**Checkers**

**Documentation**

**Arafat Qureshi**

**Quadii McMillian**

**Zabih Shinwari**

Among all the different two player games and their types, checkers is a really interesting game with a great scope and a high level of interest. Checkers has a great capacity to improve the logical and reasoning skills of its players. It challenges each player to draw upon their intellect and cognitive reasoning abilities to strategically choose a move/path that will provide the greatest possibility for success while at the same time, limiting their opponent to choices that provide the least possibilities for success.

A checkers game can be played in two different modes, single-player and two-player. Of these different types, the single user checkers game is really the most interesting and challenging, in our opinion, as a player plays the game with the computer as his or her opponent.

What makes single player checkers so challenging is the logic behind the computer’s “cognitive” abilities provided through artificial intelligence to allow the computer to make the best possible move at each board-state of the game. Artificial intelligence and, in particular, the Minimax zero-sum game searching algorithm allows the computer to look four, five, and six moves ahead when making decisions. This is impossible for a human and there in lies the challenge and intrigue in single player checkers. This is one of the reasons single player checkers is so popular among the gaming world.

The aim and objective of our research is to develop a cross platform application for a checkers game that can be played between a human player and computer using the algorithms and techniques related to artificial intelligence.

**Following are the research objectives**

* To critically review two player zero-sum computer games and how they are implemented
* To identify the role of AI in developing a checkers game and the algorithms used to create them
* To develop a single-player desktop checkers game that can be used to play against the computer
* To evaluate the performance of the AI algorithm in terms of their moves
* To create an application that offers some major features like undo and redo functions, and has an option for loading and saving the game and the application will be user friendly with a good graphical user interface).

**Minimax Algorithm**

A minimax algorithm is a recursive algorithm for choosing the next move in a two-player game. A value is associated with each position or state of the game. This value is computed by means of a position evaluation function, and it indicates how good it would be for a player to reach that position. The player then makes the move that maximizes the minimum value of the position resulting from the opponent's possible following moves. If it is A's turn to move, A gives a value to each of his legal moves.

**Alpha-Beta Pruning**

Alpha–beta pruning lies in the fact that branches of the search tree can be eliminated. This way, the search time can be limited to the 'more promising' subtree, and a deeper search can be performed in the same time. Like its predecessor, it belongs to the branch and bound class of algorithms. The optimization reduces the effective depth to slightly more than half that of simple minimax if the nodes are evaluated in an optimal or near optimal order (best choice for side on move ordered first at each node).

**User Interface**

Using Kivy library we are making the graphical interface of our game that can be run on any operating system. And one can play that game on the touch screen as well. It is easy to use with mouse clicks to designate moves, sound, and color to prompt the user; or, give feedback as appropriate, automatic generation of an algebraic notation game listing, and allow users to change sides.

**References**

Schaeffer, Johnathan & Lake, Robert - Solving the Game of Checkers. Web. 21 Oct. 2015.

URL: <https://chessprogramming.wikispaces.com/Jonathan+Schaeffer>

Balensiefer, Steven, Grenzer, Bill - Artificial Intelligence for Checkers. University of Notre Dame. Web. 20 Oct. 2015. <URL:http://www3.nd.edu/~cseprog/proj01/projects/wgrenzer/paper.pdf>

Avneet, Kaur, Dhawan & Singh, Jaswinder - Modeling and Designing of Machine Learning Procedure as applied to game using AI. 11 Jul. 2011. ACM Digital Library. Web. 22 Oct. 2015.

Benbassat, Amit, Sipper, Moshe - Evolving board-game players with genetic programming. University of Negev, Beer-Sheva, Israel. Proceedings of the 13th Annual Conference Companion of Genetic and Evolutionary computation. Jul. 2011. ACM Digital Library. Web. 22. Oct. 2015.

Markey, Luke - A Peer-To-Peer Checkers Program. Mar. 2011. ACM Digital Library. Kennesaw State University. ACM Proceedings of the 49th Annual Southeastern Regional Conference Web. 23. Oct. 2015.