北京航空航天大學
BEIHANG UNIVERSITY

Data Management and Artificial Intelligence Project Report

School of General Engineering, Beihang University

Game:2048

Members	Wei lin
	Peiyu Hu
	Yuntong Li
	Jiaqi Wan
	Yifei Yan
Professor	Sebastian Wandelt
Teaching Assistant	Weibin Dai

28/06/2020

Introduction of 2048

We select 2048, a kind of number game as our project, and our goal is to reach 2048. At the beginning of the game, the computer gives two number 2 randomly in the 4x4 square. Player can move up, down, right or left as one step. During the movement, two adjacent identical numbers in the same direction will be merged. In one step, the computer randomly gives a 2 or 4 in an empty square proportionally. If the locations of numbers do not change after a movement, this movement is invalid. When four directions are invalid, game is over.

Difficulties of playing 2048

On the one hand, a new number 2 or 4 appears randomly, which gives a high complexity. In the process of searching the game tree, the number of nodes in the search tree increases by tens of times with each additional layer of searching depth. If the search depth is set to 6, the nodes of the search tree will even reach hundreds of millions

On the other hand, it is hard to evaluate the situation. Every step needs to choose the best solution, and this is a precise evaluation of the game, which include the consolidation of figures, positional relationship of numbers and flexibility of the situation.

In conclusion, it is difficult for human player to reach the number 2048 or higher.

Solutions

We try different algorithms step by step, including MC, MCTS, NN, Minmax and Greedy algorithm. Meanwhile, we also add the result of random walk as a reference. We run these algorithms many times and get their average score, time per step and success rates respectively.

Algorithms	Average Score	Time per Step(s)	The rate of success			
			2048	1024	512	256
Random	1063.3	0.0001	0	0	0	0.09
Minmax	4785.4	1.9890	0	0.1	0.4	0.7
Greedy	5839.9	0.0007	0	0.06	0.58	0.95
NN	6656.4	0.1385	0	0.2	0.7	0.9
MCTS	6726.5	0.5785	0	0.4	0.7	1
MC	29831.4	0.1600	0.8	1	1	1

Final Choice

According to the table contents, the best algorithm is Monte Carlo method. Given a current state, this algorithm will randomly simulate for **a** times. And for each time, in order to reduce the time complexity, we only play for at most **b** steps. Then, the results of the random simulation will be divided into several sets according to the direction of their first movement. Comparing the average scores of these sets, we think the first move correspond to the set of highest average scores is the best move. In the experiment above, **a**=200 and **b**=10. Under this condition, the possibility for this algorithm to get a 2048 is 80%. So, our AI has surpassed the level of most human players.

In our presentation, we'll look at these six algorithms in more detail.