Dean&Francis

ISSN 2959-6157

Explain the combination relationship between different rotation operations of Pyraminx by using group theory, finding the solution sequence from the initial state to the target state

Mengyun Zhang

The University of Manchester, Oxford Road, Manchester, M13 9PL, UK
zhangmengyun999@outlook.com

Abstract:

This article explores the strategy of combining group theory and programming methods to study and solve the problem of irregular Rubik's Cube. Group theory, as an important branch of algebra, is a powerful tool for studying symmetry and structural transformations. In the study of Rubik's Cube, group theory is widely used to analyze various transformations and their inherent mathematical properties. Irregular Rubik's Cube, compared to traditional standard Rubik's Cube, has a more complex structure and transformation mode, which requires us to consider more subgroups, quotient groups, and their nested relationships when applying group theory.

Keywords: Pyraminx, Group theory, python.

1. Introduction

The Pyramid Cube, also known as the Pyramid Tetrahedral Cube, is a highly challenging and entertaining 3D puzzle toy. It breaks the conventional design of a cubic cube with six faces and adopts a tetrahedral (pyramid shaped) structure, with each face composed of triangles, and a total of four such faces connected to a central point. Players need to rearrange the scrambled color blocks by rotating the four sides of the pyramid cube, so that each side has a uniform and compliant color. Compared with the third-order Rubik's Cube, the solution of the Pyramid Rubik's Cube is based on strategies such as layer first method and angle first method. However, due to its special structure, the rotation rules and steps are different, requiring higher spatial imagination and logical reasoning ability.

2. Background

2.1 Structure

Structurally, the core of a pyramid cube is a fixed central axis, similar to the central block in other types of cubes, but because it is a tetrahedron, there is only one central point instead of multiple. Around this center point, four axes extend outward, with several small pieces connected to each axis. These small pieces are interconnected through specific mechanisms (such as a combination of grooves and protru-

sions) and can rotate within a certain range, thus forming the movable part of the entire Rubik's Cube. By rotating the small blocks on these axes, attempt to restore the color blocks on each face to a uniform color state, where all small blocks on each face have the same color.

2.2 Moves

In mathematical language, the rotation of a pyramid cube can be accurately described by defining its faces, axes, and rotation operations. Firstly, we clarify that the Pyramid Cube has four faces, each of which is an equilateral triangle, and assume that these faces are labeled as F respectively F1, F2, F3, F4. Due to its tetrahedral shape, the Pyramid Cube has three independent axes of rotation that pass through the center point of the pyramid and connect the centers of two opposing faces.

We can define these three axes as A respectively A1 (Connect F1 and F3). The center A2 (Connect F2 and F4). The center and a virtual axis A0 (perpendicular to the bottom, if we consider the bottom as F1 or any other surface, this axis does not exist physically, but is useful in mathematical description because it represents rotation around the center of the base surface).

Next, we define the rotation operation. For each axis Ai (i=0,1,2), we can define two types of rotations: clockwise (CW) and counterclockwise (CCW), each of which can specify the degree of rotation (but in Rubik's Cube, we usually only care about rotations of 90 degrees, 180 degrees, or 270 degrees, as smaller rotations can be achieved through a combination of these basic rotations).

3. Methodology

3.1 Conventional rotation of Pyraminx

For example, if we want to describe an axis around A1, the operation of rotating 90 degrees clockwise can be represented as R1 (CW, 90 degrees). Similarly, around axis A2, turning counterclockwise 180 degrees can be represented as R2 (CCW, 180 degrees) [1].

However, it should be noted that due to the unique structure of the pyramid cube, not all rotations are independent. Especially, around A0, the rotation of the axis (i.e. the overall rotation of the base) does not physically change the state of the cube, as it only rearranges the observer's perspective. Therefore, in mathematical descriptions, we often ignore this rotation or consider it equivalent to an identity operation.

Finally, the solution to a pyramid cube can be seen as a sequence of rotation operations that transform the cube from an initial state to a target state where all face colors match. For example, a solution may look like this: R1 (CW, 90 degrees), R2 (CCW, 180 degrees), R1 (CW, 90 degrees)… until reaching the target state.

4. Modelling

Define the state space of the Pyromania's Cube

The number of moves required to solve a tower of n discs is 2^n-1, so for n=1,2,3...10 discs, you will need to perform 1,3,7,15,31, 63, 127, 255, 511, or 1023 moves, without errors [2].

State representation: Each state can be represented by a three-dimensional color matrix (or a more efficient encoding method, such as a combination of position permutation and color permutation).

