Design Specifications for an Interactive Teaching Tool for Game AI using Gomoku

Erik Nygren
Abstract

This thesis seeks to understand and improve how students can learn the fundamentals of strategic game AI using a game-like application. The work focuses on the design specifications of a mockup application that can be used to teach a user the concepts behind the Minimax and Alpha-Beta Pruning algorithms using the strategic game Gomoku. The work aims to shed light on how technology can be used to teach students about technology and explore the possible ways it can be facilitated engagingly. The design of the mockup is based on concepts from education, human-computer interaction (HCI), and strategic game AI. The experimental learning model developed by David Kolb was used to structure the learning content, while frameworks from HCI were used to analyze the target audience and design the interface. The primary focus of the design was to rationalize the logic of the AI to the human audience using scenarios and exercises. The mockup was evaluated using a cognitive walkthrough with 5 participants. The results of the study indicate that the design can deliver an effective learning tool for teaching how game trees and the Minimax algorithm work. However, the results also suggest that the application struggles to teach the user more about the complex Alpha-Beta Pruning algorithm. Despite these results, more research and user tests are needed to determine whether the application is effective for the target audience.
# Table of Content

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>2</td>
</tr>
<tr>
<td>Background</td>
<td>4</td>
</tr>
<tr>
<td>Overview of Learning Content</td>
<td>6</td>
</tr>
<tr>
<td>Methodology</td>
<td>9</td>
</tr>
<tr>
<td>Design</td>
<td>14</td>
</tr>
<tr>
<td>Results &amp; discussion</td>
<td>26</td>
</tr>
<tr>
<td>Conclusion</td>
<td>28</td>
</tr>
<tr>
<td>Future Work</td>
<td>29</td>
</tr>
</tbody>
</table>
Introduction
Since the 1990s, the world has observed a series of events where an artificial intelligence (AI) program or agent has been able to beat the best and brightest within a game. Among the most famous were the series of chess games in 1996 and 1997 between IBM’s Deep Blue computer and the former chess world champion Garry Kasparov and in more recent years, the victory of DeepMind’s AlphaGo agent against the Go world champion Lee Sedol in 2016 [1] [2]. Each time an occurrence like this happens, journalists rush to cover the story, speculate on its significance, and attempt to explain how a computer beat a top-performing human in a game such as Chess, Go, or StarCraft.

Most people can guess that the success of such an AI lies in the incredible ability of computers to execute large numbers of calculations over a short period of time. Despite this partly explains the feat, it misses the incredible algorithms and methods used to organize the computer and its AI. Furthermore, most courses, literature, and online material intended to explain how such strategic game AI work often have a high bar of entry when it comes to understanding mathematics or computer science. This barrier makes it difficult for those without the know-how to gain a basic understanding of how a basic strategic game AI works.

This project addresses this divergence between the available knowledge and its potential audience by combining elements of human-computer interaction (HCI) and active learning to create the design specifications for a learning application. The application aims to enable the audience outside of computer science to get an introductory grasp of two central elements of strategic game AI: the Minimax and the Alpha-Beta Pruning algorithms. Gomoku was chosen as the game that links and contextualizes the use of the two algorithms for the user.

Problem Structure
The primary problem addressed in this thesis is to effectively communicate the two principal concepts within strategic game AI to an audience with little to no background within the field. The issue with the current state of available resources is that they are singularly targeted toward students studying AI. For example, multiple resources on YouTube and online websites provide context and a walkthrough of the algorithms covered in this thesis. However, these resources often fail to provide an end-to-end understanding of where the algorithm fits in and lack the interactive elements that allow students to experiment with the concept and validate their understanding without knowing how to code. On the other end, there has been a growing number of applications and websites that aid a player in becoming more proficient in a game through insights from an AI. The clearest example is where users can see scoring and suggested moves to improve their game based on an AI system. However, even though these applications allow the user to gain insight through interaction
with the AI, the user still does not get an understanding of how the AI works, just its outcomes.

**Purpose**
The purpose of this project is first, to serve as an introduction to the field of strategic game AI to those interested, and secondly, to facilitate the realization of a tool that could be used in the AI course at Uppsala University, Artificial Intelligence for Game Programming 2, 7.5 credits.

