INTEGRATION OF HEURISTIC TECHNIQUES WITH COMPUTER GO GAME

SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS OF THE DEGREE OF

BACHELOR OF ENGINEERING IN

COMPUTER ENGINEERING

BY

CHINMAY MAJITHIA

RAHUL MAKHIJA

PRAVEK KHATA

ABHAY MUKUL

GUIDE:

DR. JAYANT GADGE

(VICE PRINCIPAL,
DEPARTMENT OF COMPUTER ENGINEERING, TSEC)

COMPUTER ENGINEERING DEPARTMENT
THADOMAL SHAHANI ENGINEERING COLLEGE
UNIVERSITY OF MUMBAI

2022-2023

CERTIFICATE

This is to certify that the project entitled **“Integration of Heuristic Techniques with Computer Go Game”** is a bonafide work of

Roll No.	Name
1902090	Chinmay Majithia
1902092	Rahul Makhija
1902074	Pravek Khata
1902106	Abhay Mukul

Submitted to the University of Mumbai in partial fulfillment of the requirement for the award of the degree of **“BACHELOR OF ENGINEERING”** in **“COMPUTER ENGINEERING”**.

Dr Jayant Gadge
Guide

Dr. Tanuja Sarode
Head of Department

Dr.G.T.Thampi
Principal

Project Report Approval for B.E

Project report entitled (**Integration of Heuristic Techniques with Computer Go Game**) by

Roll No.	Name
1902090	Chinmay Majithia
1902092	Rahul Makhija
1902074	Pravek Khata
1902106	Abhay Mukul

is approved for the degree of ***"BACHELOR OF ENGINEERING" in "COMPUTER ENGINEERING"***.

Examiners

1.________________________________

2.________________________________

Date:

Place
:

Declaration

We declare that this written submission represents our ideas in our own words and where others 'ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will because for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

1) ______________________

(Signature)

Chinmay Majithia - 1902090

2) ______________________

(Signature)

Rahul Makhija - 1902092

3) ______________________

(Signature)

Pravek Khata - 1902074

4) ______________________

(Signature)

Abhay Mukul - 1902106

Date:

Abstract

Go is a strategy board game for two players in which the aim is to surround more territory than the opponent. The game was invented in China more than 2000 years ago and is believed to be the oldest board game continuously played to the present day. Our project is a Computer Go game that one can play against another person on a single device while also implementing some very useful and important features that would enhance the experience one has during playing the game free of cost and also be accessible to everyone.

Our project also includes a bot that will be able to play the game against the user skillfully and thus it can also be used to improve the user's game and showcase some excellent intellect.

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1

Introduction

1.1 Introduction

The game of Go is a very ancient game that was invented more than 2,000 years ago in China and is believed to be the oldest board game that is still played in the present. It has a basic strategy that to win one must cover more territory than the opponent. Go demands great skill, strategy, and subtlety and is capable of infinite variety, yet the rules and pieces are so simple that children can play. Special handicap rules allow players of unequal skill to play together. Aspiring professionals typically begin apprenticeships at a young age and train for years. A Japanese Go Association, founded in 1924, supervises tournaments and rules and ranks players, both professional and amateur. The European Go Federation was founded in 1950, and other regional and national organizations subsequently appeared. The first annual world go championship was held in 1979, and in 1982 an International Go Federation was established in Tokyo.The playing pieces are called stones. One player uses the white stones and the other black. The players take turns placing the stones on the vacant intersections (*points*) on a board. Once placed on the board, stones may not be moved, but stones are removed from the board if the stone (or group of stones) is surrounded by opposing stones on all orthogonally adjacent points, in which case the stone or group is *captured*. The game proceeds until neither player wishes to make another move. Our project is a Computer Go game that one can play against another person on a single device while also providing some very useful and important features that would enhance the experience of playing the game of Go and also giving the user a choice to play against a fully functional bot that would play the game skillfully.

1.2 Aim

The aim is to develop a fully functional Computer Go game that one can play against another person free of cost and that is accessible by everyone. Our aim is also to create a fully functional go-playing program that would be able to play the game skillfully. By doing this, we also try to promote this game as it is one of the oldest board games while also being one that uses intellect and knowledge.

