Mini AlphaGo

Term Project

EE488B, Autmun 2016

Hampus Adamsson, Moritz Luszek, Rafael Cartenet

20166398 20166469 20166490

School Of Computing

Supervised by

Sae-Young Chung



Korean Advanced Institute of

Science and Technology

December 28, 2016

Contents

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | Introduction | | | 2 |
| 2 | Neural Networks training | | | 3 |
|  | 2.1 | Value Networks . . . . . . . . . . . . . . . . . . . . . . . . . . . . | | 3 |
|  | 2.2 | SL Policy Networks . . . . . . . . . . . . . . . . . . . . . . . . . . | | 3 |
|  | 2.3 | RL Policy Networks . . . . . . . . . . . . . . . . . . . . . . . . . | | 3 |
| 3 | Monte Carlo Tree Search | | | 3 |
|  | 3.1 | Understanding . . . . . . . . . . . . . . . . . . . . . . . . . . . . | | 3 |
|  |  | 3.1.1 | Selection Step . . . . . . . . . . . . . . . . . . . . . . . . . | 3 |
|  |  | 3.1.2 | Backpropagation . . . . . . . . . . . . . . . . . . . . . . . | 4 |
|  | 3.2 | Implementation . . . . . . . . . . . . . . . . . . . . . . . . . . . . | | 4 |
| 4 | Conclusion | |  | 11 |

1

* Introduction

This paper is a summary of the attempt to replicate the work made by Google's AlphaGO. This is the term project for the course "Deep Learning with Al-phaGO"

Deep learning is an increasingly interesting topic in which companies world-wide strife to utilize the power that is known as machine learning. Deep learning is traditionally known as neural networks, with the slight alteration that more layers are utilized, and the training is performed layer-wise (traditionally). In regards to games in general, and Go in particular, this project mainly focus on a speci c deep learning set up suitable for the game Go. Convolutional layers are used to maintain topological ordering, while fully connected layers are added to ne tune the solution and perform a speci c action, given a particular game state.

2

* Neural Networks training

There are a number of ways to train neural networks. We have explored both supervised and reinforcement learning algorithms to obtain a good result. The main outlier that came to act as a steering wheel was the result-graph pre-sented in test players. There is a clear correlation between complexity and performance; although with the deviation that player 7 manage to beat players of higher order. This is mainly because we are comparing reinforcement learning and supervised learning by letting them play each other. See gure .

1. Value Networks
2. SL Policy Networks
3. RL Policy Networks

* Monte Carlo Tree Search

1. Understanding

In order to estimate the best move given a certain board, we can represent the next moves in a game tree. Finding the best play consist of exploring the tree in order to nd the best reward. As the tree is too big to be completely explored, we have to design an algorithm in order to explore the tree in an e cient way. The proposed solution by the AlphaGo paper is the APV-MCTS algorithm. APV stands for Asynchronous Policy and Value Networks MCTS stands for montecarlo tree search. In other words, algorithm use policy and value networks in a multi threading way through a montecarlo search inside the game tree. Technically, the CPUs are used to do the tree search and the GPUs are used in order to evaluate the neural networks.

1. Selection Step

We choose the next action according to

at = argmaxa[Q(st; a) + u(st; a)]

For u, they use a rough prediction P (s; a) that is determined with either :

Tree policy

SL policy network RL policy network

When selecting a path, they take in consideration the Q value of course, but also a bonus value that is a combination of P (s; a) and coe cients in order to do some exploration.

u value is for exploration. Otherwise it's greedy.

3

p

P

u(s; a) = cpuctP (s; a)

bNr(s; b)

1 + Nr(s; a)

1. Backpropagation

As teacher said, for simplicity reasons, we don't do the tree search in a multi threading way so we don't add virtual loss when we execute the backpropagation.

1. Implementation

Below is our implementation of the Monte Carlo Tree Search. We didn't nish it so we didn't include it inside our code. We developed a skeleton for the algorithm. We developed 3 classes, Edge class, State class, and MCTree.

State class contains the information about a state, board, player, previous explored actions and Value Network estimation of the board. It also contains the functions that choose the best action to take from this state, according to Q value and u value.

Edge class contains the action, which means the board after the action, and the di erent value of the MCTS updates.

MCTree class contains the starting State (which contains the current board) and the MCTS functions in order to explore the tree.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***import numpy a s np*** | | | |  |  |  |  |  |  |  |  |  |  |
| ***import*** | | ***t e n s o r f l o w*** | | ***a s*** | ***t f*** |  |  |  |  |  |  |  |  |
| ***# S t a t e c l a s s*** | | |  |  |  |  |  |  |  |  |  |  |  |
| ***#*** | ***c o n t a i n s t h e*** | | ***board*** | | ***o f*** | ***t h e c u r r e n t S t a t e*** | ***and*** | ***a l i s t*** | | ***o f e d g e s t h a t*** | | ***l e a d*** | ***t o t h e*** |
| ***#*** | ***next*** | ***s t a t e s .*** |  |  |  |  |  |  |  |  |  |  |  |
| ***c l a s s S t a t e ( o b j e c t ) :*** | | | | |  |  |  |  |  |  |  |  |  |
|  | ***d e f*** | ***i n i t*** | ***( s e l f*** | | ***,*** | ***board , p l a y e r ) :*** |  |  |  |  |  |  |  |
|  |  | ***# Board*** |  |  |  |  |  |  |  |  |  |  |  |
|  |  | ***s e l f . board*** | | ***=*** | ***board*** | |  |  |  |  |  |  |  |
|  |  | ***# A c t i o n s*** | |  |  |  |  |  |  |  |  |  |  |
|  |  | ***s e l f . e d g e s*** | | ***=*** | ***[ ]*** | ***# L i s t o f Edge*** |  |  |  |  |  |  |  |
|  |  | ***s e l f . nbedges*** | | | ***=*** | ***0*** |  |  |  |  |  |  |  |
|  |  | ***# P l a y e r t u r n*** | | |  |  |  |  |  |  |  |  |  |
|  |  | ***s e l f . p l a y e r = p l a y e r # e q u a l s 1 i f b l a c k . 2 i f w h i t e*** | | | | | | | | | |  |  |
|  |  | ***# Value*** | ***Network ESTIMATION*** | | | |  |  |  |  |  |  |  |
|  |  | ***s e l f . ValNetOutput = None # e q u a l s*** | | | | | ***t o*** | ***None*** | ***when not*** | | ***e s t i m a t e d y e t .*** | | |
|  | ***# Chooses t h e b e s t*** | | | | | ***a c t i o n a c c o r d i n g t o*** | ***Q*** | ***v a l u e*** | ***o f*** | ***each*** | ***edge . +*** | ***bonus*** | ***u*** |
|  | ***# PUCT a l g o r i t h m*** | | | | ***c f l e c t 2 2 .*** | |  |  |  |  |  |  |  |

