

# Optimization in Pawnless Chess Endgame Analysis

---

Saravana Bhavanandam, Jeeva

12/09/24



College of Natural Science  
MICHIGAN STATE UNIVERSITY

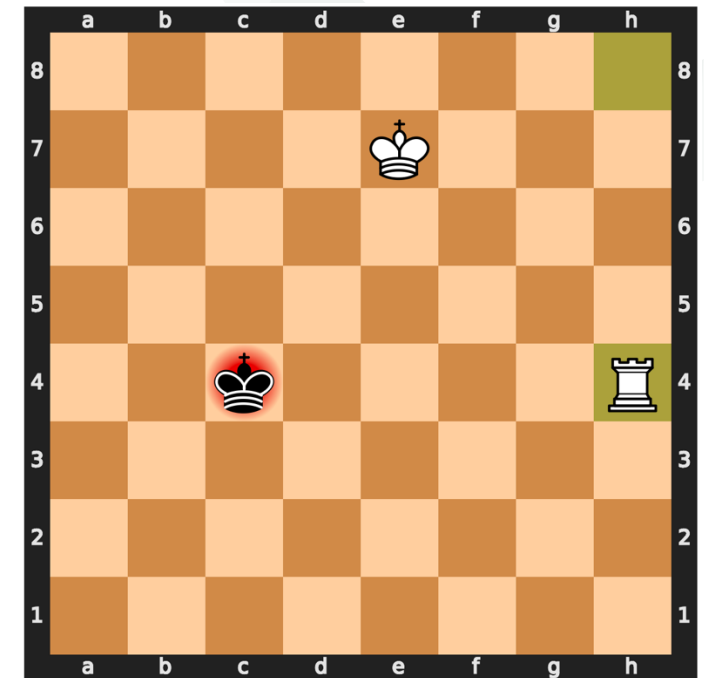
# Background & Motivation

Chess endgames without pawns highlight the blend of art and science, focusing on tactical precision since typical pawn strategies don't apply. These endgames center on key concepts like opposition and Zugzwang, demanding exact moves.

## Project Objectives:

- Develop Models:** Create computational models for pawnless chess endgames.
- Optimize Moves:** Use algorithms to find optimal moves.
- Test Algorithms:** Evaluate performance in key endgame scenarios.

This project aims to refine decision-making in these simplified chess situations using computational optimization for best-move strategies.



