PROJECT PROPOSAL
Project Title: Game Development – Rithmomachia with AI

4308281 | Athullya Roy | psyar7
Supervisor: Venanzio Capretta
23rd October 2019

1.1 Motivation and Background

From the conception of board games in the early dynastic period till today, board games have been played and enjoyed by individuals of all race and nature. From royalty to deprived masonry workers, board games have been significant to both education and recreation throughout history. Especially, due to the increased use of the internet and a recent surge in the gaming industry, more people are becoming aware and investing their time into playing board games. This is reflected on the fact that the global board games market is predicted to rise over $12 billon by the year 2023.[1]

Observing this recent growth in the industry, I decided to base the project around the development of such a game – Rithmomachia. Also known as “The Philosophers Game” or the “Battle of Numbers”, Rithmomachia was once popularised by students and academia for its representation of Boethian mathematical philosophy.[2] The popularity of the game has been fluctuating since its formation in medieval Europe. At its peak, the complex mathematical game was once a strong rival to Chess. Nonetheless, the game lost its flare in societies and even among students by the 17th century.

Through this project, I want to reintroduce Rithmomachia and adapt it to suit today’s board game culture by creating an AI player for it. AI players for board games are quite famous and have been very successful in development. DeepMind’s AlphaGo Zero is an AI developed to play the Chinese game of Go. By the year 2017, AlphaGo became the best Go player in the world by beating the world champion in all 3 games played.[3] AlphaGo learned using reinforcement learning – a neural network learned by playing against itself till it couldn’t be defeated in the game. As it plays, the neural network is tuned and parameters updated so that it can better predict the opponents moves, hence has a higher chance of winning each time. [3] Inspired by this level of progress in the field of AI opponents in games, I decided to attempt to create such an AI for Rithmomachia. Ideally, the AI should win all games that it plays against human opponents. However, due to the time limits and demands of this project, a high probability of winning is acceptable.

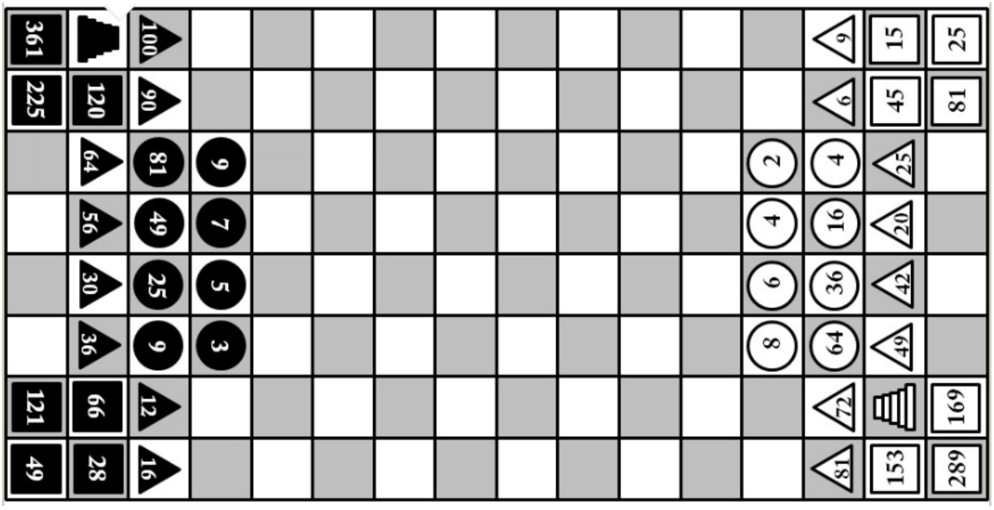


Figure 1: Rithmomachia Board - Initial State [4]

1.2 Rules and structure of Game

There are many variations to the game, due to the fact that it has been recorded by several authors in different ways throughout history. The version, I am contemplating on developing is described below. A sixteen by eight checkered board is required to play the game. Players are given sets of pieces which consist of the shapes - squares, circles and triangles. Pyramids are formed by stacking certain pieces together. The pieces all contain inscribed unique numbers. Similar to chess, the different shapes have unique directions they can move in. [5]

- Squares : Allowed to move four steps horizontally, vertically or diagonally
- Circles : Allowed to move one step horizontally, vertically or diagonally
- Triangles : Allowed to move three spaces horizontally, vertically or diagonally
- Pyramids : Follow the rules of the pieces it consists of

The game consists of two forms of victories – common and glorious. A common victory occurs when a certain number of pieces (decided by players beforehand) is captured by either parties. On the other hand, glorious victories occur when 3 pieces from a player reaches the opponents side of the board and they are arranged in either an arithmetic, geometric or harmonic progression. The shapes has to form an unobstructed line (horizontal or vertical) or a triangle on the board.

1.3 AI Opponent - Algorithms

There are many different approaches and algorithms that could be used to develop the AI component. Due to the lack of popularity of Rithmomachia, especially on the internet, there has not been any research in terms of developing an AI for it. However, AI development for other board game such as Chess and Go have been very successful in the past. From researching algorithms used to create these, I have come across the following which seems to be feasible and applicable to this problem.

