FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

BACHELOR THESIS

Devyanshu Koirala

Artificial Intelligence for Quoridor game

Department of Software and Computer Science Education (KSVI)

Supervisor of the bachelor thesis: Mgr. Klára Pešková, Ph.D.
Study programme: Computer Science
Study branch: Artificial Intelligence (AI)

Prague 2023

I declare that I carried out this bachelor thesis independently, and only with the cited sources, literature and other professional sources. It has not been used to obtain another or the same degree.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Sb., the Copyright Act, as amended, in particular the fact that the Charles University has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 subsection 1 of the Copyright Act.

In .... date ....

Author's signature

Dedication.

Title: Artificial Intelligence for Quoridor game

Author: Devyanshu Koirala

Department: Department of Software and Computer Science Education (KSVI)

Supervisor: Mgr. Klára Pešková, Ph.D., KSVI

Abstract: Quoridor presents a challenging terrain for strategic decision-making, making it a suitable testing ground for various AI algorithms. This thesis explores the implementation and evaluation of three distinct AI algorithms, namely A*, Minimax with Alpha-Beta Pruning, and Monte Carlo Tree Search (MCTS), within the realm of Quoridor gameplay.

The research begins with a comprehensive overview of the Quoridor game, its rules and strategies. Subsequently, we delve into the theoretical foundations and practical implementation details of the aforementioned AI algorithms and conduct a thorough evaluation in an effort to determine the best one in this context.

Keywords: Quoridor Artificial Intelligence AI Board Game

Contents

Introduction

Artificial Intelligence has brought a revolutionary transformation to the gaming world, pushing the boundaries of what's achievable in both single and multiplayer gaming experiences. In recent years, AI has taken center stage with its remarkable accomplishments in mastering age-old games such as Chess, Go, and many others. AI-driven games now offer users the opportunity to hone and enhance their skills, providing varying difficulty levels and offering optimal moves to guide players through each step if desired.

Designed by Mirko Marchesi, Quoridor stands out as an engaging and strategic board game that pits two or four players against each other in a thrilling race to traverse a maze-like board and reach the opposite side. Played on a chess-like board, this game introduces a fascinating twist with walls that players strategically place to obstruct their opponents' paths, compelling them to navigate longer alternate routes. Despite its seemingly simple rules, Quoridor demands a unique blend of strategic foresight and the ability to anticipate the moves of opponents. It's a dynamic and thought-provoking game that challenges the mind.

The primary objective of this thesis is to construct a well-structured framework and user-friendly interfaces that seamlessly integrate AI algorithms into the Quoridor game. The development of AI algorithms customized to Quoridor's unique rule set will not only enhance our understanding of the game's intricate nuances but also facilitate the creation of an intuitive interface for simulating these AI agents. Furthermore, a comprehensive evaluation will be conducted to identify the top-performing AI agent.

In addition to this, the project will encompass the creation of a user-friendly interface that empowers players to engage with an AI opponent of their choice, thereby bolstering the game's accessibility and inclusivity.

0.1 Acknowledged Works

Quoridor, being a widely popular game, has attracted a fair number of attention from the research community, resulting in successful AI agent developments. Some of the notable works include:

- **Mastering Quoridor [Lisa Glendenning, 2002]** The writer implements and assesses various algorithms like Negamax, Alpha-beta negamax among others. Additionally, they utilized a genetic algorithm to refine the weights within a linear weighted evaluation function, employing 10 distinct features suggested by the author, some of which include player's position towards their goal side, the opponent's position towards their respective goal, the remaining count of walls available to the player, etc.

In sharp contrast to the aforementioned paper, our thesis takes a distinctly different path by delving deeply into the architectural aspects of AI. Our approach emphasizes abstraction to the greatest extent possible, with an eye on facilitating seamless integration into a broad spectrum of games. We prioritize creating an interface that is adaptable to diverse game environments, setting our research apart from the game-specific focus of the prior works.