***d e f*** ***c h o o s e a c t i o n ( s e l f ) :***

***PUCTs = np . z e r o s ( s e l f . nbedges )***

4

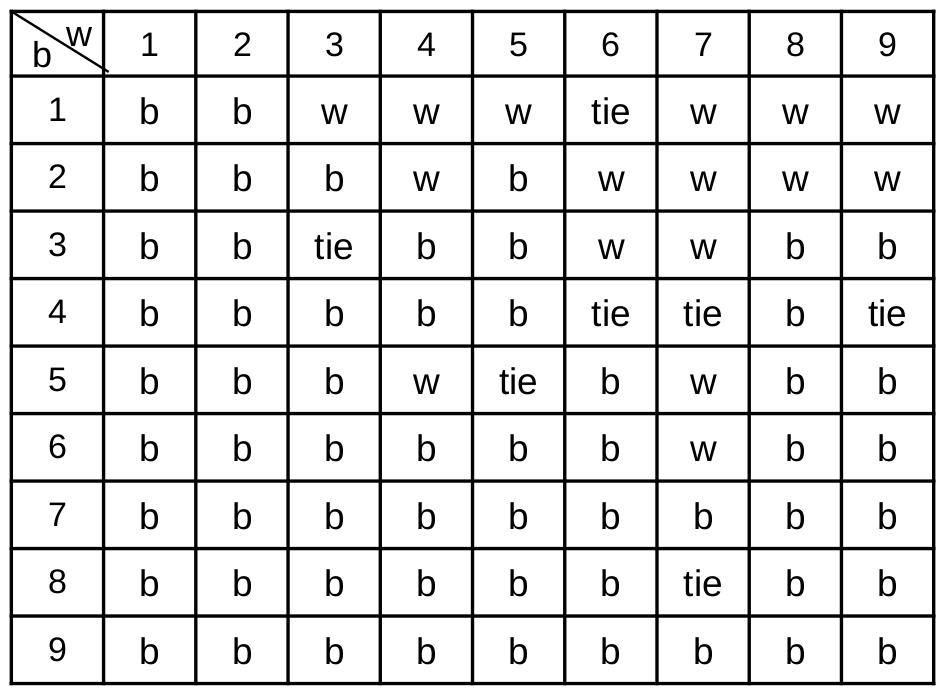


Figure 1: Graph showing the result when di erent networks play each other. X-/Y axis are ID to a particular player. The result is shown in the aligned state. The outcome is either black (B), white (W), or tie.

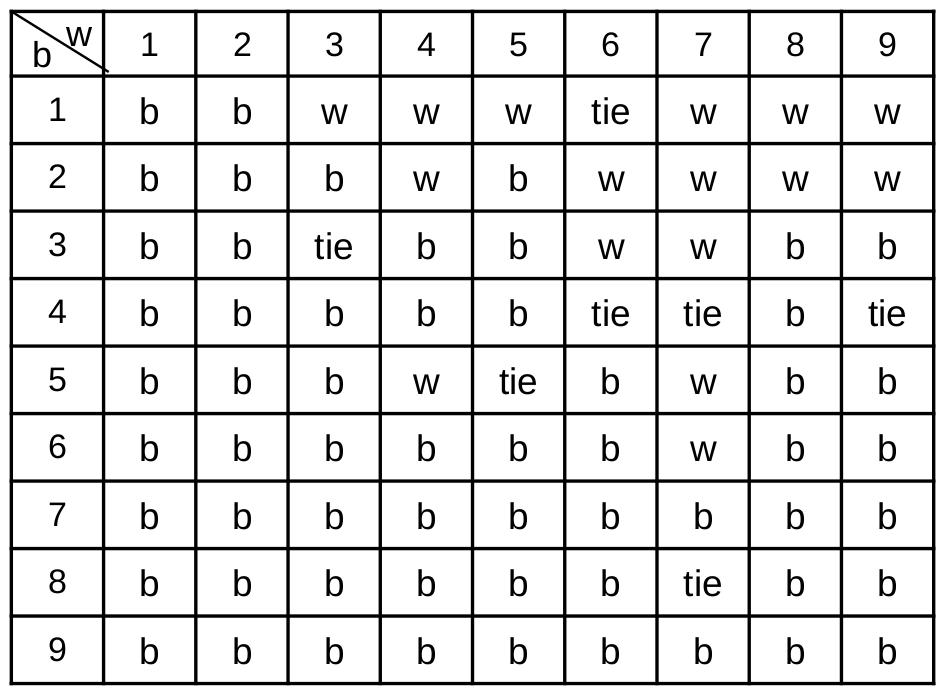


Figure 2: Graph showing the result when di erent networks play each other. X-/Y axis are ID to a particular player. The result is shown in the aligned state. The outcome is either black (B), white (W), or tie.

