ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 2.114

IJESRT
INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

SOLUTION FOR RUBIK'S CUBE BY USING GENETIC ALGORITHM

Kulkarni Sameer Anil
M.Tech student Dept. of ISE BMS College of Engineering Bengaluru-India

ABSTRACT

Solutions calculated by Genetic Algorithms have come to surpass exact methods for solving various problems. The Rubik's Cube optimization problem is one such area. In this work we present a different approach to solve the Rubik's Cube with a low

Number of moves by building upon the genetic algorithm approach. We provide a group theoretic analysis of the sub problem complexity induced by genetic algorithm approach, transitions and design a Genetic Algorithm from the ground up including detailed derivation of our custom Fitness functions. By using this genetic algorithm approach we can find optimized solution any problem, especially for NP-Hard problem we need to find a robust and optimized solution, Rubik's Cube is also one of the such type of problem, Hence in this paper our focus is to carry various experiment by using Rubik's Cube and to find the number of minimum moves in which we can solve this problem, The experiments will carry by using Rubik's Cube physically or by simulation, After getting Optimized solution paper will talk about results and conclusion that whether we got an optimized solution for this problem or not.

KEYWORDS: Basic Term Genetic Algorithm (GA).

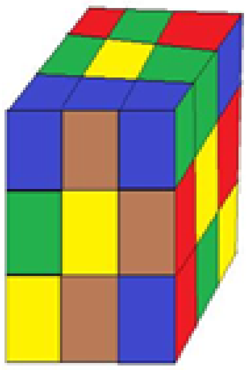
INTRODUCTION

Rubik's Cube is the puzzle developed by the professor

Erno Rubik, Standard version consists of 3*3*3 cube with having different color stickers on exposed sub cubes any 3*3*1 plane can be twisted or rotated in 90,180,270 degrees related to rest of the cube, and our goal is to have all the squares on the each side are of the same color.

A corner cube shows 3 facelets, an edge 2 and a center 1. Each side of the

Cube can be rotated clockwise (CW) and counterclockwise (CCW). Each single move changes the position of 4 edges and 4 corners. The center facelets remain fixed in position. They determined their face's color. For each edge and corner we distinguish between position and orientation: i.e. an edge can be in its right position (defined by the two adjacent center colors) but in the wrong orientation (flipped). There are several known notations for applying single moves on the Rubik's Cube. We will use F, R, U, B, L, D to denote a clockwise quarter-turn of the front, Right, up, back, left, down face and Fi, Ri, Ui, Bi, Li, Di for a counterclockwise Quarter-turn. Every such turn is a single move. In Cube related research, half turns (F2, R2, U2, B2, L2, and D2) are also counted as single moves. This notation is independent of colors but depends on the viewpoint. A sequence of moves, an Operation is created by concatenating single moves and is called operation (i.e. FRBiL2).



A corner cube shows 3 facelets, an edge 2 and a center 1. Each side of the Cube can be rotated clockwise (CW) and counterclockwise (CCW). Each single move changes the position of 4 edges and 4 corners. The center facelets remain fixed in position. They determined their face's color. For each edge and corner we distinguish between position and orientation: i.e. an edge can be in its right position (defined by the two adjacent center colors) but in the wrong orientation (flipped). There are several known notations for applying single moves on the Rubik's Cube. We will use F, R, U, B, L, D to denote a clockwise quarter-turn of the front, Right, up, back, left, down face and Fi, Ri, Ui, Bi, Li, Di for a counterclockwise Quarter-turn. Every such turn is a single move. In Cube related research, half turns (F2,

ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 2.114

R2, U2, B2, L2, D2) are also counted as single moves. This notation is independent of colors but depends on the viewpoint. A sequence of moves, an Operation is created by concatenating single moves and is called operation (i.e. FRBiL2)[1].

GENETIC ALGORITHM

Puzzle can be scrambled by making random number of twists and to solve this puzzle there are various algorithm available, focus of this paper is to solve this puzzle within a polynomial time with the help of Genetic Algorithm. Genetic Algorithm finds the good and robust solution for any problem including NP-Hard problems. We can solve classical problem of Algorithm like Travelling Salesman problem that is Minimum Spanning tree problem, Knapsack problem.

