数式処理を用いたルービックキューブの素数位数操作の探求

藤本, 光史
福岡教育大学

泊 , 昌孝
日本大学文理学部

出版情報：COE Lecture Note. 49, pp.69-77, 2013-08-09. 九州大学マス・フォア・インダストリ研究所
バージョン：
権利関係：

数式処理を用いたルービックキューブの素数位数操作の探求

A hunting of operations with prime order on Rubik's Cube using computer algebra

藤本 光史
Mitsushi Fujimoto
福岡教育大学
Fukuoka University of Education *

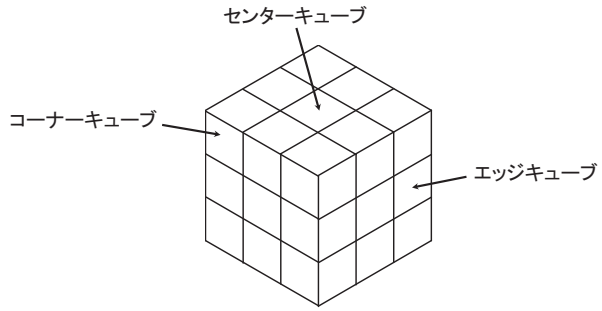
泊 昌孝
Masataka Tomari
日本大学文理学部
Nihon University †

Abstract

Rubik's cube is a suitable teaching material which connects puzzles to mathematics. In particular, every student will be surprised the phenomenon which any sequence of moves returns the Rubik's cube to the original position by repeating the operations. This phenomenon is able to be explained by group theory. In this article, we would like to report a result for a hunting of operations with prime order on Rubik's Cube using computer algebra systems.

1 はじめに

ルービックキューブは、ハンガリーの建築学者 Ernö Rubik によって 1974 年に考案された立方体パズルである。3 × 3 × 3 のタイプが「ルービックキューブ」として良く知られているが、この他に 2 × 2 × 2（ポケットキューブ）、4 × 4 × 4（ルービックリベンジ）、5 × 5 × 5（プロフェッサーズキューブ）などのタイプがある。3 × 3 × 3 のタイプは、8 個のコーナーキューブと、12 個のエッジキューブと、6 個のセンターキューブから成り、各面を回転させることで様々な模様を作ることができる。

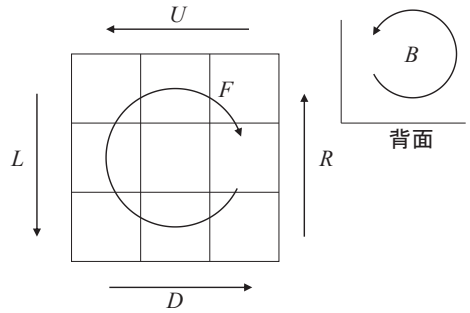


ルービックキューブは「バラバラになった状態から 6 面を揃える」ことが基本的な遊び方であり、6 面が簡単に揃えられるようになれば、「様々な模様を作る」という応用的な遊び方もできる。しかし、ここでは 3 × 3 × 3 のタイプのルービックキューブを使った「**同一の操作を繰り返すと必ず元に戻る**」という現象に着目する。

*fujimoto@fue.ac.jp
†tomari@math.chs.nihon-u.ac.jp

2　ルービックキューブの操作表現

ルービックキューブの左側面、右側面、背面、前面、下面、上面を時計回りに 90 度回転する操作をそれぞれ L, R, B, F, D, U で表す。



ルービックキューブのあらゆる操作は、この L, R, B, F, D, U とそれらの逆の積として表現できる。

例 1

$R'UL U'RUL'U'$ は「上面の右奥・右前・左前のコーナーキューブをこの順で置換する操作」である。ただし、R' は操作 R の逆（つまり、右側面を反時計回りに 90 度回転）を表す。

ルービックキューブのセンターキューブは、L, R, B, F, D, U の操作で回転するだけでその位置は変化しない。そこで、センターキューブを除いて以下の図のように 1 から 48 の番号を付ける。

```
                +--------------+
                |  1    2    3 |
                |  4   Up    5 |
                |  6    7    8 |
+---------------+---------------+---------------+---------------+
|  9   10   11  | 17   18   19  | 25   26   27  | 33   34   35  |
| 12  Left  13  | 20  Front 21  | 28  Right 29  | 36  Back  37  |
| 14   15   16  | 22   23   24  | 30   31   32  | 38   39   40  |
+---------------+---------------+---------------+---------------+
                | 41   42   43  |
                | 44  Down  45  |
                | 46   47   48  |
                +--------------+
```

