Rapport de TIPE
Morpion

Peng-Wei Chen, MP, 2017-2018

Avant-propos

Le morpion est un jeu très connu dans le monde dont la stratégie gagnante a été trouvée dans les années 90. Or, on ne sait pas encore si une stratégie gagnante existe lorsque la taille de la grille varie. L'objectif de ce TIPE est de déterminer les tailles de la grille où le premier joueur a une stratégie gagnante.

1 Introduction

1.1 La règle du morpion

Deux joueurs jouent sur une grille de taille 15×15 sur le papier. Chacun prend un symbole et on dessine au tour par tour son symbole sur la grille. Le but est d'aligner 5 symboles verticalement, horizontalement ou en diagonale pour gagner. Dans le cas généralisé, nous appelons (m, n, k)-jeu où la taille de la grille est $m \times n$ et il faut k symboles dans une ligne pour gagner.

1.2 Méthode

Il existe deux méthodes pour déterminer s'il existe une telle stratégie :

— On cherche tous les cas possibles.

— On apparie les points de la grille. Si le deuxième joueur peut toujours prévenir la réussite du premier joueur en jouant le pairage, alors une telle stratégie n'existe pas.

Dans le cas où $k \leq 7$, on utilise la première méthode avec l'arbre de décision. Chaque nœud de cet arbre est un état de la grille. Par exemple (Figure. 1) , la racine de cet arbre est la grille sans symbole. Ensuite, les fils de chaque nœud sont les états de la grille où on place un symbole de plus au dessus. Enfin, les feuilles sont les cas où un des joueurs a gagné ou où personne n'a gagné (c'est-à-dire que la grille a été remplie et que aucun joueur n'a aligné k symboles).

Or, la complexité de chercher tous les cas possibles est en $O((m \times n)!)$. Ainsi, on utilise une fonction gloutonne pour chercher le cas le « plus »possible. Avec cette fonction, on donne une note à chaque point. Si un point est plus possible pour gagner, alors sa note est plus élevée.

On utilise la deuxième méthode lorsque $k > 7$.

2 Exploitation

2.1 $k \leq 5$

Afin de diminuer la complexité, on essaie de créer une fonction qui a l'état de la grille comme entrée et renvoie dix positions : ce sont les positions possibles pour gagner. Ainsi, à chaque

nœud de l'arbre de décision, on a diminué le nombre de cas possibles de tous les positions à dix positions.

2.1.1 Fonction naïve

L'idée est simple : on considère le nombre de symboles non séparés dans une ligne. Pour chaque point sur la grille non occupé par les symboles, on définit la « note » à valeur dans $\mathbb{N}$ en ce point. La note est caculée selon les quatre directions, dépend du nombre de symbole du premier joueur (On n'utilise cette fonction que pour le premier joueur). Précisément, il y a trois cas possibles. Pour les deux premiers cas, on appelle i-vivant ou i-mort (Figure. 2) où i est le nombre de symboles déjà aligné, vivant si les deux cotés sont libres, mort si seul un des deux coté est libre. On donne 3^i au cas de i-vivant et 3^{i-1} au celui de i-mort car i-mort est en effet le cas de (i+1)-vivant. En particulier, on donne 3^{k+1} au cas où le premier joueur est sûrement gagné, c'est-à-dire k-mort ou (k-1)-vivant. Tenant compte des quatre directions, il y a des autres cas autre que k-mort ou (k-1)-vivant que le premier joueur gagne sûrement. Par exemple (Figure. 3), si k = 5, le cas 4-mort-4-mort est un cas gagnant (le deuxième joueur ne peut que prévenir la réussite d'une direction, les deux x). Dans le troisième cas, les deux cotés ne sont pas libres, alors on donne 0 selon cette direction (impossible de compléter en k symboles dans la même ligne). La « note » est la somme de ces valeurs selon les quatre directions. Par exemple (Figure. 4), au point A, on somme 81 (3^4, horizontale) et 3×3 (3^1,verticale et les deux directions diagonales). La note au point A vaut 90.

On a ainsi une fonction naïve.