We will solve the Rubik's Cube by using genetic algorithm. Genetic algorithm used to find the optimal solution for the problem[2]. Genetic algorithm works on the principle of Darwin's theory:

1) Survival of the fittest selection is the population improvement or the survival of the fittest that is Structures with the highest finesses deletes the Structures with the lower finesses.

2) Crossovers results in good component with the good structures combining to reproduce even better structures than them that is crossovers recombine the different chromosomes from different genomes.

3) Mutation which creates new structures those are similar to current structure.

How to make decision based on genetic algorithm? We will see with the help of pseudo code & flow chart

PSEUDO CODE:

```
Algorithm for GA is

// start with an initial
    time t := 0;

// initialize a usually random population of
individuals

    Initial population f (t);

// evaluate fitness of all initial individuals of
population

    Evaluate f (t);

// test for termination criterion (time, fitness, etc.)
    While not done do

    // increase the time counter
        t: = t + 1;

    // select a sub-population for offspring production
        f'(t):= select parents f (t);

    //recombine the "genes" of selected
        parents Recombine f' (t);

    // perturb the mated population
        stochastically Mutate f' (t);

// evaluate its new
    fitness Evaluate f'
    (t);

// select the survivors from actual
    fitness f(t) := survive f(t),f' (t);

End GA.
```

RELATED WORK

To get more knowledge about genetic algorithm, I referred various papers [1][2], which talks about the solution of "0-1 knapsack problem" and "n-Queens problem", The complexity of Dynamic approach is of the order of 0(n3) whereas the Greedy Method doesn't always converge to an optimum solution, so the solution for this problem is GA[1]. Genetic algorithm is applicable to wide range problems like an n-queens problem [2] also. In the absence of specialized solution for a particular problem, genetic algorithm would be efficient.

There are several computational approaches for solving the Rubik's Cube, the three most important being the work of Thistlethwaite, Kociemba and Rokicki Their advanced algorithms are based on group theory concepts and apply advanced concepts such as symmetry cancelation and dedicated traversal methods (E.g. Iterative Deep Searching, IDA). Thistlethwaite's Algorithm (TWA) works by dividing the problem into 4 sub problems - specifically subgroups and subsequently solving those. By using precalculated lookup-tables, sequences are put together that move a Cube from one group into another until it is solved.

Kociemba's Algorithm takes the idea of dividing the problem into subgroups from Thistlethwaite, but reduces the number of needed subgroups to only 2[1]. This method uses an advanced implementation of IDA, generating small maps, calculating and removing symmetries from the search tree and tends to solve the Cube close to the shortest number of moves possible. Kociemba made his approach available in form of a program called Cube Explorer which can be found at. Rokicki realized that the initial parts of the pathways computed by Kociemba's Algorithm are solutions to a large set of related configurations. He exploits this property by dividing the problem into 2 billion cosets, each containing around 20 billion related configurations. With this method he was able to push the upper bound to 200 moves sufficing to solve the Cube from any initial scrambled configuration[3].

PROPOSED SYSTEM:

The basic idea of the GA is to divide the problem of solving the Cube into four independent sub problems by using the following four nested groups:
G0=<F,R,U,B,L,D>,G1=<F,U,B,D,R2,L2>,
G2=<U,D,R2,L2,F2,B2>,G3=<F2,R2,U2,B2,L2,D2>
,G4=I. Obviously, G0 = GC. The functional principle of Thistlethwaite's Algorithm is to put the Cube into a state where it can be solved by only using moves from Gi which again has to be achieved by only using moves from Gi−1 for i = 1, . . . 4, thus named nested groups.

Specifically, every stage of the algorithm is simply a lookup table showing a transition sequence for each element in the current coset space Gi+1\ Gi to the next one (i = i+1). These coset spaces, each describing a reduced form of the 33

Rubik's Cube puzzle, induce different kinds of constraints. This directly results in the total number of attainable states being reduced by using only moves from some subgroup Gi+1. The exact orders for each group are calculated as follows:

G0

|G0| = 4.33* 10^19 represents the order of the Cube Group.