この番号を用いて、L, R, B, F, D, U の各操作は次のような巡回置換の積で表現できる。

$$
\begin{aligned}
L &= (9,11,16,14)(10,13,15,12)(1,17,41,40)(4,20,44,37)(6,22,46,35)\\
R &= (25,27,32,30)(26,29,31,28)(3,38,43,19)(5,36,45,21)(8,33,48,24)\\
B &= (33,35,40,38)(34,37,39,36)(3,9,46,32)(2,12,47,29)(1,14,48,27)\\
F &= (17,19,24,22)(18,21,23,20)(6,25,43,16)(7,28,42,13)(8,30,41,11)\\
D &= (41,43,48,46)(42,45,47,44)(14,22,30,38)(15,23,31,39)(16,24,32,40)\\
U &= (1,3,8,6)(2,5,7,4)(9,33,25,17)(10,34,26,18)(11,35,27,19)
\end{aligned}
$$

3　ルービックキューブと群

前節の結果により、ルービックキューブの操作の全体は、L, R, B, F, D, U を生成元とする置換群（48次対称群 S_{48} の部分群）となる。数式処理システム GAP [1] では、次のように簡単にルービックキューブ群を定義できる。

```
gap> L := (9, 11, 16, 14)(10, 13, 15, 12)(1, 17, 41, 40)(4, 20, 44, 37)
(6, 22, 46, 35);;
gap> R := (25, 27, 32, 30)(26, 29, 31, 28)(3, 38, 43, 19)(5, 36, 45, 21)
(8, 33, 48, 24);;
gap> B := (33, 35, 40, 38)(34, 37, 39, 36)(3, 9, 46, 32)(2, 12, 47, 29)
(1, 14, 48, 27);;
gap> F := (17, 19, 24, 22)(18, 21, 23, 20)(6, 25, 43, 16)(7, 28, 42, 13)
(8, 30, 41, 11);;
gap> D := (41, 43, 48,46)(42, 45, 47, 44)(14, 22, 30, 38)(15, 23, 31, 39)
(16, 24, 32, 40);;
gap> U := (1, 3, 8, 6)(2, 5, 7, 4)(9, 33, 25, 17)(10, 34, 26, 18)
(11, 35, 27, 19);;
gap> Cube := Group(L,R,B,F,D,U);
```

次の定理は、群論を学んだことがある人には馴染み深いものであろう。

定理 1 (Lagrange)
G:有限群, $H \leq G \Rightarrow |H|$ は $|G|$ の約数

$Cube$ をルービックキューブ群、s を L, R, B, F, D, U とその逆の積で表した操作とする。上の Lagrange の定理により、$|\langle s \rangle|$ は $|Cube|$ の約数となる。すなわち、操作 s は有限の位数 d を持つ。故に、操作 s を d 回繰り返すと $s^d = e$ により元の状態に戻る。

例 2
操作 $R'\,L\,U\,D$ は何回繰り返すと元に戻るか、*GAP* で位数を求めてみよう。

```
gap> Order(R^-1*L*U*D);
12
```

この結果から、操作 $R'\,L\,U\,D$ のルービックキューブ群における位数は *12* であり、この操作を *12* 回繰り返すと元に戻ることがわかる。

4　素数位数問題

前節で「同一の操作を繰り返すと必ず元に戻る」理由が群論の初等的な結果から証明されたが、この現象に気が付いた者は、次の疑問を抱くことだろう。

問題 1 (最大位数問題)
ルービックキューブ群における元の最大位数を求めよ。また、最大位数を持つ元はどのようなものか？

これは「どんな出鱈目な操作を行っても、それを最悪何回繰り返せば元に戻るか？」という問題と見ることもできる。この問題に対する解答は、1981 年に発行された書籍 [2] で与えられた。ルービックキューブ群における元の最大位数は $1260 = 2^2 \cdot 3^2 \cdot 5 \cdot 7$ であり、その操作は $R\,F^2\,B'\,U\,B'$ である。GAP で位数を計算すると、確かに 1260 であることがわかる。このような短い操作 (6 操作) で最大の位数になるという事実には驚かされる。

