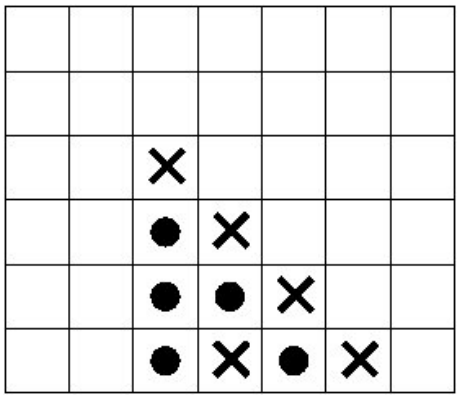
Informatique - Master 1
Module IA
TP 3 & 4 - Puissance 4

Sujet original de Tassadit BOUADI

2014-2015

Lors de ce TP, vous allez vous familiariser avec l'algorithme Min-Max ainsi que son extension avec élagage $\alpha\beta$. Vous allez appliquez cette méthode au jeu du puissance 4.

Il s'agit d'un jeu très connu dans lequel il faut aligner au moins 4 pions, soit en diagonale, soit horizontalement, soit verticalement :

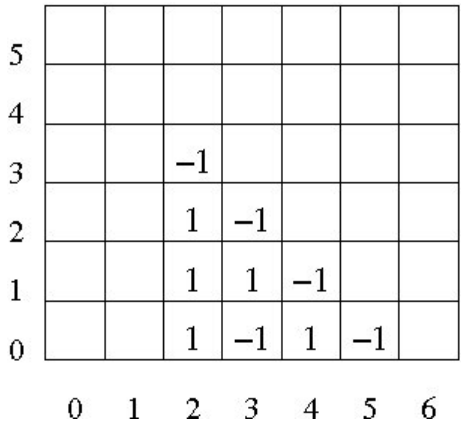


La grille est représentée ici par une matrice de 6 lignes et de 7 colonnes d'entier. Chaque case peut prendre 3 valeurs :

— 1 si le joueur 1 a joué (joueur avec les ronds) ;
— -1 si le joueur 2 a joué (joueur avec les croix) ;
— 0 si la case est vide.

La grille présentée plus haut est alors représentée ainsi (les cases vides contiennent zéro) :

Recopier les sources du TP. Vous devez ajouter, dans votre projet Eclipse, les .jar fournis : *Lecture.jar* et *FonctionEvaluationProf.jar*.



1 Boucle de jeu du puissance 4

Complétez la fonction ***jeu(Joueur joueur1, Joueur joueur2)*** du programme principal, qui permet de faire jouer joueur1 et joueur2 au puissance 4.

Pour tester cette fonction, vous pouvez utiliser les joueurs *JoueurAleatoire* (choisit un coup aléatoirement parmi les coups possibles) et *JoueurHumain* (permet d'entrer au clavier l'indice de la colonne dans laquelle on veut jouer).

2 Création d'un joueur utilisant l'algorithme Min-Max

Créer une classe ***JoueurMinMax*** qui implémente l'interface *Joueur* et qui définit la fonction *coup(Grille grille, int joueur)*, en utilisant l'algorithme Min-Max. Cet algorithme utilise une fonction d'évaluation pour attribuer une valeur à chacune des grilles générées : utiliser la fonction *FonctionEvaluationProf*.

Un des paramètres de l'algorithme Min-Max est la profondeur de l'arbre de recherche. Regardez l'influence de ce paramètre (nombre d'appels récursifs, grille choisie) en faisant varier sa valeur.

3 Création d'un joueur utilisant l'algorithme Min-Max avec élagage $\alpha\beta$

Afin d'éviter d'explorer des grilles non-intéressantes, créer une classe ***JoueurAlphaBeta*** qui étend la classe *JoueurMinMax* et qui définit la fonc-

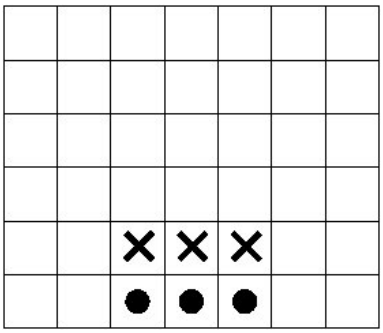
tion *coup(Grille grille, int joueur)*, en utilisant l'algorithme Min-Max avec élagage $\alpha\beta$.

Compare maintenant les résultats obtenus avec les joueurs *JoueurMinMax* et *JoueurAlphaBeta*, en termes de nombre d'appels récursifs pour la même profondeur d'exploration choisie.

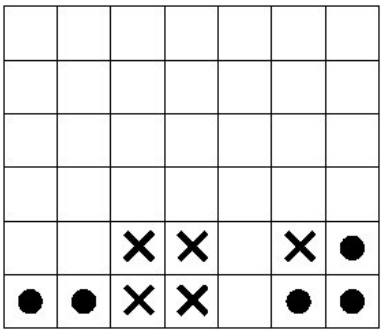
4 Recherche d'une fonction d'évaluation des grilles de puissance 4

La fonction d'évaluation qui permet d'évaluer la grille du point de vue d'un joueur est importante. Cette fonction doit rendre une valeur dans l'intervalle $[MIN, MAX]$: plus la grille semble bonne pour le joueur considéré et plus la valeur donnée par la fonction d'évaluation doit être élevée.

Par exemple, pour le joueur possédant les ronds, la fonction d'évaluation doit donner une valeur élevée pour la grille (a) et une valeur faible pour la grille (b), ci-dessous.



(a)



(b)

Définissez maintenant votre propre fonction d'évaluation en implémentant l'interface *FonctionEvaluation* et en définissant la fonction ***evaluation(Grille grille, int joueur)***. Vous pouvez utiliser certaines fonctions de la classe *Grille*. Il peut également être avantageux de modifier l'ordre de génération des coups, dans la fonction *generationCoups()*.

Comparez votre fonction d'évaluation avec la fonction *FonctionEvaluationProf* ainsi qu'avec celles créées par les autres binômes.