5

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***f o r*** | ***i*** | ***i n r a n g e ( s e l f . nbedges ) :*** | | | |  |  |  |  |  |  |  |  |  |
| ***PUCTs [ i ]*** | | | ***= s e l f . e d g e s [ i ] . PUCT( s e l f . board )*** | | | | | | | | | ***# PUCT( s , a ) = Q( s , a ) + u ( s , a ) . c*** | |  |
| ***r e t u r n*** | | ***e d g e s [ np . argmax (PUCTs ) ]*** | | | |  |  |  |  |  |  |  |  |  |
| ***# Choose*** | ***t h e b e s t a c t i o n a c c o r d i n g*** | | | | | ***t o*** | ***Q*** | ***v a l u e*** | |  | ***( Greedy )*** | |  |  |
| ***d e f c h o o s e g r e e d y a c t i o n Q ( s e l f ) :*** | | | | | |  |  |  |  |  |  |  |  |  |
| ***Qs = np . z e r o s ( s e l f . nbedges )*** | | | | | |  |  |  |  |  |  |  |  |  |
| ***f o r*** | ***i*** | ***i n r a n g e ( s e l f . nbedges ) :*** | | | |  |  |  |  |  |  |  |  |  |
|  | ***Qs [ i ] = s e l f . e d g e s [ i ] . Q # Q( s , a )*** | | | | | | |  |  |  |  |  |  |  |
| ***r e t u r n*** | | ***e d g e s [ np . argmax ( Qs ) ]*** | | | |  |  |  |  |  |  |  |  |  |
| ***# Choose*** | ***t h e b e s t a c t i o n a c c o r d i n g*** | | | | | ***t o*** | ***Nr*** | ***v a l u e*** | | | ***( Greedy )*** | | |  |
| ***d e f c h o o s e g r e e d y a c t i o n N r ( s e l f ) :*** | | | | | |  |  |  |  |  |  |  |  |  |
| ***Nrs =*** | | ***np . z e r o s ( s e l f . nbedges )*** | | | |  |  |  |  |  |  |  |  |  |
| ***f o r*** | ***i*** | ***i n r a n g e ( s e l f . nbedges ) :*** | | | |  |  |  |  |  |  |  |  |  |
|  | ***Nrs [ i ] =*** | | ***s e l f . e d g e s [ i ] . Nr # Nr ( s , a )*** | | | | | | |  |  |  |  |  |
| ***r e t u r n*** | | ***e d g e s [ np . argmax ( Nrs ) ]*** | | | |  |  |  |  |  |  |  |  |  |
| ***# bonus*** | ***u ( s , a ) depends on s*** | | | | ***s o n e e d s*** | | ***t o*** | ***be*** | ***updated from s .*** | | | | |  |
| ***d e f u p d a t e u e d g e s ( s e l f ) :*** | | | | |  |  |  |  |  |  |  |  |  |  |
| ***sumNr = 0*** | | |  |  |  |  |  |  |  |  |  |  |  |  |
| ***f o r*** | ***edge i n*** | | ***s e l f . e d g e s :*** | |  |  |  |  |  |  |  |  |  |  |
|  | ***sumNr += edge . Nr*** | | | |  |  |  |  |  |  |  |  |  |  |
| ***f o r*** | ***edge i n*** | | ***s e l f . e d g e s :*** | |  |  |  |  |  |  |  |  |  |  |
|  | ***edge . update u ( sumNr )*** | | | | |  |  |  |  |  |  |  |  |  |
| ***d e f i s l e a f ( s e l f ) :*** | | | |  |  |  |  |  |  |  |  |  |  |  |
| ***r e t u r n*** | | ***s e l f . nbedges == 0*** | | | |  |  |  |  |  |  |  |  |  |
| ***##*** | | | |  |  |  |  |  |  |  |  |  |  |  |
| ***# MCTS FUNCTIONS*** | | |  |  |  |  |  |  |  |  |  |  |  |  |
| ***##*** | | | |  |  |  |  |  |  |  |  |  |  |  |
| ***d e f expand ( s e l f ) :*** | | | |  |  |  |  |  |  |  |  |  |  |  |
| ***# expand*** | ***a*** | ***new node*** | | ***thanks*** | ***t o p o l i c y*** | | ***network*** | | | ***!*** |  |  |  |  |
| ***# e s t i m a t e s*** | | | ***next*** | ***a c t i o n*** | ***thanks*** | ***t o*** | ***p o l i c y*** | | | ***network*** | | |  |  |
| ***s e s s = t f . g e t d e f a u l t s e s s i o n ( )*** | | | | | | |  |  |  |  |  |  |  |  |
| ***s o f t o u t = np . z e r o s ( nx ny )*** | | | | | | ***board*** g | |  |  |  |  |  |  |  |
| ***f e e d*** | ***d i c t =*** f***" c u r r e n t board " :*** | | | | |  |  |  |  |  |  |  |
| ***i f s e l f . p l a y e r == 1 :*** | | | | |  |  |  |  |  |  |  |  |  |  |
|  | ***s o f t o u t [ : ] =*** | | | ***s e s s . run ( netP1 ,*** | | | ***f e e d*** | | ***d i c t*** | | ***=*** | ***f e e d*** | ***d i c t )*** |  |
|  | ***s e l f . ValNetOutput =*** | | | | ***s o f t o u t [ 1 ]*** | | |  |  |  |  |  |  |  |
| ***e l s e :*** | |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | ***s o f t o u t [ : ] =*** | | | ***s e s s . run ( netP2 ,*** | | | ***f e e d*** | | ***d i c t*** | | ***=*** | ***f e e d*** | ***d i c t )*** |  |
|  | ***s e l f . ValNetOutput =*** | | | | ***s o f t o u t [ 1 ]*** | | |  |  |  |  |  |  |  |
| ***# i n i t i a l i z e*** | | | ***new*** | ***board*** |  |  |  |  |  |  |  |  |  |  |