1.3 Scope

The scope of our project involves various goals set by us that were necessary to deliver the product we are building. The various goals were to implement the Graphical User Interface and also implement the rules of Go in the same. We also implemented a fully functional go-playing program that would play the game skillfully. The board size can be of various sizes, but a 9x9 board has been chosen to keep the game simple for development purposes.

Chapter 2
Literature Review

2.1 Domain Explanation

The user interface of the game has been implemented using HTML, CSS and Javascript. We give the user various options or choices that he can make, for example, he can choose to play as white, black and even pass his turn. The user can also see the move number on the stones themselves to analyze and record the moves made by him and the opponent. The board is of a 9x9 size where a total of 81 stones can be placed. The game will automatically end when there is no space left on the board. A scoreboard is maintained on top which would indicate what advantage white is holding against black or what advantage black is holding against white. A Go playing program has also been implemented for the sake of the user so that one can play against it and improve their game.

2.2 Existing Solution

Existing solutions to this project include the IOS App named SmartGo, in which one can analyze their games and also play against an engine in increasing order of difficulty and thus improve their game of play. Sabaki GUI is another existing solution to this project in which one can play against another person but the disadvantage of using this user interface is that it offers a pro version which would require one to buy it whereas our project is free of cost. Another existing solution would be GoGUI, another interface created in java but it lacks the ability to keep track of the moves which we have incorporated in our project.

2.3 Hardware and Software Requirement

Hardware Requirements:

- Windows 10/11 PC
- Minimum 8 GB RAM
- Solid-state drive (SSD) is recommended for faster loading times and better performance.
- Stable Internet Connection is necessary

Software Requirements:

- **HTML**: HTML is the foundation of a website it contains the information that tells the browser what is on the page in terms of text, links, where to find images
- **CSS**: CSS is used for defining the styles for web pages. It describes the look and formatting of document which is written in a markup language. It provides an additional feature to HTML. It is generally used with HTML to change the style of web pages and user interfaces.
- **Javascript:**JavaScript is used for Show or hide more information with the click of a button, change the color of a button when the mouse hovers over it, slide through a carousel of images on the homepage, zooming in or zooming out on an image, displaying a timer or count-down on a website, playing audio and video in a web page, displaying animations and using a drop-down menu.
- **Visual Studio Code (VSCode):** VSCode is a popular and powerful code editor that is widely used by developers for web development, including HTML, CSS, and JavaScript. It provides a wide range of features such as syntax highlighting, code completion, debugging, and version control integration, making it easier to write and maintain code.

Chapter 3
Proposed System

3.1 Analysis

This project has several rules that were implemented according to the game. They include:-

The board is empty at the onset of the game (unless players agree to place handicap).

Black makes the first move, after which White and Black alternate. A move consists of placing one stone of one's own color on an empty intersection on the board. Players may pass their turn at any time. A stone or solidly connected group of stones of one color is captured and removed from the board when all the intersections directly adjacent to it are occupied by the enemy. (Capture of the enemy takes precedence over self-capture). No stone may be played so as to recreate a former board position. Two consecutive passes end the game. A player's area consists of all the points the player has either occupied or surrounded. The player with more area wins.

Basic strategic aspects include the following:
Connection: Keeping one's own stones connected means that fewer groups need defense.

Cut: Keeping opposing stones disconnected means that the opponent needs to defend more groups.

Life: This is the ability of stones to permanently avoid capture. The simplest way is for the group to surround two "eyes" (separate empty areas), so that filling one eye will not kill the group and therefore be suicidal.

Death: The absence of life coupled with the inability to create it, resulting in the eventual removal of a group.

