Solving the Rubik's Cube without Human Interaction

Victoria Som de Cerff
vs2606@columbia.edu

Michael Kovalski
mhk2160@columbia.edu

Abstract

We intend to investigate known methods of solving a Rubik's Cube without the need for human-tuned heuristics and reproduce similar results.

1. Problem / Goal

A generally intelligent agent must be able to teach itself how to solve problems in complex domains with minimal human supervision. Developing heuristics has proven to limit the search space for these problems and find good solutions, but may sometimes be too complex to come up with and may not necessarily yield optimal performance. An agent without human interaction would be able to learn promising moves by exploring the search space on it's own and determine what states are the most optimal to be in. In the work done in "Solving the Rubik's Cube Without Human Knowledge" they propose a method to solve a self play game without defined heuristics as rewards for intermediate states leading up to the ultimate goal state.

Their algorithm is able to solve 100% of randomly scrambled cubes with a scramble distance of 15, while achieving a median solve length of 30 moves. For the first milestone the goal would be to solve a Rubik's cube using a non-heuristic based algorithm efficiently. I.e. not having to search the entire state space of $4.3x10^{19}$ possible configurations. The second milestone would be to achieve a success rate as close as possible to that achieved in the reference paper given GPU resource constraints.

2. Previous Work

There are previous rule based methods used to solve rubik's cubes in a number of steps, such as knowing which layers to rotate first and how to rearrange given certain states [1]. However, this requires human intuition up front. The upper bound was proven on how many steps it takes to solve a rubik's cube from any starting configuration [2], although there is no simple rule based method to solve this problem. There have been previous attempts to solve this problem computationally. Heuristic based methods have been successful by using a variation of A* [3] which finds the median optimal steps away to be 18 moves. Recently, a DNN was used with featuring engineering to find the optimal route to the solution [4] but these methods take some time to run. Additionally, genetic programming has been used but fail to make optimal decisions when more than 5 steps away from the optimal solution [5]. Recent work has been done to figure out how to solve a rubik's cube without human interaction by using the Autodidactic Iteration algorithm [6], which is similar to the AlphaZero algorithm [7] which uses Monte Carlo Tree Search to help find solutions. This algorithm can solve the rubik's cube with 100% success when moved 15 states away from the solved state, while on average being able to solve these puzzles in 30 steps. However, there is currently no source code for how to implement this solver. In addition, the paper uses 3 GPUs to search through the space and run neural networks for function approximation.

3. Dataset / Collection Method

To be able to collect data, a simulator must be built. The simulator will allow for an agent to take actions on the current state (the rubik's cube configuration) to end up at a new state. In addition, the simulator must know when to stop searching when a solution is found. The rubik's cube simulator must also be able to start at random states or states which are part of some distribution to equally cover as many possible combinations of the states as possible without needing to cover all $4.3 * 10^{19}$ possible configurations. Additionally, methods to generate random configurations that are a specified certain "scramble distance" from the goal state will be used to evaluate accuracy. This simulator can be built in any programming language where the data will be given to the learning algorithm for assessment.

4. Evaluation Criteria

To be successful, the algorithm must get the rubik's cube to the solved final state. However, this is not simply a constraint satisfaction problem. In addition, it must try to do so in as little moves as possible to reach an end goal. God's number is referred to as the maximum number of moves that one can solve a rubik's cube in, which is 20. Ideally, a learning agent would be able to solve this in 20 steps, but we will aim for as little moves as possible. The paper notes 30 moves as the median, so we will attempt to push this number as low as possible with a feasible search space ($<< 4.3 * 10^{19}$). Additionally as the scramble distance increases from the goal state, the time to solve, average number of steps, and failure to find a solution all increased. The paper was able to solve 100% of cubes scrambled 15 moves. We would also attempt to meet that success rate.

References

[1] https://ruwix.com/the-rubiks-cube/how-to-solve-the-rubiks-cube-beginners-method/

[2] https://epubs.siam.org/doi/10.1137/120867366

[3] https://www.cs.princeton.edu/courses/archive/fall06/cos402/papers/korfrubik.pdf

[4] Robert Brunetto and Otakar Trunda. Deep heuristic-learning in the rubik's cube domain: an experimental evaluation. 2017.

[5] https://dl.acm.org/citation.cfm?id=2908887

[6] Stephen McAleer, Forest Agostinelli, Alexander Shmakov and Pierre Baldi. Solving the Rubik's Cube Without Human Knowledge, 2018; arXiv:1805.07470.

[7] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan and Demis Hassabis. Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm, 2017; arXiv:1712.01815.