CLEMENT Théo & FAUCHON Margot - IMAC1 (2020-2021)

Reversi

Programmation et Algorithmique 1 - Projet de fin de semestre

MÉTHODE DE TRAVAIL

Nous avons utilisé github et gitKraken pour nous transmettre les fichiers et suivre l'historique des modifications.

CARACTÉRISTIQUES DU PROJET

Les structures "Liste"

Notre programme a pour particularité d'utiliser une structure de plus que celles demandées dans le sujet.

Dans le but de pouvoir proposer une assistance aux joueurs qui ne connaîtraient pas le jeu, nous voulions calculer (et afficher si besoin) toutes les cases jouables avant de permettre au joueur de choisir une case. Nous avons donc une structure **liste**, qui comporte un tableau de coordonnées. Tout le fonctionnement du jeu repose sur cette structure puisque les fonctions qui déterminent si le joueur peut jouer, si la case choisie par le joueur est valable et si la partie est terminée utilisent les listes des cases potentielles (adjacentes à un pion adverse) et des cases jouables (adjacentes à un pion adverse et permettant un gain).

Deux fonctions permettent de les calculer :

- **chercheCasesPotentielles** : parcourt le tableau, si une case est vide et adjacente à un jeton de la couleur adverse, elle ajoute ses coordonnées à la liste
- **chercheCasesJouables** : parmi les cases potentielles retenues, ajoute toutes celles qui permettent de faire un gain

Des fonctions récursives

Plusieurs fonctions du programme sont récursives :

- la fonction qui teste pour une case vide si un gain est possible dans une certaine direction **testGainDirection** : cette fonction est lancée par **TestGainCase** dans chacune

des directions où il y a un jeton adverse, tant que la case existe et contient un jeton de la couleur adverse, elle se rappelle elle même pour regarder la case suivante dans cette direction. La gain est alors augmenté de 1 pour chaque jeton adverse trouvé.

- la fonction qui retourne les pions adverses encadrés une fois qu'un joueur a joué **retournePionsDirection** : cette fonction est appelée par **retournePions** pour toutes les directions qui permettent un gain, tant qu'elle est exécutée dans une case contenant un jeton adverse, elle change la couleur du pion et se rappelle elle même dans la case suivante, jusqu'à tomber sur un pion de la couleur du joueur dont c'est le tour.

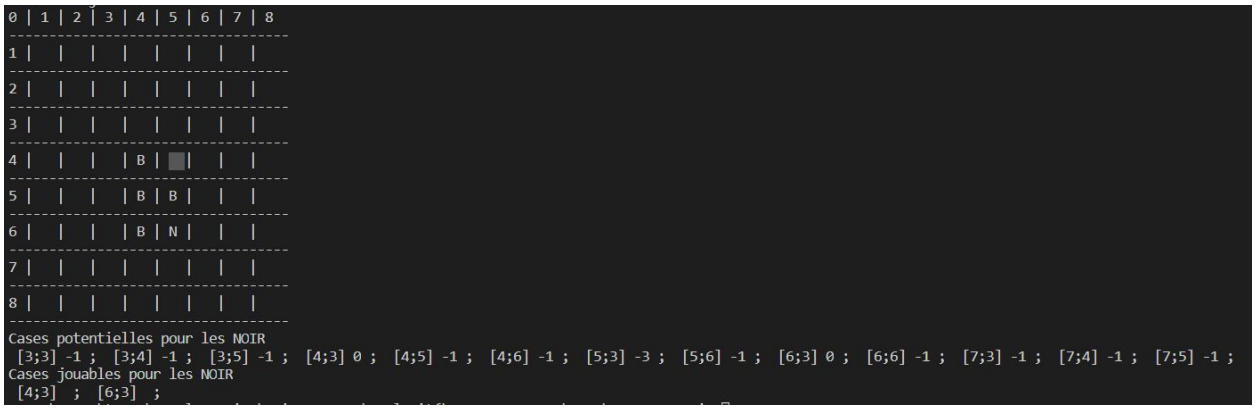
DIFFICULTÉS RENCONTRÉES

Les pointeurs sur jetons

Au départ, nous voulions que les listes contiennent des pointeurs sur jeton. Nous avons changé leur définition pour ne garder que les coordonnées parce que nous avions des difficultés à vider les listes d'un tour à l'autre, ainsi que pour les parcourir et récupérer les informations qui nous intéressaient. On se retrouvait par exemple avec des cases jouables de coordonnées [1, 1.481975].

Le calcul des cases potentielles et jouables

En jeu, il arrivait que le programme refuse une case qui aurait dû être jouable. Certaines configurations de jetons nous ont posé problème. Par exemple, dans la configuration en T, l'algorithme oubliait toujours une case sur les trois jouables :



Le problème était une erreur de raisonnement dans les fonctions de calcul de gain qui rendaient parfois des nombres négatifs ou nul.

AMÉLIORATIONS POSSIBLES

Jouer contre l'ordinateur

Il aurait été possible de créer une intelligence artificielle pouvant jouer contre l'utilisateur. Pour cela, il faudrait inclure dans le programme une fonction qui joue à la place de l'un des deux joueurs et qui choisisse la case qui lui rapporte le plus gros gain ou les bords du plateau par exemple.

Ajouter une interface graphique

Une autre amélioration possible serait bien sûr l'ajout d'une interface graphique. Grâce à un logiciel de CAO, nous pourrions concevoir un plateau similaire au jeu physique du Reversi, permettant ainsi à l'utilisateur de mieux se repérer au long de la partie et de rendre le jeu plus fluide et plus agréable. En plus de cet ajout, nous aurions pu permettre au joueur de cliquer sur la case qu'il souhaite utiliser pour améliorer l'immersion dans le jeu.

ANNEXE

Vue d'ensemble des fonctions

Beaucoup de fonctions s'appellent les unes les autres :