1. Description

The game begins with an NxN chess-like board where each square represents a potential position for the game pieces. It contains grooves that runs between the squares where players can place their walls. Each player is represented by a pawn represented by a character label that start on opposite sides, with their pawns located at the center of their respective edge. The primary objective is to be the first player to move their pawn to the row of squares on the opposite side of the game board avoiding any walls deterring its path to the goal while strategically placing walls to deter opponents from reaching their goal squares.

Walls are a fundamental element of the game, allowing players to strategically block their opponent's path and influence the course of the game. Each wall spans across two cells and occupies exactly four cells either horizontally or vertically, effectively creating a barrier between them. At the beginning, each player starts with a set number of walls that they can use during their turn.

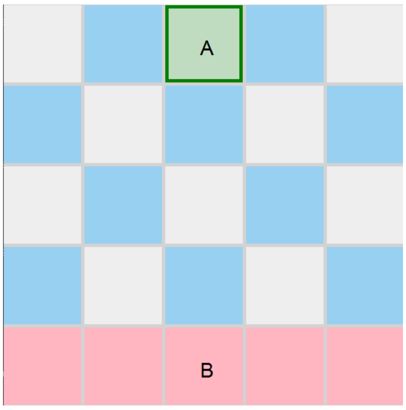


Figure 1.1: A 5x5 game board

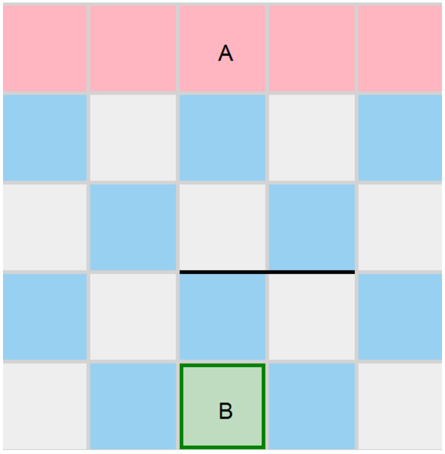
As seen in figure 1.1, In the beginning, player A starts at cell (2, 0) and player B starts at cell (2, 4). A's goal is to reach the cells highlighted in pink. The gray areas between the cells are grooves for wall placement.

1.1 Notation

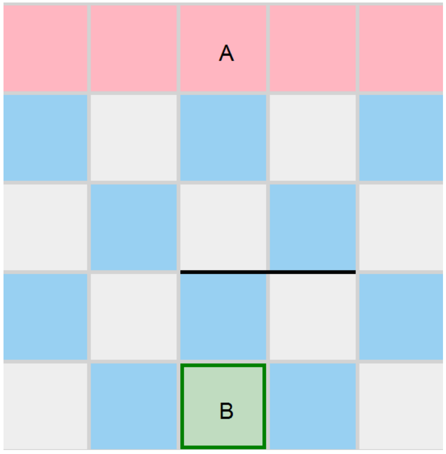
There are no official notations for this game. However, some popular ones recognized by the Quoridor community include **Glendenning's Notation** ([Lisa Glendenning, 2002]) and the **Quoridor Strats Notation** ([Quoridor Strats, 2014]).

Let $R = \{a, ..., i\}$ and $C = \{1, ..., 9\}$
Both notations follow the same principle of labelling each cell by C_{ij} where $i \in R$ and $j \in C$.
Move M and Wall W are denoted algebraically, where
$M(C_{ij}) = ij$ where $i \in R$ and $j \in C$,
$W(C_{ij}) = ijD$ where $i \in R$, $j \in C$ and $D \in \{h, v\}$

The difference between the two notations is the way the walls are represented.



(a) Glendenning's Notation: **c3h**



(b) Quoridor Strats Notation: **c4h**

Figure 1.2: Notation differences

As seen in Figure 1.2, the difference lies in the point of reference of the wall. In **Quoridor Strats Notation**, each wall coordinate is represented by the lower-left cell the wall follows along, whereas in **Glendenning's Notation**, each wall is represented by the upper-left cell the wall follows along.