**Objective**
The objective is two-fold:
1. To effectively convey Minimax and the intricacies of search in a tree structure by Alpha-Beta Pruning.
2. To present specifications for an implementation that can be used in the AI target course.

**Delimitations**
This project will only focus on the fundamental implementations of Minimax and Alpha-Beta Pruning for the game Gomoku. As such, the project will not focus on sorting tree nodes in connection with Alpha-Beta Pruning.

**Contribution**
This specification aims to contribute to the specifications for a practical application of technology-enhanced learning. The aspiration is that the specifications can serve as a sample and source of learning for future applications intended to teach students Minimax and Alpha-Beta Pruning using a game-like environment.
Background

The problem addressed in this thesis is part of an age-old question about effective teaching. In recent years, gamifying content for teaching purposes has become increasingly popular. This thesis aims to add to this field of study by developing a teaching tool for strategic game AI.

There is widespread acceptance among scholars that games can be an effective tool for teaching, as noted by research such as Sara de Freitas. However, because of the widespread use of different methodologies, there remains insufficient long-term general evidence to conclude how a game should be structured for teaching purposes [3]. This is inherent to the field as it draws on concepts and ideas from game design, education, and the topic that it is facilitating. For example, scholars such as Malliarakis et al. have found that many educational games often are ineffective because they fail to successfully engage with the player and take on the "worst characteristics" of game design and education when attempting to teach. In those cases, the game lacks the journey and reward system and focuses entirely on the learning outcomes [4].

Nevertheless, many scholars have suggested principles and techniques to ensure that educational games are effective. Bellotti et al., for instance, suggest that creativity and decision-making should explicitly be part of the game design to make sure that the user is involved in the game journey [5]. Likewise, Iliya et al. find that stimulating emotions such as a feeling of success or achievement are vital to ensuring that the user remembers and truly learns the content provided to them [6].

Within the field of strategic games, there is wider adoption of game-based learning. This is especially true for teaching how to play and improve a player's performance in strategic games. The most notable example of this is the game of chess. Applications such as Chess.com or app.DecodeChess.com allow users to track, review and learn chess through AI-based tools. Such apps interactively link moves to articles and different types of scenarios and provide suggestions on strategies, thereby improving the player's game and awareness of their mistakes. Based on evidence from their own research, Chess.com, for example, found that their content had a "significant positive association with [the student's chess] rating change" compared to a control group that played the same number of games without the learning content [7]. However, much like the rest of the field of game-based learning, there is no unified methodology or framework that these games or applications follow to facilitate their content.
However, there are two noteworthy sources that have developed practical tools for teaching Minimax and Alpha-Beta Pruning. Firstly, Aleks Kamko’s interactive web application of the Alpha-Beta Pruning algorithm is the most similar to the application proposed in this project. Kamko’s application allows users to build their own game trees and see an animation of how the Alpha-Beta Pruning works [8]. Secondly, Sebastian Lague has developed a series of online videos whereof Minimax and Alpha-Beta Pruning are presented and walked through thoroughly. Unlike this project, Lague gives the user a step-by-step explanation of the algorithms by comparing them to a code implementation of the algorithms [9].
Overview of Learning Content

The core content of the application is divided into the following sections: the environment of the AI technique, the representation of alternatives (game tree), and Minimax and Alpha-Beta Pruning.

Environment

Game Environments for AI Systems

The characteristics of a game are critical to developing a strategy to reach a desired outcome. So is the case when building a game AI. One method for reaching the objective may dominate another depending on how the game works. A few definitions that could come in handy for a better understanding of strategic game environments are summarized below [10]:

- **Perfect vs. imperfect information:** Classification according to the amount of information available to the player. A game has perfect information if the game environment has all the information to make an optimal choice at all times. A game has imperfect information if the environment only reveals partial information.
- **Move:** In games, the term “move” is often used as a synonym for “action.”
- **Position:** The term “position” is often used as a synonym for the game “state,” e.g., the placement of the stones on a Gomoku board at a given time.
- **Randomness:** A game is stochastic or chance-based if randomness plays a part in how future states are derived. Examples of stochastic games are dice and card games such as poker.
- **Available moves:** A game is discrete if it has a finite number of actions and, similarly, continuous if it has an infinite number of available actions. In practice, however, continuous games are solely theoretical artifacts that are not implemented in modern computer games (which, as an example, work with data types with limited accuracies, such as the double-precision floating-point number “double”, using 64 bits to represent a decimal number).

