Puzzles of colored cubes

Wim H. Hesselink
whh595, January 22, 2023

Bernoulli Institute, University of Groningen, The Netherlands

1 Introduction

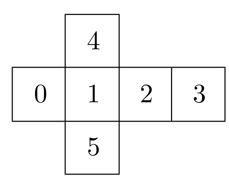
Recently, I read in a diary of mine an item of more than 50 years ago about solving a puzzle with four colored cubes. I still had the cubes, but I did not remember the puzzle. Today, we can search the internet. I found the puzzle in the wikipedia item "instant_insanity", and read that the puzzle with *four colored cubes* was first patented by F. A. Schossow in 1900 and marketed as the Katzenjammer puzzle. Let me describe it in the generalized form where the number *four* is generalized to a number H.

Each puzzle consists of H cubes with faces colored with H colors. The objective of the puzzle is to stack these cubes in a column so that each side of the stack (left, front, right, and back) shows each of the H colors.

2 Representation

One needs exhaustive search to solve this problem. Deviating from wikipedia, I do not reduce the problem to a graph problem, but instead argue about the discrete orientations of the cubes.

The first task is to represent a colored cube. Every cube has 6 faces, numbered 0, ..., 5. The horizontal faces left, front, right, and back get the indices 0, 1, 2, 3, respectively. The specification of the required column *ignores* the two vertical faces top and bottom. These faces get the remaining indices 4 and 5, respectively.



A colored cube is thus represented by an sequence of six colors. More precisely, this is an *oriented* colored cube, because a cube can be rotated and then its faces are permuted.

Assuming $H \leq 26$, the colors are represented by subsequent capital characters A, B, ... Let Col be the set of these H characters. So, the oriented colored cubes are the elements of $Col[6]$. As the colors are characters, the oriented colored cubes are strings of length 6. In particular, my set of four colored cubes can be represented by the strings $ACBDCC$, $ACDBDC$, $ABBCDD$, $ABBDCD$, if A stands for green, B for yellow, C for red, and D for blue. In general, every column of oriented cubes is an sequence of H oriented cubes, and thus an element $x \in Col[H][6]$. Therefore, my initial column is the sequence

$$(ACBDCC, ACDBDC, ABBCDD, ABBDCD).$$

A column x has two small faces at the top and the bottom, and four "tall" faces of length H, with indices 0, 1, 2, 3. For $0 \leq f < 4$, the tall face of index f has the contents $x[*, f] = \{x[j][f] \mid 0 \leq j < H\}$. Column x solves the problem iff $P_f(x) : x[*, f] = Col$ holds for all f with $0 \leq f < 4$. The specification of the puzzle is therefore the conjunction

$$P = P_0 \wedge P_1 \wedge P_2 \wedge P_3.$$

3 Solution

Each puzzle consists of a given initial column x, say of height H. The question is to find a column w, such that cube $w[j]$ is a rotation of cube $x[j]$ for each $j < H$, and that $P(w)$ holds, i.e., that the transformed column w satisfies predicate P.

As the order of the cubes in the column is irrelevant for predicate P, we can keep the order of the cubes constant.

As the specification ignores the faces 4 and 5 (top and bottom), we first consider all choices for the vertical orientations. Let a column x be called *acceptable* iff, for each color c, the total number of horizontal faces with color c equals 4:

$$Acc(x): \quad \forall\, c \in Col : \#\{(j, f) \mid j < H, f < 4 : x[j][f] = c\} = 4 \ ,$$

where $\#S$ denotes the number of elements of a set S. Indeed, predicate P implies Acc. More precisely, P is equivalent to the conjunction $Acc \wedge P_0 \wedge P_1 \wedge P_2$.

In order to discuss the rotations of the cubes, we fix a coordinate system with coordinates ξ_1, ξ_2, ξ_3, and corresponding standard basis vectors e_1, e_2, e_3. Let the solid cube be given by the inequalities $|\xi_1| \leq 1$, $|\xi_2| \leq 1$, $|\xi_3| \leq 1$. Let its faces be represented by the midpoints $\pm e_1$, $\pm e_2$, $\pm e_3$. Writing $F(i)$ for the face with index i as described above, we have

$$F(0) = -e_1,\ F(1) = -e_2,\ F(2) = e_1,\ F(3) = e_2,\ F(4) = e_3,\ F(5) = -e_3.$$

