Chessmaster

A new way to play chess

Table of Content

Motivation

State of the Art

Algorithm

Procedure

Motivation

Main Goal

- Create an powerful AI, that is capable to beat the best chessplayers in the world
- Provide a platform to train against a strong opponent
- Give the „real“ chessplayers the opportunity to step up their gametactics

State of the Art

IBM - Deep Blue

- First real chess engine to beat a reign world champion
- Developed by IBM in 1989
- It implemented the alpha beta search on very large-scale integration but in a brut-force method
  - Developers even denied it being an AI

Stockfish

- Open source UCI chess engine
  - UCI = Universal Chess interface
- As of 2018 it has become the world strongest chess entity not relying on AI
- It uses Alpha-Beta-Search and Bitboards
  - Alpha-Beta-Search is a optimized version of the MiniMax Algorithm
  - A bitboard is a specialized bit array data structure where each bit corresponds to a game board space or piece.

DeepMind - AlphaZero

- Uses a Deep Neural Network in combination with a reinforcement learning algorithm
  - Deep Neural Networks are Neural Networks with two or more hidden layers
- The trained network is used to guide the Monte-Carlo Tree Search Algorithm to select the most promising moves
- In the beginning the algorithm does not know anything about the game besides the basic rules
  - After 4 hours of training it was able to beat Stockfish
- Besides Chess AlphaZero also learned Shogi and Go