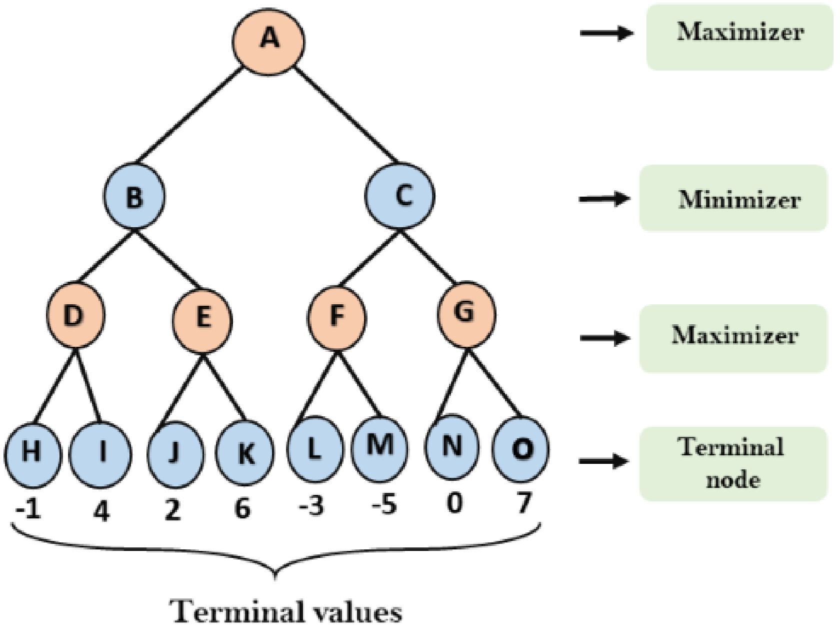
Invasion: Setting up a new living position inside an area where the opponent has greater influence, as a means of balancing territory.

Reduction: Placing a stone far enough into the opponent's area of influence to reduce the amount of territory he/she will eventually get, but not so far in that it is cut off from friendly stones outside.

The strategy involved can become very abstract and complex. High-level players spend years perfecting understanding of strategy.

3.2 Algorithm

The algorithm that has been incorporated in our project is known as the min-max search algorithm. It is one of the most commonly used algorithms in game theory used for the purpose of creating a game playing agent.



The algorithm assumes positions in the game to be nodes in the state tree, generates available moves, evaluates them using an evaluator function that uses the space occupied as a major factor and parameter for assigning values to the nodes in the game state tree. It then finds an optimal move by traversing the tree and maximizes the values for the game playing agent and minimizes them for the opponent.