また、ルービックキューブ群の位数は、GAP で簡単に確認できる。

```
gap> Size(Cube);
43252003274489856000
```

$43252003274489856000 = 2^{27} \cdot 3^{14} \cdot 5^3 \cdot 7^2 \cdot 11$ であり、このことから次の問題が考えられる。

問題 2 (素数位数問題)
ルービックキューブの素数位数を持つ操作を求めよ。

ルービックキューブ群の位数の素因数分解から、この問題は「位数 2, 3, 5, 7, 11 の操作を求めよ」ということになる。素数位数を持つ操作の存在性は、以下の有名な定理から保証される。

定理 2 (Cauchy)
G:有限群, $|G| = pk$ (p は素数) $\Rightarrow$ G は位数 p の元を持つ

5 位数 2 と 3 の操作

ルービックキューブの位数 2 の操作は、キュービストと呼ばれるルービックキューブ愛好者なら誰でも知っている。以下にそのいくつかを紹介する。

- 最短操作 R^2
- 格子模様 $L^2\,R^2\,B^2\,F^2\,D^2\,U^2$
- Rubik's Maneuver（上面の 2 個のエッジキューブを位置を変えずに 180 度入れ替える操作）
  $R'\,L\,F'\,R'\,L\,D'\,R'\,L\,B'^2\,R\,L'\,D'\,R\,L'\,F'\,R\,L'\,U'^2$
- Superflip（12 個の全エッジキューブを位置を変えずに 180 度入れ替える操作）
  $F\,R\,F'\,R\,F\,B\,L'\,U\,R\,L\,B\,U\,F'\,D^2\,R\,D'\,B\,U'\,L\,D'\,F\,D'\,B'$

次の位数 3 の操作も、バラバラの状態から 6 面を揃える際に用いられる技の中に現れる。

- 上面の右奥・右前・左前のコーナーキューブの位置をこの順で置換する操作
  $R'\,U\,L\,U'\,R\,U\,L'\,U'$
- 上面の右奥・右前のコーナーキューブを位置はそのままで向きだけを順に置換する操作
  $R'\,U^2\,R\,U\,R'\,U\,R\,L\,U'^2\,L'\,U'\,L\,U'\,L'$

6　位数 11 の操作の探求

前節で見たように、6 面を揃える際に用いられる技の中には、位数 2 や位数 3 の操作が数多く見られる。しかし、位数 5, 7, 11 の操作は、6 面を揃える際には通常使用されない。それは、位数 5, 7, 11 の操作もいくつかの小キューブの巡回置換（または巡回置換の積）であり、ルービックキューブのプレーヤーがこれらの操作を用いて目的の配置を得るには、位数 2 や 3 の操作と比較して繰り返す回数が多くなってしまうためである。

つまり、位数 5, 7, 11 の操作はキュービストが精通していない操作と言える。これらの操作を数式処理システムの GAP/Magma/Sage を用いて見つけたい。

6.1　GAP の利用

有限群論で最も有名な定理は、次の Sylow の定理であろう。

定理 3 (Sylow)
G:有限群, $|G| = p^m k$ (p は素数, p と k は互いに素) $\Rightarrow$ G は位数 p^m の部分群を持つ

GAP は Sylow 部分群を求めることができる。

```
gap> SylowSubgroup(Cube,11);
Group([ (4,7,12,39,44,21,26,31,42,20,29)(5,45,23,13,36,10,18,37,47,15,28) ])
```

このように、ルービックキューブ群の 11-Sylow 部分群を得た。この部分群の生成元

$$s = (4, 7, 12, 39, 44, 21, 26, 31, 42, 20, 29)(5, 45, 23, 13, 36, 10, 18, 37, 47, 15, 28)$$

は位数 11 の元であり、この置換 s を L, R, B, F, D, U を用いて表現することが次のステップである。

このためには、操作 L, R, B, F, D, U を生成元とする自由群からルービックキューブ群 $Cube$ への準同型写像

$$\phi : \langle L, R, B, F, D, U \rangle \ni word \rightarrow permutation \in Cube$$

