TCG HW2 Description

November 30, 2013

HW2 Description

- Implement the 7×7 kill all go.
- Require: UCB, UCT and progressive pruning.
- Bonus: other techniques.
- Grading policy:
  - Implement the basic requirement.
  - Pass the test data.

Basic Command

- reset: reset game.
- time i: set the thinking time.
- put b/w x y: put the b/w piece at (x, y)
    - if the game is start, the put is recorded.
- display: show the current game board.
- start game: start the game
    - all the move after this command is recorded.
- think b/w: make a move of b/w
    - only work after game is start
- quit: end the program.

About the Template Code

- The variable in the template code is naming as follows:
    - Define constant: all upper letters.
        - BOARDSIZE, BOUNDARYSIZE.
    - Array: Initial character is upper letter.
        - Board, MoveList
    - Non-array variable: all letter is lower case
        - There are two exceptions, X and Y.
        - game_length, num_legal_move

Board structure: Board[BOUNDARYSIZE][BOUNDARYSIZE]

	0	1	2	3	4	5	6	7	8
0	*	*	*	*	*	*	*	*	*
1	*	.	.	.	.	.	.	.	*
2	*	.	.	.	.	.	.	.	*
3	*	.	.	.	.	.	.	.	*
4	*	.	.	.	.	.	.	.	*
5	*	.	.	.	.	.	.	.	*
6	*	.	.	.	.	.	.	.	*
7	*	.	.	.	.	.	.	.	*
8	*	*	*	*	*	*	*	*	*

- BOUNDARYSIZE: 9
- BOARDSIZE: 7
- Board[i][j] is (x,y) = (j, 8-i) in the game board

Think Function

- $gen\_legal\_move(Board, turn, game\_length, GameRecord, MoveList)$
    - generate all the legal move
    - return the number of legal moves.
- $random\_pick\_move(num\_legal\_moves, MoveList)$
    - randomly pick one legal move
    - return the selected move.
- $do\_move(Board, turn, move)$
    - update the current board with “move“

gen_legal_move Function

- For each empty intersection
    - Check if the empty intersection is a legal move
    - Check if the legal move will result in a repeat board
    - Add the move to move list.
        - each move is a 3 digit integers *eij*
        - *e* denote this is a capture move (1) or not (0).
        - *ij* denote the location of Board[i][j]
        - e.g. 123: put stone in Board[2][3] is a capture move.
        - e.g. 056: put stone in Board[5][6] is not a capture move.

Function for Checking Legal Move

- $count\_neighboorhood\_state(Board, X, Y, turn, *empt, *self, *oppo, *boun, NeighboorhoodState)$
  - return the number of
    - Empty intersection
    - Self intersection
    - Opponent intersection
    - Boundary intersection
  - Record the state of each neighborhood in NeighboorhoodState.
- $count\_liberty(X, Y, Board, Liberties)$
  - count the number of liberties in each direction's string.
  - The result is saved in Liberties.
  - Using DFS method.

Legal Move

- A move is legal if
  - At least one neighborhood intersection is empty.
  - One of the self string has more than one liberty.
  - One of the opponent string has only one liberty.

Do the move

- update the Board with put b/w piece at (x, y)
- update_board(Board, X, Y, turn)
    - put turn's piece in (X, Y)
    - will not check if (X, Y) is a legal move.
- update_board_check(Board, X, Y, turn)
    - put turn's piece in (X, Y)
    - will check if (X, Y) is a legal move.
    - return 1 if (X, Y) is a legal move
    - return 0 if (X, Y) is a inlegal move

Avoid the repeat board

- GameRecord[MAXGAMELENGTH][BOUNDARYSIZE][BOUNDARYSIZE]
- game_length
- Check the all the board in the GameRecord.

Random Select a Move and Do the Move

- Randomly choose one of the legal move.
- Update the current board.