3.3 Flowchart

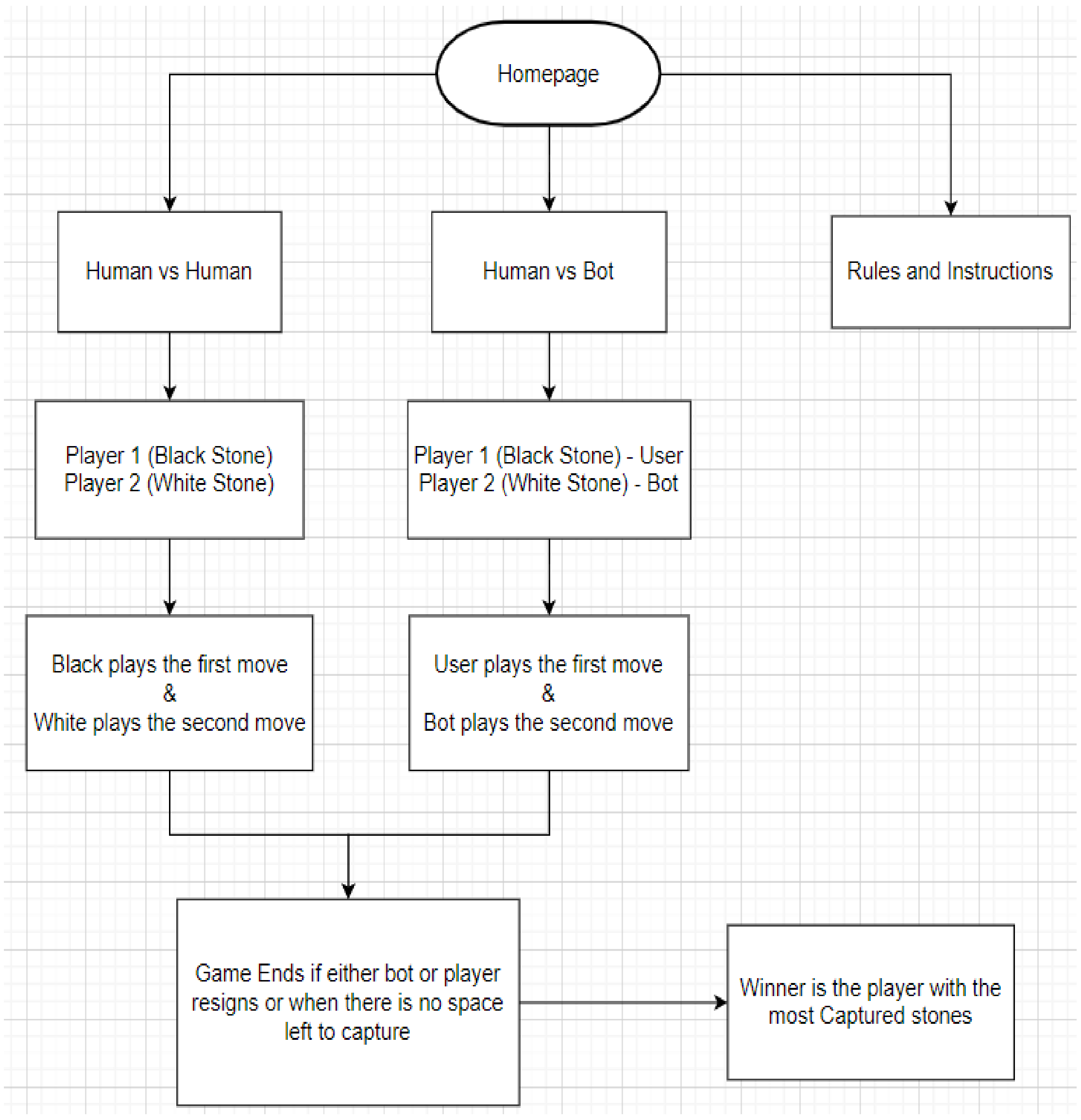

Homepage
Human vs Human
Human vs Bot
Rules and Instructions
Player 1 (Black Stone)
Player 2 (White Stone)
Player 1 (Black Stone) - User
Player 2 (White Stone) - Bot
Black plays the first move
&
White plays the second move
User plays the first move
&
Bot plays the second move
Game Ends if either bot or player resigns or when there is no space left to capture
Winner is the player with the most Captured stones

Fig. 3.2 - Flowchart of GO Game

3.4 Methodology

We thus propose a system for the game of Computer Go that would be very easy to use, simple, as well as have useful features, functions and an enhanced graphical user interface that would be responsive and interactive, also being accessible to everyone and free of cost. We have used various algorithms to implement the Graphical user Interface with the rules of the game, as they are listed below: -

1. The capture algorithm – This algorithm checks for the capture rule and implements it, that is, it checks whether a group of stones are completely surrounded by the enemy's stones. This is explained by the help of an example, as given below.

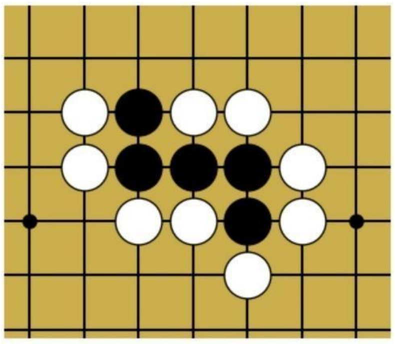
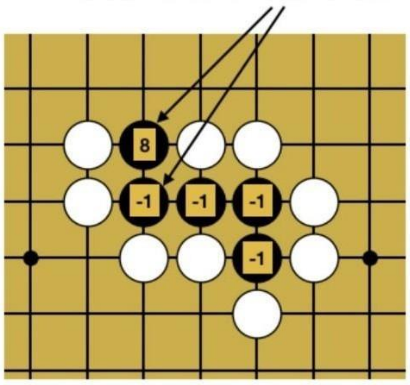


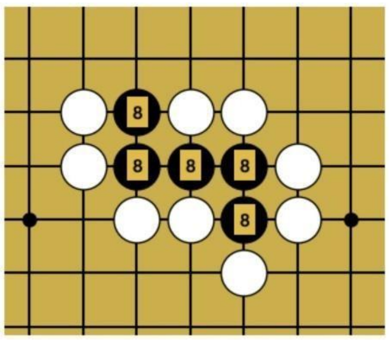
Fig. 3.3 - Capture Rule Implementation

The algorithm assigns values to the stones.

The stones with liberties should infect other stones.



This infection should spread one by one.



We need to repeat the process until all other stones of black colour are infected.

Thus we implement the Capture rule by given methodology, i.e,

the methodology of infection mechanism.