Gomoku

This study focuses on Gomoku, a two-player zero-sum game where a player’s objective is to be the first player to place five pieces in a row. The player can win by having the five pieces lined up diagonally, horizontally, or vertically. The game is turned based and played on a 15×15 board, with black starting first. Gomoku is a perfect information game as all game states are fully observable. The game is also non-random and discrete.

Gomoku serves as the “anchor” in the application design developed in this thesis to which students can relate the process behind the AI. In other words, the game serves as a straightforward way to connect the theory of AI to its practice in a real game environment [11].
Game Tree
A game tree is a hierarchical graph representation of the possible actions that a player can take in sequential order. The root node serves as the initial game state before the first action, and each of the child nodes represents the possible actions following the game state represented by the parent node. The depth of the tree refers to the number of actions that can be taken one after the other during any given search process.

Minimax
Minimax is a decision rule algorithm for two-player turn-based games. The algorithm works by choosing the action that minimizes the possible gain from the opponent's available action and maximizing the minimum gain from the player's own possible actions. The mathematical theorem was proven by John von Neumann in 1928 and is foundational to Game Theory [12]. The derived Minimax algorithm has a time complexity of $O(b^d)$, and can be described using the following pseudo-code [10]:

```
Minimax (node, p)
if (leaf) return x
if (p is a max-player)
    \alpha = -\infty
    for each child, \alpha = \max (\alpha, Minimax (child, p^+))
else
    \alpha = +\infty
    for each child, \alpha = \min (\alpha, Minimax (child, p^+))
return \alpha
```

Initial call: Minimax (start node, player)

Figure 1: Game tree at starting position

Figure 2: A Pseudo code for Minimax [13]
**Alpha-Beta Pruning**

Alpha-Beta Pruning is a non-destructive algorithm for the optimization Minimax by pruning the search tree of redundant nodes. The algorithm will not evaluate an action node if at least one possibility proves that the action is equal to or worse than the evaluated actions. In other words, the algorithm aims to reduce the number of searches in the decision tree by excluding subtrees that are equal to or worse than the "least bad" action found so far [10].

![Figure 3: A Pseudo code for Alpha-Beta Pruning [13]](image)

Alpha-Beta Pruning has a best-case time complexity of $O(\sqrt{b^d})$ or a worst-case asymptotic complexity of $O(b^d)$, i.e., the same as the nominal time complexity of Minimax.
Methodology

Hypothesis
This application's primary goal is to facilitate the key learning outcomes behind Minimax and Alpha-Beta Pruning. The thesis behind this project is to assess and attempt to validate that gamification can be an effective tool for teaching two principal concepts within strategic game AI.

Requirements
The following requirements were developed to ensure and clarify that the user gains the necessary skills and knowledge to understand the learning content on a fundamental level.

Game Environment
For the users to even begin to use the application and understand the context effectively, they must demonstrate that they:

- Understand the rules of the game.
- Can play an effective game, i.e., motivate why they choose one move over another.

Game Tree
As a prerequisite for working with the two algorithms, the user must understand the following about a game tree:

- A tree’s hierarchical structure and each node’s relationship to future actions, i.e., each level corresponds to an action in the sequence.
- Which player owns each level of the tree and the objective of each player.
- How the values of the leaf nodes are determined.
- The connection between the tree and a move on the board.

Minimax
The critical learning elements in the case of Minimax are to:

- Be able to select the value of the nodes based on the leaf nodes.
- Understand that the algorithm applies to all available positions.
- Understand why the agent chooses the highest valued position.
Alpha Beta Pruning
The critical learning element in the case of the Alpha-Beta Pruning relates to ensuring that the user understands:

- Why Alpha-Beta Pruning is needed, i.e., understanding the time complexity that can arise with large game trees.
- How the AI selects the value of the nodes based on the leaf nodes using alpha ($\alpha$) and beta ($\beta$).
- To understand why a branch can be discarded based on the decision rule.
**Tools and Methods**

The tools and methods used in this project draw on a range of the disciplines involved. The following are the main concepts that were used to develop the application design.

