TDLOG – séance n° 4
Compléments sur Python & interfaces graphiques : TP

Xavier Clerc – xavier.clerc@enseignants.enpc.fr
Thierry Martinez – thierry.martinez@inria.fr
Olivier Cros – olivier.cros0@gmail.com

17 octobre 2016

À rendre au plus tard le 27 octobre 2016 sur educnet

On continue lors de cette séance à travailler sur le jeu commencé lors de la séance précédente. L'objectif est d'améliorer le code existant, d'ajouter la possibilité de lire la grille depuis un fichier, et enfin d'ajouter une IA sommaire.

1 Refactoring

La première tâche consiste à améliorer le code écrit lors de la séance précédente à partir des annotations et de la proposition de corrigé.

La seconde tâche consiste à ajouter des exceptions afin de gérer les différentes erreurs (*p. ex.* taille de la grille paire) :

- définition de la ou les exceptions ;
- levée d'exception lorsqu'une erreur se produit ;
- rattrapage des exceptions aux endroits propices.

2 Utilisation de fichiers

Afin de tester facilement l'algorithme de la section suivante, il est utile de pouvoir charger des grilles depuis des fichiers. Nous proposons d'utiliser le format CSV [1] qui consiste en une représentation simple de données tabulaires, séparant les valeurs d'une même ligne par des virgules. Par exemple une grille de taille 5 peut se représenter au format CSV de la manière suivante :

1. https ://fr.wikipedia.org/wiki/Comma-separated_values

```
20,200,10,20,20
100,200,10,5,10
20,20,0,10,50
50,200,50,50,5
100,200,5,100,5
```

3 IA

Afin de pouvoir affronter l'ordinateur, nous proposons d'implémenter l'algorithme min-max [2], qui permet de déterminer le *meilleur* coup dans une situation donnée.

L'algorithme repose sur les opérations suivantes :
- détermination du caractère final d'une position (*i. e.* la partie est terminée) ;
- génération des coups possibles dans une position donnée ;
- application d'un coup à une position, pour créer une nouvelle position ;
- évaluation d'une position donnée, sous la forme d'un *score*.

L'algorithme peut s'écrire en pseudo-code de la manière suivante :

```
min_max(position, profondeur, is_max)
    si est_finale(position) ou (profondeur <= 0) alors
        renvoyer evaluation(position)
    sinon
        si is_max alors
            val = -infini
            pour tout coup dans generer_coups(position)
                val = max(val,
                          min_max(appliquer(coup, position),
                                  profondeur - 1,
                                  faux))
            renvoyer val
        sinon
            val = +infini
            pour tout coup dans generer_coups(position)
                val = min(val,
                          min_max(appliquer(coup, position),
                                  profondeur - 1,
                                  vrai))
            renvoyer val
```

Vous devez implémenter l'algorithme, en effectuant toutes les adaptations nécessaires pour qu'il soit possible à un joueur humain de jouer contre l'ordinateur.

2. https ://fr.wikipedia.org/wiki/Algorithme_minimax