Dossier de validation
IA

Jeu Antinomy

ALTUN Zeynel
BOUIGADER Salima
DOMI Ronald
LEVASSEUR Bastien
MARMEY Loïc
RÉGNIER Léo

Groupe 10

Explication de l'algorithme utilisé :

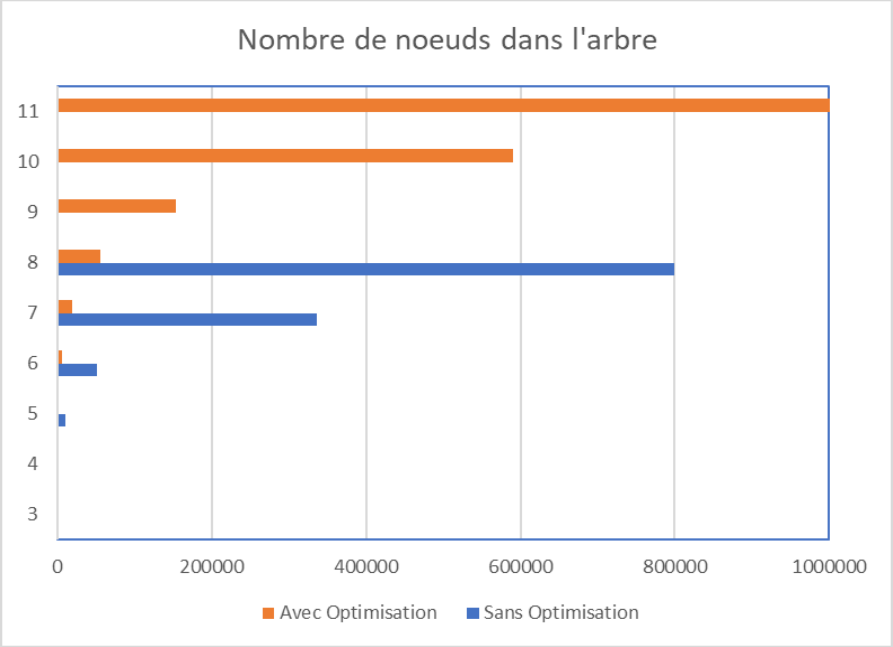
Nous avons travaillé avec un algorithme d'arbre min/max.
Lors de l'implémentation nous avons dû faire plusieurs choix et compromis.

Le premier choix a été la façon de construire l'arbre. La solution optimale serait de faire un tour complet de jeu, d'évaluer son heuristique et annuler le tour pour explorer d'autres branches de l'arbre. Dans ce projet on a implémenté une version plus basique, ou dans chaque choix, on fait un clone du jeu. Avec cette solution nous perdons en termes de mémoire et de vitesse comparé à la solution optimale.

Sans aucune optimisation, la profondeur maximale atteignable est 7. Ou nous atteignons un temps de calcul de 2-3 secondes, et avec avec profondeur 8 le calcule dépasse une minute. Avec profondeur 9 après avoir attendu 1m30, il y a un dépassement de la mémoire.

Pour optimiser nous avons implémenté la coupure alpha-beta.

	Sans Optimisation	Avec Optimisation
3	230	104
4	1468	334
5	9556	1453
6	50397	5427
7	335580	18462
8	803327	55862
9	Non calculé	153798
10	Non calculé	590709
11	Non calculé	1776017



Choix lié à l'aléatoire :

Lors d'un paradoxe, les cartes de la main du joueur sont mélangées et échangées avec les 3 premières cartes à gauche ou à droite du joueur.

Pour simplifier la création de l'arbre, nous avons choisi de ne pas générer toutes les possibilités à la fin du paradoxe.
Pour gagner du temps de calcul, nous avons choisi de prendre une seule permutation parmi celles possibles.
Nous aurions aussi pu générer tous les mélanges possibles et les parcourir un à un.
Ou nous aurions pu utiliser la seed du jeu, l'IA aurait ainsi pu connaître le future, nous gagnons alors en calcule et en précision de l'IA mais cela peut être considéré comme de la triche.

Dans le cas d'égalité de clash, on tire aléatoirement une carte de la main de chaque joueur, la carte la plus haute déterminant le gagnant. La défaite ayant des conséquences trop élevées, on a décidé de ne pas parcourir ni d'explorer l'arbre si on rencontre un clash avec une égalité. L'IA est déterministe, elle va donc éviter ce choix, par prudence.

Pour simplifier encore, l'IA connaît toutes les cartes du joueur adverse même lors des premiers tours. Lors d'une partie classique, nous connaissons très vite la main de l'adversaire, cette simplification n'a donc pas beaucoup d'impact sur la suite de la partie.

Heuristique:

Ici l'heuristique est le score qui va nous permettre de choisir les bons coups à jouer,
nous avons donc accorder des points en fonction de plusieurs paramètres :

- La somme de notre main
- Notre score
- Le score adverse
- Existence d'un gagnant

La formule de l'évaluation est généralement comme suit:

- +points*100
- -pointsAdversaire*100
- +sommeMain*10
- +infine si on a gagné
- -infinie si on a perdue

On a aussi essayé d'autres heuristiques, comme la présence de "doubles" en main ou la position du joueur sur l'axe, mais elles ne se sont pas avérées très concluantes et les privilégier n'amenait pas ou trop peu à la victoire.

Les différentes difficultés d’IA :

Nous avons implémenté dans le jeu 3 difficultés. Pour faire varier ces difficultés, nous changeons le nombre de coups qu'il anticipe (la profondeur de l’arbre parcouru).

Tout d'abord le niveau facile, nous avons choisi de mettre une profondeur de 1. L'ordinateur va choisir le meilleur coup qu'il a sous la main sans anticiper les coups adverses. Cette IA est battable plutôt facilement pour un bon joueur qui anticipe au minimum l'état futur du jeu.

Le niveau moyen utilise une profondeur de 3. L'ordinateur anticipe donc un coup à l'avance. Cette IA est battable mais il faut beaucoup réfléchir avant de jouer un coup car l'ordinateur anticipe notre prochain coup.

Pour finir l’IA difficile nous avons choisi de mettre une profondeur de 7, cette IA n'a pas encore été battue par un joueur.

Performances entre IA :

Pour vérifier que les difficultés soient cohérentes entre les différents niveaux nous avons fait un programme qui fait s'affronter les différentes IA sur plusieurs parties.

```
L IA moyenne gagne a 84.0 contre la facile.
L IA difficile gagne a 90.0 contre la facile.
L IA difficile gagne a 70.0 contre la moyenne.
```

Les résultats sont cohérents, la différence de profondeur reflète la difficulté. On peut se demander comment cela est possible qu'une IA facile puisse gagner contre une autre plus difficile. Cela est lié au fait qu'il y ait de l'aléatoire dans ce jeu.