Project Report

Project Title: Sentinel Chess: An 8x8 Chess Variant with the Sentinel Pawn

Submitted By: Maaz Kashif [22K-4518]
Abdul Rafay [22K-4192]
Umer Ahmed [22K-4213]

Course: AI
Instructor: Mr. Abdullah Yaqoob
Submission Date: 11/05/2025

---

1. Executive Summary

- **Project Overview:**
  This project introduces "Sentinel Chess," a variant of the classic 8x8 chess game that incorporates a new piece called the Sentinel Pawn. The objective is to develop an AI that can play the modified game using the Minimax algorithm enhanced with Alpha-Beta Pruning. This AI evaluates not just traditional chess strategies but also the unique offensive and defensive mechanics of the Sentinel Pawn. The project explores the impact of this new piece on strategic gameplay, adapting heuristics and evaluation functions to accommodate its shielding abilities and attack range.

2. Introduction

- **Background:**
  Chess is a widely known strategy game that has evolved over centuries. It features a fixed set of pieces with well-defined movements and objectives. We chose to innovate upon this classic game by introducing a new piece “**the Sentinel Pawn**” that alters conventional tactics. The addition allows for a richer strategic space, as it both protects adjacent allies and attacks enemies in all directions within a 1-square radius.

- **Objectives of the Project:**
  - Develop a playable version of the Sentinel Chess game.
  - Implement an AI agent that can compete using modified Minimax and Alpha-Beta pruning.
  - Analyze how the Sentinel Pawn influences game strategies.
  - Test the AI's performance against human and simulated opponents.

3. Game Description

- **Original Game Rules:**
  Traditional chess is played on an 8x8 board with two players. Each player controls 16 pieces: pawns, knights, bishops, rooks, a queen, and a king. The goal is to checkmate the opponent's king. Standard rules include pawn promotion, castling, and en passant.

- **Innovations and Modifications:**
  - The Sentinel Pawn:
    - Moves like a traditional pawn (one square forward).
    - Attacks all eight surrounding squares (one-square radius).
    - Shields all adjacent friendly pieces from being captured unless the Sentinel Pawn is taken first.
    - Cannot be captured by regular pawns.
    - Promotes to a Guard Queen, a powerful variant of the queen, upon reaching the final rank.
  - Standard rules are adjusted to accommodate these changes while preserving the overall objective of checkmate.

4. AI Approach and Methodology

- **AI Techniques Used:**
  - **Minimax Algorithm:**
    - Extended to evaluate Sentinel-specific tactical and positional considerations.
    - Version 4 of the Minimax algorithm integrates Transposition Tables and Zobrist Hashing to cache and retrieve scores of previously evaluated board positions, drastically reducing redundant computation.
  - **Alpha-Beta Pruning:** Used to reduce the computation tree and improve decision speed.
  - **(Optional)** Reinforcement Learning logic was planned for future expansion via self-play.

- **Algorithm and Heuristic Design:**
    - Custom evaluation function includes:
        - Material balance (piece values reweighted with higher value for Sentinel Pawns).
        - Positional control: extra weight given to zones protected by Sentinel Pawns.
        - Board threat analysis: proximity to king and ability to block check paths.
        - Promotion potential of Sentinel Pawns to Guard Queens.
        - Each board position is encoded using a Zobrist key. When evaluating positions, the AI checks this key in the transposition table to avoid reevaluating previously seen states.

- **AI Performance Evaluation:**
    - Performance was measured through:
        - Win ratio against random and human players.
        - Time taken per move.
        - Number of positions evaluated.
        - Qualitative gameplay analysis (strategic depth, defense quality, offensive impact).

5. Game Mechanics and Rules

- **Modified Game Rules:**
    - Sentinel Pawn can move one square forward and attack in a 1-square radius in all directions.
    - Adjacent pieces are shielded from capture unless the Sentinel is captured first.
    - Sentinel Pawn promotes to a Guard Queen on the last rank.
    - Cannot be captured by regular pawns.

- **Turn-based Mechanics:**
    - Players take alternate turns as in traditional chess.
    - Moves are validated against custom rule logic for Sentinel Pawns.
    - The AI uses Minimax to evaluate best moves each turn.

- **Winning Conditions:**
    - Standard checkmate rules apply.
    - Stalemate and draw conditions are also preserved.

6. Implementation and Development

- **Development Process:**
    - The game was implemented using Python.
    - Pygame was used for rendering the graphical board and animations.
    - The chess engine logic was customized to support new piece mechanics.
    - Minimax with Alpha-Beta pruning was coded from scratch to support depth-based evaluation.
    - Sentinel Pawn logic was integrated into move generation and validation systems.

- **Programming Languages and Tools:**
    - **Programming Language:** Python
    - **Libraries:**
        - Pygame – for GUI and game loop
        - NumPy – for board state representation
        - Collections (deque) – for search algorithms
    - **Tools:**
        - GitHub for version control
        - YAML configuration was used to define adjustable parameters for the AI engine, game settings, and UI design.

- **Challenges Encountered:**
    - Integrating Sentinel Pawn mechanics with existing move logic.
    - Balancing evaluation heuristics to consider shielding effects.
    - Ensuring fair play and avoiding infinite loops in AI decisions.
    - Debugging edge cases like Sentinel Pawn promotion and shielding logic.

7. Team Contributions

- **Team Members and Responsibilities:**
    - ○ **Maaz Kashif [22K-4518]:**
      Implemented the Minimax and Alpha-Beta Pruning algorithms, including heuristic design and state evaluation logic.
    - ○ **Abdul Rafay [22K-4192]:**
      Designed the modified chess rules, implemented Sentinel Pawn mechanics, and balanced the game logic for fair gameplay.
    - ○ **Umer Ahmed [22K-4213]:**
      Developed the GUI using Pygame, handled user interactions, and integrated the AI into the gameplay loop.

8. Results and Discussion

AI Performance:

- ○ Against human players (on Easy mode): ~65% win rate
- ○ Against random move generator: 95%-win rate
- ○ Average move decision time: ~1.8 seconds with Alpha-Beta pruning enabled

- ○ Qualitatively, the AI:
    - ○ Effectively used Sentinel Pawn for shielding key pieces.
    - ○ Prioritized promotion paths and protected zones well.
    - ○ Occasionally overvalued shield zones, which could be improved in future iterations.

9. References

- Standard Chess Rules and Strategy Guides
- Pygame Documentation – https://www.pygame.org/docs/
- Python collections module – https://docs.python.org/3/library/collections.html
- BFS and Grid Pathfinding Techniques – Online AI Research Articles
- AI and Game Theory Lecture Notes – [CS188 Berkeley & Other Online Courses]