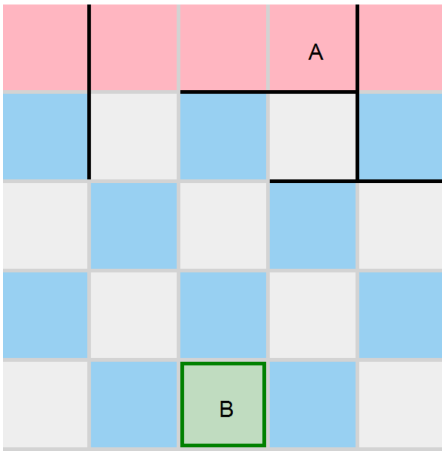
Even though they are widely used, they are very easy to get confused with since they have the same wall representations, and unless specified explicitly, it is difficult to tell which representation is being used.

This is why I have decided to use a different representation for walls. Instead of vertical and horizontal wall with respect to a cell, we define a direction explicitly.
$W(C_{ij}) = ijD$ where $i \in R$, $j \in C$ and $D \in \{N, S, E, W\}$
Looking back at Figure 1.2a, the walls can now be represented by **c4N**, i.e. a Northern wall from the cell C_{c4}.

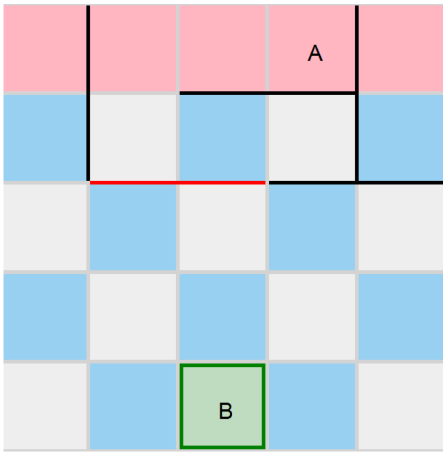
1.2 Rules

1.2.1 Wall placement rules

- Walls cannot be placed diagonally.
- A placed wall must not completely block any player's path to victory. Each player must have at least one path to victory *(See figure 1.3)*
- A placed wall cannot intersect any of the previously placed walls.
- Walls cannot be placed along the edges of the board. Walls must be placed to create a barrier for exactly 4 cells
- Every player possesses a limited supply of walls, and once they exhaust these walls, they are unable to place any additional ones. Consequently, the player is only allowed to maneuver their pawn on the board.



(a) Valid game state



(b) Invalid game state

Figure 1.3: Example of an invalid wall placement

The game state represented by Figure 1.3a shows the situation after 5 turns, with it currently being player B's turn to move. Since both **A** and **B** have viable paths to their respective goal rows and all walls have been placed according to the rules (see *Section 1.2.1*), the game state shown in Figure 1.3a is considered valid.

However, player B disrupts the rules by placing the red wall, violating the specified wall-placement rules (see *Section 1.2.1*), consequently rendering the game state represented by Figure 1.3b invalid.

1.2.2 Player movement rules

- Players are allowed to move their pawn one unit at a time in the North, South, East, or West directions during their turn. Diagonal movements are not allowed.
- **Jump**
  - If an opponent is at to the cell a player intends to move to, the player can jump over the opponent provided there is no wall between them or behind the opponent they intend to jump over. In the latter case, the player can jump to a cell on either side of the opponent's cell, given the cell is accessible from the opponent's cell.
  - Players cannot jump over walls
  - A Player is allowed to jump over multiple opponents as long as they adhere to the aforementioned conditions.

