Algorithme de Monte-Carlo en stratégie à information parfaite.

L'algorithme de Monte-Carlo appliqué à des jeux de stratégie tel que le Go utilise une approche probabiliste pour déterminer le meilleur coup à jouer lors d'une partie et permet de faire fit des contraintes auxquelles les algorithmes traditionnels ne peuvent pas répondre (Nombre de coups énorme, pas de fonction d'évaluation).
L'intelligence artificielle me fascine, aujourd'hui elle est de plus en plus developpée et arrive à réaliser des taches hier impossible pour un ordinateur. L'algorithme de Monte-Carlo utilise une approche probabiliste pour faire un choix dans des jeux où il n'y a à priori pas de hasard.

Positionnement thématique (étape 1)

INFORMATIQUE (Informatique pratique), MATHEMATIQUES (Mathématiques Appliquées), INFORMATIQUE (Informatique Théorique).

Mots-clés (étape 1)

Mots-Clés (en français)	**Mots-Clés** (en anglais)
Méthodes stochastiques	*Stochastic methods*
Intelligence artificielle	*Artificial intelligence*
Probabilités	*Probability*
Structures de données	*Data Structure*
Complexité	*Complexity*

Bibliographie commentée

L'intelligence artificielle est un domaine d'étude qui a connu son premier réel succès en 1997 quand DEEP BLUE, l'ordinateur d'IBM a battu le champion du monde d'échec. Récemment l'algorithme AlphaGo a obtenu la 2ème place au classement de go mondial, il s'agit d'un exploit car ses prédécesseurs n'arrivaient pas à battre des joueurs de niveau moyen à cause de la complexité du jeu (10^600 parties possibles). [1]

Ma recherche s'est concentrée sur des jeux de stratégie combinatoire abstrait. Bien qu'ils soient complexes, ils restent cependant accessibles car il s'agit de jeux à information parfaite donc toutes les informations pour choisir le coup à jouer sont disponibles à l'algorithme.

Il existe alors plusieurs méthodes pour créer de telles intelligences artificielles, la plus célèbre se base sur l'utilisation d'une fonction d'évaluation (par exemple algorithme min-max) pour donner un poids à une position sur un plateau de jeu. Le principe de l'algorithme sera alors de jouer en maximisant repose sur la maximisation de la valeur de cette fonction d'évaluation. L'inconvénient de cette méthode est la fonction d'évaluation elle-même qui est difficile à écrire car cela neccesite une solide connaissance du jeu et des diffèrenes stratégies possibles. [3]

L'algorithme AlphaGo utilise une méthode de Monte-Carlo ce qui lui permet de se passer de fonction d'évaluation en utilisant une fonction UTC et de rechercher le coup à jouer grâce à une approche probabiliste. [2] Cette méthode est la recherche par arborescence de Monte-Carlo, elle

peut être utilisée dans tous les jeux de stratégie combinatoire abstrait, par exemple au hex ou au puissance4. [4]

Cette méthode repose sur des simulations aléatoires et sur un arbre qui grandit par itérations. Le temps d'exécution est déterministe et le résultat peut être incorrect avec une certaine probabilité, cependant cette probabilité est faible. [5]

J'ai choisi de reproduire l'algorithme sur le puissance 4.

Problématique retenue

Le but est d'écrire un algorithme de recherche par arborescence de Monte-Carlo appliqué au jeu de puissance 4 capable de jouer contre des joueurs ayant un niveau intermédiaire avec un grand pourcentage de victoire.

Objectifs du TIPE

Dans un premier temps je vais écrire un algorithme de recherche arborescente de Monte-Carlo de manière naïve en m'appuyant sur l'idée générale du programme. Ensuite il me faudra optimiser cet algorithme en utilisant différents outils mathématiques tel que le dénombrement et l'analyse probabiliste.

Abstract

The aim is to write a powerful artificial intelligence in Connect 4 based on the Monte-Carlo algorithm which can only be used on abstract strategy games. The principle is to simulate a great deal of random games and to use the collected data to ascertain the best move. The different state of the game are represented by a tree and the best move is determined using a formula that give each node of the tree a value depending of the win rate on the move (Exploitation value) and the number of simulations (Exploration value).

Références bibliographiques

[1] Rémi Coulom : Le jeu de go et la révolution de Monte-Carlo :
https://interstices.info/jcms/c_43860/le-jeu-de-go-et-la-revolution-de-monte-carlo

[2] Cameron Browne : Recherche arborescence Monte-Carlo :
http://www.cameronius.com/research/mcts/about/index.html

[3] L. Ralaivola : Fonction d'évaluation : *cours L2 informatique de L. Ralaivola (Marseille)*

[4] Tristan Cazenave - Abdallah Saffidine : Utilisation de la recherche arborescente Monte-Carlo au Hex : *http://www.lamsade.dauphine.fr/~cazenave/papers/hex-ria.pdf*

[5] Cameron Browne : Etude des méthodes de recherche arborescente Monte Carlo :
http://www.cameronius.com/cv/mcts-survey-master.pdf