Solving 2048 Puzzle

Jun Hui Teo ID: junhui96

junhui 96@stanford.edu

Stanford

Motivation

Since games have well-defined utility functions and known rules, using AI to solve games allow for a consistent benchmark to compare inference & learning algorithms.

Challenges

1. Stochastic placement and value of tiles in each turn
2. Identify promising board states efficiently in move generation.

References

Nie, Hou, An, AI Plays 2048: http://cs229.stanford.edu/proj2016/report/NieHouAn-AIPlays2048-report.pdf

Yulin Zhou From AlphaGo Zero to 2048 From: http://www.cse.msu.edu/~zhaoxi35/DRL4KDD/2.pdf

Stanford University

Problem Definition

Game Dynamics

- Agent chooses one of: UP, DOWN, LEFT, RIGHT
- Tile of value either '2' or '4' placed in an empty tile in each turn
- Merge two tiles of the same value aligned horizontally/vertically to produce a tile of double the value.
- Gain a score of 2^{k+1} if 2 tiles of same value 2^k are merged.

Objective

Choose an action in each turn to:

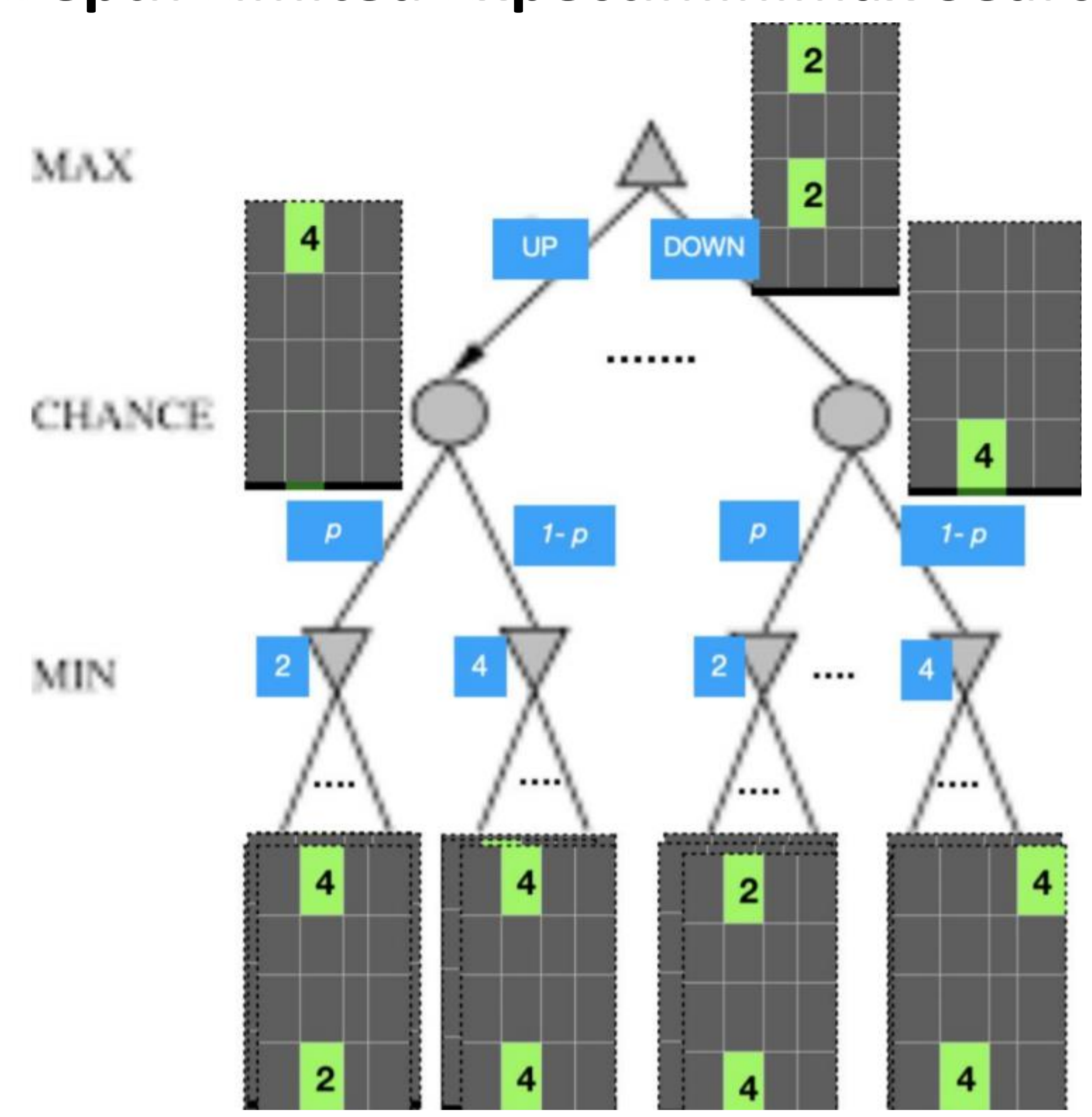
- Obtain the '2048' tile
- Maximise the game score

Model of problem

Two-player, adversarial, turn-taking, zero sum game with stochastic actions.