2.1.2 Améliorer notre fonction avec l'élagage alpha-beta

Dans notre fonction, on ne considère que le nombre de symboles dans une ligne et elle n'est pas correcte dans certains cas. Par exemple (Figure. 5.1), au début du jeu, le premier joueur place son symbole au centre de la grille. Ensuite, le deuxième joueur place en haut à gauche du centre. Dans ce cas-là, la fonction renvoie 4 positions (5.2) avec la même note. Cependant, l'une de ces positions donne l'avantage au deuxième joueur. (5.3) Dans ce cas-là, le deuxième joueur peut défendre et essayer d'aligner ses symboles en même temps.

On utilise l'arbre de décision pour améliorer notre fonction, de sorte que notre fonction « pense » à l'étape d'après. Pour le faire, on utilise l'algorithme de minimax. Dans cet algorithme, on utilise notre fonction naïve et la note est calculée par la difference de celle du premier joueur et celle du deuxième joueur. Cette note peut être eventuellement négative. Ainsi (Figure. 6), dans le niveau du premier joueur (niveau max), on veux que la note soit plus élevée tandis que la note est plus faible dans le niveau du deuxième joueur (niveau min).

Dans l'arbre de décision, on donne la note de la position si c'est une feuille. Lorsqu'on est dans un nœud du niveau max (resp. min), on compare les notes de ses fils et en choisit la plus grande (resp. petite). En pratique, on se donne une hauteur et on fait un parcours en profondeur.

On peut encore améliorer cet algorithme en utilisant l'algorithme de l'élagage alpha-beta. Dans l'algorithme de minimax, on peut couper les arêtes dans certains cas. Par exemple (Figure. 7), dans le niveau max, on a déjà déterminé la note d'un nœud 81. Or dans le nœud A, sa note est donnée par le minimum du niveau prochain. Ainsi, dans le niveau prochain, on peut s'arrêter dès qu'on a une note inférieure à 81. On appelle coupure alpha (resp. coupure beta) lorsqu'on est dans le niveau max (resp. niveau min) et une des notes des fils d'un nœud est inférieure (resp. supérieure) à celle de son frère.

On utilise la fonction améliorée pour chercher la stratégie gagnante du premier joueur.

2.2 $k \geq 8$

On peut montrer que le premier joueur n'a pas de stratégie gagnante lorsque k = 8, et donc le premier joueur n'en a pas pour tout $k \geq 8$ car la réussite du (m, n, k)-jeu où $k \geq 8$ implique la réussite du (m, n, 8)-jeu. On donne notamment la preuve du cas k = 9 (Figure. 8) car il est plus intuitif. On voit que si le deuxième joueur suit le pairage, alors le premier joueur ne peut jamais réussir (les lignes rouges).

Lorsque k = 8, on divise le jeu en des sous-jeux. Ces sous-jeux se jouent sur la grille de la forme comme Figure. 9. Il y a trois façons de gagner ce sous-jeu (Figure. 10) :

— Aligner trois symboles en diagonale.
— Aligner verticalement deux symboles.
— Aligner horizontalement quatre symboles.

Ainsi, puisque le premièr joueur ne peut jamais gagner ce sous-jeu, il ne peut pas non plus gagner le (m, n, 8)-jeu. (Figure. 11)

3 Conclusion

— k = 3
— k = 4
— k = 5
— k ≥ 8
Une telle stratégie n'existe pas.