**PACT**

People, Activities, Context, and Technologies (PACT) is an HCI design framework and analysis tool for understanding how, when, what, and where the user interactions occur and the desired outcomes [14].

- People refer to understanding the users of the technology, i.e., the who.
- Activities refer to the desired goals, tasks, and actions that the user is intended to reach through the actions.
- Context is the environment and the circumstance in which the interaction will occur.
- Technologies refer to how the HCI will take place.

**Experiential Learning**

Experiential Learning refers to a type of Active Learning that aims to "learn through reflection on doing". The technique is comparable to learning to ride a bike. The student learns by repeatedly trying until the student is successful. This study will use David Kolb's Experiential Learning Model (ELM) to structure the content of the learning application. The ELM model can be described in the following method [15]:

- The learner is part of the experiment.
- The learner reflects on the experience.
- The learner analyses the cause and results.
- The learner applies the learnings to the experiment.

**HCI**

Human-Computer Interaction (HCI) is the interdisciplinary study of technology designs with human interaction. HCI seeks to understand and identify design techniques and methods that lead to an intended interaction between humans and technology. The field has its origins in the 1980s and draws many of its findings from behavioral science, design, and computer science [16].

**Cognitive Walkthrough**

A cognitive walkthrough is a method to assess the functionality of an application using a range of reviewers that represent the application's target audience [17]. The cognitive walkthrough of this project was done using five persons within the target age range and with little to no formal background in computer science or strategic game AI. The walkthrough
was conducted using a simple mockup of the final application. The questions and results of the cognitive walkthroughs can be found in the appendix A and table 1 [18].

Participants
Student, Female, Age 16
Student, Male, Age 20
Student, Female, Age 24
Office Worker, Male, Age 25
Office Worker, Female, Age 28

Methodology
This project utilized a range of methodologies to create the application specifications and assess its efficacy. Firstly, the target audience was analyzed using the PACT framework. This was followed by the design of the learning content, which was done using a variation of the ELM model. In addition to concepts from HCI, the learning content was then used to create the Graphical User Interface (GUI) design and its logic. Finally, to evaluate that the content and design accurately captured the learning requirements and had done so in an engaging way, they were evaluated by the participants in a cognitive walkthrough and an evaluation by the author using experience from the target course.
Analysis of Target Audience - PACT Analysis

Target group: The application's primary target audience is students with an interest and a curiosity within the field of game AI but no formal education within it. The target audience can be characterized as within the age range of 16–29 and brought up in western society. The assumption comes from the fact that most (65%) higher education students in Sweden are within that age range [19]. This demographic group statistically has a high degree of experience with using online tools for learning and is well versed in quickly getting familiar with a new GUI [20].

Activates: As the application will only rely on visual elements such as text, diagrams, and symbols, the user’s actions are limited to clicking, typing, dragging, and dropping elements within the application.

Context: A study context such as a school, library or home were the only environments considered in this Thesis.

Technologies: The only device needed for this application is a computer. The application can certainly remain valid for smart phones and other technologies. However, this will not be considered in this study.
Design

Learning Content

The learning content is organized into five sections. The beginning and end sections are meant to introduce and incentivize the user to compete against the AI and become interested in its capacity. Each middle section intends to explain and teach the user about the AI using the ELM framework. In each middle section, the user is first asked to use their own intuition and experiment given a task, then they are presented with the content, an example, and finally tested before moving on to the next section. The design aims to ensure that the user gradually builds their understanding of the process.

![Diagram of Learning Approach](image)

**Figure 4:** Learning approach

Pre-Learning Content

Prior to introducing the three main sections of the application, the user is asked to try to beat the AI in one of three games. During this section, the user is also given a brief introduction of the application and its aim. The point of doing so is to assure that the player knows how Gomoku works, allow them to see the power of the AI in use, and build an expectation of what will be presented in the application. The player is not intended to win any games in the pre-learning part. This is to build intrigue about the game and build up a rewarding sentiment for when the user beats the AI in the Post Learning section.