G1

The first coset space G1\G0 contains all Cube states, where the edge orientation does not matter. This is due to the impossibility of flipping edge cubies when only using moves from G1. As there are 2^11 possible edge orientations,
|G1\G0|=2^11=2048…… (1)

The order of |G1| is
|G1|≡|G0|÷|G1\G0|=2.11*10^16……..
(2)

G2

Using only moves from G2, no corner orientations can be altered (eliminating 37 states). Additionally, no edge cubies can be transported to or from the middle layer (eliminating 12! (8! ·4!) States). The coset space G2\G1 thus depicts a reduced puzzle of the order |G2\G1|=3^7*(12! ÷ (8! \4!))=1082565…
(3)
And
|G2|≡|G1|÷|G2\G1|=1.95*10^10 ………. (4)

G3

Once in the coset space G3_G2, the Cube can be solved by only using moves from G3, here the edge cubies in the L,R layers cannot transfer to another layer (eliminating (8÷(4!·4!)) * 2 states) and corners are put into their correct orbits, eliminating (8! ÷ (4! ·4!)) * 3 states).

Thus,
|G3\G2|= ((8! ÷ (4!*4!))^2)*2*3=29400………. (5)

And
|G3|≡|G2|÷|G3\G2|=6.63*10^5……… (6)

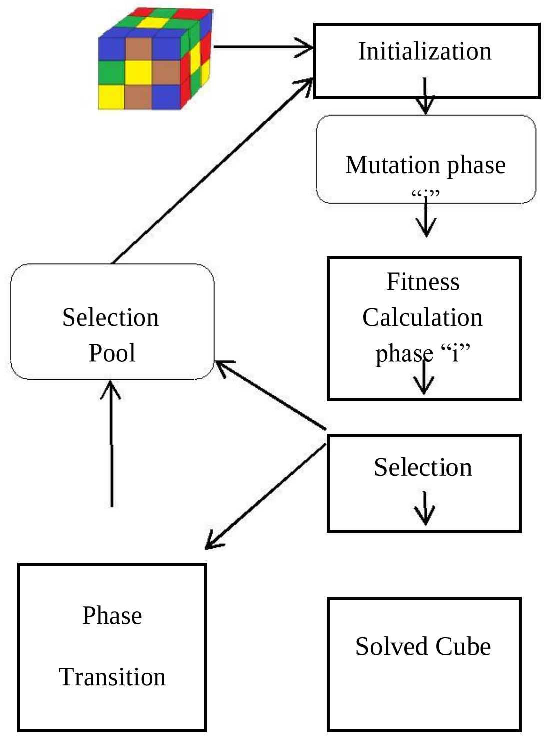
G4

As G4 represents the solved state - obviously |G4| = 1 which means there exist a mere |G3| possible states for which a solution needs to be calculated to transfer from G4_G3 to solved state. Most essential to TWA are the groups G1, G2, G3 as G0 simply describing the Cube Group GC and G4 the solved state. To further exemplify how the coset spaces simplify the Rubik's

Cube puzzle the following may prove helpful. When looking at the constraints induced by G2\G1\G0 carefully (combining the constraints induced by G2\G1 and G1\G2) it is essentially a Rubik's Cube with only 3 colors - counting two opposing colors as one. This representation can be reached for each distinct coset space by examining and applying its effect to the complete Rubik's Cube puzzle.

While solving the Rubik's Cube in a divide and conquer manner, breaking it down into smaller problems (by generating groups and coset spaces) is effective, there exists one major flaw. Final results obtained by concatenating shortest subgroup solution do not necessarily lead to the shortest solution, globally.

Basic Work-Flow of Genetic Algorithm for this Puzzle:



FITNESS FUNCTION

Translation phase which contains, the translation of our algorithm in to an appropriate fitness function is mandatory, Survival the fittest so the functions having large running time should be discarded[1].

