CSE318 Artificial Intelligence Sessional
Assignment 3: Adversarial Search

Chain Reaction AI Evaluation

2105066 – Sayaad Muzahid Masfi

June 16, 2025

Abstract

This report evaluates different heuristic strategies for an AI agent playing Chain Reaction, a two-player deterministic board game. We implement five distinct heuristic evaluation functions and test them in various configurations using alpha-beta pruning minimax search. The experiments compare performance across different search depths and heuristic combinations, with results showing that aggressive and cell control strategies generally outperform other approaches.

1 Introduction

Chain Reaction is a deterministic, turn-based, perfect information game for two players, where each aims to eliminate their opponent by triggering cascading chain reactions. Players place colored orbs on a 9×6 board; when a cell reaches its *critical mass* (determined by its number of orthogonal neighbors), it explodes, distributing orbs to adjacent cells and potentially triggering chain reactions.

Developing a competent AI for Chain Reaction involves:

- Crafting intelligent heuristic evaluation functions
- Implementing efficient adversarial search with alpha-beta pruning
- Designing a robust game engine with proper chain reaction resolution

2 Problem Definition

The project implements a complete Chain Reaction game with two main components:

2.1 Game Engine (Back End)

- Implements core game logic and rules
- Features alpha-beta pruning minimax search
- Includes five distinct heuristic evaluation functions

- Handles explosion resolution and game state transitions
- Communicates with UI via `gamestate.txt` file protocol
- Implemented in `backend_2105066.py`

2.2 User Interface (Front End)

- Interactive Pygame-based visualization
- Handles user input and displays game state
- Communicates with backend via file protocol
- Features animated menu system for configuration
- Implemented in `2105066.py`

3 Implementation Details

3.1 Technical Specifications

- **Language:** Python 3 with Pygame for UI
- **File Protocol:** Game board format with move headers
- **Board Dimensions:** 9 rows $\times$ 6 columns
- **Critical Mass:** 2 for corners, 3 for edges, 4 for center

3.2 Game Features

- Three game modes:
  - Human vs AI
  - AI vs AI (with configurable heuristics)
  - Random vs AI
- Five selectable heuristic strategies
- Adjustable AI search depth (1-6)
- Animated, interactive menu system
- Robust frontend/backend separation

4 Heuristic Evaluation Functions

Five distinct heuristic functions were implemented, each evaluating board states differently:

ID	Name	Evaluation Function
1	Simple	score = (player orbs−opponent orbs)+ $0.5 \times$ (player cells − opponent cells)
2	Cell Control	score = player cells − opponent cells
3	Edge Priority	score $= \sum_{\text{player cells}} \text{weight}(r,c) - \sum_{\text{opponent cells}} \text{weight}(r,c)$ where weight is 3 for corners, 2 for edges, 1 for center
4	Critical Mass	score $= \sum_{\text{player cells}} \frac{\text{orbs}}{\text{critical mass}+1} - \sum_{\text{opponent cells}} \frac{\text{orbs}}{\text{critical mass}+1}$
5	Aggressive	score = $2 \times$ player orbs $- 3 \times$ opponent orbs

Table 1: Heuristic Functions Summary

5 Experimental Setup

5.1 Configuration

- **Board Size:** 9×6 grid
- **Search Depths:** 1 through 6
- **Time Limits:** 10s per move (6000s for batch tests)
- **Platform:** Linux, Python 3.10, Pygame 2.1

5.2 Experiment Design

Three experiment types were conducted:

1. **Depth Analysis:** Random agent vs. Aggressive heuristic at depths 1-3
2. **Heuristic Comparison:** Random agent vs. all five heuristics at depth 4
3. **Head-to-Head:** All heuristic pairs competing at depth 3

6 Experimental Results

6.1 Random vs. Aggressive Heuristic (Varying Depth)

Depth	Red Player	Blue Player	Red Wins	Blue Wins	Avg. Time (s)
1	Random	Aggressive	0	5	26.60
1	Aggressive	Random	5	0	16.80
2	Random	Aggressive	0	5	32.40
2	Aggressive	Random	5	0	29.00
3	Random	Aggressive	0	5	132.00
3	Aggressive	Random	5	0	119.60

