Rubik's Cube USRA Project Notes

Benjamin Kyle
Concordia University

Supervisors: Dr. Tiberiu Popa, Dr. Denis Pankratov

https://github.com/cosmic-bkyle/usra-project

---

1 Terminology

- **FMC**: Fewest Moves Challenge
- Rubik's Cube legal moves: {F, F', F2, R, R', R2, L, L', L2, B, B', B2, D, D', D2, U, U', U2}
- **EO**: Edge Orientation. The first step for most competitors' FMC strategies. WLOG all possible states with edges oriented are generated by {U, D, R, L}.
- **DR**: Domino Reduction. The next major reduction of the cube after EO. There are several procedural strategies to go from EO to DR. All domino states are generated by {U, D, R2, L2, F2, B2}
- **HTR**: Half-Turn Reduction. Once performed, the cube is solvable with {F2, B2, R2, L2, D2, U2}, which conversely generates all HTR states. Typically, from HTR humans solve all but the E-slice ("leave slice"). This is because one may backtrack through the domino solution and insert slice moves to solve the E slice with small cost (∼1 move).

2 Research Directions

We aim to learn features of the Rubik's Cube states that correlate with promising optimal solution lengths. We are specifically interested in efficient trajectories from the Rubik's Cube group into structured subgroups. We will apply ML techniques to solutions generated by Nissy and GAP-based (Groups, Algorithms, Programming library) tools to improve human heuristics for the Fewest Moves Challenge. The following are some other research directions of our initial brainstorming:

- using ML to predict how many states are visited (time cost) in a human-heuristic-algorithm's search for $\leq$ optimal +1 for solving from the DR state to solved state
- helping humans with scrambled → DR without passing through EO so early
- using ML to predict an optimal first move on a scrambled cube / fmc substep
- speeding up solvers e.g. Nissy, Mallard
- (group-theoretic) discover properties of general algorithms which take random group elements to the nearest element of a particular subgroup, until the identity element is reached. (In the case of the cube and the current FMC meta, this is: Scrambled → EO → DR → HTR → Solved

Related papers:
https://deepcube.igb.uci.edu/static/files/SolvingTheRubiksCubeWithDeepReinforcementLearningAndSearch_Final.pdf uses quarter-turn metric and deep learning to find efficient solutions.

https://arxiv.org/pdf/2502.13266 A Machine Learning Approach That Beats Large Rubik's Cubes
https://github.com/enricotenuti/h48thesis/blob/main/thesis.pdf Computational analysis of Nissy

3 Research Plan

3.1 Short-term Goals

My immediate short-term goals are outlined below.

1. Generate HTR subset stats, where the subsets are defined by chosen features: optimal corner solution length, htr subset.
2. Get my work environment set up: have a visual representation of the cube states, affectable by move sequence strings. In addition, support nissy commands in the same environment
3. Enact machine learning models (e.g. Supervised Regression), and FMC steps along with cube features suspected to correlate with good optimals for the steps (e.g. blockiness for DR $\rightarrow$ solved)
4. Discuss how to get the machine to learn the features (Deep learning)

3.2 Long-term Goals

The project's long-term goals are outlined below.

1. Write a report of findings
2. In the discovery of such important features, contribute to the strategies of world-class FMC competitors

4 Research Journal

May 20

- Discussed high-level goals of the project
- Reviewed "HTR Subset Stats" and asked:
    - What distribution on DR states is the data from?
    - Can I replicate the same numbers?
    - What other features can we think of to predict the optimal solution length from the DR?
    - Task for next week: Compute similar statistics for two other features; corner solution length and blockiness

May 27

- Reviewed code and the statistics generated for HTR subset X optimal length, as well as corner optimal length X optimal length
- Discussed G, a new approach towards the blockiness feature:

$$G = (U, V, E, w)$$

where

$$\begin{aligned} U &= \text{corner pieces of the cube} \\ V &= \text{border pieces of the cube} \\ E &= \{(u, v) \mid u \text{ is touching } v \text{ in the cube state}\} \\ w &: E \rightarrow R \quad \text{assigns weights to the edges.} \end{aligned}$$

- The graph is undirected. Observe that without the weights, every DR state corresponds to a unique such graph
- We would like the total weight of the graph to represent the quality of the DR state, in terms of its optimal solution length.
- If we could learn an algorithm to weigh the edges, then we could see what kind of blocks (e.g. RGY connected to GY) correlate with short solutions.
- Task for the week: develop a representation of this graph, given a dr scramble string. Try to run a linear regression to fill weights.

June 3

- Met with Denis and reviewed code from the week:
  - An object-oriented cube representation, affectable by input move sequences.
  - The bipartite adjacency graph of any given cube state
  - The generation of the dataset (X = 100k bipartite graphs, y = solution lengths)
- Some problems encountered include:
  - Segmentation faults on nissy's end while generating the data.
  - Some relationships were less prominent even though they are equivalent up to symmetries to ones which were strong.

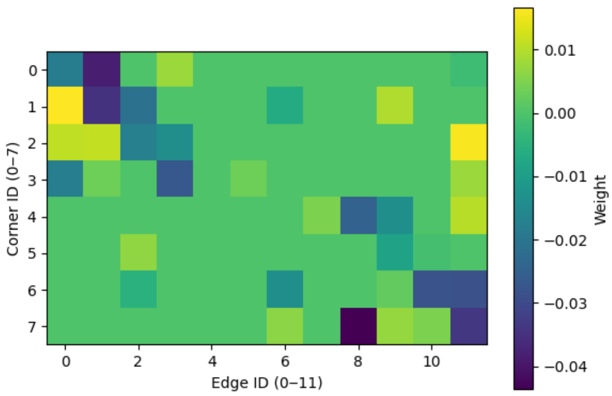

Figure 1: Learned weights of corner-edge relationships in 100k domino scrambles. Details of the scikit Lasso model used can be found in the source code.

- This week: Set up a more classic ML pipeline. Generate 300k scrambles and partition into train, val, and test sets. Tune hyperparameters / model selection by residual error.
- Implement modified cube state representation and adjacency graphs ignoring middle slice.
- The next feature will be triple-blocks, of which there exist two shapes (ignore E-slice). Try and throw these into the regression and compare their weights.

References