

# Halma

Durée de l'épreuve : 4 heures

Juillet 2004

**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 Description du problème

**Le jeu de Halma.** Ce problème est consacré à la conception d'algorithmes jouant au jeu de Halma. C'est un jeu de plateau datant de l'époque victorienne, se jouant sur un plateau carré quadrillé.

**Règle de victoire.** Dans ce problème, on s'intéresse au jeu de Halma à un seul joueur, avec un plateau carré de taille arbitraire  $n$  et un nombre de pions arbitraire  $p$ . On appelle *configuration* du jeu la position des pions. La distance d'un pion se situant sur la case  $(x, y)$  est par définition  $x + y$ . La mesure de l'*insatisfaction* d'une configuration des pions sur le plateau de jeu est la somme des distances des pions. Si l'insatisfaction d'une configuration est minimale parmi toutes les configurations ayant le même nombre de pions, on dit que la configuration est *rangée*. Le but de ce problème est de trouver des algorithmes permettant d'arriver à une configuration rangée, de préférence en minimisant le nombre de coups entre la configuration de départ et cette configuration d'arrivée.

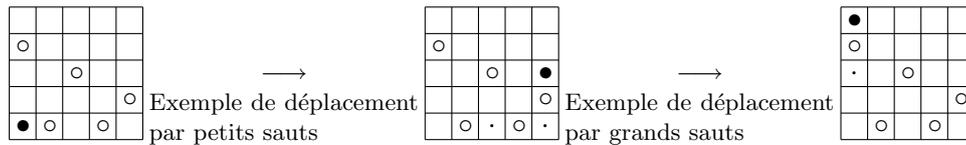
**Règles de déplacement des pions.** À chaque coup, on déplace un pion. Plusieurs types de déplacements sont possibles.

On peut choisir de déplacer un pion vers l'une des quatre cases voisines, si celle-ci est libre.

Un pion peut aussi être déplacé par une succession de *petits sauts*. On définit un *petit saut* comme un déplacement de deux cases, dans l'une des quatre

directions. Ce déplacement est autorisé si la case d'arrivée est libre, et si la case intermédiaire est occupée. Contrairement aux dames, le pion par dessus lequel on saute reste sur le plateau de jeu. Une succession de petits sauts, en nombre arbitraire, est considéré comme un unique coup.

Une règle optionnelle permet non seulement de faire de petits sauts (définis ci-dessus) mais aussi de *grands sauts*. Un grand saut est un déplacement d'un nombre pair de cases dans l'une des quatre directions, tel que la case au milieu de ce trajet soit occupée, et toutes les autres soient libres.



**Organisation du problème.** Dans la section 2 on étudie quelques propriétés des configurations rangées. Dans la section 3 on définit une configuration de départ. Ces deux sections sont indépendantes. Dans la section 4 on étudie comment, à partir d'une configuration donnée, choisir un déplacement de pion qui diminue le plus possible l'insatisfaction. Cette section ne dépend que partiellement des résultats de la section 2. Dans la section 5 on construit des algorithmes de rangement utilisant les techniques de la section précédente.

**Rappel sur la représentation de tableaux bidimensionnels.** On peut représenter les tableaux bidimensionnels dans des tableaux simples. Par exemple, pour un tableau de dimension  $n$ , on peut représenter l'élément de coordonnées  $(x, y)$  par  $T[x+n*y]$ .

## 2 Préliminaires : configurations rangées

Dans cette section, la taille du plateau est arbitrairement grande. Autrement dit, les pions peuvent être sur toute case de coordonnées  $(x, y)$  avec  $x \geq 0$  et  $y \geq 0$ . On étudie quelques propriétés élémentaires des configurations rangées.

**Question 1** Montrez que dans une configuration rangée avec  $p$  pions, la distance du pion le plus loin est  $\min\{k | \frac{(k+1)(k+2)}{2} \geq p\}$ .

**Question 2** Décrire comment calculer l'insatisfaction d'une configuration rangée à  $p$  pions. Quelle est la complexité de votre algorithme en fonction de  $p$ ? Quelle est l'insatisfaction d'une configuration rangée à 1000 pions?

**Question 3** Montrez que si le nombre de pions n'est pas un entier de la forme  $\frac{k(k-1)}{2}$  alors il y a plusieurs configurations rangées.

### 3 Préliminaires : configuration de départ

On utilisera la suite (pseudo-aléatoire) d'entiers positifs  $(u_n)$  définie 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 4** Que valent  $u_{992}$  et  $u_{9992}$  ?

Dans la suite du problème, on s'intéressera aux cas où le couple  $(n, p)$  est pris dans l'ensemble  $\mathcal{E} = \{(4, 6), (6, 10), (15, 20), (25, 10)\}$ , et aux deux variantes du jeu. La *configuration de départ* est définie en plaçant  $p$  pions selon l'algorithme ci-dessous : on pose  $(x_i, y_i) = (u_{2i} \bmod n, u_{2i+1} \bmod n)$  ; les positions initiales des  $p$  pions sont les  $p$  premières valeurs distinctes de la suite  $(x_1, y_1), (x_2, y_2), \dots$

On remarque que le premier pion posé sur le plateau a donc pour coordonnées  $(x_1, y_1)$ , mais que le  $p$ -ième pion posé a pour coordonnées  $(x_\pi, y_\pi)$  pour une valeur  $\pi \geq p$ .

**Question 5** Pour chaque choix de paramètres  $(n, p) \in \mathcal{E}$ , quelles sont la valeur  $\pi$  et les coordonnées  $(x_\pi, y_\pi)$  ?