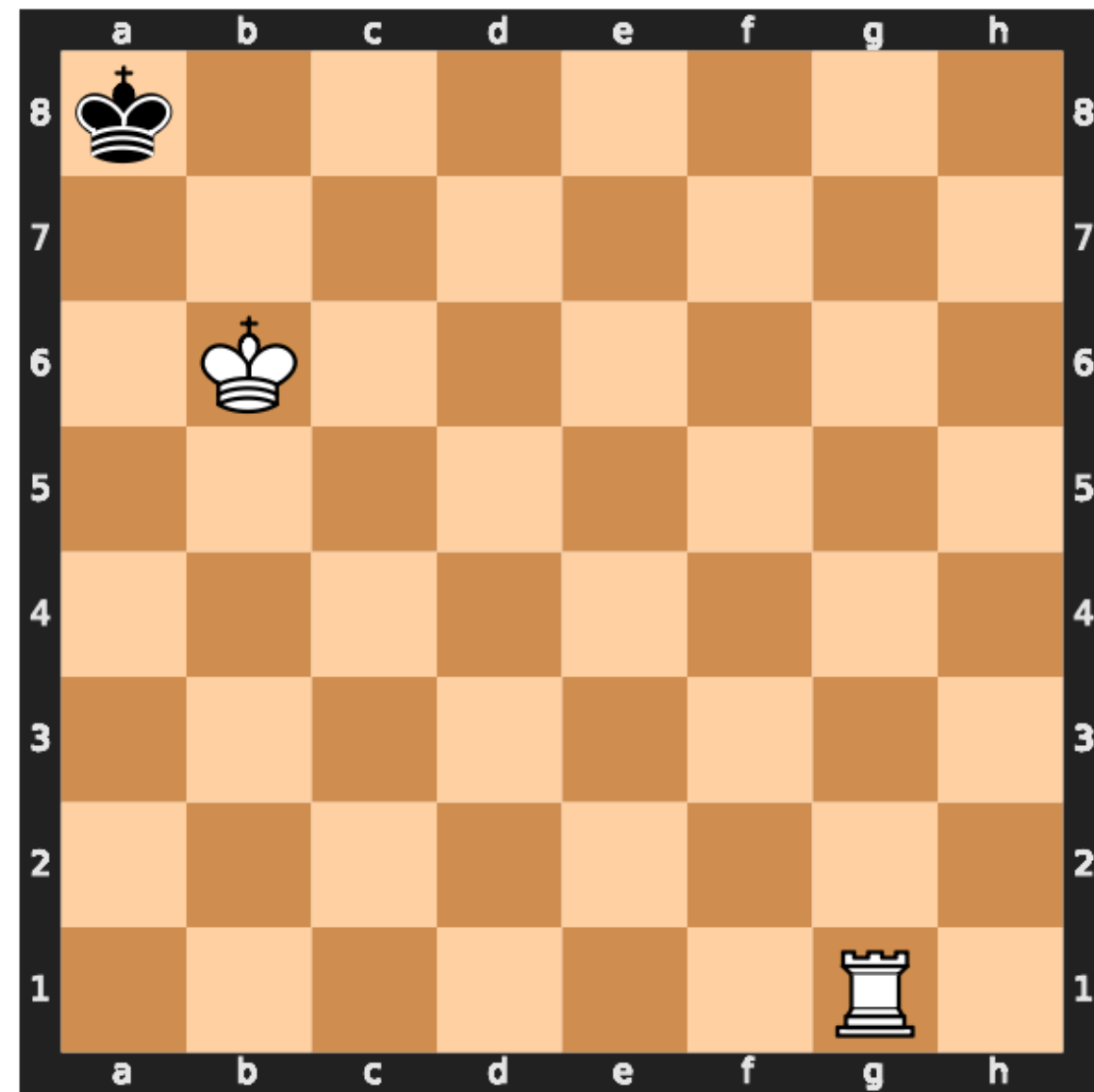
# Endgame Modeling

## Assumptions for Pawnless Endgames:

- Focus on classic scenarios adhering to official chess rules.
- No Pawns Start
- Finite Analysis Depth
- Simplified Metrics
- Ideal Play Scenarios

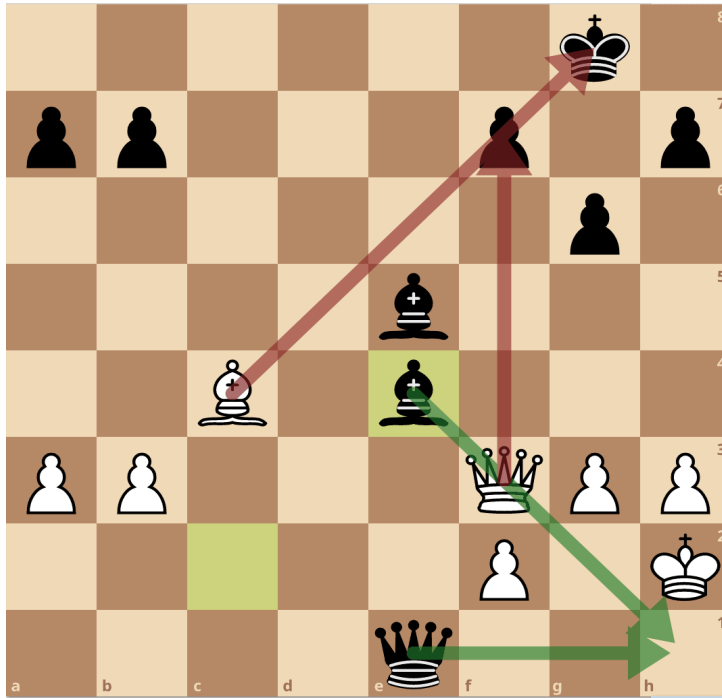
I will set up the board for famous pawnless endgames using the python-chess library. Here are three well-known scenarios:

- **King and Rook vs. King (KR vs. K)**
- **King and Two Bishops vs. King (KBB vs. K)**
- **King and Knight vs. King (KN vs. K)**



*Chess board setup of King Knight vs King*

# Designing the Objective Function



Components of the Objective Function:

- **Material Value:** Scores based on piece type and count.
- **King Safety and Opposition:** Assesses king security and strategic positioning.
- **Zugzwang Detection:** Identifies when a move worsens an opponent's position.
- **Terminal States:** Evaluates major outcomes like checkmate or stalemate.

