****

A REPORT ON GAME DEVELOPMENT PROJECT OF ARTIFICIAL INTELLIGENCE LABORATORY

**CANADIAN CHECKERS BOARD**

By

Rahat Mahmud Khan (1707040)

Talha Ibne Mahmud (1707057)

Rakibul Haque (1707060)

**COURSE TEACHERS:**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Abdul Aziz**

Assistant Professor

CSE, KUET

**S.M. Taslim Uddin Raju**

Lecturer

CSE, KUET

**Introduction**

The focus of AI is creating intelligent machines capable of solving problems as a human would. Besides AI is used in making several types of games. A simple example is two player games.

In two player games, the players alternately play the game. It is the game of perfect information. This game is used the min-max algorithm which determine the best moves for the computer. This is a Canadian Checkerboard game which will determine many things of the Artificial intelligence in the game.

**Deterministic**- A game is deterministic if there is no element of randomness. The next events should be perfectly predictable given the knowledge of the previous game states and the player’s actions.

**Canadian Checkerboard**- Canadian Checkers Board games involve pure competition. A player’s loss is another player’s gain and vice versa. One player tries to maximize one single value, while the other tries to minimize it. So, when every piece of a player will be removed then the opponent will win the game gain the highest point.

**Give Perfect Information**- A game gives perfect information if each player knows all the events that previously occurred.

**Game Description**

In this project, we have built a game called ‘Canadian Checker’s Board’ is an extended version of Checker’s board game.

* In which two players alternately put black piece and white piece in the opposite edge cells of a board formed with 12 x 12 shaped matrix.
* Each player (AI & Human) tries to make maximum numbers of pieces to remove from the opposite side (only by diagonally) using the cornering moves.
* The player who removes all the pieces from the opposite side is known as the winner also who will gain the most point from that table checker.
* If any piece moves to the end of the opposite corner, it will be crowned as king and can move any direction as well as any position of the checkerboard.
* If there is no move for a player, then the game will be declared as dead or draw position.

**Game Rules**

Now, there are simple rules in this game. These are:

**Moves:**

* The general rule is that all moves and captures are made diagonally.
* The player with the dark pieces moves first. Then turns alternate.
* Ordinary pieces move one square diagonally to an unoccupied square.

**Capture:**

* Enemy pieces can and must be captured by jumping over the enemy piece, two squares forward or backward to an unoccupied square immediately beyond. If a jump is possible, it must be done, even if doing so incurs a disadvantage.
* Multiple successive jumps forward or backward in a single turn can and must be made if after each jump there is an unoccupied square immediately beyond the enemy piece.
* It is compulsory to jump over as many pieces as possible. One must play with the piece that can make the maximum number of captures.

**Crowning:**

* A piece is crowned if it stops on the far edge of the board at the end of its turn.
* Crowned pieces, sometimes called kings, can move freely multiple steps in any direction and may jump over and hence capture an opponent piece some distance away and choose where to stop afterwards, but must still capture the maximum number is possible.

**Win Vs Draw:**

* A player with no valid move remaining loses. This occurs if the player has no pieces left, or if all the player’s pieces are obstructed from moving by opponent pieces.
* A game is a draw if neither opponent has the possibility to win the game.
* The game is considered a draw when the same position repeats itself for the third time.
* A king-verses-king endgame is automatically declared a draw.

**Algorithm**

function minimax(

node, depth, isMaximizingPlayer, alpha, beta

):

if node is a leaf node or game end:

return value of the node

if isMaximizingPlayer:

best value = -INFINITY

for each child node:

value = minimax(

node, depth+1, false, alpha, beta

)

best value = max(best value, value)

alpha = max(alpha, best value)

if beta <= alpha:

break

return best value

else:

best value = +INFINITY

for each child node:

value = minimax(

node, depth+1, true, alpha, beta

)

best value = min(best value, value)

beta = min(beta, best value)

if beta <= alpha:

break

return best value

**Outputs:**

**Background pattern

Description automatically generated**

Figure 1: Initial State of the game

Background pattern

Description automatically generated

Figure 2: Movement detection of the game

Background pattern

Description automatically generated

Figure 3: Captured pieces by using move

A screenshot of a game

Description automatically generated with low confidence

Figure 4: Piece that will be crowned

A screenshot of a game

Description automatically generated with low confidence

Figure 5: Piece captured by king

**Result:**

**A screenshot of a game

Description automatically generated with medium confidence**

Figure 8: Result of the game (Human won this time)

**Conclusion:**

This is our overall game project. In this project, AI is implemented perfectly with different AI algorithms. In the game, sometimes AI wins and sometimes Human. To optimize the result, in minimax procedure with a specific maximum search depth (k=3) was used. For larger k value AI performs better. But larger k makes the system slower. Alpha Beta pruning algorithm was implemented. Draw situation rarely happens as well.