Programmation modulaire en C – Joueur d'Othello intelligent

1 Modalités

Ces projets sont à réaliser par groupes de 2 personnes (idéalement). Une phase préliminaire du projet consistera ainsi pour chaque étudiant à trouver un partenaire avec qui il réalisera le travail.

Le travail s'étalera sur l'ensemble de la période janvier 2013–avril 2013. Afin d'assurer une progression constante de chaque groupe de travail, 3 évaluations auront lieu, espacées de 4 semaines environ (le détail de ces jalons sera fourni ultérieurement). Chacune de ces évaluations se basera sur un rapport de 8 à 10 pages sur le travail effectué et sur le programme développé. Chaque rapport contiendra une description et un rappel du projet, les difficultés rencontrées, les structures de données utlilisées, les résultats obtenus et des indications sur le points à améliorer. Concernant le programme, la notation prendra en compte la présence des fonctionnalités demandées, le fonctionnement sans erreur du programme et sa modularité : ces éléments seront évalués au cours d'une soutenance sur machine.

L'évaluation finale du projet (c'est-à-dire la troisième évaluation) tiendra compte, en plus des points précisés auparavant, de la qualité « othellistique » du joueur programmé : les programmes réalisés s'affronteront au cours d'un tournoi et les résultats obtenus compteront, comme points bonus, dans la notation du projet.

Au cours de l'avancement de l'année, des documents concernant plus précisément les tâches à effectuer et l'organisation du projet seront régulièrement déposés à l'url suivante :

http://www.lif.univ-mrs.fr/~liva/doku.php?id=teaching:20122013:projetalgo

2 Description du sujet

2.1 But du projet

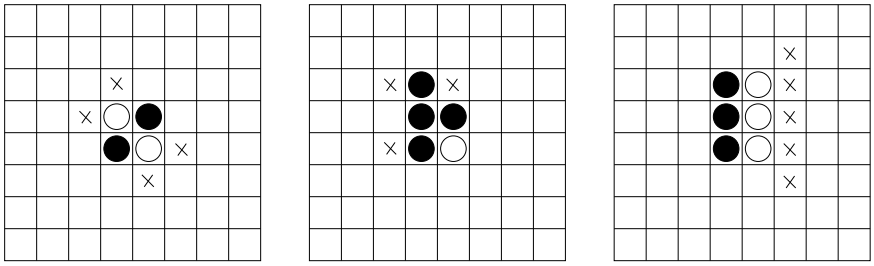
L'objectif de ce projet est de créer un joueur artificiel capable de jouer à Othello à un niveau de jeu élevé. En particulier, à la fin du projet il doit être pratiquement impossible aux étudiants de battre le programme.

2.2 Othello-Reversi

2.2.1 Présentation

Othello est un jeu de réflexion à deux joueurs qui se joue sur un damier de 64 cases, et où les pions sont de deux couleurs différentes. Le but d'une partie est d'avoir au final le plus grand nombre de pions de sa couleur posés sur le damier.

Les règles de ce jeu peuvent se trouver partout sur internet, aussi on se contente de rappeler la position de début de partie et l'état du damier après 2 coups (les pions noirs débutent la partie) :



2.2.2 Les algorithmes MinMax et AlphaBeta

La conception d'un algorithme permettant à l'ordinateur de jouer constitue un des axes principaux de ce projet. Pour parvenir à faire jouer l'ordinateur, on prendra soin d'élaborer une fonction d'évaluation permettant de mesurer la qualité d'une position, fonction qu'on utilisera dans le cadre d'une recherche du meilleur coup du type MinMax. La recherche MinMax basique consiste à simuler le raisonnement d'un être humain : à une position donnée, un joueur regarde toutes les possibilités de jeu qui lui sont offertes et pour chacune d'entre elles, évalue la qualité des ripostes possibles de son adversaire. Le joueur essaie alors de maximiser la qualité de la position sachant que son adversaire fera tout pour la minimiser. La figure 1 représente un arbre de jeu sur lequel on fait une recherche MinMax.