The **Objective function**  $f(x)$  :

$$f(x) = w_1 \times g(x) + w_2 \times k(x) + w_3 \times z(x) + \text{Terminal State Bonus}$$

Where,

$w_1, w_2$  and  $w_3$  are the weights for each component,

$g(x)$  is function of material value,

$k(x)$  is function of King Safety and Opposition,

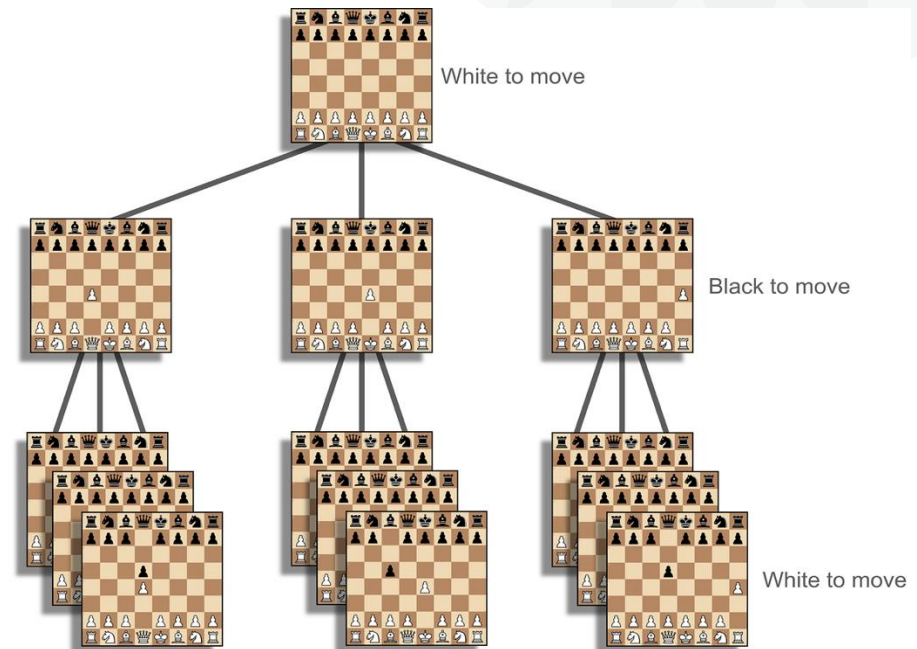
$z(x)$  is Zugzwang Detection

# Optimization Algorithms

## Implementing Optimization Algorithms

### 1. Minimax with Alpha-Beta Pruning

- **Approach:** Depth-first adversarial search that evaluates future game states to decide optimal moves.
- **Alpha-Beta Pruning:** Prunes branches that cannot influence the final decision, significantly reducing computation time.
- **Key Benefits:**
  - Explores fewer nodes than standard Minimax.
  - Ensures optimal play when both sides play perfectly.
  - Suitable for deep analysis in well-defined game trees.



# Optimization Algorithms

## 2. Monte Carlo Tree Search (MCTS)

• **Approach:** Combines random sampling and iterative search to build a decision tree.

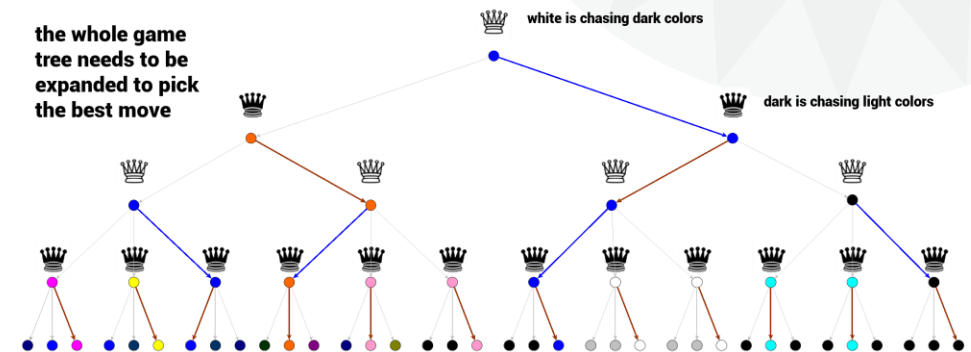
• **Balance:**

- **Exploration:** Searches uncharted game states.
- **Exploitation:** Focuses on previously successful moves.

