

Othello

Intelligence Artificielle

1 Généralités

L'objectif de ce projet est de construire, à vous tous, un jeu d'Othello sur ordinateur, à l'aide d'OCaml. Votre dévoué enseignant a passé ses vacances à écrire (et réécrire) un logiciel, à le diviser et le modulariser. Cet Othello peut être joué à deux sur une machine ou sur deux machines ou encore contre une intelligence artificielle disposant de nombreux niveaux de difficulté. Vous avez maintenant sous la main toutes les briques nécessaires pour reconstruire le programme, documentation y compris. Tout, mais pas le code source.

Comment obtenir un Othello en état de marche ? Dans une invite de commande, allez dans le répertoire qui contient tous ces fichiers et tapez

```
make .cmd
```

La commande produira un fichier `reversi.exe`, sur lequel vous pouvez double-cliquer ou que vous pouvez lancer en ligne de commande. Si vous tapez

```
reversi.exe --help
```

un mini-manuel s'affichera, avec la liste des options de cet Othello.

Que devez-vous faire ? Dans l'état actuel des choses, le jeu entier a été programmé par votre tyrannique enseignant. Il s'agit d'une injustice à faire cesser aujourd'hui même, en remplaçant, pièce par pièce ce qu'il a fait.

Pour ce faire, choisissez un module, consultez sa documentation, fournie dans le fichier `index.html` du répertoire `doc`, supprimez le fichier `.cmo` correspondant, qui ne vous servira plus à rien et implantez ce module dans un fichier `.ml`.

Votre objectif est d'obtenir un fichier `.ml` tel que

```
make .cmd
```

produise un nouveau `reversi.exe` qui fonctionne aussi bien – voire mieux – que l'original.

À quoi avez-vous droit ? Vous avez 10h. Vous avez le droit à tous les documents, à Internet, à prendre des pauses, à discuter, à poser des questions. Par contre, vous n'avez pas le droit de faire faire votre travail par quelqu'un d'autre.

Vous avez le droit de travailler en binôme. Par contre, les notes seront individuelles. En d'autres termes, pour chaque fonction ou structure de données que vous écrivez, notez dans les commentaires (balise `@author`) le nom de la personne qui s'en est occupée. Seule cette personne recevra des points !

Comment serez-vous notés ? Pour chaque module, vous trouverez un barème. Pour avoir le maximum de points, il faut

- que le module compile
- que le module respecte son interface et s'intègre à un Othello fonctionnel
- que les difficultés que vous avez rencontrées et les limitations de vos algorithmes soient notés dans les commentaires
- que vos fonctions soient propres et lisibles – une fois que vous avez fait fonctionner quelque chose, n'hésitez pas à le retravailler pour améliorer la clarté

- que vos algorithmes soient documentés – ceci compte pour la moitié des points
- que vous ayez marqué votre nom devant chacune de vos fonctions écrites
- éviter de parler politique.

2 L'Intelligence Artificielle (40 à 60 points)

Le module `Ia` (fichier `ia.ml`) définit l'Intelligence Artificielle d'Othello. Plus précisément, ce module contient les fonctions nécessaires pour permettre à l'Intelligence Artificielle de jouer avec plusieurs coups d'avance. Pour ce faire, on emploie des techniques assimilées à "si je joue ceci, alors tu peux jouer cela, auquel cas je pourrais jouer ceci mais toi tu pourrais jouer cela, ce qui va finir par m'amener dans une situation que je n'aime pas – par contre, si j'avais joué autre chose..."

Le module `Ia` lui-même n'a pas pour rôle de déterminer quelles sont les situations appréciables ou redoutées – ceci est du domaine du module `Evaluateur`. Par contre, il a pour tâche de regarder ce qui se passe s'il joue un coup, puis si l'adversaire joue un coup, puis...

Pour ce faire, on emploie l'algorithme Minimax ou une de ses variantes plus complexes mais plus rapides, telles que l'élagage α - β (ou Alpha-Beta Pruning), NegaScout...

Ce module demande beaucoup d'initiative et de recherches bibliographiques.

Dépendances Le module `Ia` utilise

- `Evaluateur`, pour évaluer l'intérêt d'une situation.
- `Plateau`, pour manipuler les couleurs et les plateaux.
- `Arbitre`, pour déterminer quels sont les coups possibles ou le vainqueur dans une situation donnée.

Vous êtes donc encouragés à consulter la documentation de ces modules et à vous servir de leur contenu. Vous n'avez pas à les réimplanter. Précisons que le module `Ia` n'a pas à appeler directement les fonctions du module `Evaluateur`.

Utilisateurs Le module `Ia` est utilisé par le module `Main`, pour initialiser l'Intelligence Artificielle d'une partie.

Suggestions

- Commencez par vous documenter sur l'algorithme Minimax.
- Pour le type `t`, vous pouvez utiliser

```
type t = {
  couleur : Plateau.couleur; (**La couleur du joueur IA *)
  evaluateur : Evaluateur.t; (**La fonction d'évaluation*)
  profondeur : int (**Le nombre de coups à considérer d'avance *)
}
```

BON COURAGE !