INF8175
Intelligence Artificielle : Méthodes et Algorithmes

Projet - Agent intelligent pour le jeu Abalone

Equipe Thomaxel
Axel Baudot 2297081
Thomas Perrin 2229377

Travail remis à
Prof. Quentin Cappart

En date du:
7 Décembre 2023

Introduction

Ce rapport détaille le développement et l'implémentation d'un agent intelligent pour le jeu d'Abalone. On présentera plusieurs heuristiques utiles pour ce jeu, et la manière dont elles ont été utilisées dans le cadre d'une stratégie de recherche.

On détaillera ensuite la stratégie de recherche basée sur une variante du Minimax appelée Negamax augmentée d'un élagage alpha-beta, d'une table de transposition et d'une recherche de quiescence. Une telle recherche peut nécessiter beaucoup de temps de calcul et on présentera la méthode utilisée pour ne pas dépasser le temps alloué à l'agent.

Enfin, on évaluera les performances des agents lors de différentes parties. Cela comprend l'évaluation des différentes heuristiques avec des agents simplistes et celle des stratégies de recherche, basée à la fois sur des configurations de jeu simplifiées et des parties complètes.

Méthodologie

Heuristiques

L'heuristique utilisée est obtenue par la combinaison de plusieurs heuristiques, à la fois défensives et offensives. L'heuristique finale est ensuite calculée comme étant la différence entre l'heuristique pour le joueur et l'heuristique pour l'adversaire. Les heuristiques choisies sont les suivantes :

1. **Le nombre de billes éjectées** : critère évident puisque déterminant le vainqueur en fin de partie. Cette heuristique va privilégier les actions éjectant une bille, comportement souhaitable dans la plupart des cas.

2. **La distance normalisée au centre du plateau** : Les billes au centre du plateau ne peuvent pas être éjectées immédiatement et ont plus d'options de déplacements qu'au bord du plateau. Occuper le centre tend également à repousser vers les bords les billes de l'adversaire, qui seront donc plus faciles à pousser vers les côtés et à éjecter.
   Enfin, en cas d'égalité, il est important d'avoir ses billes au centre car c'est l'équipe ayant la plus petite somme des distances entre ses billes et le centre du plateau qui gagne.

3. **L'adjacence** décrit la proximité des billes entre elles en comptant le nombre moyen de billes alliées autour de chaque bille. Lorsque nos billes sont groupées, il est plus difficile pour l'adversaire de les pousser, et on sera plus souvent en supériorité numérique stricte pour pousser l'adversaire.
   A l'inverse, minimiser cette métrique pour l'adversaire revient à diviser ses billes, permettant de prendre l'avantage et le pousser vers l'extérieur.
   Cette heuristique est une variante de la **cohésion**, une autre heuristique étudiée qui consiste à calculer la distance totale entre les billes, mais l'adjacence a été préférée pour ses meilleurs résultats à un coût calculatoire moindre.

Une autre heuristique qui a été explorée est **la pression**, donnée par une combinaison linéaire du nombre de possibilités de pousser et d'éjecter les billes de l'adversaire. Cependant, le calcul implique

de considérer toutes les actions possibles et les états qui en résultent, ce qui revient à pousser la recherche au niveau de profondeur suivant sans pour autant retenir d'autres informations utiles ou remplir la table de transposition. Cette heuristique n'a donc pas été retenue à cause de son coût calculatoire et de son intérêt limité.

Stratégie de recherche

La stratégie de recherche peut être résumée ainsi : c'est un negamax avec élagage alpha-beta, table de transposition et recherche de quiescence. On détaillera par la suite chacun de ces termes.

Negamax

Le negamax est une variante du minimax qui tire parti du fait que le jeu est à somme nulle : le score de l'état du plateau pour un joueur est l'opposé du score de ce même plateau pour l'autre joueur. Il simplifie le code car on peut utiliser la même fonction pour les nœuds max et min.

Il permet d'améliorer l'utilité de la table de transposition (cf. plus bas) car on retrouvera un score basé sur la configuration du plateau indépendamment du joueur dont c'est le tour.