Table 2: Performance vs. Search Depth

6.2 Random vs. All Heuristics (Depth 4)

Opponent	Red Player	Blue Player	Red Wins	Blue Wins	Avg. Time (s)
Simple	Random	Simple	0	5	162.00
Simple	Simple	Random	5	0	156.60
Cell Control	Random	Cell Control	0	5	184.40
Cell Control	Cell Control	Random	5	0	188.60
Edge Priority	Random	Edge Priority	0	5	232.60
Edge Priority	Edge Priority	Random	5	0	193.80
Critical Mass	Random	Critical Mass	0	5	198.80
Critical Mass	Critical Mass	Random	5	0	224.00
Aggressive	Random	Aggressive	0	5	212.00
Aggressive	Aggressive	Random	5	0	219.60

Table 3: Heuristic Performance Comparison

6.3 Heuristic vs. Heuristic (Depth 3)

Matchup	Red Player	Blue Player	Red Win	Blue Win	Time(s)
Simple vs Cell Control	Simple	Cell Control	0	1	49.00
Cell Control vs Simple	Cell Control	Simple	1	0	54.00
Simple vs Edge Priority	Simple	Edge Priority	0	1	64.80
Edge Priority vs Simple	Edge Priority	Simple	1	0	47.00
Simple vs Critical Mass	Simple	Critical Mass	0	1	98.80
Critical Mass vs Simple	Critical Mass	Simple	1	0	121.40
Simple vs Aggressive	Simple	Aggressive	0	1	113.00
Aggressive vs Simple	Aggressive	Simple	0	1	89.00
Cell Control vs Edge Priority	Cell Control	Edge Priority	0	1	57.20
Edge Priority vs Cell Control	Edge Priority	Cell Control	1	0	106.80
Cell Control vs Critical Mass	Cell Control	Critical Mass	0	1	144.20
Critical Mass vs Cell Control	Critical Mass	Cell Control	1	0	93.40
Cell Control vs Aggressive	Cell Control	Aggressive	0	1	105.20
Aggressive vs Cell Control	Aggressive	Cell Control	1	0	115.80
Edge Priority vs Critical Mass	Edge Priority	Critical Mass	0	1	121.20
Critical Mass vs Edge Priority	Critical Mass	Edge Priority	0	1	123.00
Edge Priority vs Aggressive	Edge Priority	Aggressive	0	1	88.60
Aggressive vs Edge Priority	Aggressive	Edge Priority	1	0	55.20
Critical Mass vs Aggressive	Critical Mass	Aggressive	0	1	139.80
Aggressive vs Critical Mass	Aggressive	Critical Mass	1	0	124.80

Table 4: Head-to-Head Heuristic Performance

7 Analysis

7.1 Key Findings

- **Depth Impact:** Higher search depths consistently improve AI performance, though with increased computation time
- **Heuristic Effectiveness:** Aggressive and Edge priority heuristics showed strongest overall performance
- **Strategic Differences:** Edge Priority excels early game, while Critical Mass performs better in mid-late game

7.2 Performance Trade-offs

- **Simple Heuristic:** Fastest computation but weakest against specialized strategies
- **Aggressive Heuristic:** Best win rate but longest computation time
- **Cell Control:** Good balance between performance and speed

8 Conclusion

The project successfully implemented and evaluated multiple heuristic strategies for Chain Reaction AI:

- Developed five distinct heuristic evaluation functions
- Implemented robust game engine with alpha-beta pruning
- Created interactive visualization and testing framework
- Demonstrated that Aggressive and Edge Priority heuristics perform best
- Showed clear trade-offs between search depth and computation time

Future work could explore:

- Hybrid heuristic approaches
- Optimizations for faster search
- Machine learning approaches to heuristic development

Sayaad Muzahid Masfi
Roll: 2105066