Step1:
Phase 0 → Phase 1

to reach phase 1 from any scrambled Cube, we have to orient all edge pieces right while ignoring their position. The fitness function for this phase simply increases the variable phase0 by 2 for each wrong oriented edge. Furthermore, we add the number of moves that have already been applied to the particular individual in order to promote shorter solutions. Finally, we adjust the weight between w (number of wrong oriented edges) and c (number of moves applied to current Cube individual). This will be done similarly in all subsequent phases.

$$phase0 = 5*(2w) + c \qquad (7)$$

With a total of 12 edges which can all have the wrong orientation this gives max {2w} = 24. The Cube has been successfully put into Phase1 when phase0 = c.

Reaching Phase1 is fairly easy to accomplish, thus making the weight-factor 5 a good choice.

Step2:

Phase1 → Phase2 In order to fulfill Phase 2 the 8 corners have to be oriented correctly. Edges that belong in the middle layer get transferred there. Tests with the Classical solution showed it somewhat problematic to do this in one step. Oftentimes, the algorithm would get stuck in local optima. To solve this, the process of transferring a Cube from Phase1 to Phase2 has been divided into two parts. First, edges that belong into the middle layer are transferred there. Second, the corners are oriented the right way. The first part is fairly easy and the fitness function is similar to that from phase0 except for w (number of wrong positioned edges), i.e. edges that should be in the middle layer but are not.

$$phase1 = 5* (2w) + c \qquad (8)$$

In the second part, for each wrong positioned corner, 4 penalty points are assigned as they are more complex to correct than edges. Obviously, in order to put the Cube from G1 to G2 both phases described here have to be fulfilled, which yields:

$$phase2 = 10 * (4v) + phase1 \qquad (9)$$

Where v represents the number of wrong oriented corners. The weighing factor is increased from 5 to 10 to promote a successful transformation into G2 over a short sequence of moves.

Step3:

Phase2 → Phase3 We now have to put the remaining 8 edges in their correct orbit. The same is done for the 8 corners which also need to be aligned the right way. Thus, the colors of two adjacent corners in one circuit have to match on two faces. In G3 the Cube will only have opposite colors on each face. Let x (number of wrong colored facelets) and y (number of wrong aligned corners), then

$$phase3 = 5* (x + 2 *y) + c \qquad (10)$$

Step4:

Phase3 → Phase4 (solved) The Cube can now be solved by only using half-turns. For the fitness function we simply count wrong colored facelets. Let z be the number of wrong colored facelets, then

$$phase4 = 5 \cdot z + c \qquad (11)$$

To summarize, 5 different fitness functions are needed for the Genetic algorithm. Phase i is solved if phase i = c, i = 0, ..., 4 and with the properties of nested groups we can conclude, given the above, a solved Cube

ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 2.114

implies:
4
Σ phase i =
c i=0
Fulfilling the above equation satisfies the constraints induced by the groups G0,….G4, with the final fitness value c describing the final solution sequence length. The weight factors chosen are based on consecutive testing throughout development. The ratio is dictated by the size of the nested groups.

Finding optimal weights presents a separate optimization problem and may be subject to future work.

MUTATION OPERATORS

The mutation operators are dictated by the subgroups used[2]. Conveniently, the maximum sequence length (s) needed to transform the Cube from one subgroup to another is given by Thistlethwaite [13]. Those lengths are 7,13,15,17 (the sum of which is 52, hence"52 Move Strategy") for each group transition respectively. An individual in phase i is mutated by:

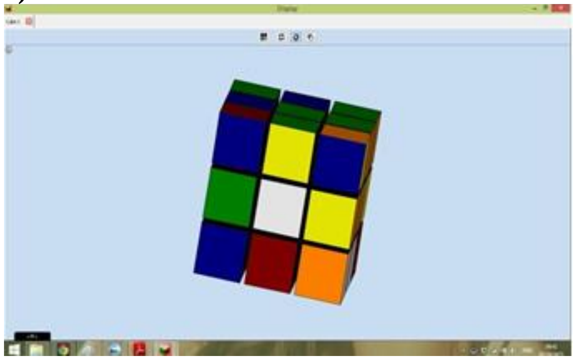
1. Generating a random length (l) with 0 ≤ l ≤ s, according to i (i = 0 → s = 7, i = 1 → s = 13, i = 2, 3 → s = 15, i = 4 → s = 17)
2. Concatenating l random single moves from the according group Gi
3. Applying this sequence to the current Cube individual