2. The KO Rule – This algorithm checks for the KO rule an implements it, that is, it checks whether a single capture move can lead to a repetition of moves that lead to an infinite loop. The algorithm is as follows:

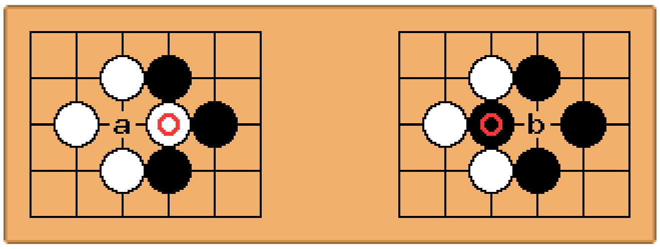


Fig. 3.4 - KO Rule Implementation

The algorithm first check if the previous move is played

It then checks if the previous move exists and if the stone played was of the opposite color.

Then, it checks if the stone is not an overwrite and it captures exactly one stone.

If this move captured exactly one stone at the location of the previous stone, then we reject the stone of the opposite color to capture that chain again.

3. Implementation of the bot – We have implemented a go playing program that would use the minimax algorithm with alpha beta pruning to find an optimal move in a particular position. This is done by considering the position to be a node itself in the game state tree and finds the optimal move, but in order to do so, it calculates best possible move for the opponent. If depth of calculations has not yet been reached then this process is repeated until the depth is reached. The parameters on which the algorithm depends are current node state of the board, depth of calculations and player's stone color. We assign values to the nodes in the game state tree and these values are used to determine optimal moves. These values are assigned using an evaluation function that uses certain parameters. These parameters were the sum of the number of stones inserted on the board by given player and number of empty

intersections which are surrounded from every direction by given player's stones. We also designed a move generation function that would determine the possible moves. These moves would satisfy the condition that every intersection that is empty and not surrounded by opponent's stones can be a valid place to play the stone on that spot on the board and thus can be played.

3.5 GUI Design

The tentative GUI design for the major modules of the project have been shown below :