4 Annexe

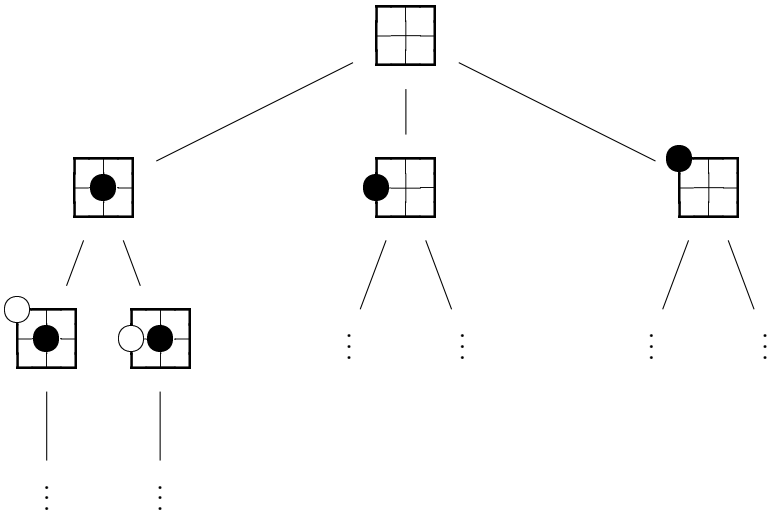
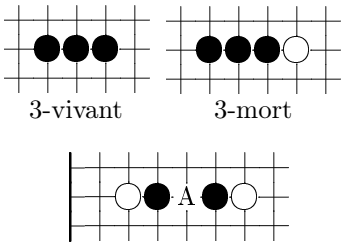


Figure 1 – Arbre de décision (le cercle noir est le symbole du premier joueur)



La note au point A est nulle pour le noir.

