Minah Kim

CS-UY 4613

Project-Checkers: Report

Due: 25 April 2018

**Instruction**

1. The game is programmed in python. In order to properly run the game, use python version 2.x.

2. This game is a command-line program and does not use any special python modules other than ones that already come with python 2 package.

2. Run file “Start.py” to start the game.

**High Level Design/Program Description**

This checkers game is a player vs. computer game whose AI that backs up the computer’s plays is implemented using the alpha-beta search algorithm.

This game package contains a Board class, an ABTree class, and a Game class.

The Board class contains information about the state of the checkerboard per turn. This information is stored in a list whose individual element is whether a location (r, c) is a white or a black tile and whether a black tile contains a checker piece (AI or player) or not.

Terminal States:

1. AI has captured all 6 player pieces

2. Player has captured all 6 AI pieces

3. There are no more legal moves

a) There’s a draw (player score = AI score)

b) AI has won (AI score > player score)

c) Player has won (player score > AI score)

Utility Values (with respect to terminal states):

1. 60 = 10 \* number of player pieces captured

2. -30 = -5 \* number of AI pieces captured

3. Three utility values for no legal moves:

a) 15 = 10 \* number of player pieces captured – 5 \* number of AI pieces captured

b) 28 = 15 (draw) + 13

c) 2 = 15 (draw) – 13

Evaluation Function:

When there is a cutoff at a certain depth level *l*, the an

1. When cutoff leaves are “max” level:

10 \* number of player pieces captured – 5 \* number of AI pieces captured

+ number of jumps possible by AI

2. When cutoff leaves are “min” level:

10 \* number of player pieces captured – 5 \* number of AI pieces captured

– number of jumps possible by player