**Khulna University of Engineering & Technology**

Department of Computer Science and Engineering

****

**Course Title:** Artificial Intelligence Laboratory

**Course No:** CSE 4110

Report on “**Chess Game”**

**Submitted To:**

Abdul Aziz

Lecturer, Department of CSE, KUET

&

S M Taslim Uddin

Lecturer, Department of CSE, KUET

**Submitted By:**

Nishat Anjum Lea

Roll: 1607009

Md. Rezwan Al Kaoser

Roll: 1607017

Projapoti Roy

Roll: 1607041

4th Year 1st Semester

**Date of Submission: 16 March 2020**

**Objectives:**

1. To check mate the opponent king.
2. To seek draw by repeated moves if the game is not making progress.
3. To draw the game after some evaluation.

**Introduction:**

A board game for two players who move their pieces according to specific rules; the objective is to checkmate the opponent's king.

There will be two opponents placed face to face. They differ from each other by colors like black, white. Players will play by turns. Players take pieces when they encounter an opponent in their movement path. Players cannot take or move through their own pieces.

1. Pawns only move forward. On the first move a pawn can move one or two spaces, every subsequent move can only be one space. Pawns move diagonally to take opponents.
2. Rooks move in a continuous line forwards, backwards and side-to-side.
3. The queen moves in continuous diagonal and straight lines. Forward, backward and side-to-side.
4. The king can move in any direction, one square at a time. A king cannot move to a square that is under attack by the opponent.
5. A king is in check when an opponent's piece is in a position that can attack the king. A player must move their king out of check, block the check or capture the attacking piece. A player cannot move their king into check.
6. Putting an opponent's king in "checkmate" is the only way to win the game. A king is in checkmate if it is in check, the opponent's piece that has the king in check cannot be captured, the check cannot be blocked, and the king cannot move to a square that is not under attack.

We will implement this game using min-max algorithm with alpha-beta pruning. Alpha-beta pruning is an optimization method to the minimax algorithm that allows us to disregard some branches in the search tree. This helps us evaluate the minimax search tree much deeper, while using the same resources. A search tree will be created from which the algorithm can choose the best move. This is done by using minimax algorithm. The alpha-beta pruning is based on the situation where we can stop evaluating a part of the search tree if we find a move that leads to a worse situation than a previously discovered move.The alpha-beta pruning does not influence the outcome of the minimax algorithm — it only makes it faster.

Methodology:

We made some changes from the original game.

Super Queen:

In original game, Queen and Knight had different types of move in the board but we made a new piece named Super Queen which alone moves like Queen and Knight. The traditional Queen can move in every direction – horizontally, vertically and diagonally. The traditional knight moves unconventionally compared to other chess pieces. Whereas other pieces move in straight lines, knights move in an “L-shape”—that is, they can move two squares in any direction vertically followed by one square horizontally, or two squares in any direction horizontally followed by one square vertically. Our Super Queen can move in any direction along with “L-shape” move.

Super Rook:

In our game, A new type of Rook is made which we will call Super Rook. It will be able to move like Rook and Bishop. The traditional bishop chess piece moves in any direction diagonally. The traditional rook moves horizontally or vertically, through any number of unoccupied squares. Our super rook can move diagonally, horizontally and vertically.

Board:

Traditional Chess board is 8x8 but our game’s board is 8x4

In our game the pieces are Super rook, super queen, King and Pawn

We used min-max algorithm with alpha-beta pruning in this game. The game is between the player vs computer. The player tries to checkmate the computer.

Discussion:

In our game there are mainly 4 files. Main.py, AI.py, Board.py and Pieces.py. In AI.py we applied the min-max approach. We used heuristic technique by using evaluate function in heuristics class. Here we also used piece position score and material score. In AI class, we used minimax and alpha-beta. The best move for every piece is calculated here by using minimax and alpha beta pruning approach.

**Conclusion:**

We are going to program a chess-playing-algorithm that can play basic chess. With alpha-beta, we get a significant boost to the minimax algorithm. The alpha-beta algorithm also is more efficient if we happen to visit **first** those paths that lead to good moves. The strength of this simple chess-playing algorithm is that it doesn’t make stupid mistakes.