La recherche est faite avec une profondeur limitée. Les nœuds terminaux (correspondant à une fin de partie) auront un score de $+\infty$ en cas de victoire, $-\infty$ en cas de défaite et 0 pour une égalité. Quand la profondeur limitée de la recherche fait qu'elle s'arrête à un nœud non-terminal, le score du nœud correspond au calcul de l'heuristique choisie pour cet agent.

Élagage alpha-beta

L'élagage alpha-beta permet de réduire le nombre de nœuds à explorer en éliminant les branches qui ne peuvent pas améliorer le score du nœud parent. Il a amélioré considérablement la performance de l'algorithme et permet d'explorer des arbres de plus grande profondeur.

Table de transposition

La table de transposition permet de stocker les configurations déjà explorées et leur score. L'implémentation est un dictionnaire : la clé est la représentation textuelle de la configuration du plateau; la valeur est le score de cette configuration, associée à la profondeur de l'exploration qui a permis de déterminer ce score.

Cette implémentation simple (pas de hashing de Zobrist pour la clé par exemple) a été jugée suffisante au vu des caractéristiques de performance des dictionnaires en Python, notamment écriture et lecture de complexité linéaire par rapport à la longueur de la clé, et les mécanismes de gestion des collision déjà implémentés.

Il est important de conserver la profondeur d'exploration associée à un score afin de pouvoir affiner le score dans des recherches suivantes.

Lorsque l'on recherche un score dans la table, on doit également prendre en compte la limite de tour de jeu : une configuration jugée équilibrée en milieu de partie peut assurer un victoire à deux tours de la fin du jeu.

Recherche de quiescence

Lorsqu'un nœud de la recherche est le résultat d'une poussée, il peut être utile de calculer des scores plus profonds pour vérifier que cette poussée (a priori souhaitable) est une action judicieuse (on peut potentiellement se mettre en danger pour les tours suivants). Les nœuds résultant d'une poussée sont donc sujet à une recherche de quiescence, d'une profondeur unitaire. Autrement dit, on va calculer les scores de tous les nœuds suivants pour avoir une meilleure estimation du score du nœud parent.

Gestion du temps et de la profondeur de recherche

Outre la recherche de quiescence, la profondeur de recherche est fixée pour l'agent de manière à pouvoir maintenir cette profondeur pendant toute la partie sans dépasser le temps alloué de 15 minutes.

Si l'on est à moins d'une minute de la fin du temps alloué, la profondeur de recherche est fixée à 2, valeur qui permet à coup sûr de finir la partie dans la limite de temps, sans toutefois dégrader totalement la performance de l'agent.

Résultats et évolution de l'agent

Evaluation des heuristiques

Pour évaluer simplement les heuristiques, on utilise un agent "glouton" qui joue le coup le mieux noté par l'heuristique. Le choix de l'action est par conséquent quasiment instantané et permet d'itérer rapidement lors de l'implémentation des heuristiques. De manière simpliste (heuristique dirait-on) on considère que l'heuristique de l'agent glouton vainqueur est de meilleure qualité.

Selon ce protocole, les heuristiques par ordre croissant de qualité sont :

- Billes éjectées
- Billes éjectées / distance au centre
- Billes éjectées / distance au centre / cohésion
- Billes éjectées / distance au centre / adjacence

Ce classement a été confirmé par les expérimentations avec des agents utilisant la recherche negamax.

Evaluation des stratégies de recherche

Configuration simplifiée

Dans un premier temps, on a évalué la correction des algorithmes utilisés par l'agent en le faisant jouer sur une configuration simplifiée pouvant être résolue (c'est-à-dire, permettant une recherche exhaustive). La configuration simplifiée implique moins de billes (3 par joueur) et un nombre très bas de tours (3 à 7 tours selon les expériences).

Les métriques d'intérêt sont alors le nombre de nœuds explorés et le nombre de scores retrouvés dans la table de transposition.