による置換 s の逆像（の一つ）を求めればよい。これを求める関数を GAP のプログラミング言語で書くと次のようになる。

```
GetWordOfElements:=function(G,GenName,x)
  local gen,F,hom;
  F:=FreeGroup(GenName);
  gen:=GeneratorsOfGroup(G);
  hom:=GroupHomomorphismByImages(F,G,GeneratorsOfGroup(F),gen);
  return PreImagesRepresentative(hom,x);
end;
```

この関数を用いれば、どんなルービックキューブの状態からでも 6 面を揃える操作を求めることができる[1)]。実際、藤本はこの関数を利用したルービックキューブ解法表示ソフト [3] を開発した。

この関数を用いて、上で求めた位数 11 の元 s から L, R, B, F, D, U による操作を求める。

[1)] ただし、この関数から得られるルービックキューブの解法が最小手数であることは保証されない。

```
gap> GetWordOfElements(Cube,["L","R","B","F","D","U"],
(4,7,12,39,44,21,26,31,42,20,29)(5,45,23,13,36,10,18,37,47,15,28));
L^-1*U*B^-1*U^-1*B^-1*F^-1*U^-1*B*L^2*D^-1*R*F^-1*R^-1*D*L^2*R^-1*D^-1*R*D*F^-1
*D^-1*U*L^-1*U^-1*F^-1*L^-1*F*U*F*L*F^-1*U^-1*F*D*F^2*D^-1*F^-1*L*D^-1*L^-1*D*F
*L*F^-2*D*F*D^-1*F*L*F^-1*L^2*F^-1*L^-1*F*L*F*U^-1*F^-1*U*F*L*F^-1*L*D^-1*L*D
*L^-2*F*L^-1*U*L*U^-1*L^-1*F^-2*D^-1*L^-1*D*L*F*L^-1*D^-1*L^-1*B^-1*L*B*D*L*F*U
*L*U^-1*L^-1*F^-1
Length(last);
100
```

6.2 Magma の利用

GAP を利用して求めた位数 11 の操作は 100 手もあり、とても覚えられるものではない。この結果は、GAP の組み込み関数 PreImagesRepresentative が出力したものであり、GAP のソースを読んでそのアルゴリズムを解析するのは最後の手段にして、別の数式処理システム Magma [4] を試してみることにしたい。

Magma 上で、ルービックキューブ群は次のようにして定義できる。

```
magma> s48:=Sym(48);
magma> L := s48!(9, 11, 16, 14)(10, 13, 15, 12)(1, 17, 41, 40)
(4, 20, 44, 37)(6, 22, 46, 35);
magma> R := s48!(25, 27, 32, 30)(26, 29, 31, 28)(3, 38, 43, 19)
(5, 36, 45, 21)(8, 33, 48, 24);
magma> B := s48!(33, 35, 40, 38)(34, 37, 39, 36)(3, 9, 46, 32)
(2, 12, 47, 29)(1, 14, 48, 27);
magma> F := s48!(17, 19, 24, 22)(18, 21, 23, 20)(6, 25, 43, 16)
(7, 28, 42, 13)(8, 30, 41, 11);
magma> D := s48!(41, 43, 48,46)(42, 45, 47, 44)(14, 22, 30, 38)
(15, 23, 31, 39)(16, 24, 32, 40);
magma> U := s48!(1, 3, 8, 6)(2, 5, 7, 4)(9, 33, 25, 17)
(10, 34, 26, 18)(11, 35, 27, 19);
magma> Cube:=PermutationGroup<48|L,R,B,F,D,U>;
```

さらに、準同型 $\phi : \langle L, R, B, F, D, U\rangle \ni word \to permutation \in Cube$ を定義し、置換

$$s = (4,7,12,39,44,21,26,31,42,20,29)(5,45,23,13,36,10,18,37,47,15,28)$$

の ϕ による逆像として、位数 11 の操作を求める。

```
magma> w<L,R,B,F,D,U>:=FreeGroup(6);
magma> images := [w.i -> Cube.i : i in [1..6] ];
magma> phi:=hom< w -> Cube | images >;
magma> s:=s48!(4,7,12,39,44,21,26,31,42,20,29)(5,45,23,13,36,10,18,37,47,15,28);
magma> s @@ phi;
```