6

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***new board*** | | | ***= np . z e r o s ( s e l f . board . shape )*** | | | | | | | | | | |  |  |  |  |  |  |  |  |  |  |
| ***new board [ : ,*** | | | | ***: ]*** | ***= board [ : ,*** | | | | ***: ]*** | |  |  |  |  |  |  |  |  |  |  |  |  |  |
| ***n e w s t a t e =*** | | | | ***np . z e r o s ( s t a t e . shape )*** | | | | | | | |  |  |  |  |  |  |  |  |  |  |  |  |
| ***n e w s t a t e [ : ] = s t a t e [ : ]*** | | | | | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| ***#*** | ***c h o o s e*** | | ***b e s t move a c c o r d i n g*** | | | | | | | ***t o*** | ***r e s u l t*** | | | ***o f*** | ***p o l i c y*** | | ***network*** | | |  |  |  |  |
| ***t e m p s o f t o u t*** | | | | ***=*** | ***s o f t o u t [ : ] / np . sum ( s o f t o u t [ : ] )*** | | | | | | | | | | | |  |  |  |  |  |  |  |
| ***n v a l i d m o v e s = 0*** | | | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| ***w h i l e True :*** | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | ***c u m s o f t o u t = np . cumsum ( t e m p s o f t o u t )*** | | | | | | | | | | | | | |  |  |  |  |  |  |  |  |  |
|  | ***s o f t r*** | | ***=*** | ***rand ( 1 )*** | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | ***s o f t x y = np . sum ( s o f t r*** >***c u m s o f t o u t )*** | | | | | | | | | | | | |  |  |  |  |  |  |  |  |  |  |
|  | ***rxy*** | ***=*** | ***np . a r r a y ( s e l f . xy ( s o f t x y ) )*** | | | | | | | | |  |  |  |  |  |  |  |  |  |  |  |  |
|  | ***r j*** | ***=*** | ***i n t ( rxy [ 0 ] [ 0 ] )*** | | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | ***rk*** | ***=*** | ***i n t ( rxy [ 0 ] [ 1 ] )*** | | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | ***i s v a l i d ,*** | | | ***,*** |  | ***=*** | ***s e l f . v a l i d ( b [ : ,*** | | | | |  | ***: ,*** | ***s t a t e [ : ] ,*** | | | ***rxy ,*** | | ***s e l f . p l a y e r )*** | | | |  |
|  | ***i f i n t ( i s v a l i d [ 0 ] ) :*** | | | | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | ***i s v a l i d*** | | | ***,*** | ***bn ,*** | ***sn*** | ***=*** | ***s e l f . v a l i d ( b [ : ,*** | | | | | | ***: ] , s t a t e [ : ] ,*** | | | | | ***rxy ,*** | ***s e l f . p l a y e r )*** | | |
|  |  | ***new board [ : ,*** | | | | | ***: ]*** | ***= bn*** | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | ***n e w s t a t e [ : ] = sn*** | | | | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | ***x =*** | | ***r j*** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | ***y = rk*** | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | ***n v a l i d m o v e s += 1*** | | | | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | ***break*** | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | ***e l s e :*** | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | ***t e m p s o f t o u t [ s o f t x y ] = 0*** | | | | | | | | | |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | ***norm = np . sum ( t e m p s o f t o u t )*** | | | | | | | | | |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | ***i f norm*** | | | ***!= 0 :*** | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  | ***t e m p s o f t o u t*** | | | | ***=*** | ***t e m p s o f t o u t /norm*** | | | | | | | |  |  |  |  |  |  |  |
|  |  | ***e l s e :*** | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  | ***break*** | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| ***i f*** | ***not*** | ***i n t ( i s v a l i d [ 0 ] ) :*** | | | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | ***i s v a l i d , bn , sn =***n | | | | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | ***s e l f . v a l i d ( b [ : , : ] , s t a t e [ : ] , np . o n e s ( ( 1 , 2 ) ) , s e l f . p l a y e r )*** | | | | | | | | | | | | | | | | | | | | |  |
|  | ***n e w s t a t e [ : ] = sn*** | | | | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| ***# SHOULD TRANSFORM BOARD AND STATE INTO A NEW STATE.*** | | | | | | | | | | | | | | | | | | |  |  |  |  |  |
| ***r e t u r n STATE*** | | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| ***d e f r o l l o u t ( s e l f ) : #*** | | | | | | ***R o l l o u t*** | | ***p o l i c y*** | | | ***i n o r d e r*** | | | | ***t o*** | ***d e t e r m i n e a*** | | | | ***reward 1 0 o r 1*** | | |  |
| ***# a c c o r d i n g*** | | ***t o t h e*** | | | ***r e s u l t o f*** | | | ***t h e game .*** | | | | ***p o l i c y*** | | | ***=*** | ***p l a y s random*** | | | | ***move*** | ***u n t i l t h e*** | | ***end .*** |
| ***# u s e*** | ***s t r u c t u r e*** | | | ***from*** | | ***g o*** | ***t r a i n v a l u e*** | | | | ***t o*** | ***g e t*** | | ***t h e*** | | ***f u l l y random*** | | | | ***game*** | ***e v a l u a t i o n*** | |  |
| ***#game = game1 ( )*** | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| ***#*** | ***r a l l*** | ***= np . o n e s ( ( n t r a i n ) )*** | | | | | | | | ***# random*** | | | | ***moves*** | | ***f o r*** | ***a l l*** | ***games*** | | |  |  |  |
| ***#*** | ***[ d1 ,*** | ***w1 , wp1 ,*** | | | ***d2 ,*** | | ***w2 ,*** | ***wp2 ]*** | | ***=*** | ***game . p l a y g a m e s ( [ ] ,*** | | | | | | | ***[ ] ,*** | ***r*** | ***a l l ,*** | ***[ ] ,*** | ***[ ] , r*** | ***a l l , n*** |
| ***#*** | ***w b l a c k :*** | | | ***nb1 1 ,*** | | | ***0 : t i e ,*** | | | ***1 :*** | ***b l a c k*** | | ***wins ,*** | | | ***2 : w h i t e*** | | ***wins*** | |  |  |  |  |
| ***#*** | ***wp*** | ***black : win*** | | | | ***p r o b a b i l i t i e s*** | | | | | ***f o r b l a c k*** | | | |  |  |  |  |  |  |  |  |  |
| ***#*** | ***d w h i t e :*** | | | ***4 d*** | | ***matrix*** | | ***o f s i z e*** | | | ***nx ny 3 nb2*** | | | | | ***c o n t a i n i n g*** | | | ***a l l moves*** | | | ***by w h i t e*** | |
| ***#*** | ***w white :*** | | | ***nb2 1 ,*** | | | ***0 : t i e ,*** | | | ***1 :*** | ***b l a c k*** | | ***wins ,*** | | | ***2 : w h i t e*** | | ***wins*** | |  |  |  |  |

