Adaptive Battleship AI using CSP and Heuristic Search Strategies

Syed Farhan Syed Sathik Basha (CS23B2039)
Mohamed Amjad (CS23B2013)

April 8, 2025

Abstract

This project focuses on designing and implementing an intelligent Battleship game agent using Artificial Intelligence strategies. We employed Constraint Satisfaction Problems (CSP) to place ships and determine potential target zones combined with heuristics search strategies to optimize targeting decisions. The aim is to demonstrate intelligent decision-making in a familiar game setting.

1 Introduction

Battleship is a classic two-player strategy game where each player places ships on a hidden grid and tries to guess the locations of the opponent's ships by firing at grid coordinates. Our AI system enhances the game by integrating strategic reasoning, pattern recognition, and CSP principles for smarter gameplay.

2 Objective

The objective of this project is to develop a Battleship-playing AI that:

- Places ships validly and efficiently using CSP.
- Makes intelligent guesses using knowledge from previous hits/misses.
- Utilizes heuristics to efficiently target and sink ships.

3 Domain Study

The domain selected is the game of Battleship. The AI operates in an environment defined by:

- A 10x10 grid per player.
- Five ships of varying lengths Carrier (5 cells), Battleship (4 cells), Cruiser (3 cells), Submarine (3 cells), Destroyer (2 cells)
- Rules that prevent ships from overlapping .

Key AI Concepts Applied:

- **Constraint Satisfaction Problems (CSP)**: Used for placing the ships initially and also determining valid ship placements by pruning impossible configurations based on the grid.
- **Heuristics and Pattern Recognition**: Used to prioritize follow-up shots when hits are detected, favoring horizontal/vertical continuation.
- **Stateful Memory**: The AI maintains a knowledge base of all shots, hits, sunk ships, and continues targeting patterns based on this state.

4 System Design and Implementation

The project is implemented in Python. Major components include:

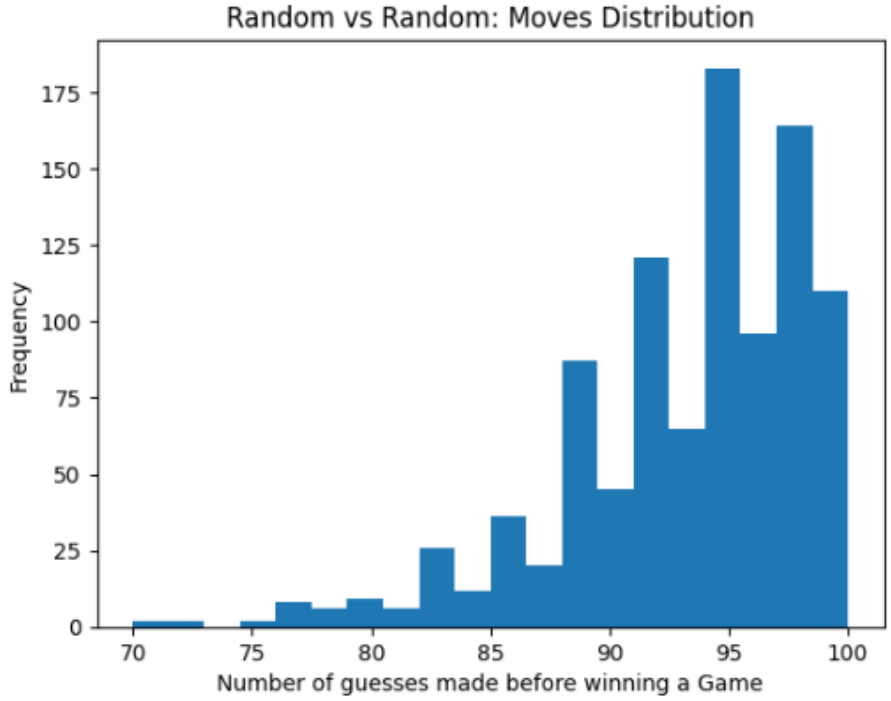
- **Ship Placement using CSP**: Each ship is placed on a 10x10 grid. Valid placements are computed dynamically such that no overlapping occurs and all ship cells remain within bounds. This is modeled as a CSP where:
  - **Variables:** Ships
  - **Domains:** Valid horizontal and vertical placements.
  - **Constraints:** No overlapping and within the boundaries of the grid.