Game Tree

Once the user has demonstrated that they can play three effective games against the AI, they are allowed to move on to the Game Tree section. The user is firstly tasked with using their own intuition to build a game tree. The user is given a specific move, a brief description of a game tree, and what components they can use. The user then drags and drops the tree elements to represent a given board configuration. The user is not graded or assessed in this attempt. However, they can only proceed once they have used all components. Once the user has guessed what a game tree looks like, they are presented with a step-by-step explanation of what it is and how it works. The explanation is done in sequential order, showing the root
node selection and how the children’s nodes are added until the correct depth has been reached.

The user is then asked to recreate a similar tree using a new root node on the board. Once the user has completed the tree with the correct number of nodes and depth, then the user is introduced to the idea of scoring the nodes in the tree. The user is tasked with filling in the values of the root nodes of the tree that they just created based on their own thinking. Once the user has filled the nodes, the user is given a detailed explanation of how a move is valued and a step-by-step example of how all nodes in a tree are valued. Once the user has gone through the example, they are tasked with creating a new tree and assigning the values to each node. Once the tree is correctly filled out and valued, they can proceed. The user is finally given a detailed explanation of the intricacies of the game tree. The content highlights the great sizes of a game tree and the in-order transversal method used in this application. The user is assigned two final tasks before moving to the Minimax section. Firstly, they are asked to identify the number of available moves given a scenario. Secondly, they are tasked with clicking on the nodes using the depth-first in-order transversal method. Once both are completed correctly, the user can move to the Minimax section.

**Minimax**

This section asks the user to identify the best move on a given board scenario with three different moves and associated trees. The user can see each tree by clicking on its available square. The user selects their chosen move by double-clicking on the select square. Then application asks them to motivate why an agent would act the way that the user has guessed. Once the student has made their guess and motivation, the user is given a step-by-step approach to solve the same case using the Minimax. First, the user is shown how the value of each of the three available spots is determined and then explains that the highest scored square is chosen. The walkthrough describes how the algorithm starts by traversing the tree from left to right and displays the points that are listed in the tree. Since three squares are considered, the user is given three different examples of how the Minimax algorithm works.

Once the user has gone through the explanation and seen how the algorithm works, they are then tasked with finding the best move when provided three new squares in the same scenario. The game trees of each of the three squares only have the leaf values filled in, and it is the task of the user to drag the value of the correct child node up the game tree. When the three game trees are completed and the Minimax values of each of the three moves have been found, the user must double click on the highest valued square. The user cannot move on to Alpha-Beta Pruning until all three trees are correct and the most optimal move has been chosen.
**Alpha-Beta Pruning**

In this section, the user is first asked to select any edges they thought were redundant in the game tree of the chosen node from the Minimax. The user does so by clicking on the edges of the redundant branches. Once the user has made their own attempt to see why a branch can be ignored, they continue to get a step-by-step explanation of the Alpha-Beta Pruning algorithm. The explanation is conveyed by the three examples used in the Minimax section. The user is given an annotated presentation of the algorithm as it proceeds through the three trees. The example begins by showing a selected point and its game tree. Then it progressively walks through the transversal of the tree and fills in the alpha and beta values until the tree has been fully explored. The annotation is made using text to aid the user in understanding why alpha is compared on one level and why beta is used on others.

Once the user has walked through the three examples, the user is given three new potential moves and asked to apply the Alpha-Beta Pruning technique to each tree. The task requires that the values are correct, the unexplored branches are marked, and the optimal action is chosen. If the user has filled in all three, they can finish the learning content and proceed to the end of the application.

**Post Learning Content**

Once the user has completed all sections, the user is congratulated and asked to try out if they can now beat the AI with their new insights as a player. The player can play as many times as they wish, and the score of their games should be used to motivate and display their progress.
GUI Design

The following GUI design was developed using the learning flow, the requirements, and insights from HCI and game design.

Colors

The choice of color is vital in providing the user with visual cues and associations between elements in the GUI. Therefore, the design of this GUI follows the design rules of Aaron Marcus based on the following color schemes [21]:

The game utilizes five primary colors: black, white, blue, red, and yellow, and one secondary color: beige. The board is beige with black borders to resemble the physical board of the game. Player one uses red stone (circles) to make moves on the board. These moves are the same color in the game tree. The stones and nodes are shaded if they are future moves. Player two uses blue stones on the board and blue triangles in the game tree. Similarly, they are shaded if they are future moves. Finally, the game also uses a yellow outline on the stones and nodes to signify a selected move and a black outline to highlight a leaf node.

