Cube Algorithm

Bobitsmagic

1 Move definitions

$M = \{$ **L, L', L2, R, R', R2, D, D', D2, U, U', U2, B, B', B2, F, F', F2** $\}$

2 IndexCube

We are in need of class that represents a 3 by 3 rubiks cube. The IndexCube stores the permuation and orientation state of the corners and edges seperately.

2.1 Edge Permuation

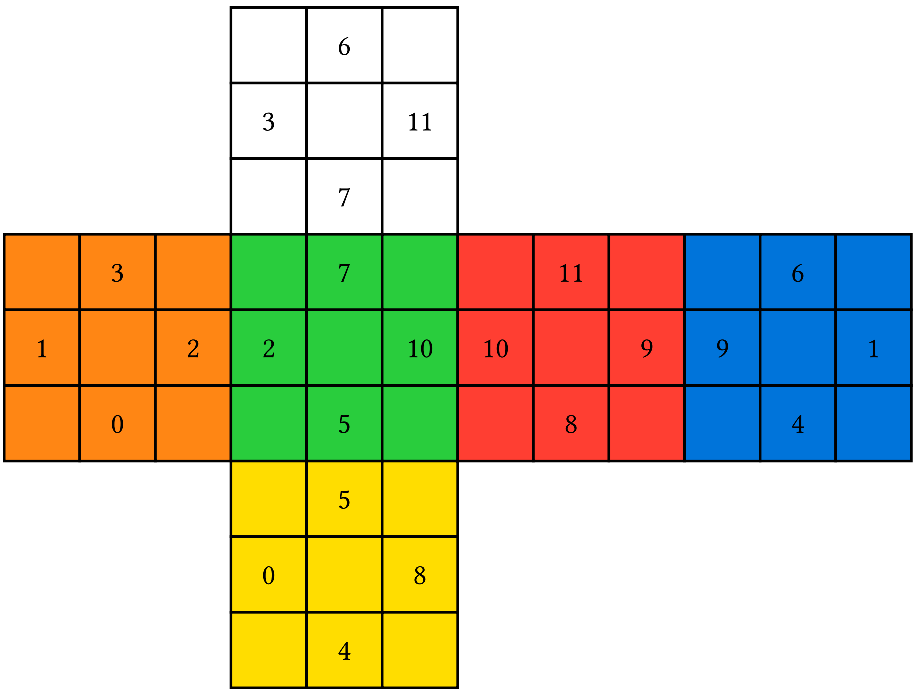
The position of all edges is a permuation of 12 elements. Instead of using 12 integer to store the position of every edge we use one integer that stores the lexicographic index of the permuation. There are $\lceil \log_2(12!) \rceil = 29$ bits needed to store all possible indices and we use a 32 bit integer to store this value. We use the family of bijective functions $E_i : \{0, 1, ..., 11\} \to \{0, 1, ..., 11\}$ where $i \in \{0, 1, ..., 12! - 1\}$ is the lexicographic index of the permuation that E_i represents.

A 12 element permuation will be represented by the following 2 row matrix:

$$\begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 \\ x_0 & x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & x_8 & x_9 & x_{10} & x_{11} \end{pmatrix}$$

.

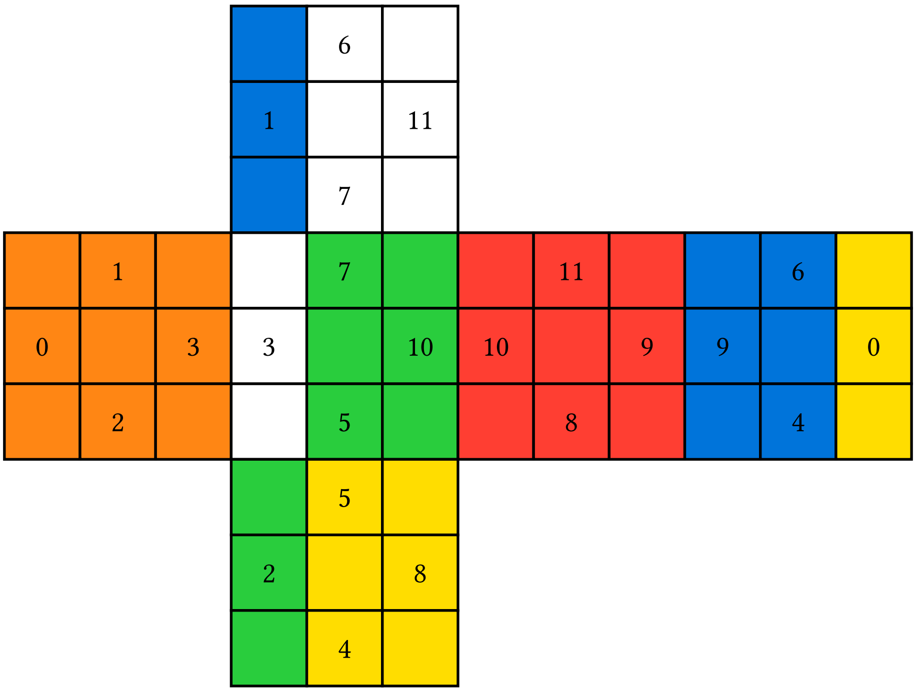
Every edge has to corrospond to an index between 0 and 11 now. We sort the edges by their x, y and z components on a right handed coordinate system with the green facing towards positive z and white facing towards positive y. In the following cube net the resulting indices can be seen.



