Rubik’s Cube

Project

**Consider a Rubik cube. How can we model it?**

First, it is reasonable to set one of the corners to be stationary and reduce all rotation operations to three basic quarter rotations (namely, for counterclockwise rotation).

(Possible question: why is this the case?)

A picture containing container, box

Description automatically generated

Rotation

A picture containing container, box

Description automatically generated

Rotation

Rotation

A picture containing container, box

Description automatically generated

A picture containing text, green

Description automatically generatedA picture containing container, box

Description automatically generated

Stationary block

Back of the cube

Rotation

Rotation

Rotation

One block face

**Question: How to abstract this into string, arrays, or matrices (to form a group, )?**

To approach this problem, consider in previous page, we stated that one of the blocks could be considered as stationary. This leaves us with individual block faces.

So, a string of length may be used to represent any possible permutation of the cube:  
 abcd efgh ijkl mno pqr stu

This says, the string shown above can be permuted with matrices. However, algorithms involving permutation matrices are not very efficient. In the program I built, permutations are simply done by methods permuting specific letters in the string.

To form a group, consider the rotation actions as the elements of the group, all possible combinations of rotation form the whole action group.

**It is obvious that the group can be generated by the 3 basic rotations .  
In the following, write down the 3 conditions for the 3 basic elements (identity is given).**

A picture containing shoji

Description automatically generatedA picture containing shoji

Description automatically generated

g

u

t

s

m

r

l

k

q

p

j

o

n

i

h

f

e

d

c

b

a

A

E

abcd efgh ijkl mno pqr stu

A picture containing shoji

Description automatically generatedA picture containing shoji

Description automatically generated

C

B

**Let the group we just built be .**

**Question: What is the size of this group?**

Text, letter

Description automatically generated

All permutations of blocks, and corner rotation ignoring one block.

**Question: Why is corner turn impossible?**

Parity is wrong. All turns are odd permutation, but corner turn is even.

Assume no information of one of the block. Can take in corner turn.

**Question: Is this group abelian? cyclic?**

Not abelian, by counter example or .

Not cyclic:   
 Theorem 4.9: Every cyclic group is abelian.

**Question: Is it possible to ignore the color and only model the blocks? (only 7?)  
 (hint: assume isomorphic and try the action )**

No, because R U R’ U R U2 R’ U2 while .  
 This action does not permute the blocks, but scrambles the colors.

The subgroup that does this is actually a normal subgroup and can be proven easily.  
 It is also abelian. (unsure if cyclic though, not proven yet.)

I do not think it is cyclic. There will always be one block not permuted by any combination. The generating element will need to be even permutation.

**Try the previous element, pay attention to its order.  
Question: Why is it possible to have an element of order 3 when the whole group is generated by 3 order 4 elements?**

The previous element is order 3.  
 The group is not abelian.

**Before approaching the problem, we should introduce the concept of Graph and Cayley Graph here.**

Why is graph important? (very important data structure in computer programming)

What is a vertex? What is an edge?   
 Why we like undirected graph for Cayley graph (inverse exists and easier traversal?)?  
 How to represent and traverse a Cayley graph for .  
 put an example of subset and different generating set here

Why is it too difficult to produce a Cayley graph for mini-Rubik cube (too many vertex, not very helpful)?

Can make a small graph traversal here to show the exponential growth of vertex number.  
 Can show or ask for a small subset of the Rubik cube graph.

Question: If we solve this problem by brute force, and the maximum number of quarter turns needed was 14, how many cases do we need to check?

Try to develop a tree class to solve this problem on Java.

What does this look like in graph traversal?

(a matrix representation here might be better than string)

Why do we want to avoid brute force BFS? Why use algorithms and subsets?  
 the efficiency class is n! we are trading steps for better efficiency  
 this is a lot like steepest descent and descent by step (divide and conquer)

Why is hash table bad?

Explain here.

Preferred approach: decrease and transform

Question: What if we divide the problem into some subgroups? Will this reduce the total number of cases we need to check? Will this increase the number of total quarter turns in the final solution? (Yes and Yes)

Show this with the program.

What does this mean in the sense of graph traversal (reach into a specific smaller subset of the graph, then go in closer and closer in each stage.).

What is the overall efficiency of this method? Why is it a lot better? What is the trade off?

Question: Any other ways to set the subgroups?

Can try the here.

More onto larger even Rubik cubes…

A picture containing indoor, red, close, tile

Description automatically generated