The colors of the players were chosen with regards to two main reasons: firstly, the contrast makes it easy to distinguish the stone and see patterns. Secondly, red is typically associated with danger, so the color of the AI is easily associated with the opponent.

Simplifications

As noted by Benyon, humans have clear limitations with keeping the information in their active memory and gaining information from unstructured content [16]. For that reason, three main design choices were made to simplify the concepts behind the game tree, Minimax and Alpha-Beta Pruning: firstly, given that a game tree has exponential properties, it is important to limit the depth of three and branching factor of the tree, so that the user can understand the game tree without an overwhelming amount of information. Therefore, a depth of two and branching factor of three was chosen. A depth lower than 3 would make it difficult to showcase the value of Alpha-Beta Pruning. Meanwhile, any higher would create more work without adding more value for the user. Similarly, a branching factor below two would not demonstrate the complexity of the many options of the player, and a higher is more likely to confuse the player.

Secondly, to keep the user engaged, the application must balance repetition while introducing new elements and giving the user choices. Therefore, the examples and learning content at most can have three forms of repetitive content, such as examples or tests for the user. The
exception is if the user fails to pass the test sections to ensure that the content has been properly grasped.
GUI Layout
The core interface has two primary purposes: first, to provide a game context, and second, to facilitate the learning. In this case, the game context is Gomoku, while the learning is primarily done through text, images, and the game tree. The core GUI has two main focus points, as shown in figure 5. In addition to these two elements, the application also has a toolbar to navigate the game and the content.

Figure 5: The Interactive GUI
**Game Board**

The Gomoku board consists of a 15×15 square board. The board is present throughout the game and will only marginally change as the user goes through the three main sections of learning content. As stated, the blue stones represent the player, while the red represent the AI. Winning lineups are highlighted using a line through the winning stones. The AI will not be active during the three middle sections, so that the user can explore the board and the given scenarios. This ensures that the user can interact with the game tree as they are presented with new information about the game tree, Minimax, and Alpha-Beta Pruning.

![Figure 6: Board and all different board elements](image)

The game is based on the following logical rules to ensure that it works in accordance with the learning content and the rules of Gomoku.

**Board Logic**

- Red always starts the game
- The players lay a move after the other player.
- A move is committed if the player double clicks on an available tile.
- If the player is in pre-or post-learning mode, then the AI plays for the red player.
- If the player is in the learning mode and clicks on a given tile, then the opposing side’s two best options are displayed using the striped stones with the value of the action.
• If five stones are lined up diagonally, horizontally, or vertically, then the game ends and the winning lineup is highlighted. No more stones can be placed after a win.
• When a node in the game tree is selected by the mouse the corresponding stone on the board is highlighted using a yellow outline.

Navigation Bar
The navigation bar, like the board, is a permanent element of the application. The bar has three functions: reset the board, to advance, or go back in the learning content. The colors used in the examples below aim to have the starkest contrast that is possible and make sure that the focus is on the game and the content.

Figure 7: Navigation Bar

Navigation Bar Mechanics
• If the reset button is pressed, then create a new board.
• If the left arrow button is pressed, then the previous page will be shown, unless it is at the start page, then stay there.
• If the user has not filled in the information correctly, they will not be able to proceed. An error message will appear in the learning content.
• If the right arrow button is pressed, then proceed to the next content page and update the board accordingly. If the user is on the last content page and clicks on the button, do nothing.

Learning Content
The learning content changes as the user proceed throughout the application. The learning content is a combination of images, animations, text, and/or the game tree. The learning content aims to explain and simplify the concepts behind Minimax and Alpha-Beta Pruning. Ideally, this would be done using a step-by-step explanation of an event, idea, or feature. Inspiration can be drawn from the animation approach used in Kamko’s application [8].
**Game Tree**

The blue user’s nodes are blue triangles pointing up, while the red nodes are pointing down. The triangles indicate if the user is attempting to minimize or maximize the goal function. In the first part of the Game Tree learning, the edges and nodes are movable as the user must build their own tree.

