Solving Rubik's Cube

Keith H. Randall [1]
NE43-207
randall@lcs.mit.edu

Rubik's cube has fascinated mathematicians and laypeople alike since its introduction by Ernö Rubik [2]. Despite the fact that a large amount of research has been done on Rubik's cube, some important questions remain unanswered. This abstract will focus on one particular unanswered question — what is the hardest position to solve? Although we will not answer this question definitively, we will present some strong evidence that the hardest cube to solve is the one pictured in Figure 1, called "superflip" because it is the solved cube with each edge piece flipped in place.

In order to find the hardest position, we need to define what it means for a position to be hard to solve. We examine the problem of solving Rubik's cube using the *quarter-turn metric*. In this metric, each quarter-turn of a face of the cube counts as one move. Thus, there are twelve possible moves, turning each of the six faces either clockwise or counterclockwise.[2] We denote the six clockwise moves as F,B,U,D,L,R for the front, back, up, down, left, and right faces of the cube, and the counterclockwise moves are denoted with a prime ('). With this definition of a move, a cube is hard to solve if the minimum number of moves to solve that position, called its *depth*, is large. The depth of the superflip cube is known to be 24, and it has been conjectured that it may be the hardest Rubik's cube position. Although the depth of superflip is known, little else is known about other deep cubes. This work is the first to produce an approximate histogram of cube depths.

Determining the depth of a position is a difficult task. There are $43,252,003,274,489,856,000$ possible positions of Rubik's cube, which is too many to exhaustively search with today's computers. We have developed a program, however, that can find the minimum depth of most cubes in a at most few hours.

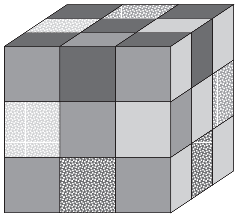

Figure 1: The superflip cube, conjectured to be the hardest Rubik's cube position to solve. It is formed by flipping in place each of the edge peices of the solved cube.

We will describe how this program works and how we have used it to provide evidence that the superflip cube is in fact a very hard cube.

1 Search

In this section, we describe the methods we use to find minimal solution sequences (and hence depths) of cube positions. Our solution is basically a brute-force search algorithm with very agressive pruning heuristics. Our program can solve an average cube in about an hour on a four-processor 466MHz Alpha machine using about 1.2 gigabytes of memory.

Our search algorithm is simple. From a cube position, we try all possible move sequences of a certain length and see if any reach the solved position. Of course, some move sequences are redundant. For instance, the move sequence F F and the move sequence F' F' both generate the same cube. In order to reduce the branching factor of our search, we try to eliminate as many of these redundant move sequences as possible. We do this pruning using four rules:

- Do not undo the last move.
- Do not play three of the same move in a row.
- Do not play two counterclockwise moves in a row.
- Play "parallel" moves in increasing order. Parallel moves are moves of opposite faces. For instance, the move sequences F B and B F both generate the same position. We do not play the lexicographically larger of these two sequences.

[1] Keith Randall is supported in part by a Department of Defense NDSEG Fellowship. His faculty supervisor is Charles E. Leiserson.

[2] Other metrics include the *half-turn metric*, in which half turns of the faces count as one move, and the *half-slice metric*, in which quarter and half turns of the center slices also count as one move.

These rules reduce the branching factor from 12 to 9.374. Why only these four rules? Interestingly, there are no more simple rules like the ones above. The above rules can all be enforced by disallowing certain triples of moves. If we can disallow quadrouples or quintuples of moves, we get no further improvement. In other words, all sequences of 5 moves are distinct when filtered with the above four rules. Additionally, less than 1% of all 6 move sequences are redundant. Thus, these four rules provide all the branching factor pruning we can reasonably expect.

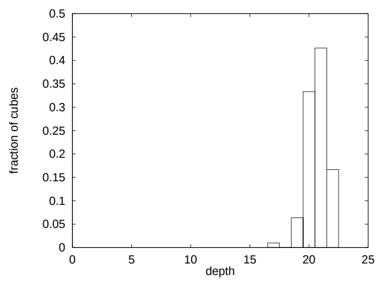

Figure 2: Sampled histogram of depths of random cubes. This histogram represents over 200 random cubes.

2 Pruning Heuristics

Once we have removed most redundant sequences, we apply pruning heuristics that eliminate certain positions from possible consideration for a minimal move sequence. Our rules generate assertions of the form "this position cannot be solved in less than x moves." When searching a particular position to a particular depth, we apply all of our rules, and if one of them asserts that the position in question cannot be solved in the given number of moves, we prune the search. We have considered many different pruning rules, but only two rules are currently used by our solver.

Our first rule is that a position cannot be solved in less moves than the corners of the cube can be solved. The "corners cube", consisting of Rubik's cube minus the edge pieces, can be solved by brute force because there are only $88,179,840$ possible positions. We use a table of the depths of these positions, stored at 4 bits per entry ($\approx$ 42MB), to determine the number of moves required to solve the corner cube.[3]

Our second rule uses a hash table to determine a bound on the depth of a cube. Each hash table entry contains (a lower bound on) the minimum depth of all cubes that hash to that entry. To fill the table, we use an exhaustive enumeration of all cubes near the solved cube. When searching, we hash our unknown cube and determine the minimum depth of that cube using the hash table. If we are looking for a shorter solution than the hash table tells us, we can prune the search. We currently use a 1 GB hash table, stored with 2 bits per entry.

3 Symmetry

We also take advantage of the symmetries of the cube. There are 48 symmetries of the cube, defined by the rigid transformations of space that take the cube to itself. For instance, there is a symmetry obtained by rotating the cube 90 degrees around a face, 120 degrees around a corner, etc. We can apply these symmetries to a cube position to obtain another "symmetric" position.

An important property of symmetric positions is that they have the same depth. We take advantage of this fact by storing only a canonical representative of each equivalence class in the hash table. Because most equivalence classes are of size 48, we save almost a factor of 48 in hash table space. In order to calculate canonical representatives quickly, we again use the corner cube and an auxiliary 84 MB table to figure out which symmetry operation generates the canonical representative for a particular position.

[3] We would use an edge cube rule as well, but the edge cube has $490,497,638,400$ positions, slightly too large for us.

4 Conclusion

Using our search and heuristic techniques, we can solve a random cube in about an hour on a four-processor 466MHz Alpha machine using about 1.2 GB of memory. We use the parallel language Cilk [1] to exploit all four processors of the machine. We have used our program to generate an approximate histogram of cube depths (see Figure 2). After solving over 200 random cubes, we have found no cubes with depth greater than 22, showing statistically that few cubes are depth 23 or greater. In fact, the only known depth 23 equivalence classes are the one adjacent to superflip, superflip + F U F, and superflip + F F B. Since the depth of the superflip cube is known to be 24, we conjecture that it is likely to be the hardest position of Rubik's cube.

References

[1] Cilk-5.1 (beta 1) reference manual. Available on the Internet from `http://theory.lcs.mit.edu/ cilk`.

[2] David Singmaster. *Notes on Rubik's Magic Cube*. Enslow Publishers, Hillside, New Jersey, 1980.