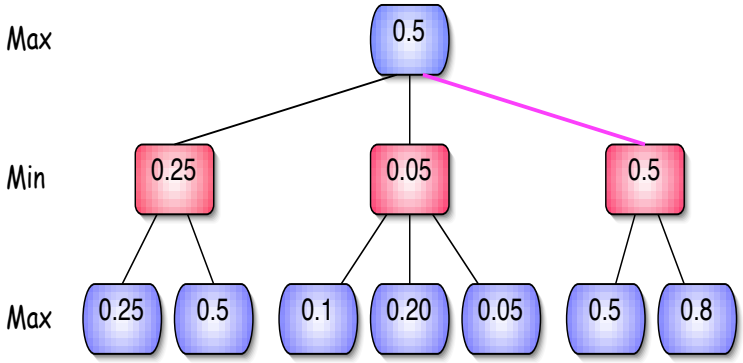


Figure 1 – Recherche MinMax : chaque branche de l'arbre représente un coup possible et chaque nœud une position de jeu. Les nombres dans les rectangles arrondis sont les estimations des positions par une fonction d'évaluation. Lorsqu'on est à un niveau Max, le joueur (en l'occurrence l'ordinateur) cherche à Maximiser la valeur de la position alors que c'est l'inverse dans le cas où on se trouve à un niveau Min (adversaire de l'ordinateur). Le coup choisi par l'ordinateur est le dernier (branche de droite) car il lui permettra d'atteindre une position de valeur au moins égale à 0.5.

Plusieurs améliorations de l'algorithme MinMax (très coûteux en fonction de la profondeur de l'arbre) existent dans la littérature, dont la plus connue est l'algorithme AlphaBeta qui permet des gains en rapidité de recherche. De plus, la programmation de MinMax peut se faire en implantant l'algorithme NegaMax qui possède un code plus compact. Voici plusieurs liens sur la programmation de jeu de stratégies et sur les paramètres à prendre en compte pour réaliser une bonne fonction d'évaluation pour le jeu d'Othello :

- http ://turing.cs.pub.ro/auf2/html/chapters/chapter3/chapter_3_4_3.html description de l'algorithme Alpha-Beta avec quelques questions-réponses ;
- *http ://users.informatik.haw-hamburg.de/~owsnicki/search.html* fait une revue des méthodes de recherche dans les arbres de jeu et la programmation de la méthode NegaMax ;
- *http ://www.rohleder.de/~mike/schach/etc/search.html* résume les méthodes utilisées dans la recherche d'arbres de jeux pour les échecs ;

– *http ://www.ffothello.org/* est le site de la Fédération Française d'Othello et propose des liens intéressants pour la programmation d'un joueur d'Othello dans la section informatique ;
– Ajax (`http://www.onlinespiele-sammlung.de/othello/othello-reversi-games/ajax/`) est une applet Java qui vous permettra de mesurer les progrès de votre joueur d'Othello. Vous pourrez commencer à considérer votre programme comme bon lorsqu'il sera en mesure de vaincre Ajax en mode expert – avec, bien entendu, la même vitesse de réflexion.

2.3 Travail à faire

Le travail à effectuer consiste en la réalisation d'une application en mode texte permettant de jouer à Othello contre l'ordinateur. Les points suivants devront en particulier être respectés :
– le programme doit détecter la fin de la partie,
– la gestion du joueur qui doit jouer doit être faite par le programme – dans le cas où un joueur doit passer son tour, l'ordinateur doit le signaler et passer la main à son adversaire,
– pour recommencer une partie, l'application ne devra pas être relancée,
– le niveau de jeu de l'ordinateur doit être ajustable,
– le programme doit offrir la possibilité d'annuler tous les coups joués,
– il doit être possible d'arrêter une partie en cours de jeu, de l'enregistrer, puis de la reprendre,
– les paramètres de la fonction d'évaluation seront enregistrés dans un fichier,
On notera que les parties algorithmique (algorithme alpha-beta) et 'analytique' (création d'une fonction d'évaluation) sont les points essentiels de ce projet.

Extensions

Comme exemples d'extensions, on pourra :
– faire en sorte que l'ordinateur s'améliore au cours des parties qu'il joue en prenant en compte les erreurs qu'il a pu faire ou en apprenant en jouant contre lui-même (apprentissage par renforcement),
– modéliser la fonction d'évaluation à l'aide d'un réseau de neurones qui sera appris grâce à de l'apprentissage par renforcement,
– faire une interface graphique.