Figure 2 – 3-vivant et 3-mort

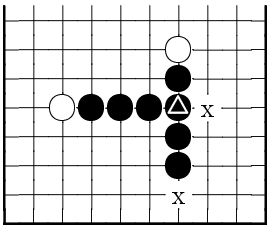


Figure 3 – L'effet des directions différente

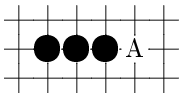


Figure 4 – Un exemple de la note

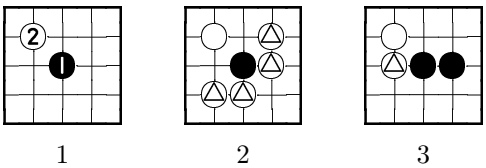


Figure 5 – Insuffisance de notre fonction naïve

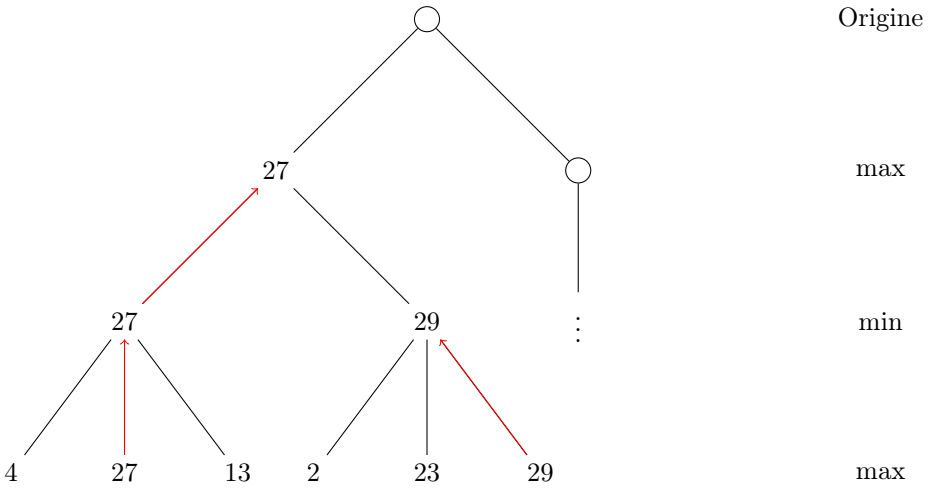


Figure 6 – L'algorithme de minimax

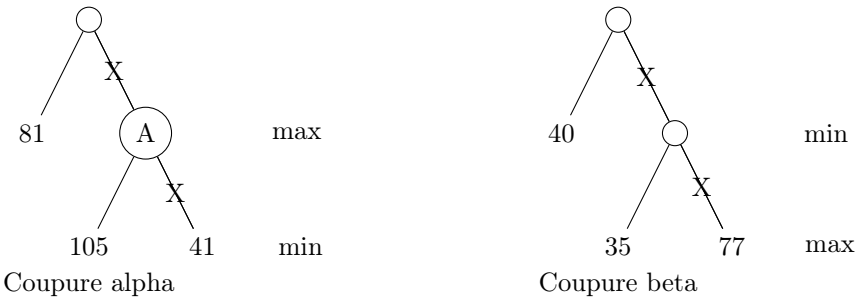


Figure 7 – L'algorithme de l'élagage alphaa-beta

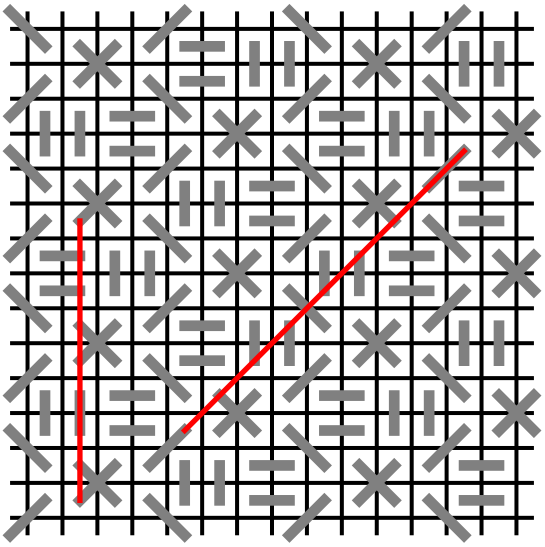


Figure 8 – (m, n, 9)-jeu

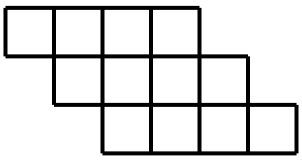


Figure 9 – La grille du sous-jeu

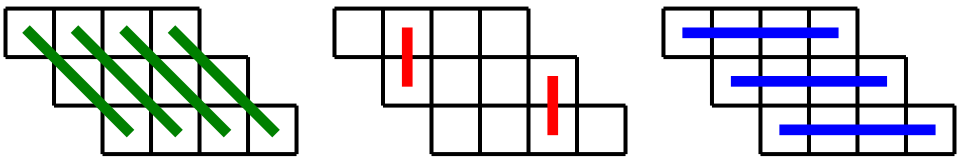


Figure 10 – Les trois façons pour gagner le sous-jeu.

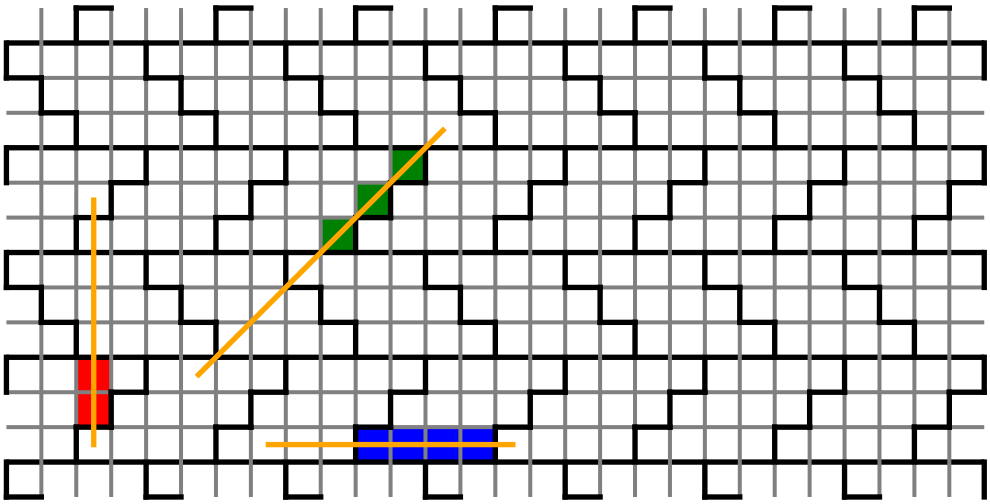


Figure 11 – (m, n, 8)-jeu : toutes les lignes de longueur 8 passent une des lignes gagantes du sous-jeu.