0.1 Proof of the Upper Bound

The upper bound of the Rubik's Cube is the lowest number of twists needed to solve the *Rubik's Cube*n the worst case. The upper bound has been proven to be equal to or less than a certain number of twists. Such proofs have been published several times, and the upper bound has been lowered each time. A major breakthrough was when Thistlewaite's algorithm was proven to be able to solve an arbitrary cube in 52 twists or under. Since then a lot of progress has been made in the field. This section describes this progression.

0.1.1 Set Solver

The algorithm used to prove the current lowest upper bound is known as a set solver and uses Kociemba's algorithm. The set solver is a viable method for proving the upper bound because it does not solve every single cube but a whole set of cubes at the time as the name suggests. This means that it solves approximately 19.5 billion cubes at a time. The set solver does this by finding all the move sequences of a relabeled cube of the distance d that transforms the cube into H.

Algorithm 1 Set Solver [1]

```
1:  f = null
2:  d = 0
3:  while true do
4:    if d ≤ m then
5:      for b ∈ S^d do
6:        if r(ab) = r(e) then
7:          f = f ∪ ab
8:        end if
9:      end for
10:   end if
11:   if f = H then
12:     return d
13:   end if
14:   d = d + 1
15:   f = f ∪ fA
16:   if f = H then
17:     return d
18:   end if
19: end while
```

First in the set solver two variables are initialized. The first one f is a set that can hold all the positions of H is set to *null*. The second variable is the distance d, which is the distance from a scrambled position a to a position in H. This distance d is set to 0.

Next the while loop is run. It will run until d is returned, which is when all positions in H has been found or d has reached it's maximum limit, m.

If d is lower than or equal to m a for loop will be run. This loop performs all possible move sequences of the the length d, and adds the position ab to f if it is a position in H. For efficiency's sake move sequences that give the exact same position more than once are not used. If a move sequence contained F F', that part would be unnecessary.

if f is equal to H all positions in H have been found, d is returned and the algorithm has finished. If not d is incremented by one. The different A moves are performed on all the current H positions in f and the new H positions are saved in f. If f contains all positions in H, d is returned – if not the while loop continues.

When the algorithm has finished all the different possible H positions should be saved, if the maximum distance m is set sufficiently high. The theoretically highest number of twists needed to transform any scrambled cube to H is 12, but more positions are found if it is set higher. This is because the set solver both performs A moves on a cube in H, which gives more H positions. The set solver also performs moves that transforms a cube in H to a cube not in H and then back again by using S moves.

0.1.2 The current and present upper bounds

This is a slow moving field, but some events have occurred the last couple of years. The reason why it is a slow process to prove the upper bound, is because there is a vast amount of different positions a Rubik's Cube can assume. Even with todays computer power there is simply to much data to process. This had the effect that a small group of people dedicate a lot of time to create and improve algorithms to solve arbitrary Rubik's Cubes. We have looked at the algorithm that Herbert Kociemba made and now we will look at the set solver, which Tomas Rokicki helped create.

The set solver has a special way of testing the Rubik's Cubes. It does not solve them to the unit position e, instead it finds a move sequence for a subgroup of the Rubik's Cube this way it can solve approximately 19.5 milliard cubes at a time and not just one. The reason for this is that if you relabel an arbitrary cube, that given cube can be unlabeled to approximately 19.5 milliard different cube positions. Recall that there are approximately 19.5 milliard positions in the set H and all these positions are equal to e when relabeled. The same logic applies to any other given position.

It started moving fast when that set solver proved the first upper bound of 25 moves. This was done on home computers from October 2007 to March 2008. They only needed to solve 6000 sets, but after this they got contacted by John Welborn from Sony Pictures Imageworks and he offered a lot of idle computers from a computer farm to help on the project.

After this the process of lowering the bound sped up, not long after they proved the upper bound of 24 and 23. As the upper bound is lowered they need

to solve more and more sets to ensure that it is the upper bound, and they needed to test almost 27000 and 180000 sets for 24 and 23.

The newest upper bound is on 22 moves and was proved in 2009 (KILDE med tidspunkt på!!!). To prove this they needed to computer 1,265,326 different sets. At the moment they have not proven that the upper bound can be 21, but the computer farm is currently working on it, and they expect that it is possible to lower the upper bound to 20. This means that any arbitrary Rubik's Cube could be solved in just 20 moves.

Bibliography

[1] Tomas Rocicki. Twenty-two moves suffice for rubik's cube. *Mathematical Entertainments*, November 2009. 1