Rapport : Intelligence Artificielle et Jeux Adversariaux

UE Sécurité et aide à la décision

Licence Informatique

Emmanuel GARREAU

21700336

Groupe 1B

Sujet : Jeu de l'infection et algorithmes d'Intelligence Artificielle

2019 - 2020

Table des matières

1 Introduction

Bienvenue sur le projet "Jeu de l'Infection et Intelligence Artificielle" conçu dans le cadre de l'UE Sécurité et Aide à la Décision.
Il s'agit d'un projet ayant pour objectif la mise en place du jeu de l'infection ainsi que l'implémentation d'algorithmes d'Intelligence Artificielle tels le MiniMax, l'élagage AlphaBêta ou encore leur version NégaMax, réputés dans le monde du jeu.

J'ai choisi de réaliser ce projet en Java, langage statiquement typé et qui nécessitera donc, contrairement au Python, une classe supplémentaire pour la mise en oeuvre de l'Intelligence Artificielle.
Nous allons, ici, faire le bilan concernant les différents algorithmes, leurs avantages et inconvéniants. Vous pouvez effectuer ces expérimentations via la classe *Test*. Nous mentionnerons également les améliorations pouvant être mises en place pour accélérer le temps de calcul.

2 Etude du nombre de noeuds parcourus

Cette partie portera sur l'étude du parcours des noeuds. On calculera le nombre de noeuds parcourus au total par les deux joueurs.
On considère un terrain de taille 5x5.

2.1 Algorithme MiniMax/Négamax

On effectuera cette étude sur une profondeur allant de 1 à 5 pour les deux joueurs. Au-delà, le temps de calcul s'avère trop long.

Profondeur J1	Profondeur J2	Noeuds parcourus au total
1	1	192
2	2	2872
3	3	30896
4	5	624007
5	5	5439450

TABLE 1 – Tableau récapitulatif du nombre de noeuds parcourus par MiniMax

2.2 Elagage AlphaBêta

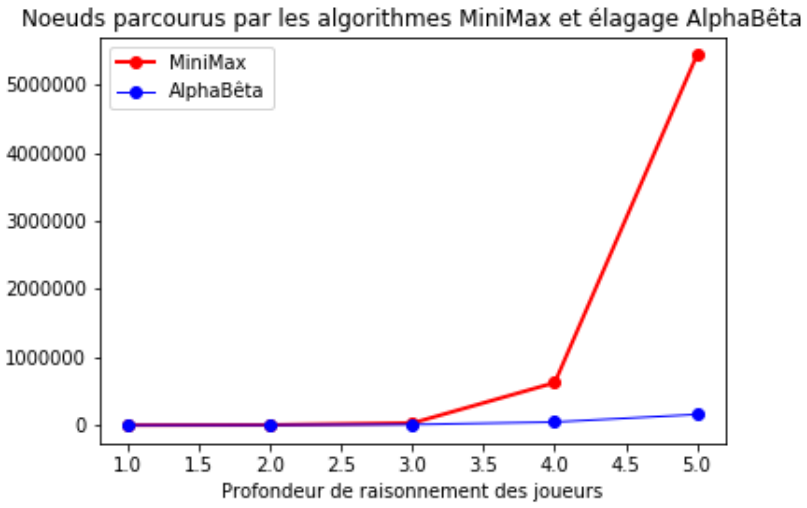
On effectuera cette étude sur une profondeur allant de 1 à 5 pour les deux joueurs dans un esprit d'équité vis à vis de l'algorithme MiniMax. Néanmoins,

l'élagage AlphaBêta pourrait aller bien au-delà avec un temps de calcul raisonnable.

Profondeur J1	Profondeur J2	Noeuds parcourus au total
1	1	192
2	2	1191
3	3	8514
4	5	45779
5	5	156980

TABLE 2 – Tableau récapitulatif du nombre de noeuds parcourus par AlphaBêta

2.3 Conclusion



On constate sur ce graphique à quel point l'élagage AlphaBêta est nettement plus optimisé et permet donc d'obtenir de très bons résultats avec des profondeurs de raisonnement plus grandes et des calculs plus courts.
Quand l'algorithme MiniMax atteint plus de 5 millions de noeuds parcourus, l'élagage AlphaBêta n'atteint que 156 980 noeuds pour un résultat équivalent.
Il est clairement le meilleur des deux et celui à privilégier pour la conception d'Intelligences Artificielles dans le cadre de jeux adversariaux.

3 Profondeur de raisonnement vs coups d'avance

3.1 Observations

Nous allons effectuer cette observation sur un terrain de jeu de taille 5x5.
Pour cette expérimentation, la profondeur de raisonnement du joueur 1 va augmenter tandis que le joueur 2 va gagner des coups d'avance supplémentaires lors de son premier tour.

Profondeur J1	Profondeur J2	Coups d'avance J2	Vainqueur
1	1	0	J1
2	1	1	J1
3	1	2	J1
4	1	3	J1
5	1	4	J1
6	1	5	J2
7	1	6	J2
8	1	7	J2
9	1	8	J1
10	1	9	J2

TABLE 3 – Tableau récapitulatif pour des coups d'avance progressifs

Observons maintenant ce qui se passe avec des coups d'avance statiques :

Profondeur J1	Profondeur J2	Coups d'avance J2	Vainqueur
1	1	3	J2
2	1	3	J2
3	1	3	J1
4	1	3	J1
5	1	3	J1
6	1	3	J1
7	1	3	J1
8	1	3	J1
9	1	3	J2
10	1	3	J1

TABLE 4 – Tableau récapitulatif pour des coups d'avance statiques

3.2 Analyse

On remarque qu'à partir d'un certain palier, la profondeur de raisonnement n'apporte plus grand avantage par rapport aux coups d'avance. En effet, lorsque

le joueur 2 atteint 5 coups d'avance, il est largement dominant et remporte la plupart des parties.
Cela paraît normal étant donné que le joueur 2 va pouvoir effectuer plus d'actions que le joueur 1 et augmenter son nombre de pions de façon conséquente dès le début de la partie, s'attribuant un avantage certain sur son adversaire.

Dans une partie équitable, la profondeur de raisonnement a une grande importance. Mais quand des coups d'avance sont attribués à l'un ou l'autre des joueurs, la partie devient inégale.
Lorsque les coups d'avance n'évoluent pas et reste à un faible niveau, on constate néanmoins que la profondeur de raisonnement se montre plus forte la plupart du temps.

4 Amélioration

Afin d'améliorer ce programme, nous pourrions par exemple concevoir une classe Joueur afin d'avoir une instance pour le joueur 1 et une instance pour le joueur 2. Ces objets pourraient ainsi contenir une variable représentant le nombre de pions du joueur sur le plateau. Elles seraient alors incrémentées ou décrémentées en fonction du mouvement effectué. Ces variables pourraient tout aussi bien être ajoutées à une classe pré-existante bien entendu.
De cette façon, la fonction *getScore* pourrait être modifiée : sa complexité quadratique (deux boucles for imbriquées pour le parcours du terrain) deviendrait quasi-constante (un simple calcul : nombre de pions du joueur / nombre de pions total). On rappelle que la fonction *getScore* est souvent appelée par les algorithmes d'Intelligence Artificielle.
Ainsi, le programme serait grandement optimisé et gagnerait en vitesse de calcul.