Approaches

Depth Limited Expectiminimax Search



$$V_{expectiminimax}(s,d) = \begin{cases} Utility(s) & IsEnd(s) \\ Eval(s) & d=0 \\ max_{a\in Action_{agent}(s)} V_{expectiminimax}(Succ(s,a),d) & Player(s)=Agent \\ min_{a\in Action_{computer}(s)}[p * V_{expectiminimax}(Succ_{computer}(s,a,"2"),d-1) & \\ +(1-p) * V_{expectiminimax}(Succ_{computer}(s,a,"4"),d-1)] & Player(s)=Computer \end{cases}$$

- Learn p via Monte Carlo

$p = (1-\eta)p + \eta u, \eta = \frac{1}{\#turns+1}$; u = ['2' in turn]

- Learn depth d via gird search from d = 1 to 4

Heuristics

4^{15}	4^{14}	4^{13}	4^{12}
4^{8}	4^{9}	4^{10}	4^{11}
4^{7}	4^{6}	4^{5}	4^{4}
4^{0}	4^{1}	4^{2}	4^{3}

S-shaped weight matrix to enforce monocity of aligned tiles

- Pruning of game tree by dynamically ordering successor states by heuristic function.
- Limit branching to a fixed factor f, by selecting top f states for exploration.

Results

For 100 samples, distribution of max tiles are:

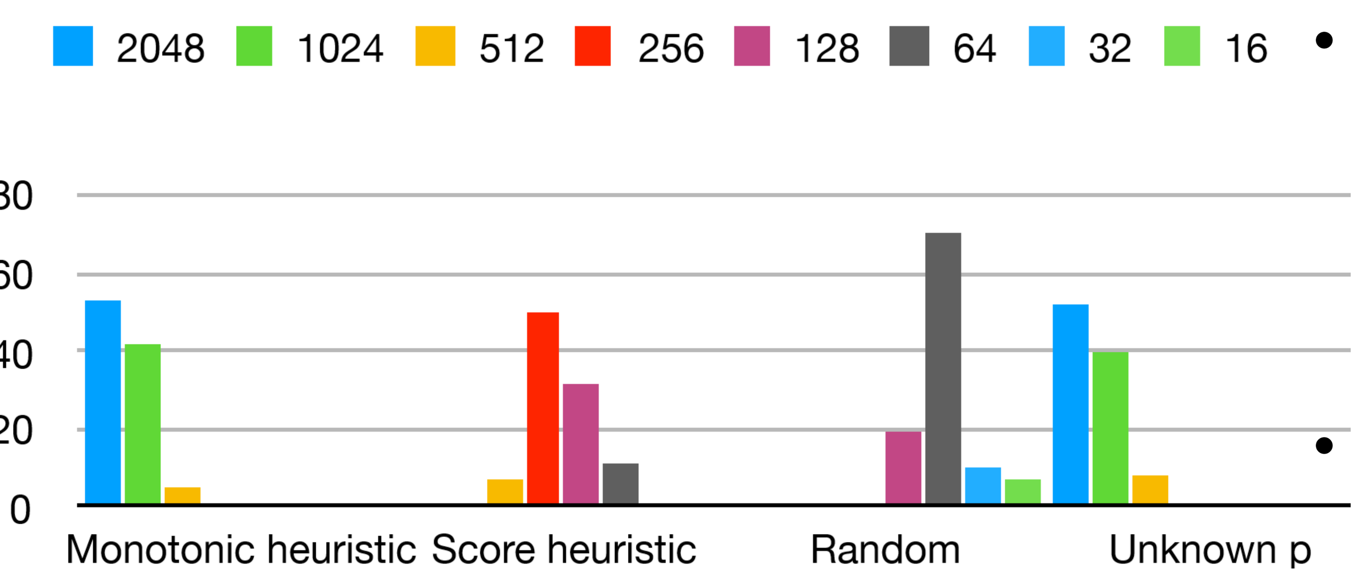


Fig 1: Expectiminimax search to depth 3 with different heuristics v/s baseline

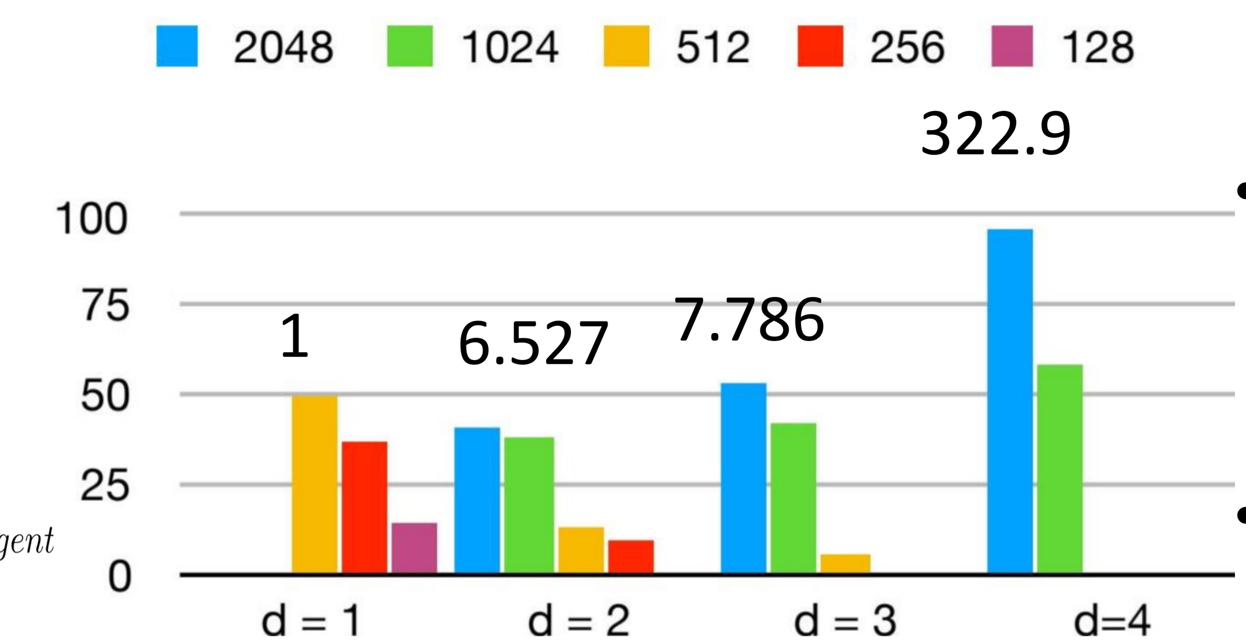


Fig 2: Grid search results with avg normalised time* for each turn

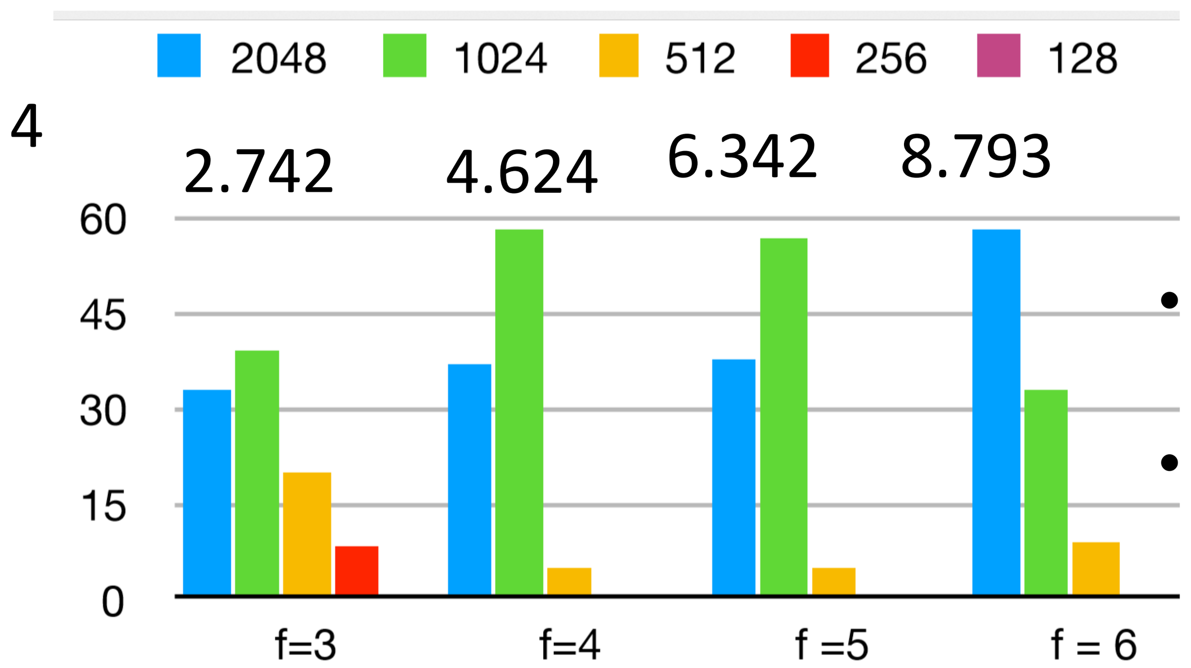


Fig 3: Plot of branching factor f against max tile distribution with normalised time* for a turn

*time is normalised w.r.t d =1 of unpruned case

Analysis

- Monotonic heuristic performs significantly better than the naive score heuristic or the baseline of random moves - success rate of 53%
- Monte Carlo updates learns p well & has comparable performance.

- Success rate increases with depth as further look-ahead ensures more accurate estimate of minimax value.
- d = 3 best manages the success, time-efficiency trade-off

- Limits to branching early on mean that agent neglects states that seem less promising initially but have higher true minimax values
- Most substantial improvement in performance seen from f=5 to f=6.
- But average time for a step similar to unpruned case, due to time to evaluate heuristic and order states.