—State set: All possible states form a state space, which is vast but finite for a Pyraminx's Cube.

Define rotation operation

—Rotation type: Define six basic rotation operations corresponding to single-layer rotation of the six faces of the Pyraminx's Cube (upper U, lower D, left L, right R, front F, rear B). Double-layer or full-layer rotation can be further defined, but these can typically be represented as a combination of single-layer rotations [3].

—Rotation function: Write a function for each rotation type that takes a Pyraminx's Cube state as input and returns the new state after applying the rotation.

Establish a transformation group

—Group element: Each rotation operation (or combination thereof) is an element of the group. Group elements can be composite, meaning multiple rotations can be applied continuously.

—Unit element: The operation that does not change the state of the Pyraminx's Cube (i.e. does not perform any rotation) is the unit element of the group.

—Inverse element: Each rotation operation has an inverse operation, which rotates the cube back to its original state.

—Union law: The compound of rotation operations satisfies the union law, which means that the order of rotation does not affect the final result (although in practical operations, some rotation orders may not be feasible due to physical limitations).

The properties and structure of groups

—Order: The order of a Pyraminx's Cube group (i.e. the number of elements in the group) is equal to the number of possible Pyraminx's Cube states, which is a huge number.

—Generative set: A Pyraminx's Cube group can be generated from a small generative set, which typically includes six basic rotation operations (or fewer equivalent operations).

—Subgroup: A subgroup of a Pyraminx's Cube group

ISSN 2959-6157

can be defined, such as a group that only includes even step rotations (because any rotation twice will return to its original position).

Application and Implementation

—Algorithm: Design a Pyraminx's Cube algorithm using the theory of Pyraminx's Cube groups, such as the Kociemba algorithm, which utilizes some properties of groups to accelerate the search process.

—Software implementation: Implement the rotation functions and group operations mentioned above in programming languages such as Python and C++, as well as data structures for representing and manipulating Pyromania's Cube states.

—Visualization: Develop a user interface that allows users to see real-time updates of the cube's status and the effects of rotation operations.

Further optimization

—State compression: finding more effective ways to represent states to reduce memory usage and improve algorithm efficiency.

—Heuristic search: Combining heuristic information to guide the search process and reduce the size of the search space.

—Parallel processing: Utilizing multi-core processors or distributed computing to accelerate the search and solve Pyromania's Cube process.

5. Python Methods

Implementing a representation of a Rubik's Cube's state, rotation functions, and group operations in Python involves defining a suitable data structure for the cube's state and then writing functions to manipulate that state. Below is a simplified version focusing on a 2x2x2 Rubik's Cube for brevity and clarity.

1. Data Structure for Cube State

We present the group Rubik(M) as a permutation group on the corners and side edges. This way of presenting the group follows where the standard Rubik's cube was considered [1-3].

For a 2x2x2 cube, we can represent the state as a 3D array (or a list of lists of lists) where each inner list contains the colors of the stickers on that face, organized by layer and row. However, for simplicity and efficiency, we'll use a flat list and some indexing logic to simulate the 3D structure.

2. Constants for Colors and Faces

Let's define some constants to represent colors and faces for clarity.

1) COLORS = ['W', 'R', 'G', 'B', 'Y', 'O'] # White, Red, Green, Blue, Yellow, Orange

2) FACES = ['U', 'D', 'L', 'R', 'F', 'B'] # Up, Down, Left, Right, Front, Back

3) # For a 2x2x2 cube, we can flatten the 3D structure into a 1D list

4) # Each face has 4 stickers, ordered top-left, top-right, bottom-right, bottom-left

5) # Faces are ordered U, D, L, R, F, B

6) FLAT_CUBE_SIZE = len(FACES) * 4

3. Initial Cube State

Let's define a function to create a solved 2x2x2 cube state.

```python
def create_solved_2x2x2():
    # Solved state for a 2x2x2 cube
    return [
        'W', 'W', 'W', 'W',  # U
        'Y', 'Y', 'Y', 'Y',  # D
        'R', 'R', 'B', 'B',  # L
        'G', 'G', 'O', 'O',  # R
        'R', 'B', 'R', 'B',  # F
        'G', 'O', 'G', 'O'   # B
    ]
```

1) Rotation Functions

Rotation functions can be implemented by swapping the relevant stickers in the flat list. For simplicity, we'll only implement single face rotations (90 degrees clockwise).

Note: This rotation function is very simplified and only handles 90-degree clockwise rotations for the outer faces of a 2x2x2 cube. For a 3x3x3 or larger cube, or to handle counterclockwise rotations, additional logic would be needed.