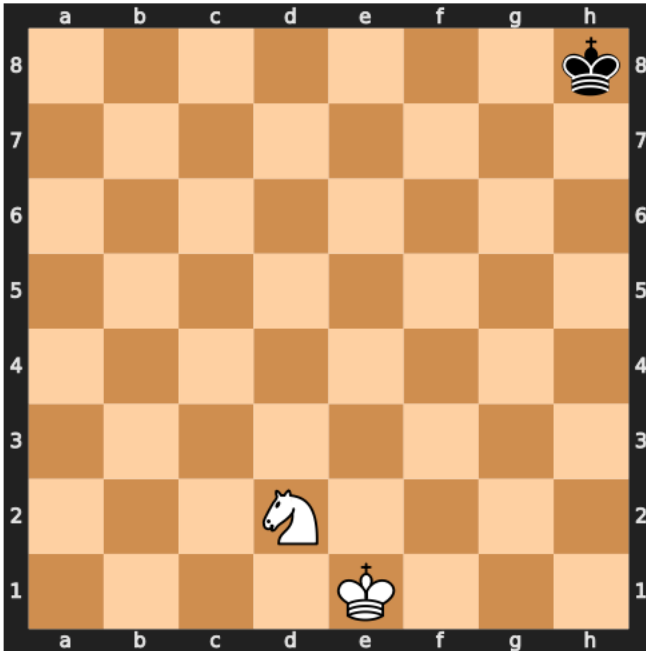
• **Key Benefits:**

- Adaptable to dynamic and complex decision spaces.
- Learns from simulations to refine move choices.
- Effective in situations with uncertain outcomes.

Both algorithms leverage their strengths to navigate the complexity of chess endgames, offering different trade-offs between computational depth and adaptability.

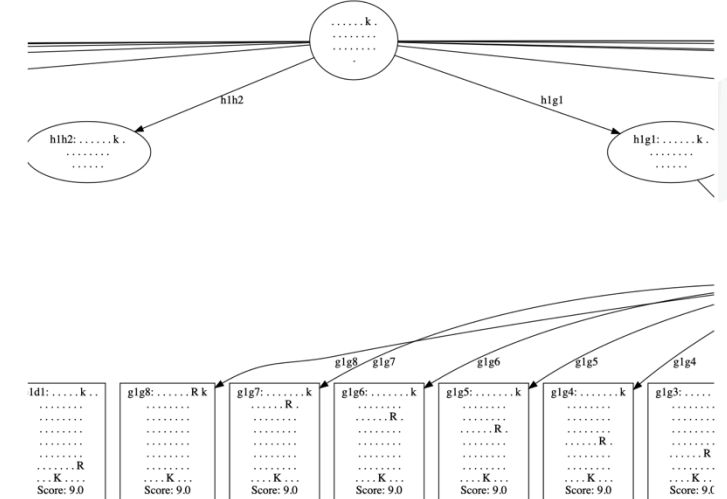


# Results - Minimax



.....k  
 .....  
 .....  
 .....  
 .....  
 .....  
 .....  
 .....  
 ....N....  
 ....K...  
 Score: -97.0