Minimax is a backtracking algorithm which recursively searches through a game tree to find the optimal move for a player, assuming that the opponent's moves are always optimal as well. An evaluation function can be developed to assess the current game state and the possible end states to deem which player is likely to win with each move. Based on these values returned from the function, the minimax algorithm can be applied by assuming that at each node, the player whose turn it is chooses the value which maximises their chance of winning, and the opponent would choose the value which minimises the player's chance.

However, the main drawback of the minimax algorithm is that since it uses uninformed searching (depth-first search), the search tree will grow rapidly with each move the algorithm analyses; hence, it is computationally infeasible. This would affect storage space and will decrease the speed of the program significantly. However, a solution to this issue lies in alpha-beta pruning, where certain branches are pruned (not evaluated) to prevent wasting computation time or storage space.

A more advanced machine learning algorithm is the Monte-Carlo Tree Search. MCTS is a heuristic reinforcement learning algorithm which attempts to find the optimal move by only exploring certain nodes in the search tree. It consists of 4 stages – selection, expansion, simulation and backpropagation. The selection stage uses an evaluation function to choose the child nodes which has a higher chance of winning. When a node has been selected, the MCTS moves onto the expansion stage, where child nodes of the selected node is added to the search tree. In the simulation stage, a random playout is carried out from the selected child node till the game is completed. Finally, the result of the playout is used to update information on all the nodes that has been visited so far – this is the backpropagation stage. [6]

1.4 Technology

The game will be implemented in Python 3.7 using the IDE PyCharm. Various approaches could be taken to develop the graphics for the game. The most basic would be to use Pygame, which is an open-source Python programming library build on top of SDL and is used for graphics development. Another approach would be to use a game engine, such as OpenGL. If this is the case, I will have to install PyOpenGL, which is a Python OpenGL binding. The main benefits of using Pygame is that it is easy to learn and there is lots of support and tutorials available online. As well as that, Pygame should be sufficient for a 2d game development. Therefore, I have decided to use Pygame over game engines as it will be easier and more convenient for this scenario.

To create the neural network for the MCTS, I will use Keras which is a high level neural networks API in Python. I will also need to use libraries such as NumPy and math to calculate formulas for functions such as the evaluation function. All code will be uploaded to a Gitlab repository.

1.5 Aims and Objectives

The aim of this project is to develop a game of Rithmomachy and to produce an AI to play as the user's opponent. The core objectives which determines the success of the project are:

a) *Through extensive research, gain an understanding of the various approaches and algorithms used to develop AI for board games.*
Research into min-max algorithm, alpha-beta pruning, Monte-Carlo searching, machine learning etc. This research should provide sufficient information to decide which implementations will be the most feasible.

b) *Develop a game of Rithmomachia which implements the rules and multiple forms of victories stated above.*
Research into the libraries that can aid the game development and implement the game so it is authentic to the traditional board game.

c) *Design and develop a graphics interface for the game.*
Ensure that the representation of the game is easy and simple for the user's to understand.

d) *Enable two human players to compete against each other.*
This should be indication that the first part of game development is complete – The next focus should be developing the AI for the game.

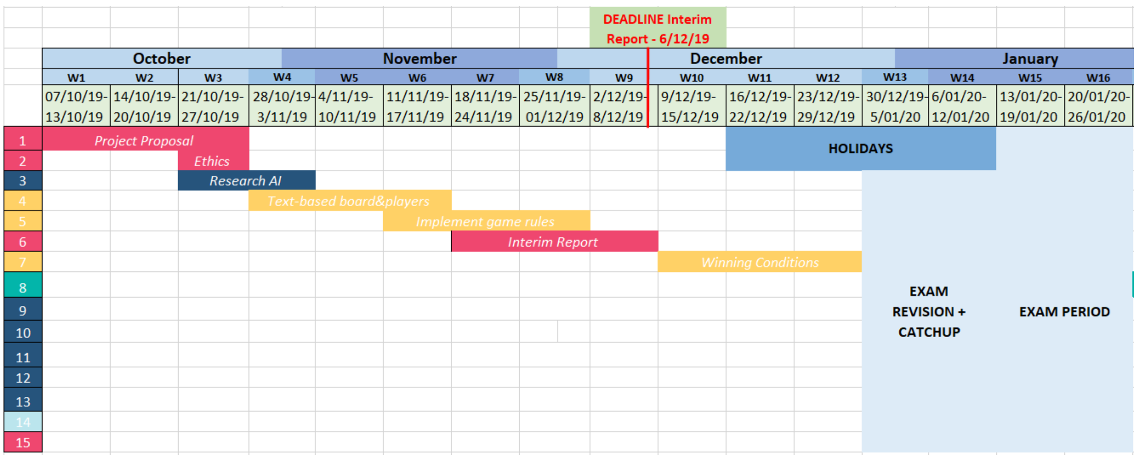
e) *Develop an AI to play against a human player.*
Start by designing a minimax algorithm. Continue developing better AI so that there is a higher chance of the computer winning the game.