Par exemple, pour une partie de 5 tours, le minimax résout le jeu en explorant près de 1,8M de nœuds, tandis que l'élagage alpha-beta réduit immédiatement ce nombre aux alentours de 20k. L'ajout d'une table de transposition permet environ de retrouver 4k score et réduit le calcul à 16k nœuds. Enfin, le passage au negamax a doublé le nombre de scores retrouvés dans la table, soit 8k score retrouvés et 12k nœuds explorés.

Ces résultats en configuration simplifiée nous ont permis de valider l'intérêt (et la correction) de la plupart de nos améliorations avant de les tester en situation réelle.

Parties complètes

Les parties complètes ont validé l'intérêt des tables de transposition et de recherche quiescente, puisqu'un agent disposant des ces améliorations remportait la majorité des parties contre des agents n'en disposant pas (étonnamment, la victoire n'était pas systématique pour autant).

L'élagage alpha-beta et la table de transposition permettent d'effectuer des recherches de profondeur plus grande, tandis que la recherche de quiescence augmente la profondeur d'exploration pour des nœuds judicieusement choisis, économisant ainsi du temps de calcul.

Ainsi les différents agents peuvent être décrits et classés par performance de jeu croissante dans le tableau suivant:

Agent	Profondeur de recherche	Profondeur de recherche de quiescence
Glouton	1	NA
Negamax	2	NA
Negamax, alpha-beta	2	NA
Negamax, alpha-beta, table	3	NA
Negamax, alpha-beta, table, quiescence	2	2
Negamax, alpha-beta, table, quiescence	3	1

A chacune de ces stratégies de recherche, il faut associer une heuristique, et la performance des différents agents résultant de ce choix pour une stratégie de recherche donnée suivra le même classement que celui décrit dans la section "Evaluation des heuristiques".

Résultats au concours

L'agent sélectionné pour le concours utilisait donc une recherche negamax de profondeur 3 avec table de transposition et recherche de quiescence de profondeur 1, avec l'heuristique combinant le nombre de billes éjectées, la distance au centre et l'adjacence.

L’agent a remporté toutes les rencontres en poules et a perdu sa première rencontre des phases finales. C’est somme toute un résultat très satisfaisant.

Discussion

Comme on l’a vu dans les sections précédentes, les améliorations apportées au minimax améliorent les performances de l’agent final : l’élagage alpha-beta accélère la recherche en évitant d’explorer les sous-arbres peu avantageux, la table de transposition nous dispense de recalculer certains scores, et le negamax augmente l’utilité de cette table. Enfin, la recherche quiescente augmente la profondeur de la recherche pour les configurations les plus intéressantes uniquement, augmentant l’intelligence de l’agent pour un coût calculatoire raisonnable.

Certaines omissions peuvent limiter la performance de l’agent. Par exemple, la table de transposition aurait dû permettre d’explorer les nœuds par score croissant, augmentant ainsi l’élagage, mais l’implémentation de ce mécanisme était incorrecte, nous forçant à l’abandonner. Ce même mécanisme devait servir de base à une recherche par approfondissement itératif, qui n’a pas pu être implémentée.

La profondeur des recherches à été sélectionnée pour pouvoir être maintenue tout au long de la partie, mais elle a été fixée à partir d’expérimentation sur nos machines. Une machine plus rapide conduira l’agent à ne pas utiliser tout le temps alloué, tandis qu’une machine plus lente pourra déclencher son mécanisme de sécurité et le forcer à finir la partie avec une profondeur de recherche de 2 et sans recherche de quiescence. Des mécanismes plus flexibles de gestion du temps seraient souhaitables.

Conclusion

En conclusion, ce rapport a présenté une méthodologie pour développer un agent intelligent pour le jeu d'Abalone, en combinant différentes heuristiques et en utilisant une stratégie de recherche Negamax, incorporant plusieurs améliorations évoquées en cours et dans les ressources de programmation de jeux d’échecs.

Les expérimentations ont permis de sélectionner l’agent le plus prometteur, qui a réalisé une performance respectable lors du concours, attestant ainsi la pertinence de l'approche adoptée. Les travaux décrits ici nous ont permis d’acquérir une base de connaissance solide pour de futurs travaux dans la conception d'agents intelligents pour le jeu d'Abalone et d’autres jeux de plateaux.