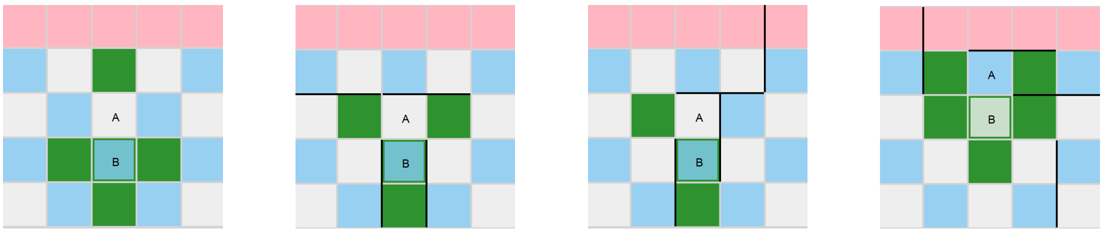


Figure 1.4: Examples of possible moves for player B in the game state

2. Game Analysis

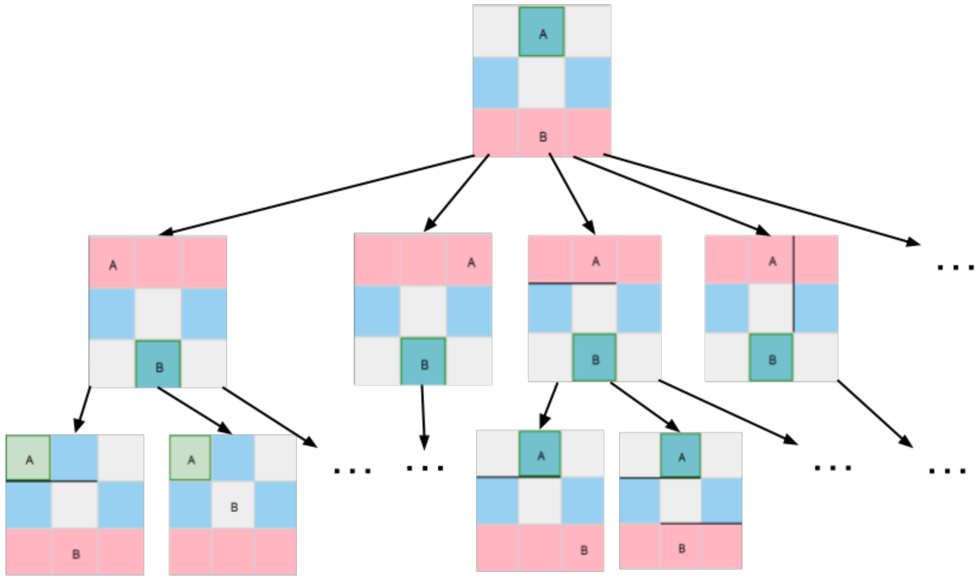


Figure 2.1: A partial game tree for a 3x3 game board.

2.1 Minimax

The Minimax algorithm is a decision-making algorithm commonly used in two-player, zero-sum games, such as chess, checkers, and tic-tac-toe. Its primary objective is to determine the best move for a player in a given game state by considering the potential outcomes of each move and selecting the one that minimizes the maximum possible gain for the opponent.

A high-level overview of the Minimax algorithm ((ADD PICTURES))

- **Game Tree**
  The algorithm constructs a game tree, where each node represents a possible game state, and the edges represent possible moves. The tree extends to various depths, representing different future moves and their consequences.

- **Evaluation Function**
  It can be computationally expensive for games with a large number of possible moves and deep game trees. For this reason, after a certain depth, a static evaluation function is used to determine the value of that game state. This function assigns a score to the game state, reflecting how favorable it is for the player whose turn it is.

- **Alternating Players**
  The algorithm alternates between two players, maximizing and minimizing. The player who is currently maximizing (MAX) aims to choose moves that maximize their own score, while the player minimizing (MIN) aims to choose moves that minimize the score of the maximizing player.

- **Backtracking**
  As the algorithm traverses the tree, it backtracks and carries information about the values of the nodes up to the root node. The root node's children are the possible moves, and the algorithm selects the move that leads to the highest value for the maximizing player.

- **Pruning**
  To optimize the search process and reduce the number of nodes to evaluate, various pruning techniques are often applied, such as the Alpha-Beta Pruning, which eliminates branches of the tree that are guaranteed not to affect the final decision.

Conclusion

Bibliography

Lisa Glendenning. Mastering Quoridor. 2002. URL https://api.semanticscholar.org/CorpusID:18564930.

Quoridor Strats. The Official Quoridor Notation, Sep 2014. URL https://quoridorstrats.wordpress.com/notation/.

List of Figures

List of Tables

List of Abbreviations

A. Attachments

A.1 First Attachment