![Figure 8: Building the game tree](image)

Once the player has completed the task of building their own tree, then the tree elements become unmovable. However, the user can click on the node to write in a value.

![Figure 10: Assigning the values to the game tree](image)
Game Tree Content Logic

- When building the game tree, the user will be tasked to first clicking on the square they would like to explore, thereby creating the root. Afterward, the user will have to drag and drop edges to build the tree.
- If a player clicks on the highlighted square on the board, then its tree will be presented.
- If the play makes a single click on a node, its corresponding move is highlighted on the board and vice versa.
- If the user is tasked with filling in the values of the leaf nodes and clicks on the leaf node then they can write in the value. In other cases, the user is not able to write in the leaf values.
- An edge must be attached to two nodes.
- A node can be attached to one edge and at most four edges.
Minimax and Alpha-Beta Pruning

To facilitate the learning of the two algorithms, the user is given a tree and a game scenario. When using the Minimax algorithm, the user can drag and drop up the values of the child node on to the parent node.

While in the Alpha-Beta Pruning scenario, the user must drag the value of the child node to the correct parent node and mark if the edge can be excluded by clicking on the edge.
Minimax and Alpha-Beta Pruning Logic

- If the user makes a single click on a node, its corresponding move is highlighted on the board and vice versa.
- During minimax, if the player drags a value from a child node to its parent node, then the value of the child is carried to the parent.
- If the user clicks on the edges, then it will become inactive (shaded) until it is clicked on again.
- If a node is inactive, then it will not be shown on the board.
- A user cannot drag and drop the value of an inactive branch.
- For the user to proceed, all nodes in the game tree must be correctly allocated.
Results & Discussion

A mockup of the application was created using the learning content and the GUI design. The mockup consisted of images and text and was presented to the cognitive walkthrough participants. The participants were asked to try to complete the tasks assigned to the user and assess the mockup verbally. The result of the study is presented in Table 1.

<table>
<thead>
<tr>
<th></th>
<th>Learning Objectives (median)</th>
<th>Clarity (median)</th>
<th>Engagement (median)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pre-Game</td>
<td>Reached</td>
<td>Clear</td>
<td>High</td>
</tr>
<tr>
<td>Game Tree</td>
<td>Reached</td>
<td>Somewhat clear</td>
<td>High</td>
</tr>
<tr>
<td>Minimax</td>
<td>Reached</td>
<td>Clear</td>
<td>Medium</td>
</tr>
<tr>
<td>Alpha-Beta Pruning</td>
<td>Not reached</td>
<td>Unclear</td>
<td>Low</td>
</tr>
<tr>
<td>Post-Game</td>
<td>N/A</td>
<td>N/A</td>
<td>Medium</td>
</tr>
</tbody>
</table>

Table 1: Results based on five test subjects

All five of the participants understood and effectively proceeded through the application during the Pre-game and Game Tree sections. Three of the participants reached the learning objectives in the Minimax section, and four out of five expressed that the description and task could be demonstrated in a more effective manner using more text, a video, or animation. Only one of the participants could reach the learning objectives in the Alpha-Beta Pruning section. The four other participants failed in their attempts to fill in the tree and mark the redundant branches. All five participants expressed that the Alpha-Beta Pruning learning content was unclear and hard to understand. One participant pointed out a significant shift in the difficulty and level of abstraction between the Minimax section and the Alpha-Beta Pruning section. The level of engagement from the participants was either high or medium except for during the Alpha-Beta Pruning section. Two participants suggested that the content should follow more of a story to create a common thread throughout the application to keep the engagement at high levels.

The cognitive walkthrough demonstrated that the application still requires further development to reach the dual objective of being highly engaging and effectively securing the learning outcomes for the user. From the exercise, it was also clear that there is a risk that the application puts too much focus on reaching the learning objectives. Consequently, the design failed to secure that the user was engaged and also failed to convey the information that was part of the learning. This case is an example of what Malliarakis describes as “Shavian reversals,” whereof the learning content has not been appropriately adapted to fit the game.
and the users. This is undoubtedly the case for the Alpha-Beta Pruning section and less so for the other sections of the application [4].