For example: Let i = 2 (transitioning from G2 → G3). The maximum sequence length for this step is s = 15. Let random l = 4, (0 ≤ 4 ≤ 15). Next, we chose a random single move from G2, repeat this a total of 4 times and concatenate these to form a sequence. Let those 4 single moves be D, F2, and R2, U. This results in the sequence D,F2,R2,U representing the present mutation which is applied to the current Cube individual. In case of l = 0 the mutation is an empty sequence, leaving the current individual untouched. Given an appropriate fitness, this allows distinct Cubes to survive multiple generations.

OBSERVATIONS

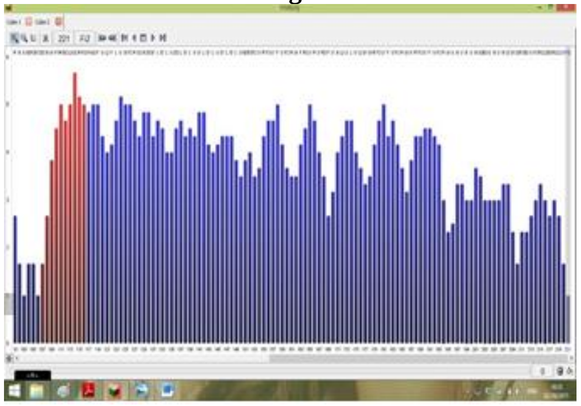
Observations are observed by using simulator named "Arcus". With the help of this software we can simulate the Rubik's cube,

1) Initial Position:



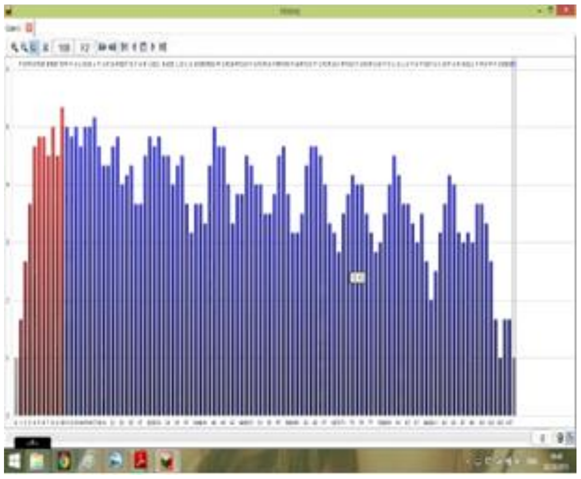
2) Final Solution by Classical Algorithm: Number of moves: 221

As the numbers of moves are higher so the running time for the solution is high.



3) Final solution by Genetic Algorithm: Number of moves: 107

As the numbers of moves are higher so the running time for the solution is lower as compared to classical solution.



ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 2.114

CONCLUSION

It can be concluded that, genetic algorithm can provide optimal solution for the Rubik's cube problem, as we have observed in above simulation results; the classical solution takes **227 moves** to solve this cube, while by applying the genetic algorithm theory we can get the optimal solution in **107 moves**, so this paper concludes that the genetic algorithm provides the efficient solution for this problem.

REFERENCES

[1] Modified Genetic Algorithm for Solving n-Queens Problem Jalal eddin Aghazadeh heris, Mohammadreza Asgari Oskoei,Intelligent Systems (ICIS), 2014.

[2] A task scheduling algorithm based on genetic algorithm

and ant colony optimization in cloud computing, Chun-Yan LIU Pei WU, 2014 13th International Symposium on Distributed Computing and Applications to Business, Engineering and Science

[3] The Enhanced Genetic Algorithms for the Optimization Design Pengfei Guo XuezhiWang Yingshi Han, 2010 3rd International Conference on Biomedical Engineering and Informatics (BMEI 2010)

[4] Test Data Generation Using Annealing Immune Genetic

Algorithm, X. B. Tan, Cheng Longxin, Xu Xiumei, 2009 Fifth International Joint Conference on INC, IMS and IDC.