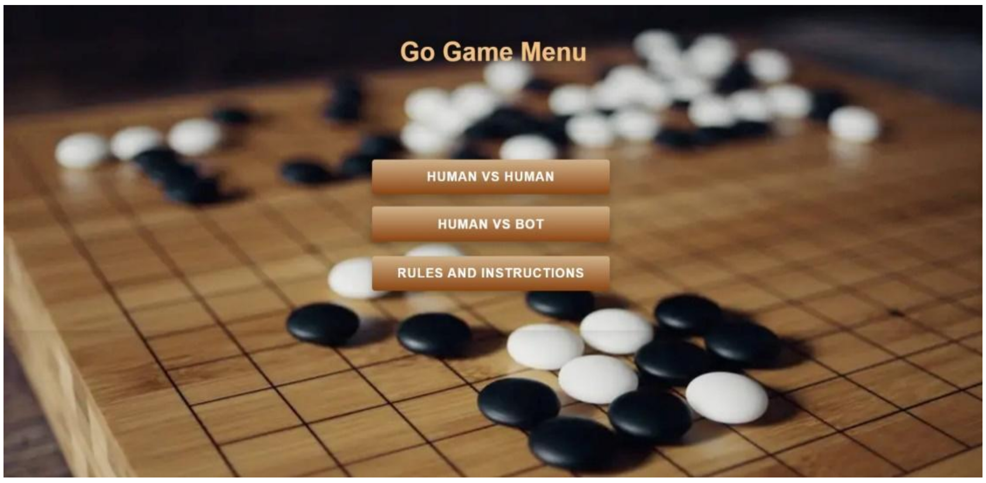


Fig. 3.5 - GUI Design for Home Page

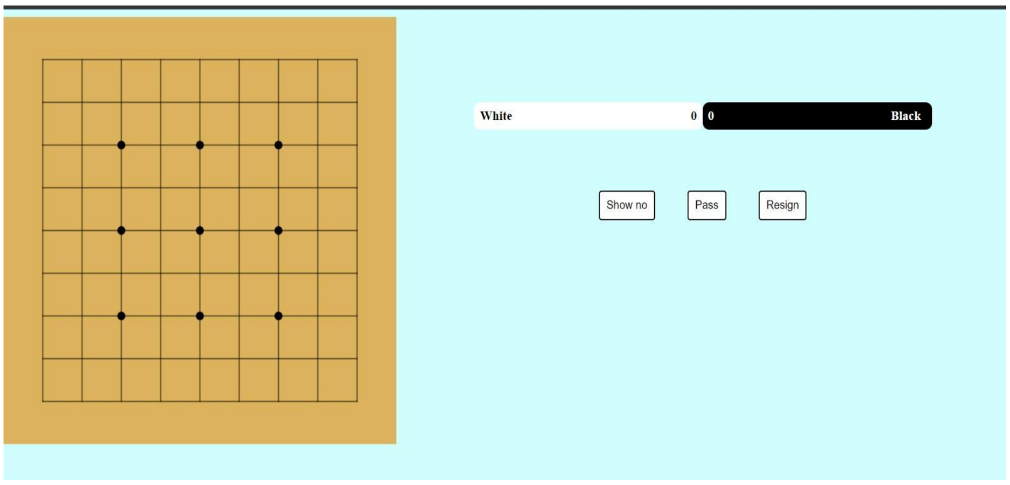


Fig. 3.6 - GUI Design of GO Board

Chapter 4

Implementation Details

This chapter describes the overall implementation phase.

4.1 Experimental Setup

The design includes implementation of various algorithms like the capture rule, which works on an infection mechanism, that is, once a stone is found that has a particular liberty, all other stones connected to that stone directly or indirectly will be infected in the sense that they will be given the same value and when a group of stones that have entirely been covered have the same value that entire group of stones is considered to be captured. The K.O rule has also been implemented that prevents repetition of moves that lead to an infinite loop and thus these two foundations or basic rules have been implemented in Javascript.

Implementation of the bot – We have implemented a go playing program that would use the minimax algorithm with alpha beta pruning to find an optimal move in a particular position. This is done by considering the position to be a node itself in the game state tree and finds the optimal move, but in order to do so, it calculates best possible move for the opponent. If depth of calculations has not yet been reached then this process is repeated until the depth is reached. The parameters on which the algorithm depends are current node state of the board, depth of calculations and player's stone color. We assign values to the nodes in the game state tree and these values are used to determine optimal moves. These values are assigned using an evaluation function that uses certain parameters. These parameters were the sum of the number of stones inserted on the board by given player and number of empty intersections which are surrounded from every direction by given player's stones. We also designed a move generation function that would determine the available moves. These moves would satisfy the condition that every intersection that is empty and not surrounded by opponent's stones can be a valid place to play the stone on that spot on the board and thus can be played.