7

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | ***#*** | | ***wp*** | | ***white :*** | | | ***win*** | ***p r o b a b i l i t i e s f o r*** | | | | | ***w h i t e*** | | |  |  |  |  |  |  |  |  |
|  |  | ***r e t u r n 0 # 1 0 o r 1*** | | | | | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| ***d e f*** | | ***v a l u e n e t w o r k ( s e l f ) : #*** | | | | | | | | ***u s e s t h e*** | | ***VAL*** | | ***f u n c t i o n t h a t*** | | | | | ***u s e s*** | | ***t h e Value network*** | | |  |
| ***#*** | ***t o*** | ***p r e d i c t win*** | | | | | | ***v a l u e .*** | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| ***# SHOULD BE CALLED ONLY ONCE PER NODE ( i f*** | | | | | | | | | | | | | | | ***not*** | | ***s a v i n g*** | | ***t h e r e s u l t*** | | | ***might be good )*** | |  |
| ***# u s e t h e v a l u e f u n c t i o n o f s t r a t e g y . py ?*** | | | | | | | | | | | | | | |  |  |  |  |  |  |  |  |  |  |
| ***#*** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| ***#*** | ***Indeed*** | | | ***:*** | ***i f*** | ***n e v e r computed*** | | | | | ***b e f o r e ,*** | | ***e s t i m a t e*** | | | | ***i t thanks*** | | | ***t o*** | ***v a l*** | ***net , o t h e r w i s e*** | |  |
| ***# j u s t r e t u r n s t h e v a l u e .*** | | | | | | | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | ***i f*** | ***s e l f . ValNetOutput == None :*** | | | | | | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  | ***s e s s = t f . g e t d e f a u l t s e s s i o n ( )*** | | | | | | | | | | |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  | ***s o f t o u t = np . z e r o s ( 3 )*** | | | | | | | | |  | ***board*** g | | | |  |  |  |  |  |  |  |  |
|  |  |  | ***f e e d d i c t*** | | | | | ***=*** f***" c u r r e n t board " :*** | | | | |  |  |  |  |  |  |  |  |
|  |  |  | ***i f s e l f . p l a y e r == 1 :*** | | | | | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  | ***s o f t o u t [ : ]*** | | | | ***=*** | ***s e s s . run ( netV1 ,*** | | | | | ***f e e d*** | | ***d i c t*** | ***=*** | ***f e e d*** | | ***d i c t )*** |  |  |  |
|  |  |  | ***e l s e :*** | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  | ***s o f t o u t [ : ]*** | | | | ***=*** | ***s e s s . run ( netV2 ,*** | | | | | ***f e e d*** | | ***d i c t*** | ***=*** | ***f e e d*** | | ***d i c t )*** |  |  |  |
|  |  |  | ***s e l f . ValNetOutput*** | | | | | | | ***=*** | ***s o f t o u t [ 1 ]*** | | |  |  |  |  |  |  |  |  |  |  |  |
|  |  | ***r e t u r n s e l f . ValNetOutput*** | | | | | | | | | ***# r e t u r n*** | | ***r e s u l t r e l a t e d*** | | | | | | ***t o*** | ***v a l u e network*** | | | |  |
| ***d e f e v a l u a t e ( s e l f ) :*** | | | | | | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| ***#*** | ***I*** | ***g u e s s*** | | ***t h i s*** | |  | ***f u n c t i o n i s about s a m p l i n g*** | | | | | | | | ***some*** | | ***p o s s i b l e*** | | | ***a c t i o n s , a*** | | | ***f i n i t e number o*** |  |
| ***#*** | ***and*** | | ***e v a l u a t e*** | | |  | ***them u s i n g*** | | | ***r o l l o u t*** | | ***and*** | ***v a l u e*** | | | ***n e t . ( That*** | | | | ***s h o u l d u s e*** | | | ***t h e f u n c t i o n edge*** |  |
|  |  | ***V = s e l f . v a l u e n e t w o r k ( )*** | | | | | | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | ***R = s e l f . r o l l o u t ( )*** | | | | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | ***r e t u r n V, R*** | | | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| ***# Edge*** | ***c l a s s*** | | | ***:*** | ***Edge*** | | ***from s t a t e S*** | | | | ***with*** | ***a c t i o n*** | | | ***A*** |  |  |  |  |  |  |  |  |  |
| ***c l a s s*** | ***Edge :*** | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| ***# GLOBAL VARIABLES*** | | | | | | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| ***s e l f . l b d a = 0 . 5*** | | | | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| ***s e l f . c p u c t = 5*** | | | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| ***s e l f . n t h r = 40 # v i s i t t h r e s h o l d*** | | | | | | | | | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |
| ***d e f*** | |  | ***i n i t*** | | ***( s e l f*** | | | ***, board ) :*** | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | ***s e l f . s t a t e*** | | | | ***=*** | | ***S t a t e ( board ) # Board*** | | | | |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | ***s e l f . p l a y e r*** | | | | | ***=*** | ***None #*** | | ***P l a y e r p e r f o r m i n g*** | | | | | | ***a c t i o n*** | |  |  |  |  |  |  |  |
|  |  | ***s e l f . p r i o r P*** | | | | | ***=*** | ***None #*** | | ***P r i o r p r o b a b i l i t y*** | | | | | | ***f o r*** | ***t h a t*** |  | ***edge*** | |  |  |  |  |
|  | ***# VALUE NETWORK*** | | | | | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | ***s e l f . Nv =*** | | | | ***0*** |  | ***# Number*** | | | ***o f l e a f*** | | ***e v a l u a t i o n s u s i n g*** | | | | | | | ***t h e*** | ***v a l u e network*** | | |  |
|  |  | ***s e l f .Wv =*** | | | | ***0*** |  | ***#*** | ***Accumulation*** | | | ***o f*** | ***v a l u e s e s t i m a t e s*** | | | | | | | ***u s i n g t h e v a l u e network*** | | | |  |
|  | ***# ROLLOUT POLICY*** | | | | | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | ***s e l f . Nr*** | | | ***=*** | ***0*** |  | ***# Number*** | | | ***o f l e a f*** | | ***e v a l u a t i o n s u s i n g*** | | | | | | | ***t h e*** | ***r o l l o u t*** | | ***p o l i c y*** |  |
|  |  | ***s e l f .Wr =*** | | | | ***0*** |  | ***#*** | ***Accumulation*** | | | ***o f*** | ***r e w a r d s*** | | | | ***u s i n g*** |  | ***t h e*** | ***r o l l o u t*** | | ***p o l i c y*** | |  |