しかし、得られた結果は 1600 行を超える異常に長い操作になってしまった。この原因は、自由群 $F_6 = \langle L, R, B, F, D, U\rangle$ に対して関係式を与えていないためだと思われる。そこで、ルービックキューブ群の

L, R, B, F, D, U を用いた生成元と関係式による群表示について過去の研究成果を調査したが、5 個の生成元と 44 個の関係式による群表示 [5] しか見つけることができず断念した。

6.3 Sage の利用

ここでは、GAP で得られた位数 11 の 100 手から成る操作を「簡約」するアプローチについて述べる。次の問題は、ルービックキューブの最小手数問題として有名である。

問題 3

$3 \times 3 \times 3$ のルービックキューブを解くために必要な手数の上限を求めよ。

ここでの手数は face turn metric であり、連続する同じ操作（つまり、180 回転する操作）は 1 手とカウントするものである。この上限の手数は God's number と呼ばれており、2010 年に Tomas Rokicki, Herbert Kociemba, Morley Davidson, John Dethridge によって、20 手であることが示された。

数式処理システム Sage [6] は、Michael Reid による Optimal Solver アルゴリズム [7] が実装されており、与えられた操作の簡約が可能である。ただし、この実装は face turn metric ではなく、quarter turn metric（90 度回転を 1 手とカウントする）での最適解を求めるものとなっている。Sage を用いて、GAP が求めた位数 11 の 100 手の操作の簡約は以下のように行う。

```
sage: rubik = CubeGroup()
sage: s = rubik.faces("L^-1*U*B^-1*U^-1*B^-1*F^-1*U^-1*B*L^2*D^-1*R*F^-1
*R^-1*D*L^2*R^-1*D^-1*R*D*F^-1*D^-1*U*L^-1*U^-1*F^-1*L^-1*F*U*F*L*F^-1
*U^-1*F*D*F^2*D^-1*F^-1*L*D^-1*L^-1*D*F*L*F^-2*D*F*D^-1*F*L*F^-1*L^2*F^-1
*L^-1*F*L*F*U^-1*F^-1*U*F*L*F^-1*L*D^-1*L*D*L^-2*F*L^-1*U*L*U^-1*L^-1*F^-2
*D^-1*L^-1*D*L*F*L^-1*D^-1*L^-1*B^-1*L*B*D*L*F*U*L*U^-1*L^-1*F^-1")
sage: ans = rubik.solve(s, algorithm='optimal')
sage: print ans
F  R' L  F  U  D  L  B  U' F' U' F' D  F' B  L  U' L' U  R
```

以上によって、位数 11 の 20 手の操作を得ることができた。

7 位数 5 と 7 の操作

前節で得られた手法を以下の手順で位数 5,7 にも適用する。

1. 【GAP】ルービックキューブ群を定義し、SylowSubgroup 関数を用いて、5-Sylow 部分群と 7-Sylow 部分群を求め、その生成元から位数 5,7 の置換表現を得る。
2. 【GAP】GetWordOfElements 関数を用いて、位数 5,7 の置換表現から対応する L, R, B, F, D, U による操作を求める。
3. 【Sage】CubeGroup().solve 関数を用いて、操作手数の簡約を行う。

これによって、次の操作が得られた。

- （位数 5）12 個のエッジキューブのうち 7 個を固定し、残り 5 個の巡回置換を行う操作
  $D\,F'\,R\,U'\,F'\,L'\,R\,U\,L\,R'\,U\,R'\,F\,D'$

- （位数 5）8 個のコーナーキューブのうち 3 個を固定し、残りの 5 個の巡回置換を行う操作
  $RU\,F\,U'\,L\,U\,F'\,U'\,R'\,F\,L'\,F'$
- （位数 7）8 個のコーナーキューブのうち 1 個を固定し、残りの 7 個の巡回置換を行う操作
  $R'\,U\,R\,D\,F^2\,R'\,D\,U'\,R\,U'\,R\,D'\,R\,D'\,U\,F^2$