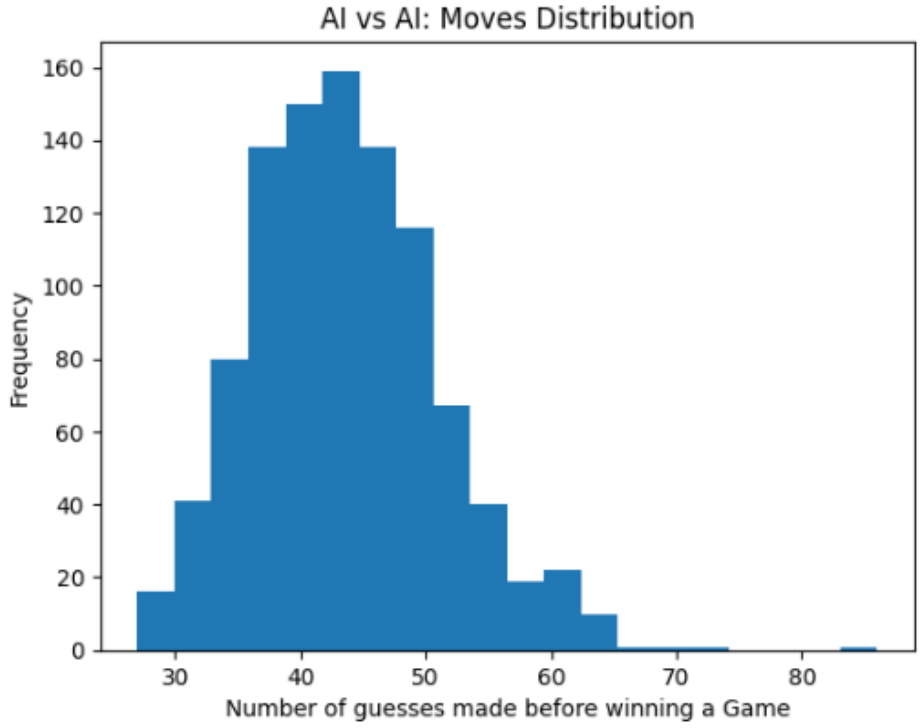
  The solver randomly selects from the set of valid placements.
- **Heuristic Targeting Strategy** : Once the game begins, the AI uses various strategies to decide the next shot:
  - **Adjacency Check:** After a hit, adjacent cells are evaluated for follow-up.
  - **Line Extension:** If multiple hits are aligned, the AI extends the line.
  - **CSP Targeting:** Incorporates remaining ship sizes and calculates valid placements.
  - **Heuristic Move:** If there are no hits, heuristic values are calculated as the number of possible placements of remaining ships for a particular cell, where CSP obtains possible placements.
- **Sink Confirmation**: A ship is only considered sunk if all expected positions based on the hit pattern are verified.
- **Interactive and AI Modes**: Supports manual or AI-driven play modes with visual display of full and discovered boards.

5 Results

5.1 Comparative Analysis of Guess Counts: Random Strategy vs. CSP and Heuristic-Based Approach

	Mode	P1 Wins	P2 Wins	Avg Moves	Median Moves
0	Random vs Random	543	457	93.018	94.0
1	AI vs AI	521	479	43.481	43.0





5.2 The AI demonstrates the ability to:

- Avoid repeated incorrect guesses.
- Target effectively around confirmed hits.
- Sink ships using adjacent cell traversal once a hit is detected.
- Maintain efficient and legal ship placement throughout.

6 Conclusion

This project successfully demonstrates how classical AI concepts can be applied to enhance gameplay in strategic environments such as as Battleship. By combining CSP for ship placement and adaptive targeting strategies, AI mimics human-like strategic thinking.