8

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | ***# Q VALUE*** | |  |  |  |  |  |  |  |  |  |  |  |
|  | ***s e l f .Q =*** | ***0*** | ***# Combined e s t i m a t e*** | | | | | | | ***o f*** | | ***t h e a c t i o n v a l u e f u n c t i o n u s i n g t h e*** |  |
|  |  |  | ***#*** | ***v a l u e*** | | ***network*** | | | ***and*** | ***t h e*** | | ***r o l l o u t p o l i c y*** |  |
|  | ***# UCT*** |  |  |  |  |  |  |  |  |  |  |  |  |
|  | ***s e l f . u =*** | ***0*** | ***#*** | ***bonus*** | | ***u (P)*** | | ***f o r*** | ***edge*** | | ***s e l e c t i o n*** | |  |
|  | ***# Rough*** | ***P r e d i c t i o n*** | |  |  |  |  |  |  |  |  |  |  |
|  | ***s e l f . P =*** | ***0*** | ***# Can*** | | ***be*** | | ***p r o b a b i l i t y*** | | | | ***from :*** | |  |
|  |  |  | ***# Tree*** | | | | ***p o l i c y*** | |  |  |  | ***P tau ( a*** j ***s )*** |  |
|  |  |  | ***# SL*** | |  | ***p o l i c y*** | | ***network*** | | |  | ***P d e l t a ( a*** j ***s )*** < ***AlphaGo*** |  |
|  |  |  | ***# RL*** | |  | ***p o l i c y*** | | ***network*** | | |  | ***P rau ( a*** j ***s )*** |  |
| ***d e f*** | ***update Q ( s e l f ) :*** | |  |  |  |  |  |  |  |  |  |  |  |
|  | ***qV = s e l f .Wv /*** | | ***s e l f . Nv*** | |  |  |  |  |  |  |  |  |  |
|  | ***qR = s e l f .Wr / s e l f . Nr*** | | | |  |  |  |  |  |  |  |  |  |
|  | ***s e l f .Q = ( 1s e l f . l b d a ) qV + s e l f . l b d a qR*** | | | | | | | | | | | |  |
| ***d e f*** | ***update u ( s e l f ,*** | | ***sumNr ) :*** | |  |  |  |  |  |  |  |  |  |
|  | ***num = np . s q r t ( sumNr )*** | | | |  |  |  |  |  |  |  |  |  |
|  | ***denom = 1 + s e l f . Nr*** | | | |  |  |  | ***num*** | |  |  |  |  |
|  | ***s e l f . u =*** | ***s e l f . c p u c t*** | | | ***s e l f . P*** | | | ***/*** | ***denom*** | |  |
| ***d e f*** | ***PUCT( s e l f , board ) :*** | | |  |  |  |  |  |  |  |  |  |  |
|  | ***r e t u r n s e l f .Q + s e l f . u*** | | | |  |  |  |  |  |  |  |  |  |
| ***# e x e c u t e s a rough*** | | | ***p r e d i c t i o n*** | | | | ***o f*** | ***t h i s a c t i o n .*** | | | | |  |
| ***# f o r w a r d s t h e board*** | | | |  |  |  |  |  |  |  |  |  |  |
| ***d e f r o u g h p r e d i c t i o n ( s e l f ) :*** | | | | |  |  |  |  |  |  |  |  |  |

