<![endif]–>

Charle Vandermies 15123

Victor Smits 16107

Projet AI Quarto

Pour notre travail de fin de 2ème Bac en ingénieur industriel à l'Ecam orientation Génie Electrique, notre professeur d'informatique nous a mis au défi de créer une intelligence artificielle jouant au jeu Quarto.

Pour ce faire, il nous a mis à disposition une librairie ainsi qu'un code pour la structure du jeu.

Fonctionnement

Pour créer notre intelligence artificielle, nous avons décidé d'utiliser la librairie EasyAI http://zulko.github.io/easyAI/index.html. Grâce à cette librairie, nous avons pu utiliser différents algorithmes du type élagage Alpha-Beta. Dans notre cas, nous utilisons plus précisément l'algorithme Negamax https://en.wikipedia.org/wiki/Negamax ainsi que la méthode Solving qui va résoudre la partie en utilisant Negamax avec différentes profondeurs de recherche.

Pour générer le coup qui sera joué, différentes intelligences sont disponibles (voir Intelligence).

L'intelligence principale, l'intelligence *client*, utilise la fonction id_solve de la class Solving qui va nous retourner plusieurs informations dont le mouvement le plus intéressant pour arriver à la victoire le plus rapidement possible.

Utilisation

Pour jouer au jeu, vous devez lancer 3 terminaux depuis le dossier où est enregistré le jeu.

1. [Fenêtre 1] : Server du jeu
2. [Fenêtre 2] : Client 1 du jeu
3. [Fenêtre 3] : Client 2 du jeu

Vous pouvez lancer 2 fois le même client.

Intelligence

1. client : Utilisation de id_solve *(recommandé)*
2. clientB : Utilisation de Negamax avec transposition table
3. user : Pour jouer contre l'IA
4. rdm : IA agis 100% aléatoirement
5. prof : IA d'origine

Lancer une partie

Server

```
./quarto.py server --verbose
```

Client

```
./quarto.py <Intelligence> <Nom> --verbose
```

Vous avez aussi la possibilité de lancer les clients et server sur différentes machines, il vous faudra donc préciser l'IP du host ainsi que le port de communication (*Par défaut le host = localhost et le port = 5000*) :

Server disant

```
./quarto.py server --verbose --host=<IP> --port=<Port>
```

Client distant

```
./quarto.py <Intelligence> <Nom> --verbose --host=<IP> --port=<Port>
```

Test AI

Grâce au Test AI, vous pouvez tester différentes combinaisons avant de programmer l'intelligence. Il suffit d'utiliser cette commande:

```
./quarto.py ai --verbose --algo=<algorithme> --depth=<depth> --tt
```

1. algorithme : Choisissez entre Negamax, SSS, solve. Default = Negamax
2. depth : Profondeur de la recherche de l'AI . Default = 3
3. tt : Enregistre True et active la table de transposition . Default = False

Ces 3 arguments ne sont pas obligatoires.

Le test AI n'a pas besoin du serveur, il peut être lancé dans une seule fenêtre de terminal.