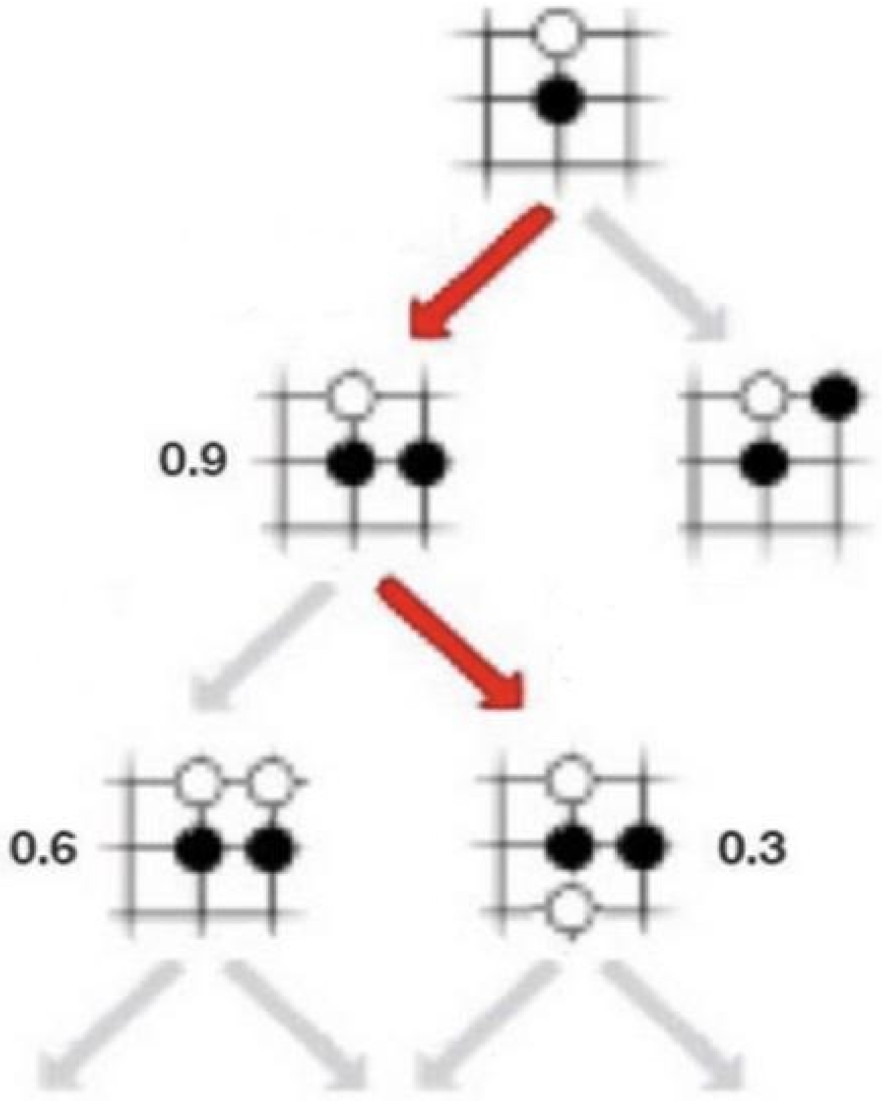


Fig. 4.1 - Minimax Algorithm Working

4.2 Software and Hardware Setup

Hardware Requirements:

- Windows 10/11 PC
- Minimum 8 GB RAM
- Solid-state drive (SSD) is recommended for faster loading times and better performance.
- Stable Internet Connection is necessary

Software Requirements:

- **HTML:**
  HTML is the standard markup language for Web pages. It can be assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript.
  HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects such as interactive forms may be embedded into the rendered page. HTML provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. HTML elements are delineated by tags, written using angle brackets. Tags such as <img /> and <input /> directly introduce content into the page.
  Other tags such as <p> surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags but use them to interpret the content of the page.
  HTML can embed programs written in a scripting language such as JavaScript, which affects the behavior and content of web pages. Inclusion of CSS defines the look and layout of content.

- **CSS:**
  As mentioned above CSS defines the look and layout of content. CSS is the language we use to style and HTML documents. CSS describes how HTML elements should be displayed. CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts. [3] This separation can improve content accessibility; provide more flexibility and control in the specification of presentation characteristics; enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file, which reduces complexity and repetition in the structural content; and enable the css

file to be cached to improve the page load speed between the pages that share the file and its formatting. As the project requires a little bit of a complex UI, CSS is used as it has a simple syntax and uses a number of English keywords to specify the names of various style properties.
A style sheet consists of a list of rules. Each rule or rule-set consists of one or more selectors, and a declaration block.

- **Javascript:**

JavaScript is an open-source programming language designed for creating web-centric applications.

It is lightweight and interpreted which makes it much faster than other languages and is integrated with HTML making it easier to implement in web applications.

JavaScript is a scripting language that is used to create and manage dynamic web pages, basically anything that moves on your screen without requiring you to refresh your browser.
JavaScript offers lots of flexibility. You can create stunning and fast web applications with tons of customizations to provide users with the most relevant graphical user interface.

JavaScript is now also used in mobile app development, desktop app development, and game development. This opens many possibilities for you as a Javascript developer.

Due to the high demand in the industry, there are tons of job growth opportunities and high pay for those who know JavaScript. The incredible thing about Javascript is that you can find tons of frameworks and libraries already developed, which can be used directly in web development. That reduces the development time and enhances the graphical user interface.

- **Visual Studio Code(VSCode):**

VSCode is a popular and powerful code editor that is widely used by developers for web development, including HTML, CSS, and JavaScript. It provides a wide range of features such as syntax highlighting, code completion, debugging, and version control integration, making it easier to write and maintain code. VSCode is cross-platform, lightweight, and customizable, making it a preferred choice for developers working on various projects.

Chapter 5

Results and Discussion

5.1 Performance Evaluation Parameters

In section 3.3 the key stakeholders of the application were identified. In section 4.2 the design details and tech-stack have been discussed. These results were obtained by the bot as listed below :-

We assumed the player with black stones starts (placing first stone in random intersection). If there is a tie (players has equal number of points), then the player with white stones should be a winner (Komi rule is not applied, so the player with white stones is in a worse situation).
Manual tests show increase of strategic thinking in algorithm and choosing better moves as depth of calculation increases (i.e. number of min-max calculations for both players), which causes more difficult to beat the algorithm. Nevertheless, it still reveals no consideration of a big strategy plan, so moves may seem to be repeated and inexperienced.