Finally, based on the feedback, it was clear that the emotional engagement of the users was not effectively reached. Stimulating an emotional response from the user is often critical for creating or sustaining the motivation to continue the work. However, in all five cases of the cognitive walkthrough, it was observed that none of the five participants experienced any emotional response to the content. It was suggested that the game could use more humor and characters to create more of a positive response from the user.
Conclusion

This thesis concludes with the specifications of an application for teaching a user how a Game Tree works and how the algorithms Minimax and Alpha-Beta Pruning are used for determining a move in a strategic game such as Gomoku. The specification outlined the three learning sections and the corresponding GUI and logic for building the application. The application was built using the experimental learning framework ELM and associated techniques from HCI and game design to develop the specifications.

The application was evaluated using the author's own perspective and experience from taking the target AI course and reflecting on how a game could facilitate the knowledge using gamification. As the application has not been tested on the intended target audience, it is not possible to determine beyond indications if it was effective in providing them with the learning objectives. It is also difficult to establish if a user is more included in gaining an interest in the field of strategic AI. To fully complete the application and verify its success, a complete prototype must be developed using the presented specifications. This prototype must be tested against a statistically significant group of people, such as 100 or more test subjects [22]. This author hopes that the project is further developed by others and tested to its completion.
Future work

There is much potential in improving and even expanding the proposed application's ability to secure the learning outcomes. There are three main types of future work that are worth highlighting: the implementation, the scope, and the facilitation of learning.

As this project did not implement the application but only created a partial prototype, it is worth recommending that future studies implement the specifications and assess its success using a test and control group of users. The application can be implemented in any programming language of choice. However, it is recommended to implement it using either JavaScript or C++. Implementing the application by JavaScript would be proper due to its popularity in creating web applications. As a web application, this teaching tool would be easier to share and test with a broader group of users. The main reason for implementing the application in C++ would be to aid in the AI course in question at Uppsala University. It could, in that case, both be used as a medium to teach the students about Minimax and Alpha-Beta Pruning and allow the students to see its direct implementation via the C++ code.

The project's scope only concerns itself with two of the principal methods within strategic game AI. Hence it would be an exciting study to implement other strategies and allow the student to compare the performance of the different algorithms. Furthermore, it would also be interesting to allow the user to explore different games rather than just Gomoku. For example, would the student find it more difficult or easier if they could compare how Minimax and Alpha-Beta Pruning worked in Gomoku, chess, and checkers? It could very well be that one game is better at teaching a student than another.

Finally, one last area to consider for future work is to assess the different methods in which the learning is facilitated. This application primarily aims to use the visual medium and physical response via typing, dragging, dropping, or clicking on an element. However, these are hardly the only ways to engage with a user, and it would be interesting to see if better learning outcomes were reached by engaging with other sensory systems such as sound.
References


Appendix A - Questions for Cognitive Walkthrough

Pre-Learning

• Is it clear how to play the game?
• Is it clear how to proceed or go backwards?
• Are the rules effectively communicated?
• Are you motivated to learn about the topics?

Game Tree

• Is the initial task clear?
• Did you complete the task successfully?
• Is the learning element clear?
• Will the user see how the tree is connected to the game?
• Will the user be able to complete the test given the information?
• What can be improved to help the user?
• What can be improved to motivate the user?

Minimax

• Is the initial task clear?
• Did you complete the task successfully?
• Is the learning element clear?
• How can learning element become more interactive?
• Will the user understand how the Minimax algorithm determine the AI’s choice?
• Will the user be able to complete the test given the information?
• What can be improved to help the user?
• What can be improved to motivate the user?

Alpha-Beta Pruning

• Is the initial task clear?
• Did you complete the task successfully?
• Is the learning element clear?
• How can learning element become more interactive?
• Will the user see why there are benefits to the Alpha-Beta Pruning Algorithm?
• Will the examples provide enough information for the user to understand the Alpha-Beta Pruning Algorithm?

• Will the user be able to complete the test given the information?
• What can be improved to help the user?
• What can be improved to motivate the user?

Post Learning
• Will the user be motivated to continue to play against the AI?
• What can be improved to aid the user in feeling a sense of accomplishment?