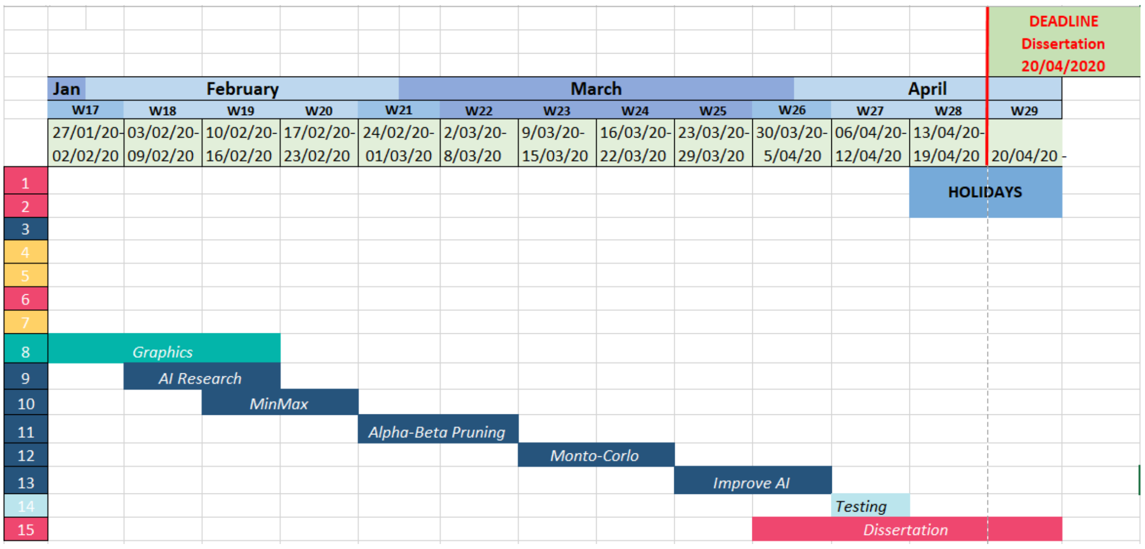
f) *Conduct the end user testing thoroughly*
Ensure that the user testing is done thoroughly so that statistics can be produced to understand how likely the AI opponent is to win a game.

1.5 Project Plan

The programming component of this project can be broken down to three significant parts – (A) developing a text-based game that works, (B) developing the graphics for the game and (C) developing the AI for the game. These are colour coded in the Gantt chart and shown in the table above.

Component	ID	Task
Planning	1	Propose a project plan to supervisor
Documentation	2	Complete the ethics checklist
C	3	Research into existing AI for games
A	4	Develop a text-based board, players and pieces
A	5	Program the enforcement of game rules
Documentation	6	Write the Interim Report for dissertation (deadline 6th December)
A	7	Implement the winning conditions – common and glorious victories
B	8	Develop the graphics for the game
C	9	Further research into AI and algorithms
C	10	Develop the MiniMax algorithm
C	11	Develop Alpha-Beta Pruning
C	12	Develop Monte-Carlo Searching
C	13	Improve the AI further
Testing	14	Test the program
Documentation	15	Write final dissertation (deadline 20th April)





1.6 References

1) Wood, L. (2018) ‘Global Board Games Market Outlook and Forecast 2018-2023: Major Players are Asmodee Editions, Hasbro, Mattel & Ravensburger’, *PR Newswire*, 07 August.
Available at: https://www.prnewswire.com/news-releases/global-board-games-market-outlook-and-forecast-2018-2023-major-players-are-asmodee-editions-hasbro-mattel--ravensburger-300693174.html

2) Moyer, A. E. (2001) ‘The Philosophers’ Game, Rithmomachia in Medieval and Renaissance Europe’, United States of America: *University of Michigan*, pp. 6-7.

3) Silver, D. and Hassabis. D. (2017) ‘AlphaGo Zero: Starting from scratch’, *DeepMind*, 18 October.
Available at: https://deepmind.com/blog/article/alphago-zero-starting-scratch

4) Tomas, H.M and Borquez. D, (2015) ‘Breve Historia de los Juegos de Mesa’, University of Cambridge: *Ludoteca de Pampala Press*.
Available at: https://ludotecapampala.wordpress.com/2016/06/01/rithmomachia-bhjm-11/

5)Newton, D. P. (1984) ‘Rithmomachia’ - Mathematics in School, vol 13, no. 2, *The Mathematical Association*, pp. 2-4. Available at: www.jstor.org/stable/30216194

6) Sharma, S. (2018) ‘Monte Carlo Tree Search’, *Towards Data Science*, 01 August.
Available at: https://towardsdatascience.com/monte-carlo-tree-search-158a917a8baa