Unfortunately, as min-max calculation depth increases, the time of these calculations also increases.

Mean calculation times for 5 games for given depth:

depth	1	2	3
calculation time [s]	0,03	1,32	33,11

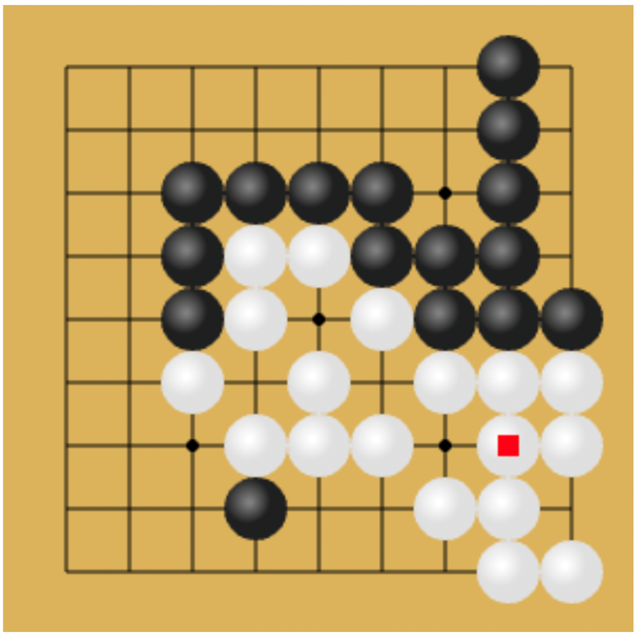
Mean value of points player with white stones obtained for 5 games for given players' depths (this is zero-sum game, thus positive outcomes for player with white stones are negative for player with black stones and vice versa)

White/black colors represents if mean outcome >= 0, then player with white stones mainly win, if mean outcome < 0, then player with black stones mainly win:

white \ black	1	2	3
1	-1,00	-17,00	-7,80
2	-5,80	-7,00	-2,20
3	16,60	3,80	-1,00

5.2 Implementation Results

Thus, the game has successfully been implemented in html, css and javascript with the go playing program using the minimax algorithm with alpha beta pruning completely written in javascript for a faster implementation. Images shown show the performance of the bot.



The above figure shows the performance of white against black.

As is visible, the performance of the white player surpasses that of the black player which thus implies that the white player (bot) is playing at a higher level and searching deeper for a better move.

Chapter 6

Conclusion and Future Scope

This was a really interesting project implemented under the guidance of Dr. Jayant Gadge. We are thankful to him and we look forward to complete the project with an engine designed for project that would use optimal go tactics and strategies to perform moves like tenuki (which means playing at a distance for greater purpose) and also include various levels of difficulty according to the requirement of the user.

References

[1] https://citeseerx.ist.psu.edu/viewdoc/download? doi=10.1.1.118.5927&rep=rep1&type=pdf

[2] https://webdocs.cs.ualberta.ca/~mmueller/ps/2013/2013-gochapter-preprint.pdf

[3] https://www.britgo.org/intro/history

[4] https://www.wired.com/2014/05/the-world-of-computer-go/

[5] https://webdocs.cs.ualberta.ca/~mmueller/ps/goeval.pdf

[6] https://www.davidsilver.uk/wp-content/uploads/2020/03/master-level-go.pdf

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to our project guide and mentor, Dr. Jayant Gadge, for guiding us throughout the semester. He has helped us by providing us with his insightful knowledge in areas of Web development. We thank them for their patience, motivation and much needed support. We will be forever grateful to him for the immense learning and experience we got while working on this project. We would also like to thank our Head of Department (HOD) of Computer Engineering, Dr. Tanuja Sarode ma'am for her support and the B.E. project coordinator, Dr. Ujwala Bharambe ma'am for approving our B.E. project topic and encouraging us to explore new technologies. Finally, we would like to thank our Principal, Dr. G.T. Thampi, for providing us with this wonderful opportunity and also we would like to thank everyone else who have directly or indirectly been of help to us in completing our B.E. project report for this semester.

Chinmay Majithia
Pravek Khata
Abhay Mukul
Rahul Makhija