Chess Endgames Solver

A Reinforcement Learning Approach

Valeria De Stasio, Christian Faccio, Giovanni Lucarelli

August 25, 2025

Project Overview

* goal of the Project
* used algorithms
* metrics overview ?

Chess Programming Background

Brief History of Chess Programming

Endgame Phase

* what is an Endgame

* usual approach to solve it in modern chess programming (syzygy and tb)

Simple Endgame: (our) Problem Definition

Simple Endgames that we want to solve:

* winning for white

* kings and one (or more) heavy piece for white

* the number of states is already huge

Rules that we use:

* piece movement

* no 3 repetition

Assessment Metrics Overview

* DTM, DTZ etc * success, top1 etc

MDP Formulation

MDP: States

* States $\mathcal{S}$: all legal endgame positions, augmented with side-to-move (no turn number, no 50 moves rule)

* Terminal State $\bar{\mathcal{S}}$: checkmate or draw (insufficient material), once reached the game ends.

MDP: Actions

* Actions: legal moves for the current player.

MDP: Transition

* Transition function: deterministic update given current state and chosen action, followed by the opponent's (black) deterministic reply

Instead of storing $\mathcal{P}$, chess engines implement functions that *define* $\mathcal{P}$ procedurally:

* $legal_moves(s)$ — generates valid actions in state s

* $apply_move(s, a)$ — returns the next state s'

* $is_terminal(s')$ — checks if game is over

MDP: Reward

* Rewards (Chess): +1 for win, -1 for loss, 0 otherwise.

Rewards (simple endgames): -2 per ply, -1000 for draw

RL Algorithms

Results

Policy interpretability through human chess principles

Video Animation of optimal ply vs our policy

maybe for a mate in 5 or more

Thank You!