8 記憶可能な操作の探求

GAP と Sage を駆使して、ルービックキューブの素数位数 2,3,5,7,11 の操作を求めることができた。これで問題 2 は解決である。素数位数の操作を求めることができたので、これを実演したいと考えるのは自然なことであろう。そこで実際のルービックキューブを用いて練習してみたが、何度やっても操作を記憶することができない。ルービックキューブの F や B の操作は、手の移動距離が大きく、できるだけ避けたい操作である。それで、求めた素数位数の操作を F や B をなるべく使用しないように置換してみたが、それでも記憶することは不可能であった。そして、我々は次の問題を考えることにした。

問題 4
ルービックキューブの記憶可能な単純な素数位数操作を求めよ。

位数 11 の操作について考えることにする。前々節で得られた手数 20 の位数 11 の操作は、12 個のエッジキューブのうち 1 個を固定し、残りの 11 個の巡回置換を引き起こす。下面と背面にまたがるエッジキューブを固定することにすると、それを動かさない操作は、U, F, L, R の 4 操作である。よって、位数 11 の操作はこの 4 操作の組み合わせで作ることが自然と考えられる。後は、4 操作の記憶しやすい組み合わせ x を作り、その位数を GAP で計算する。位数が 11 の m 倍になったとき、x^m が求める位数 11 の操作になる。この手順をまとめると次のようになる。

1. 位数 p の操作がルービックキューブ上のどのような巡回置換を引き起こすか考える。
2. 使用する操作を限定する。
3. 限定した操作から記憶しやすい単純な組み合わせを考える。
4. その組み合わせの位数を Gap で計算する。
5. 位数が p の m 倍になるものを見つける。

この手順の $3 \to 4 \to 3 \to 4 \to \cdots$ の繰り返しにより、

```
gap> Order((U*R^-1)^3*(L*F^-1)^3);
33
gap> Order(L*R^-1*F*U^-1);
44
gap> Order(R^-1*U*F^-1*L);
33
```

という組み合わせを見つけることができた。この最後の結果から、位数 11 の記憶可能な単純な操作（手数 12）が得られた。また、同様の手法により、位数 7,5 の記憶可能な単純な操作も得られた。以下に、これらの結果をまとめておく。

- 位数 11 の操作：$(R'\,U\,F'\,L)^3$
- 位数 7 の操作：$(D^2\,R)^2\,(U^2\,L)^2$
- 位数 5 の操作：$R'\,U\,R\,U$

9 おわりに

本稿ではルービックキューブの素数位数操作を数式処理システムを利用して求める手法について解説した。ここで取り扱った問題は、大学での代数学の講義や高校での出前授業などの教材として用いることが可能である。実際、我々は所属する大学での講義や高校での出前講演で使用している。ルービックキューブは、数学とパズルを結びつける格好の教材の一つと言えよう。

最後に、二つの発展的な問題を残してこの稿を終えることとする。

問題 5

ルービックキューブ群の元が取り得るすべての位数を求め、その位数に対応する最短の操作を求めよ。

問題 6

人間が記憶可能なルービックキューブの操作を自動生成することは可能か。

参 考 文 献

[1] GAP – Groups, Algorithms, Programming – a System for Computational Discrete Algebra, http://www.gap-system.org

[2] David Singmaster, Notes on Rubik's Magic Cube, Enslow Pub Inc, 1981.

[3] 田崎拓馬, 藤本光史, GAP を用いた Rubik's Cube 解法表示ソフトについて, 京都大学数理解析研究所講究録 1652, 「Computer Algebra – Design of Algorithms, Implementations and Applications」(2009) 125–131.

[4] Wieb Bosma, John Cannon, and Catherine Playoust, The Magma algebra system. I. The user language, J. Symbolic Comput., 24 (1997) 235–265. http://magma.maths.usyd.edu.au/magma/

[5] Dan Hoey, Presenting Rubik's Cube, Cube-lovers mailing list, http://www.math.rwth-aachen.de/~Martin.Schoenert/Cube-Lovers/Dan_Hoey__Presenting_Rubik%27s_Cube.html

[6] William A. Stein, et al., Sage Mathematics Software (Version 5.9), The Sage Development Team, 2013, http://www.sagemath.org

[7] Michael Reid, Optimal Rubik's cube solver, http://math.cos.ucf.edu/~reid/Rubik/optimal_solver.html