* ***#***
* ***MCTS FUNCTIONS***
* ***#***

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***#*** | ***e x e c u t e s t h e s t e p s*** | | | ***needed*** | ***i n*** | ***t h e*** | ***backprop a l g o r i t h m*** | | |  |  |  |
| ***#*** | ***do we*** | ***r e i n i t i a l i s e*** | | ***t h e v a l u e s*** | | ***o f*** | ***Nv and Wv*** > ***no ,*** | | ***t h e*** | ***l e a f*** | ***Node*** | ***w i l l a u t o m a t i c a l l*** |
| ***d e f b a c k p r o p u p d a t e ( s e l f , n o d e P r e v i o u s ) :*** | | | | | | | |  |  |  |  |  |
|  | ***#no*** | ***p a r a l l e l p r o c e s s i n g*** | | | > ***commented*** | | | ***out*** |  |  |  |  |
|  | ***#z t*** | ***= 1 # random*** | | ***v a l u e*** | ***chosen ,*** | | ***s i n c e*** | ***no p a r a l l e l*** | ***p r o c e s s i n g i n*** | | | ***o r d e r*** |
|  | ***#s e l f . Nr = s e l f . Nr + 1 # v i r t u a l l o s s a p p l i e d t o d i s c o u r a g e e v a l u a t i o n by p a r a*** | | | | | | | | | | | |
|  | ***#s e l f .Wr = s e l f .Wr + z t # v i r t u a l l o s s a p p l i e d t o d i s c o u r a g e e v a l u a t i o n by p a r*** | | | | | | | | | | | |
|  | ***s e l f . Nv = s e l f . Nv + 1*** | | | |  |  |  |  |  |  |  |  |
|  | ***i f*** | ***n o d e P r e v i o u s . i s L e a f ( ) : # we*** | | | | | ***o n l y*** | ***have t o e v a l u a t e*** | | ***t h e*** | ***winning p r o b a b i l i t i e s*** | |
|  |  | ***s e l f .Wv =*** | ***s e l f .Wv +*** | | ***n o d e P r e v i o u s . v a l u e n e t w o r k ( )*** | | | | | ***# add*** | ***t h e*** | ***v a l u e network e s t*** |
|  | ***e l s e :*** | |  |  |  |  |  |  |  |  |  |  |
|  |  | ***s e l f .Wv =*** | ***s e l f .Wv +*** | | ***n o d e P r e v i o u s .Wv*** | | | |  |  |  |  |