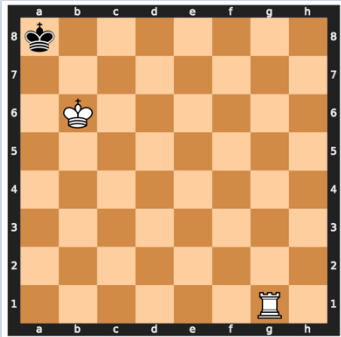
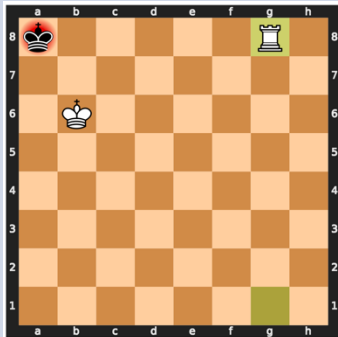
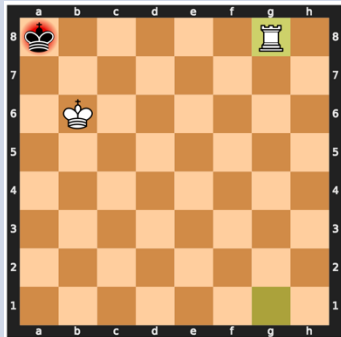
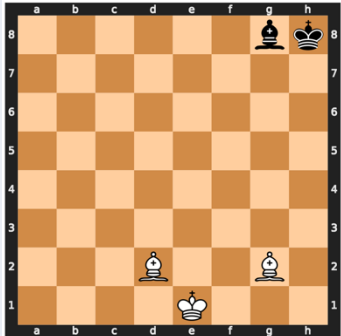
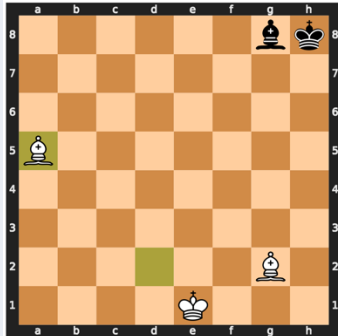
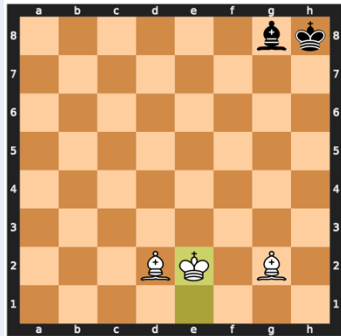
*Short overview of Minimax decision tree for KRvK Game setup*



In chess, the endgame scenario of a knight (KN) versus a king (K) is theoretically a draw when both players perform optimally. I successfully implemented the Minimax algorithm with alpha-beta pruning, which confirmed this outcome across multiple different trials.



# Results MCTS and Benchmarking with Stockfish

Endgame Board Position	Initial Board Setup	MCTS Best Move	Stockfish Best Move
KN vs K Checkmate			
KBB vs KB			



# Overall Results

Endgame Board Position	Minimax Result	MCTS Optimized Best Move	Best Move from Stockfish	Optimality Score
KR vs K	9.75	e1f2	h1f1	0.4
KBB vs K	10.75	d2a5	e1e2	0
KN vs K	-97.0	d2e4	d2e4	0.4
KN vs K Checkmate	9.75	g1g8	g1g8	1.0
KRBB vs KBN	12.0	b5b7	b5b7	1.0
KQ vs KR	7.0	e1e2	d1h5	0.9
KR vs KN	6.0	e1d2	e1e2	0.6
KQ vs KQ	4.0	d1d8	d1h5	0
KBB vs KNN	3.0	e1e2	g2d5	0
KRN vs KBN	5.0	h1g1	h1f1	0.2
KQ vs KBN	6.0	d1a1	d1d4	1.0
KQ vs KB	12.0	d1c1	e1d2	0.9
KN vs KR	-2.0	e1d1	d2b3	0.8
KBB vs KB	7.0	d2a5	e1e2	0



# Discussion and Conclusion



## Key Insights:

- **Minimax with Alpha-Beta Pruning** excels in deterministic scenarios, efficiently predicting outcomes.
- **MCTS** is effective in complex environments, favoring exploration.
- **Stockfish** sets a high benchmark, guiding the evaluation of other algorithms.

## Conclusions:

- No algorithm universally outperforms others; effectiveness depends on the specific chess endgame context.
- **Strategy Selection:** Choosing the right algorithm is crucial, based on the game's complexity and strategic demands.
- **Future Directions:** Potential for hybrid algorithms that blend Minimax's depth and MCTS's breadth, improving adaptability and performance in dynamic situations.
- **Q-Learning Implementation:** Could be particularly transformative in long-term strategy games where learning from past outcomes to optimize future decisions is crucial.

**Application:** Insights applicable to strategic decision-making beyond chess, highlighting broader algorithmic uses.



College of Natural Science  
MICHIGAN STATE UNIVERSITY