2) Using the Functions

This code provides a basic framework for representing and manipulating a 2x2x2 Rubik's Cube state in Python. Extending this to a 3x3x3 or larger cube would require a more complex data structure and rotation logic, possibly

involving.

3) Python in Pyramids Cube

To solve a "pyramid cube" might be a bit ambiguous as it's not a standard term in mathematics or computer science. However, I'll assume you mean generating a pattern that resembles a pyramid when printed to the console, where each layer of the pyramid is represented by a different number of cubes (represented by, say, asterisks '*') increasing downwards [4].

Here's a simple Python program that demonstrates how to print such a pyramid pattern. This example will create a pyramid of height n, where each layer has an increasing number of asterisks (*) from 1 to n, aligned in the center of the console:

```
def print_pyramid(n):
# Loop through each layer of the pyramid for i in range(1, n + 1):
# Print leading spaces to center the asterisks
# This is a simple way to center, assuming a console with an even character width
# For a more precise centering, you might need to adjust based on the actual console width
print(' ' * (n - i), end='')
# Print the asterisks for the current layer print('*' * (2 * i - 1))
# Example usage
n = 5 # Adjust the height of the pyramid print_pyramid(n)
```

This code snippet defines a function $print\_pyramid(n)$ that takes a single parameter n, which represents the height of the pyramid. It then iterates from 1 to n, printing the appropriate number of spaces to center the asterisks on each line, followed by an increasing number of asterisks (*) to form the pyramid shape.

Note: The calculation $2*i-1$ for the number of asterisks ensures that the pyramid has an odd number of asterisks on each layer, which helps maintain a balanced appearance when centered. If you want the pyramid to have an even number of asterisks on the bottom layer, you'd need to adjust this calculation accordingly.

Also, keep in mind that the centering method used here (printing $n-i$ spaces before the asterisks) is a simple approach that assumes your console or terminal window has a sufficient width to display the pyramid without wrapping. For more precise centering, especially in narrower windows, you might need to dynamically calculate the required number of spaces based on the actual console width and the length of the longest line of asterisks.

6. Data

The data of the Pyraminx's Cube group mainly comes from mathematical group theory research, especially the analysis of group structure for Pyraminx's Cube transformations. These studies are typically based on the physical properties and operational rules of Pyraminx's Cube, forming precise descriptions of all possible states and their transformations through mathematical abstraction and modeling. In addition, with the development of computer technology, computer-aided proof and simulation have also provided important data sources for the study of Pyraminx's Cube groups.

7. Discussion

1. The uniqueness of Pyraminx's Cube groups as examples of group theory

The Pyraminx's Cube group, as a concrete example of group theory, demonstrates the powerful power of group theory in solving practical problems. By defining the basic rotation of the Pyraminx's Cube as the generator and proving that these rotations satisfy the four fundamental properties of the group, we construct a complete group structure to describe all possible states of the Pyraminx's Cube and their transformation relationships. This method of combining abstract mathematical concepts with concrete physical objects not only makes group theory more vivid and concrete, but also promotes the application and development of group theory in a wider range of fields.

2. Analysis of the Structure and Properties of Pyraminx's Cube Groups

In the paper, we conducted a detailed analysis of the structure and properties of Pyraminx's Cube groups. By calculating the order of the Pyraminx's Cube group (i.e. the total number of states), we reveal the enormous number of states in the Pyraminx's Cube, further emphasizing its representativeness as a complex system [5]. It also provided a mathematical basis for designing efficient Pyraminx's Cube solutions.

8. Conclusion

In this article, we delve into the close connection between Pyraminx's Cube and group theory, which not only enriches the application scope of mathematical theory but also provides a new perspective for Pyraminx's Cube enthusiasts and researchers to understand the complexity and beauty of this classic puzzle toy.

ISSN 2959-6157

References

[1] MATHIEU DUTOUR SIKIRIC. (2020). A VARIATION ON THE RUBIK'S CUBE. https://www.gap-system.org/Doc/Examples/rubik.html

[2] Robinson Roulette. (1985). Robs puzzle. http://www.robspuzzlepage.com/seqmove.html

[3] Gerald Jiarong Xu. (2018). Solving Megaminx puzzle With Group Theory. New York. Pp. 2-19.

[4] D. Kunkle, G. Cooperman, Twenty-six moves suffice for rubiks cube, in: Proceedings of the 2007 international symposium on Symbolic and algebraic computationISSAC07, 2007.

[5] D. Joyner, Adventures in Group Theory: Rubik's Cube, Merlin's Machine, and Other Mathematical Toys, Johns Hopkins University Press, 2008.