**Battleship AI with Heatmap**

**Prepared by: Jeremy Levine, Robert Mahoney, Blake Diekman, Aidan Stiles**

**Date: 05/03/2024**

**Table of Contents**

**1. Project Background and Description**

**2. Project Scope + Deliverables**

**3. Timeline/Schedule**

**4. Methodology and Implementation Details**

**5. Analysis of Results and Conclusions**

**6. References**

**Improved Battleship AI with Heat Map**

**1. Project Background and Description**

In battleship, two players place an assortment of ships on a grid space hidden from the other player. Each player takes turns blindly shooting at their opponent's grid, only knowing what grid spaces they have previously shot at, and whether the shot is a hit, miss, and if a ship is sunk. The first player to sink all of their opponents’ ships wins. The game has different strategies to both how a player places their ships on the grid and what grid square they choose to shoot on their opponent’s side.

Our initial question was if an AI could be implemented to use probability and statistics to provide the most effective placement and shooting techniques. We found a few implementations of this, but one that stuck out was an unoptimized version that leaned on statistical data to place and shoot effectively [1]. The author noted that the AI’s strategy could be improved with active probability analysis and a heat map would provide a visual depiction of the AI’s choices [1]. This kind of implementation would improve the AI’s performance, as well as provide a simple means of showing why the AI performs so effectively through “educated guesses”.

**2. Project Scope + Deliverables**

Our goal is to improve the AI’s shooting with probability-defined weighting of grid spaces. What this means is that after every shot, depending on whether it is a hit or a miss and what ships have already been sunk, the probability of the AI’s next shot to hit a ship will be calculated based on assigning a weight value to each grid space. Some of the issues that can arise with this methodology that were found from our research is that edge and corner spaces will be largely avoided unless they are given arbitrarily higher weighting to incentivize the AI to check them, as this is a common strategy that can affect the AI’s performance.

With this weighting of grid spaces for each shot comes the desire for an easy way to visually show the AI’s process. This will be possible through implementing a heat map. With a heat map, each grid space will be a certain color corresponding to their given weight following each shot that the AI performs. A heat map will show certain patterns to not only shooting, but also ship placement made by the AI’s opponent [2]. When a ship is hit, a very recognizable pattern can be seen with grid spaces weighed adjacent to the hit space, working through them until the ship is sunk.

The input for the AI will be the probabilities for each grid space. The AI agent will be partially observable, as it is only aware of what a grid space has after firing on it. It will follow a sequential style with each shot progressing the agent toward its goal, for all opponent ships to be sunk. Its environment will be static as the opponent cannot move their ships after placing them, as well as discrete as there are only so many shots the agent can make on the grid space. A probability-based random search algorithm approach will be used to determine grid space weighting and probabilities.

Here is the link to our GitHub repository for the project:

<https://github.com/AidanSt458/CS461-Team3.git>

**3. Timeline/Schedule**

Complete Heat Map: 4/26/2024

Complete Probability shooting: 4/28/2024

Complete Small Improvements: 4/29/24

Complete Project Presentation: 4/29/24

Complete Project Report: 5/3/24

**4. Methodology and Implementation Details**

To create the heat map, we will follow implementations in similar projects we found in our research. These will provide a basis for what we are looking for in our own heat map, while still giving us the challenge of developing and connecting our own to the existing and updated code. The same approach will be used for implementing a probabilistic-based random search approach to grid space weight and probability, though this will be defined more by the algorithm itself. Our implementation of a heat map and improved shooting probability will largely be within the starting JavaScript code file [1]. The AI’s starting shots will be updated from a predefined set of spaces based on statistics [1] to a truly random selection to support the AI’s autonomy. Any implementations that include outside resources or locations will be updated here.

**5. Analysis of Results and Conclusions**

Our implementation of the previously mentioned improvements was largely a success. We were able to add a heat map that will update following each shot done by the AI. It correctly and effectively displays grid spaces that have already been fired upon, whether they are a hit or a miss, and resultant probabilities of each space to be a hit. In adding the heat map, we updated the user interface of the project for a cleaner and simpler view. We also improved the AI’s initial starting shots, which were determined by a predefined set of grid spaces given random weights. Now, each grid space is assigned a random weight to truly randomize the AI on its first shot and make it dependent on probability for its following shots. We took the existing project and improved the visualization and autonomy of the probability-based AI. We explored not only how to see what the AI is doing and why, but how to improve its own decision making and allow it to be fully autonomous based on weighting.

**6. References: (use ACM/IEEE reference styles.** <https://www.acm.org/publications/authors/reference-formatting>)

[1] Mei, Bill. 2024. "battleboat: A Battleship Game Implemented in Python." GitHub Repository. Retrieved April 14, 2024, from <https://github.com/billmei/battleboat>.

[2] Schwartz, Aydin. 2024. "Coding an Intelligent Battleship Agent." Towards Data Science. Retrieved April 14, 2024, from <https://towardsdatascience.com/coding-an-intelligent-battleship-agent-bf0064a4b319>