On rappelle que selon les règles du jeu, un pion peut se déplacer vers une case voisine, ou bien faire une succession de (petits ou grands) sauts. On pourrait imaginer de n'autoriser que des sauts. La question suivante montre que cela restreindrait beaucoup le déplacement des pions.

**Question 6** Montrez que si un pion ne se déplace que par sauts, il y a trois quarts des cases du damier auxquelles il n'a pas accès, indépendamment de la position des autres pions. Cette propriété définit une partition de l'ensemble des cases du damier en quatre parties. Donnez une définition caractérisant ces quatre parties, et donnez (pour la configuration de départ définie ci-dessus) le nombre de pions dans chaque partie ainsi définie.

### 4 Étude d'un coup

Dans cette section, on étudie la configuration de départ, pour savoir quel déplacement choisir. On étudie donc les coups possibles, et leur effet sur le rangement du plateau (variation de l'insatisfaction). On donnera les résultats

pour les deux variantes du jeu (petits sauts ou grands sauts) et toutes les valeurs  $(n, p) \in \mathcal{E}$ .

**Question 7** *Quelle est l'insatisfaction de la configuration de départ ? Décrivez l'algorithme que vous avez utilisé et sa complexité en fonction de  $n$  et  $p$ .*

**Question 8** *À partir de la configuration de départ, si on choisit de déplacer le pion de coordonnées  $(x_1, y_1)$ , combien de cases peut-il atteindre en un coup ?*

*Décrivez l'algorithme que vous avez utilisé et sa complexité en fonction de  $n$ .*

**Question 9** *À partir de la configuration de départ, combien y a-t-il de coups possibles (un coup étant défini par la position de départ et la position d'arrivée du pion qu'on déplace) ? Décrivez l'algorithme que vous avez utilisé et sa complexité en fonction de  $n$  et  $p$ .*

**Choix du meilleur coup.** On définit deux ordres totaux sur les cases du plateau : l'ordre  $\prec_L$  est l'ordre lexicographique et l'ordre  $\prec_D$  celui des distances. Formellement :

- $(a, b) \prec_L (c, d) \Leftrightarrow a < c$  ou  $(a = c$  et  $b < d)$
- $(a, b) \prec_D (c, d) \Leftrightarrow a + b < c + d$  ou  $(a + b = c + d$  et  $a < c)$

Parmi toutes les configurations accessibles en un coup, certaines ont une insatisfaction minimale. Le “meilleur coup” est donc l'un de ceux-là. Si plusieurs réponses sont possibles, on utilise l'un des ordres définis ci-dessus. Le meilleur coup selon l'ordre  $\prec$  est celui qui fait bouger le pion dont la case est la plus grande selon l'ordre  $\prec$ , et parmi les déplacements de ce pion celui pour lequel la case d'arrivée est la plus grande selon l'ordre  $\prec$ .

**Question 10** *Décrivez un algorithme qui à partir d'une configuration donnée, calcule le meilleur coup selon l'ordre des distances. Quelle est la complexité de votre algorithme en fonction de  $n$  et  $p$ . Quel est son résultat pour la configuration de départ, pour chaque choix de paramètres  $(n, p) \in \mathcal{E}$ .*

## 5 Algorithmes de résolution du problème

On se base sur l'algorithme glouton, qui cherche à chaque coup à minimiser l'insatisfaction. Plusieurs variantes de l'algorithme glouton sont possibles, selon la façon dont on choisit le coup parmi ceux qui minimisent l'insatisfaction à l'étape suivante.

NB : on donnera les résultats pour les deux variantes du jeu et toutes les valeurs  $(n, p)$ .

Dans l'algorithme A, si plusieurs coups minimisent autant l'insatisfaction, on choisit le meilleur coup selon l'ordre  $\prec_D$ .

**Question 11** *Après combien de coups cette variante de l'algorithme glouton s'arrête-t-elle ? Quelle est l'insatisfaction à la fin ?*

Dans l'algorithme A, si plusieurs coups minimisent autant l'insatisfaction, on choisit le meilleur coup selon l'ordre  $\prec_L$ .

**Question 12** *Après combien de coups cette variante de l'algorithme glouton s'arrête-t-elle ? Quelle est l'insatisfaction à la fin ?*

On remarque que les algorithmes ci-dessus arrivent rarement à une configuration rangée. Il faut donc trouver un algorithme de résolution de fin de partie. Pour cela, on commence par définir un *déplacement en diagonale* : c'est une succession de deux coups, le premier vers une case voisine, et le second vers une case voisine et dans une direction orthogonale à la première. On peut remarquer que certains déplacements en diagonale ne changent pas l'insatisfaction.

On propose l'algorithme suivant (algorithme C) : à partir d'une position où il existe un coup qui diminue l'insatisfaction, on applique l'algorithme A. À partir d'une position  $X$  où aucun coup ne diminue l'insatisfaction, on choisit le plus grand des pions (dans l'ordre  $\prec_D$ ) tel que, après une succession de déplacements en diagonale ne changeant pas l'insatisfaction, il est possible en un coup d'obtenir une insatisfaction plus faible que celle de la position  $X$ .

**Question 13** *Montrer que l'algorithme C arrive toujours à une configuration rangée. En combien de coups range-t-il le plateau, à partir des configurations de départ définies plus haut ?*

L'algorithme D généralise les algorithmes A et B de la façon suivante : si plusieurs coups minimisent autant l'insatisfaction, on les choisit simultanément, et on élimine dans l'arbre des exécutions possibles les branches pour lesquelles il existe une autre branche qui à la même profondeur (mesurée en nombre de coups) a une insatisfaction strictement plus faible.

**Question 14** *Après combien de coups cette variante de l'algorithme glouton s'arrête-t-elle ? Quelle est l'insatisfaction à la fin ?*