TP IA: Programmation d'un wargame simplifié
Algorithme alpha/beta

1 Cadre du jeu

Le but est de créer un "wargame" simplifié se jouant à 2 joueurs. A chaque tour, chaque joueur ne peut déplacer qu'une unité d'une case dans les 8 directions. Il est interdit de bouger vers une case occupée par une de ses unités ou de sortir de l'aire de jeu. Si la case destination est occupée par une unité ennemie, on prend l'unité ennemie à condition que la somme de ses pièces sur les 8 cases voisines soit supérieure à la somme des pièces ennemis sur ces même 8 cases voisines. Le perdant perd l'unité qui est mise en jeu. Un joueur perd s'il ne lui reste plus d'unité ou si l'autre joueur réussit à atteindre avec l'une de ses pièces, la ligne de fond de jeu de l'autre joueur.

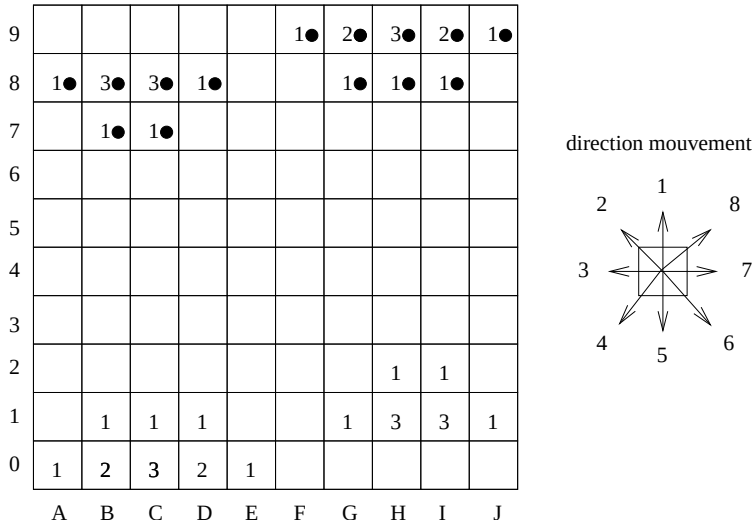

Figure 1: Situation du jeu au départ et mouvements valides de chaque unité. Le chiffre indique la valeur d'une unité. Le point noir indique les unités appartenant aux Noirs. Les autres unités appartiennent aux Blancs. Ce sont les blancs qui commencent.

Evaluation pour la prise d'une unité adverse sur la case (x,y):

$$B \ \ Gagnant \quad ssi : \sum_{i \in V_{x,y}} F_i^B > \sum_{i \in V_{x,y}} F_i^N \tag{1}$$

Afin de simplifier les problèmes d'interconnexion de 2 programmes pour un tournois, le dialogue avec la machine devra soit permettre d'afficher l'état du jeu en mode texte soit se limiter à afficher le mouvement joué par la machine.

Exemple de dialogue:

- D1D2 (on lit au clavier le coup de l'utilisateur)
- B7B6 (on affiche à l'écran la réponse de la machine)
- ...

Dans le cas d'une version permettant d'afficher la carte du jeu, on notera les pièces par leur valeur suivie d'un point ou un espace selon qu'il s'agit des noirs ou des blancs par exemple.

2 Structures de données

Déterminer les structures de données et les fonctions permettant de gérer le jeu.

3 Algorithme en profrondeur d'abord

Utiliser une recherche en profondeur d'abord (avec prof limite fixée) pour déterminer les coups de l'ordinateur. Le jeu se jouant à 2 joueurs, on utilisera MINIMAX ou NEGAMAX pour remonter l'évaluation des noeuds extrèmes et choisir le coup à jouer.

Quelle fonction d'évaluation choisissez vous? pourquoi?

4 Utilisation de l'algorithme alpha/beta

Modifier l'algorithme précédent pour introduire des coupures alpha/beta.

5 Analyse des performances de l'algorithme et tournois

5.1 Complexité

Combien d'opérations sont faites en moyenne par votre alogrithme pour l'analyse d'un coup quelconque? (justifier votre réponse).

5.2 Analyse expérimentale

Tracer la courbe du nombre de noeuds explorer en fonction de la profondeur d'analyse dans le cas de l'algorithme MINIMAX ou NEGAMAX puis dans le cas de l'algorithme alpha/beta. Pour construire ces courbes penser à moyenner sur plusieurs réalisations (vous pourrez utiliser gnuplot ou openoffice pour afficher vos courbes).

Quel devrait être le gain théorique en performance?

Pouvez vous améliorer votre programme pour qu'il devienne plus performant à niveau de jeu constant (profondeur d'analyse)? Proposer et tester une solution.

6 Améliorations

On pourrait rendre cachée la valeur des pièces pour compliquer l'évaluation de l'adversaire. On peut aussi ajouter des cases interdites (ou difficiles) et permettre à chaque jour de déplacer plusieurs unités à chaque itération (la combinatoire du jeu peut alors augmenter fortement).