour les valeurs élevées de k.

Question 3 *Combien y a-t-il de pions sur le plateau ?*

On appelle alignement (vertical ou horizontal) un ensemble maximal de pions alignés. Par exemple, $(T_{i,j}, T_{i+1,j}, ..., T_{i+n,j})$ est un alignement horizontal (de longueur n) si, et seulement si, toutes ces cases sont pleines et si $T_{i-1,j}$ et $T_{i+n+1,j}$ sont vides ou inexistantes. De même un alignement vertical regroupe des cases avec i constant. NB : un pion isolé est un alignement de longueur 0.

Question 4 *Combien y a-t-il d'alignements de chaque type dans la configuration de départ, pour chaque valeur de k ? Décrivez votre algorithme et la structure de données. Évaluez la complexité de votre algorithme en fonction de k.*

On considère aussi les alignement diagonaux : ceux où $i + j$ est constant seront appelés diagonales de type A, et ceux où $i - j$ est constant diagonales de type B.

Question 5 *Calculez le nombre d'alignements diagonaux de chaque type.*

À toute configuration on associe un score, qui est égal à la somme des carrés des longueurs des alignements (verticaux, horizontaux et diagonaux).

Question 6 *Calculez le score de la configuration de départ.*

3 Meilleur déplacement

Pour chaque question de cette partie, on donnera la réponse pour $k = 4$ à $k = 12$, $k = 100$ et $k = 1000$. Si votre programme s'avère trop lent, il est déconseillé d'attendre passivement le résultat pour les valeurs élevées de k.

Dans un premier temps, on s'autorise à déplacer un unique pion, dans une case libre voisine (comme le roi aux échecs). À partir de la configuration de départ, les meilleurs déplacements sont ceux qui maximisent le score de la nouvelle configuration.

Question 7 *Décrivez l'un des meilleurs déplacements, en indiquant la position de départ et la position d'arrivée du pion déplacé. Quel est le score de la nouvelle configuration ?*
Décrivez votre algorithme et la structure de données utilisée. Évaluez la complexité de votre algorithme.

On s'autorise désormais à déplacer un unique pion, en ligne droite selon un trajet sans obstacle (comme la dame aux échecs). À partir de la configuration de départ, les meilleurs déplacements sont ceux qui maximisent le score de la nouvelle configuration.

Question 8 *Décrivez l'un des meilleurs déplacements, en indiquant la position de départ et la position d'arrivée du pion déplacé. Quel est le score de la nouvelle configuration ?*
Décrivez votre algorithme et évaluez sa complexité.

4 Plusieurs déplacements

Le but est de maximiser le score de la configuration, en étant autorisé à déplacer chaque pion au plus une fois vers une case voisine (à la manière du roi), dans l'ordre qu'on veut.
Dans cette partie, les algorithmes ont une complexité élevée et ne seront utilisables que pour de petites valeurs de k.

Première approche : recherche exhaustive.

Il s'agit de tester toutes les possibilités et de garder la meilleure.

Question 9 *Proposez un majorant du nombre de possibilités évaluées lors d'une recherche exhaustive. Ce majorant doit être facile à calculer en connaissant la position de départ.*
Donnez la valeur de ce majorant pour $k = 4$ à 7.

Question 10 *Programmez l'algorithme de recherche exhaustive, et donnez son résultat pour les valeurs que votre majorant indique être atteignables en un temps raisonnable. Combien de possibilités sont réellement testées ?*

Deuxième approche : algorithme glouton

La recherche exhaustive étant impraticable pour des valeurs de k élevées, on peut utiliser l'algorithme glouton décrit ci-dessous, qui bien évidemment ne donne pas nécessairement le meilleur score possible.
L'algorithme glouton choisit à chaque fois le déplacement qui augmente le score le plus possible, jusqu'à ce qu'aucun déplacement n'augmente le score. Puisqu'il arrive souvent que plusieurs meilleurs déplacements soient possibles, l'algorithme glouton ainsi décrit est *non-déterministe*, ce qui signifie que son exécution dépend des choix faits lorsqu'il y a plusieurs meilleurs déplacements.

On définit comme suit un ordre sur les coordonnées : $(i, j) < (i', j')$ si $i < i'$ ou, $i = i'$ et $j < j'$.

Question 11 *On choisit à chaque fois, parmi les meilleurs déplacements, celui du pion $T_{i,j}$ de coordonnées (i, j) minimales, et parmi les déplacements de ce pion celui vers la case de coordonnées minimales.*
Quel est le score obtenu, pour allant de $k = 4$ à $k = 12$ et $k = 100$?

Question 12 *Réalisez un programme qui teste toutes les exécutions possibles de l'algorithme glouton. Utilisez votre programme pour les valeurs de k entre 4 et 12 pour lesquelles le nombre de chemins d'exécution de l'algorithme glouton n'est pas trop élevé.*
Quel est le meilleur score obtenu pour chaque valeur de k ? Combien de chemins d'exécution sont-ils testés ?

Troisième approche : construction de longs alignements

Une autre stratégie est la suivante : on cherche à construire les plus longs alignements possibles.
L'algorithme fonctionne comme suit. Au départ, tous les pions sont "vivants". Une fois déplacé, un pion devient "mort". À chaque itération, on choisit un alignement de pions vivants de longueur maximale. Pour chacune des huit directions dans lesquelles peut bouger cet alignement (d'une case), on regarde s'il est possible d'agrandir cet alignement en bougeant d'autres pions vivants (d'une case chacun). On choisit l'option qui donne le plus long alignement.
Comme précédemment, plusieurs chemins d'exécution sont possibles.

Question 13 *Réalisez un programme qui teste toutes les exécutions possibles de cet algorithme. Utilisez votre programme pour les valeurs de k entre 4 et 12 pour lesquelles le nombre de chemins d'exécution de l'algorithme n'est pas trop élevé.*
Quel est le meilleur score obtenu ? Combien de chemins d'exécution sont-ils testés ?