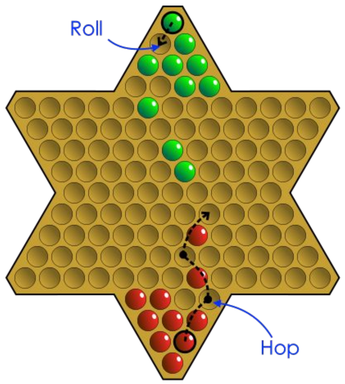
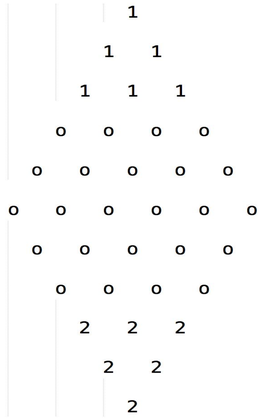
Playing Chinese Checkers with Reinforcement Learning

Sijun He, Wenjie Hu, Hao Yin

Abstract

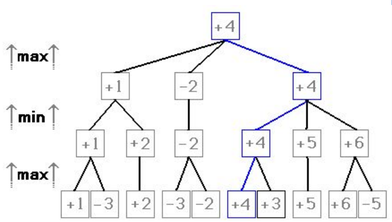
We built an AI for Chinese Checkers using Reinforcement Learning. The value of each board state is determined via minimaxation of a tree of depth k, while the value of each leaf is approximated by weights and features extracted from the board. Weights are tuned via value approximation. The performance of our modified minimax strategy with tuned weights stands out among all the other strategies.





Methodology

The AI implementation is a shallow depth-k minimax game tree. The "value" of each leaf board state is approximated by a linear evaluation function based on the features of pieces positions:



A_i: total distance to the destination corner of player i

B_i: total distance to the vertical central line of player i

C_i: Sum of vertical advances for all pieces of player i

$$\widetilde{V} = w_1(A_2 - A_1) + w_2(B_2 - B_1) + w_3(C_1 - C_2)$$

Challenges & Solutions

Weights Tuning

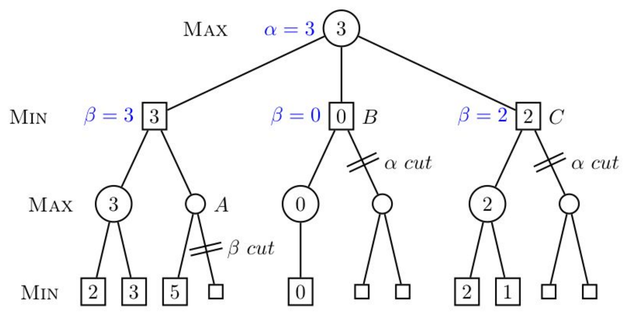
- Idea: to make the raw value consistent with the minimax value.
- Method: value approximation via stochastic gradient descent with diminishing step sizes.

Algorithm 1 Tuning weights

1: Initialize a weights $\boldsymbol{w}$;
2: **repeat**
3: Play one game with both player following Minimax-rule using weights $\boldsymbol{w}$, record the feature vectors at each turn in a matrix $\boldsymbol{\Phi}$ and the corresponding minimax scores in a vector $\tilde{\boldsymbol{v}}$;
4: $\boldsymbol{w}_{new} \leftarrow LeastSquare(\boldsymbol{\Phi}, \tilde{\boldsymbol{v}})$;
5: $\boldsymbol{w} \leftarrow \boldsymbol{w} + \gamma(\boldsymbol{w}_{new} - \boldsymbol{w})$
6: **until** converged

Computational requirement

The worst-case time complexity of a minimax tree of depth 4 is approximately 10^6 without pruning, while with alpha-beta pruning, it can be reduced to approximately 10^3.



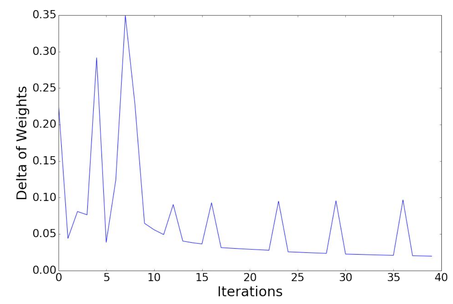
Procedure of alpha-beta pruning

	time (s)		nodes visited	
depths	Without	With pruning	Without	With pruning
2	4.07	0.85	196	27
3	84.39	7.31	4032	301
4	1763	30.68	82944	848

Performance of alpha-beta pruning

Results

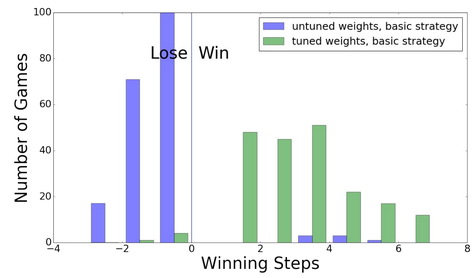
1. Convergence of weights tuning



2. Benchmarking

We benchmarked the performance of algorithms by simulating 200 games against a random look-ahead greedy algorithm. The result is measured by winning steps, which is the number of steps needed for the losing player to finish the game.

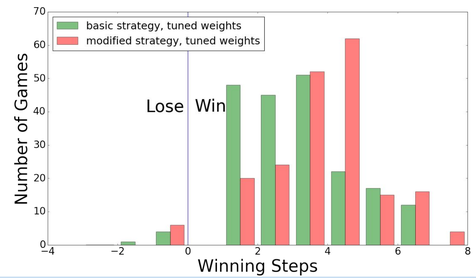
2.1. Effect of weights



2.2. Effect of search depth

The simulation with search depth 4 is underway on AWS.

2.3. Effect of modified strategy



Reference: Bell, George I. 2009. The shortest game of chinese checkers and related problems. Integers 9(1) 17–39.; Liang, Percy. 2015a. Lecture 8: MDPs II. URL http://web.stanford.edu/class/cs221/lectures/mdp2.pdf; Liang, Percy. 2015b. Lecture 9: Games I. URL http://web.stanford.edu/class/cs221/lectures/games1.pdf;