9

***s e l f . update Q ( )***

***d e f getBackpropVal ( s e l f ) : r e t u r n [ Q, Nv ,Wv]***

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***# e v a l u a t e t h i s a c t i o n*** | | | | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| ***#*** | ***e v a l u a t i o n t a k e s*** | | | | | ***i n c o n s i d e r a t i o n*** | | | | | ***two t h i n g s*** | | | ***:*** |  |  |  |  |  |  |  |  |
| ***# Value*** | | | ***Network*** | | ***E st i m at e : v*** | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| ***# R o l l o u t p o l i c y*** | | | | | | ***( random*** | | ***p l a y*** | ***u n t i l*** | | | ***end o f*** | ***t h e game )*** | | | ***with reward*** | | ***: r*** |  |  |  |  |
| ***d e f e v a l u a t e ( s e l f ) :*** | | | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | ***s e l f . Nv += 1*** | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | ***s e l f . Nr +=*** | | | ***1*** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | ***#TO COMPLETE#*** | | | | | | | | | |  |  |  |  |  |  |  |  |  |  |  |  |
|  | ***s e l f .Wv +=*** | | | ***s e l f . v a l u e n e t w o r k ( )*** | | | | | | ***# ESTIMATION FROM THE VALUE NETWORK ( c u r r e n t*** | | | | | | | | | | | | ***No*** |
|  | ***s e l f .Wr +=*** | | | ***s e l f . r o l l o u t ( ) # ESTIMATION FROM THE ROLLOUT POLICY*** | | | | | | | | | | | | | | ***( c u r r e n t*** | | ***Node*** | | ***i s*** |
|  | ***##*** | | | | | | | | | |  |  |  |  |  |  |  |  |  |  |  |  |
|  | ***#TODO*** | | ***update*** | | ***t h e r e s u l t*** | | | ***o f*** | ***t h e*** |  | ***e v a l u a t i o n*** | | ***f o r t h e u p d a t i n g o f*** | | | | | ***t h e*** | ***t r e e*** | ***i n*** | ***t h e*** | |
|  | ***#*** > ***d i d t h e*** | | | | ***change t o Wv and Wr do t h i s ?*** | | | | | | | |  |  |  |  |  |  |  |  |  |  |
| ***c l a s s MCTree :*** | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| ***d e f*** | | ***i n i t*** | ***( s e l f*** | | ***,*** | ***board ) :*** | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | ***s e l f . s t a r t i n g*** | | | | ***N o d e = S t a t e ( board)#*** | | | | | | | ***s t a r t i n g*** | | ***N o d e*** | ***i s*** | ***t h e t r e e head*** | | |  |  |  |  |
|  | ***# path h o l d s*** | | | | ***a l l t h e*** | | ***i n f o r m a t i o n needed f o r*** | | | | | | | ***t h e*** | ***b a c k p r o p a g a t i o n*** | | | |  |  |  |  |
|  | ***#*** | ***add*** | ***a v a i l a b i l i t y o f*** | | | | ***s e t t i n g*** | | | ***t h e*** | | ***v a l u e s*** | ***c a l c u l a t e d*** | | | ***i n*** | ***backprop ( n e e d s*** | | | ***t o*** | ***be*** | ***a*** |
| ***#*** | ***path*** | ***c o n t a i n s*** | | ***o n l y t h e*** | | | ***s e l e c t e d*** | | | ***path*** | | ***i n one*** | ***s e l e c t i o n*** | | | ***s t e p*** > ***board*** | | | ***s t a t e*** | ***a t t h e*** | | |
| ***#*** | ***s e l e c t one path*** | | | | ***from rootNode*** | | | | ***u n t i l*** | | | ***a l e a f*** | ***node o f*** | | ***t h e newly c o n s t r u c t e d*** | | | | | ***monteCa*** | | |
| ***d e f s e l e c t i o n ( s e l f ,*** | | | | | | ***path*** | ***=*** | ***[ ] ,*** | ***node*** | | ***=*** | ***None ) :*** |  |  |  |  |  |  |  |  |  |  |
|  | ***i f*** | ***node == None :*** | | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | ***node =*** | | ***s e l f . s t a r t i n g N o d e*** | | | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | ***i f*** | ***not*** | ***node . i s*** | | | ***l e a f ( ) :*** | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | ***n e x t t = node . c h o o s e*** | | | | | | ***a c t i o n ( ) # CHOOSE NEXT EDGE ACCORDING TO POLICY*** | | | | | | | | | | | |  |  |  |
|  |  | ***path . append ( n e x t t . s t a t e ) # Adds t h e s t a t e t o*** | | | | | | | | | | | | | ***t h e path .*** | | |  |  |  |  |  |
|  |  | ***r e t u r n s e l f . s e l e c t i o n ( path ,*** | | | | | | | | | ***n e x t t . s t a t e )*** | | | |  |  |  |  |  |  |  |  |
|  | ***r e t u r n*** | | ***path ,*** | | ***node*** | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| ***# Adds*** | | ***an*** | ***a d i t i o n a l*** | | | ***node*** | ***C*** | ***from*** | ***t h e*** | | ***s e l e c t e d*** | | ***node*** | |  |  |  |  |  |  |  |  |
| ***d e f e x p a n s i o n ( s e l f ,*** | | | | | | ***node ) :*** | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | ***# add Node t o*** | | | | ***MCTree*** | | ***i f*** | ***a c e r t a i n*** | | | | ***number*** | ***o f v i s i t s*** | | | ***t o*** | ***an !EDGE! i s*** | | ***r e a c h e d*** | |  |  |
|  | ***r e s u l t*** | | ***= node . expand ( )*** | | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | ***r e t u r n*** | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| ***#*** | ***S i m u l a t e s from*** | | | | ***t h e Node C*** | | | ***which*** | | ***was*** | | ***!EXPANDED! u n t i l*** | | | | ***t h e*** | ***end o f*** | ***t h e*** | ***game*** | ***and*** | ***com*** | |

10

|  |  |  |
| --- | --- | --- |
| ***d e f s i m u l a t i o n ( s e l f*** | | ***, node ) :*** |
| ***r e t u r n node . e v a l u a t e ( )*** | | |
| ***# change t h e*** | ***p l a y e d*** | ***/ won v a l u e f o r e v e r y node i n t h e path from C t o t h e r o o t , a c c*** |
| ***d e f backprop ( s e l f ,*** | | ***path ) :*** |
| ***p r e v i o u s*** | ***= l e n ( path )1*** | |
| ***f o r i i n*** | ***r e v e r s e d ( r a n g e ( l e n ( path ) ) ) :*** | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***# v a l u e s*** | | ***i n t h e*** | ***l e a f node*** | | ***a l r e a d y i n i t i a l i s e d*** | | | |  |
| ***path [ i ] . b a c k p r o p*** | | | ***u p d a t e ( path [ p r e v i o u s ] )*** | | | | |  |  |
| ***p r e v i o u s*** | | ***= i*** |  |  |  |  |  |  |  |
| ***d e f MCTS( s e l f ) :*** | |  |  |  |  |  |  |  |  |
| ***# S e l e c t s*** | ***a*** | ***l e a f and*** | ***r e t u r n s*** | | ***t h e*** | ***path*** | ***from t h e*** | ***r o o t*** | ***t o node .*** |
| ***path , node*** | ***=*** | ***s e l f . s e l e c t i o n ( )*** | | |  |  |  |  |  |
| ***# C r e a t e s an*** | | ***a d i t i o n a l node*** | | ***from*** | | ***t h i s*** | ***l e a f*** |  |  |
| ***node = s e l f . e x p a n s i o n ( node )*** | | | |  |  |  |  |  |  |
| ***# Add t h e c r e a t e d node i n t h e path*** | | | | | | |  |  |  |
| ***path . append ( 0 )*** | | |  |  |  |  |  |  |  |
| ***# S i m u l a t e*** | ***t h e game*** | | ***u n t i l t h e*** | | ***end*** |  |  |  |  |
| ***s e l f . s i m u l a t i o n ( node )*** | | | |  |  |  |  |  |  |
| ***# Backpropagate t h e*** | | | ***r e s u l t*** | ***o f*** | ***t h e*** | ***s i m u l a t i o n i n*** | | ***t h e*** | ***t h r e e through t h e path*** |
| ***s e l f . backprop ( path )*** | | |  |  |  |  |  |  |  |

We didn't manage to nish our MCTS algorithm

4 Conclusion

References

11