Fast and Furious Game Playing : Monte Carlo Drift Specifications report

Prateek Bhatnagar, Baptiste Bignon,
Mikaïl Demirdelen, Gabriel Prevosto,
Dan Seeruttun--Marie, Benoît Viguier

Supervisors: Nikolaos Parlavantzas, Christian Raymond

11/27/2014

One of the advantages of the MCTS algorithm is its generalization. In fact it is applied to a lot of games. We decided to keep this generalization, therefore we thougth about two solutions.

One of them is Genaral Game Playing, a formal language which allows softwares to create the model given its formal rules. However the problem of this approach is that even if it permits to play to unknow games, it prevents the use of efficient heuristic to improve the algorithms. So we decided to choose a second way which consist to exploit the properties of MCTS algorithm. It does not need to know the rules, only the moves. With this in mind we created an interface for the game which define the methods that MCTS algorithm demands. In other words our algorithm is compatible with all the two-players games implemented with this interface.

Only the following fonctions are required :

- return all the possibles moves given position
- play a random move
- play a chosen move
- play random moves until the end of the game
- return whether the game is not finished or who won

The main application of our algorithm is the Arimaa game. Therefore we will be able to specialize our algorithm for it in order to improve its efficiency. The main problem is the branching factor[1] of the Arimaa game which average is 17 281 and reaches about 22 000

[1] In a tree, the branching factor is the number of children at each node.

after 10 moves[2].

Game	Average number of possible moves
Othello	8
Chess	35
Game of Go	250
Arimaa	17 281

The reason why the branching factor of a game is so important is because it increases greatly the space that has to be searched in order to guess what will happend multiples moves ahead. In chess after 6 moves, the number of positions evaluated are about 35^6 which is roughtly equivalent to 1,8 billions. In Arimaa, after 3 turns (yours, the opponent and yours again), if you were the explore all positions, you would need to evaluate around 5,2 trillions[3] boards (2000 times more than chess with half the number of moves).

In order to decrease the space to be search, our MCTS Algorithm will perform a big number of simulations before chosing the nodes to explore. After the selection, it will prune the tree in order to optimise search speed and the memory managment.

[2]http://arimaa.janzert.com/bf_study/
[3]1 trillion = 1 thousand billions = 10^{12}.