The specific selection of indices will reduce the amount of space needed to store a matrix in discussed in section **KEK**. After the move **L** (90° clockwise rotation of the orange side) the permuation matrix has the following values:

$$\begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 \\ 1 & 3 & 0 & 2 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 \end{pmatrix}$$

.



There are 2 ways to define the relation of the state of the cube and the permutation matrix. The matrix above describes the new position of each edge. The edge 0 (orange-yellow) is now at postion 1, where the orange-blue edge belongs. The edge 1 is now positoned where edge 3 would be in a solved cube.

The other interpretation would be the inverse of this Permuation.

$$\begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 \\ 2 & 0 & 3 & 1 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 \end{pmatrix}$$

.

Here the bottom row describes which edge is now at this position. So At position 0 there is edge 2, at position 1 there is edge 0. We will use the first interpretation as it makes composition of functions more intuitive.

The function $E_{47174400}$ represents permuation after the Move **L**. The lexicographic index of any permuation $\pi : \{0, ..., N-1\} \to \{0, ..., N-1\}$ of N elments can be computed by the following formular:

$$\sum_{i=0}^{N-1} \left((N-1-i)! \cdot \sum_{j=i+1}^{N-1} \begin{cases} 1 \text{ if } \pi(i) > \pi(j) \\ 0 \text{ otherwise} \end{cases} \right)$$

2.2 Edge orientation

Every edge has 2 possible orientations. To represent all edge orientations, 12 bits are needed which are stored into a single 16 bit integer. To make the edge orientation independent from the edge permuation state the i-th value in the bit vector corrosponds to the orientation of the edge at position i. We write an edge orientation similar to a permuation with a 2 row matrix.

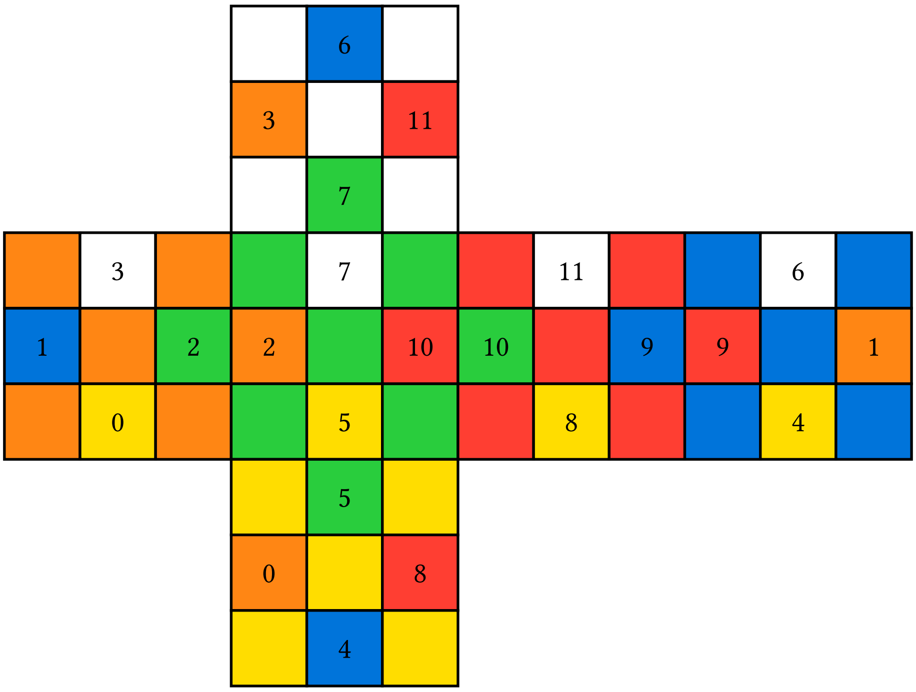
$$\begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

.

A 0 would corrospond to an oriented edge and a 1 to a flipped edge. The superflip has all edges not oriented but in the right position.

$$\begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

.



It is trivial to check whether an edge is oriented or not if it is at the original position. In any other position we have to define a logic that determines whether the edge is oriented or flipped. We choose one of the 3 main axis we base our orientation on. If we choose the Z-axis we can define the orientation of an edge as follows:

- Move the edge i to position i by without using the moves **F**, **F'**, **B** or **B'** (90° rotation on the Z-axis).
- Check whether the edge is oriented.

In practice this can be checked faster by choosing a tile for a certain edge and storing all faces this tile can be on by using our restricted set of moves. The orientation can be defined similary for the other 2 axis but we will use the Z-Axis as the base for our orientations.

2.3 Corner Permuation

The position of all corners is a permuation of 8 elements. We need $\lceil \log_2(8!) \rceil = 16$ bits to