HUREL Arnaud
Elèves-Ingénieurs en master Informatique : Système Intelligent

Métaheuristique et jeu

Projet de construction d'une IA du jeu 2048

A l'attention de Monsieur CAZENAVE

À Paris le 8 mars 2015

Table des matières

1 Introduction sur le 2048

Le jeu 2048 a été créer en 2014 par Gabriele Cirulli à l'age de 19 ans, le but de ce jeu est de faire glisser des tuiles sur un grille de 4x4 cases afin d'obtenir une tuile égale à 2048. En effet, les tuilles ont pour valeur des puissance de 2. A chaque tour, le jeu fait apparaitre de manière aléatoire soit une tuile 2, soit une tuile 4 (90% pour la tuile 2 et 10% pour la tuile 4). Pour jouer, l'utilisateur a le choix entre quatres coup : Nord, Sud, Est et Ouest. Ceux-ci auront, respectivement, un effet de comprésion des tuiles vers le haut, le bas, la droite et la gauche.

Le jeu ainsi que les IAs ont été développer en python, vous pouvez y acceder en toute liberté sur mon Github à l'adresse suivante : https ://github.com/phatryun/2048

2 Algorithmes utilisés

Une fois le jeu créé, nous avons mis en place différents algortithme en vu de créer un intelligence artificiel capable de résoudre se casse tête. Nous avons alors regarder sur internet afin de se renseigner sur les différents algorithme utilisé

Pour des raisons techniques, lors de nos test nous avons choisi de faire tourner nos algortimes que avec une profondeur de 4.

2.1 Algorithme Naif

Cet algorithme consiste à déterminer le meilleur coup parmis tous ceux possible. Il peut être considéré comme naif, car il vas tout simplement effectuer tous les coups de manière exhaustif sans logique particulière. Dans notre fichier source il corresdond aux fonctions *nextMove* et *nextMoveRecur*. Voici son derroulement :

- Tant que la profondeur de recherche n'est pas égale à celle voulut, on va effectuer chaque coup possible (Nord, Sud, Est, Ouest)
- On va, ensuite, ajouter de manière aléatoire la nouvelle tuile pour ensuite calculer le score heuristique de la grille obtenue.
- Puis on va étudier ces fils qui lui retourneront leur meilleur score calculé par l'heuristique choisit.
- De ces dernier, on retiendra le coup pour le quel le score de l'heuristique à été le plus élévè.
- Enfin on va ajouter à son propore score celui de son meilleur fils mais de façon pondéré.

2.2 Expectimax

Cet algorithme est répartie en deux étapes : la première consiste à rechercher le meilleur coup possible calculé en maximisant le score obtenu grâce à une heuristique et ensuite de minimiser l'impacte de l'ajout aléatoire des nouvelles tuiles. Dans notre code nous le retrouverons à travers la fonction *player_max* qui aura pour objectid de maximiser le choix du coup et la fonction *player_expect* qui va minimiser les placements de la nouvelle tuile.

Voici plus en détail leurs déroulement :

- Premièrement, notre condition d'arret sera quand la profondeur de recherche sera égale à 0 ont calculera notre score heuristique
-

3 Heuristiques utilisé

lol

3.1 Utilisation du score

lol

3.2 Gradient

lol

3.3 Ordre des valeurs des cases

lol

3.4 Conjugaison d'heuristique : Ordre, fusions possibles, cases libres

lol

4 Résultats obtenus

Algorithme	Naif				
Heuristique	Score	Gradiant	Gradient ++	ordre	ordre ++
0	13	3	0	0	0
1024	12	5	0	0	0
2024	0	13	0	0	0
4096	0	4	0	0	0
8192	0	0	0	0	0
temps moyen	0	239	0	0	0

Algorithme	Expectimax				
Heuristique	Score	Gradiant	Gradient ++	ordre	ordre ++
0	0	0	0	0	0
1024	0	0	0	0	0
2024	0	0	0	0	0
4096	0	0	0	0	0
8192	0	0	0	0	0
temps moyen	0	0	0	0	0