Capstone Project Specification: Strategic Decision-Making via Minimax Search

ESAP: Engineering Summer Academy at Penn

July 2025
Instructor: Joel Ramírez, Harry Smith

Overview

This capstone project constitutes the culmination of your work in algorithmic thinking and game theory. Working in groups of 2–4 students, your task is to design a competitive agent that plays a deterministic two-player game using the **minimax algorithm**. You will evaluate its performance through empirical simulations and analyze the resulting data using modern tools for computational experimentation and visualization.

The final deliverables include a formal written report documenting your implementation, simulations, and your code.

Timeline

- **Tuesday:** Introduction to minimax, recursive tree search, and adversarial logic
- **Wednesday–Thursday:** System design, simulation, data collection, and analysis
- **Friday:** Submission of the final written report

Project Objectives

Students will demonstrate the following skills:

- Implement recursive minimax-based search for strategic games
- Simulate large batches of games against multiple types of agents
- Collect and analyze empirical results across repeated trials
- Visualize outcome trends using Python libraries such as `matplotlib` or `seaborn`
- Synthesize findings into a structured technical report

Approved Game Options

Students may select one of the following adversarial games to analyze:

- **Tic Tac Toe** (3x3 grid)
- **Halving Game**
- **Connect Four** (small grid variant: 4x4 or 5x5)
- **Nim** (any consistent version with multiple heaps)

Custom games are permitted with prior approval by the instructor. The selected game must support discrete, deterministic moves and full information at all times. If you want to try something more difficult, try checkers.

Getting Started

Begin by implementing your chosen game engine. Once the rules and win conditions are fully functional, integrate the minimax-based agent. Test your implementation by playing it against a random player before scaling to full simulations.

Technical Requirements

All implementations must satisfy the following criteria:

1. A working game engine capable of managing states, enforcing rules, and detecting termination conditions.
2. An agent based on the minimax algorithm (optionally enhanced with depth limits or pruning, you're on your own here).
3. A set of automated simulations, comprising at least 100 matches between different agent types (e.g., random vs minimax). Are there any patterns you notice?
4. Persistent storage of simulation results, including win rates, game lengths, and other relevant metrics.
5. At least two clearly labeled visualizations generated using Python (e.g., bar plots, histograms, line charts). What you find to be the most interesting.
6. A formal report, authored using google docs or your editor of choice containing methodology, analysis, and critical reflection.

Suggested Experimental Questions

Your team should design experiments that yield meaningful insights about decision-making in games. Some guiding questions include:

- How does minimax perform relative to random or other agents you might makes?
- What is the average number of moves per game under different agent configurations?
- How does the depth of the minimax tree affect strategic quality or run-time efficiency? (i.e. how does the depth of the tree affect speed?)
- How does the initial state of a game affect win rate? Is there a state that always garuntees a win? Which are they?
- Are there situations where a deliberately imperfect (non-optimal) agent can outperform a perfect one due to unpredictability? Why?
- How stable are your results?
- The questions are endless...

Report Guidelines

The formal report must be submitted as a PDF, and should be 3-5 pages in length. Your report must include the following sections:

1. Introduction

- Describe the selected game and briefly explain its rules.
- Justify why the game is well-suited to analysis and why it works using a minimax policy. Make sure to formalize the game. And introduce Minimax meaning to explain it as best as you can in the report.

2. Strategy and Implementation

- Provide a high-level summary of your game engine. What are the states, how are the represented, etc.
- Explain your recursive minimax approach, including any enhancements you made if at all.

3. Simulation Design

- Specify the number of simulations conducted and the types of agents tested.
- Describe how data were recorded, stored, and processed.

4. Data Visualization

- Present at least two figures summarizing key outcomes (win rates, average length, distribution of results). If you want to convert lists to pandas dataframes, that's totally possible.
- Figures must include captions and be referenced in your text.

5. Analysis and Interpretation

- Explain any trends observed in the data.
- Discuss the strengths and weaknesses of the minimax agent if any.
- Consider both strategic effectiveness and computational efficiency.

6. Reflection

- Identify technical or conceptual challenges faced by your group.
- Discuss what you would improve given more time.
- Reflect on what you learned about adversarial reasoning and computational design. It's been a long three weeks.

Grading Guidelines

Grading for this capstone will emphasize completion, creativity, and curiosity. You'll be rewarded for pushing your thinking and engaging deeply with the project — not for perfection.

Baseline Grading:

- To be determined.

Other things we'll be looking at:

- Clear and correct implementation of the minimax agent
- Quality of experimentation and visualizations
- Creativity or ambition in design
- Clarity, effort, and honesty in your write-up

Optional Extensions

Teams are encouraged to consider creative extensions, including:

- A graphical interface using PennDraw:
- Parameterizable agents with adjustable difficulty levels. (any way you can introduce probability when choosing an action?)

Submission Instructions

Submit the following items by noon on Friday.

- A compiled PDF of your final report
- Python files for your implementation and simulation scripts
- Please name your files using the format `groupname_projectname.pdf` and `groupname_code.zip`

All submissions should be uploaded to canvas! Thanks everyone.