The game Konane, also known as Hawaiian checkers, is a two-player, zero-sum strategy board game ideally suited for this type of research. Players take turns making moves to capture an opponent’s piece. The game ends when a player does not have a move in which he can capture an opponent’s piece, and thus loses the game. In order to have a successful strategy, a player must consider many future possibilities. Our research focused on designing computer agents to play the game Konane using artificial intelligence techniques. Specifically, we aimed to create computer agents that utilized the minimax and minimax with alpha-beta pruning algorithms to select a move that would maximize the likelihood of winning the game. The outcome of this research will be an analysis of the effectiveness of each computing agent.

### The Game Konane

**Background:**
- Konane dates back to ancient Hawaii
- Traditionally shells or pebbles are used as game pieces and the board is made out of stone with indentions carved to show where the game pieces should go
- Board can be square or rectangular ranging from at least 6x6 to 16x18

**Gameplay:**
- Initially set up with game pieces filling every space in an alternating checkerboard pattern
- Black removes one of his own pieces from a corner space or a space in the very center of the board
- White then removes one of his own pieces adjacent to the now empty space
- The two players take turns making moves that involve an orthogonal jump to capture an opponent’s piece until a player cannot make a legal move and they lose the game

**Evaluation**

On an 8x8 game board, we evaluated a minimax agent and minimax with alpha-beta pruning agent both at a fixed depth of 5 against:
- each other
- a human agent (me)
- an agent that selects a random move for the list of potential moves
- an agent that selects a move based on the following criteria/strategy:
  - Avoid moving corner pieces
  - Make a move that puts the player in a position to make a future move
  - Move a piece that is in danger of being jumped
  - Avoid moving a piece that is not in danger of being jumped

## Results

<table>
<thead>
<tr>
<th></th>
<th>Minimax Agent</th>
<th>Alpha-beta Agent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wins against Human</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>Wins against Random</td>
<td>90%</td>
<td>91%</td>
</tr>
<tr>
<td>Wins against Strategy</td>
<td>89%</td>
<td>89%</td>
</tr>
<tr>
<td>Wins against other AI</td>
<td>~50%</td>
<td>~50%</td>
</tr>
</tbody>
</table>

**Evaluation**

On an 8x8 game board, we evaluated a minimax agent and minimax with alpha-beta pruning agent at various fixed depths.

### The Minimax Algorithm Cont.

Once the algorithm reaches a leaf node which indicates a terminal state, the utility value of the node is returned. This value is then backed up through the tree and compared to other potential moves’ utility values. Depending on which player’s turn it is, an internal node’s utility value becomes either the minimum or maximum utility value of all the child nodes’ utility values.

Alpha-beta pruning is an additional technique that can be used in combination with the minimax algorithm. This variant of the minimax algorithm should theoretically have the same statistical chance of winning as the basic minimax algorithm but be much more computationally efficient because it eliminates branches that cannot possibly influence the final decision. For example:

\[
\text{minimax-value (root)} = \begin{cases} 
\text{max (min(3, 12, 8), min(2, 7, 6))} & \text{if n is a terminal state} \\
\text{max(3, 2) if n is a Max node} \\
\text{min(3, 2) if n is a Min node} 
\end{cases}
\]

In other words, even if we let z be the minimum of x and y, the minimax decision for the node root is independent of the values of x and y so we can prune away these branches.

**Acknowledgements**

Foremost, I would like to express my sincere gratitude to my advisor Dr. Michael Scherger for his guidance, patience, and constant support throughout the duration of this project. I would also like to thank my thesis committee, Dr. Antonio Sanchez and Dr. Wendy Williams, for their encouragement, insightful comments, and hard questions as well as the rest of the Department of Computer Science for their support throughout my years in the Computer Science program.

\begin{itemize}
\end{itemize}