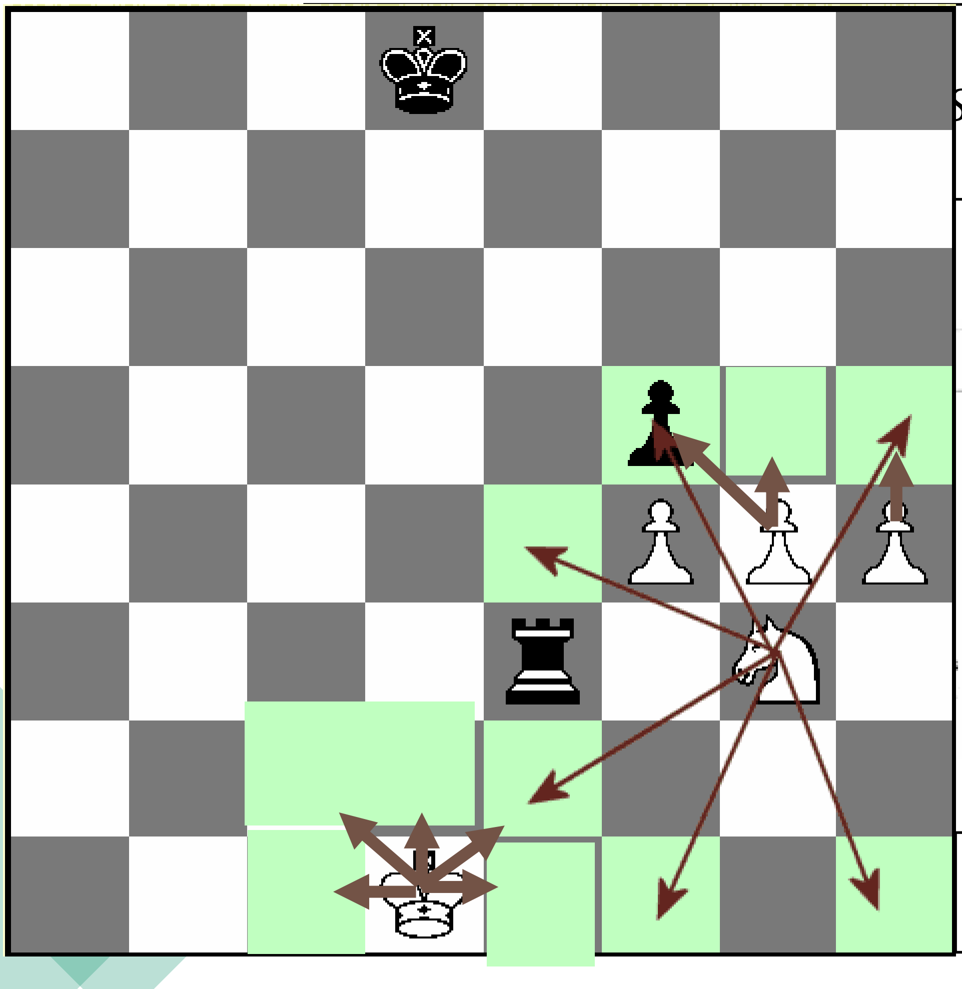
Algortihm explained

Monte Carlo Tree Search

- Statistical Algorithm
- Described in 2006 by Rémi Coulom
- Used by various AIs, like AlphaZero or MuZero (by Deepmind)
- Improved Efficiency above (true) Tree Search (Brute Force)
- Prevents a large Exploration to Exploitation Trade off!

- Used to find optimal moves

Monte Carlo Tree Search



$$S_i) = \bar{v} + c * \sqrt{\frac{\ln(N)}{n_i}}$$

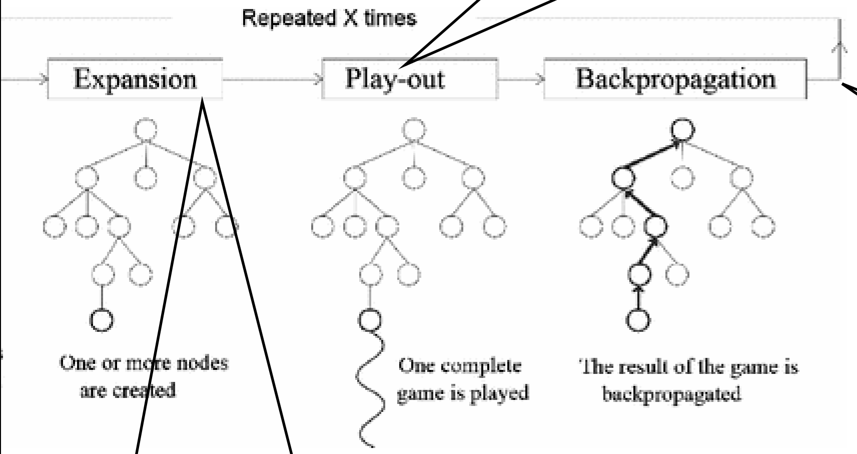
$\bar{v}$ = average value of node $\left(\frac{v}{ni}\right)$

$c = Exploration\ parameter\ (static)$

$N = times\ visited\ parent\ node$

$n_i = times\ visited\ i - th\ node$

Also called Rollout


Repeated X times
Expansion
Play-out
Backpropagation
One or more nodes are created
One complete game is played
The result of the game is backpropagated

One new node per action (here 14)

After X times:
Return moves with probabilities

V-Network | Temporal Difference Learning

- V = Value
- V-Network
  - Neural Network (Chess agent)
  - Should output a policy π (to win the game)
- Temporal Difference Learning
  - Model-Free Reinforcement Learning Algorithm
    - Learn from experience by incomplete episodes
  - $V(S_t) = V(S_t) + \alpha * (G_t - V(S_t))$
    - α = Learning Rate ∈ [0,1]
    - $G_t = \text{Estimated return}$
  - Simplest TD-Algorithm (TD(0))
    - $G_t = R_{t+1} + \gamma * V(S_{t+1})$
    - $V(S_t) = V(S_t) + \alpha * (R_{t+1} + \gamma * V(S_{t+1}) - V(S_t))$
      - R_t = Reward
      - γ = discount factor ∈ [0,1]

$R_{t+1} + \gamma * V(S_{t+1})$ = TD Target

$R_{t+1} + \gamma * V(S_{t+1}) - V(S_t)$ = TD error

Procedure

Next Steps

- Comparison of the evaluation possibilities of the algorithm
- Human versus computer (Graphical user interface)
- Summarize scientific papers
- Preparation of the elaboration

Evaluation of Algortihm

- Elo rating system (Comparison of two chess players)

- Evaluation of the chess game
  - Evaluation of Position
  - Material on the board
  - Pieces Activity

Any Questions?

Thank you for your Attention