Every cube has three pairs of opposite faces. One of them must be chosen to be ignored. These pairs are permuted by the rotation R over the angle $2\pi/3$ given by

$$R(e_1) = e_2\ ,\ R(e_2) = e_3\ ,\ R(e_3) = e_1\ .$$

This rotation permutes the indices of the faces by

$$R(0, 1, 2, 3, 4, 5) = (1, 5, 3, 4, 2, 0)\ .$$

Using this rotation the initial column x can be transformed into the column y given by $y[j] = R^{e(j)}x[j]$ for all j with $0 \leq j < H$, for all sequences of H exponents $e(j)$ with $0 \leq e(j) < 3$. We keep the exponents $e(j) < 3$ because R^3 is the identity. Predicate Acc is used to filter the set of columns y, i.e., preserve columns y with $Acc(y)$ and dismiss the columns y that do not satisfy Acc. Let S_1 be the resulting filtered set.

In general, the columns $y \in S_1$ do not satisfy predicate P, but they can be rotated in such a way that all its vertical faces (top, bottom) remain vertical. The easiest way to do so is to use the horizontal rotation L over $\pi/2$ with $L(e_1) = e_2$, $L(e_2) = -e_1$, and $L(e_3) = e_3$. This rotation permutes the indices of the faces by

$$L(0, 1, 2, 3, 4, 5) = (1, 2, 3, 0, 4, 5)\ .$$

Note that L^4 is the identity. If rotation L is applied to all cubes of a column, solutions are preserved. We therefore decide to keep one of the cubes (the one with number $j = H - 1$) unchanged and to apply rotation L to the other cubes. For each $y \in S_1$, we thus get the columns z given by $z[j] = L^{f(j)}y[j]$ for all sequences of H exponents $f(j)$ with $0 \leq f(j) < 4$ and $f(H - 1) = 0$. The columns z are filtered by the predicate $P_0 \wedge P_2$, testing the correctness of the tall faces along the e_1-axis (with indices 0 and 2). Let S_2 be the resulting filtered set.

For the columns in S_2, the tall faces on the e_2-axis can be wrong. A third rotation K is needed, which preserves the faces 0 and 2, and swaps the faces 1 and 3, and also the faces 4 and 5. It is the rotation over π with $K(e_2) = -e_2$, $K(e_3) = -e_3$, $K(e_1) = e_1$. This rotation permutes the indices of the faces by

$$K(0, 1, 2, 3, 4, 5) = (0, 3, 2, 1, 5, 4)\ .$$

Note that K^2 is the identity. If rotation K is applied to all cubes of a column, solutions are preserved. As before, we keep one of the cubes (the one with number $j = H-1$) unchanged and apply rotation K to the other cubes. For each $z \in S_1$, we thus form the columns w given by $w[j] = K^{g(j)}z[j]$ for all sequences of H exponents $g(j)$ with $0 \leq g(j) < 2$ and $g(H-1) = 0$. The columns w are filtered by the predicate P_1.

The resulting filtered set S_3 is the set of solutions. Or rather, it is a set of solutions, and every orbit of solutions under the symmetry group of the cube contains one element of S_3.

4 Programming

As indicated, oriented colored cubes are represented by strings of type `char[]`. Three procedures are constructed to rotate them, for example

```
void rotR(char s[]);
```

applies rotation R to string `s`. Similarly, *rotL* and *rotK* apply the rotations L and K to `s`, respectively.

The predicates *Acc* and P_f is represented by the functions

```
bool accept(char x[][]) {
  int cnt[] = {0, ...};
  for (int j = 0 ; j < H ; j++)
    for (int f = 0 ; f < 4 ; f++)
      cnt[x[j][f] -'A']++;
  for (int j = 0 ; j < H ; j++)
    if (cnt[j] != 4) return false;
  return true;
}

bool pred(int f, char x[][]) {
  for (int j = 0 ; j < H ; j++)
    for (int k = j + 1 ; k < H ; k++)
      if (x[j][f] == x[k][f]) return false;
  return true;
}
```

In procedure `pred`, it is assumed that all colors of column `x` belong to the set *Col* of H colors.

The main search procedure is

```
void search1(char x[][]) {
  int j, e1[] = {0, ...};
  do {
    if (accept(x)) search2(x);
    j = 0; // determine successor
    while (e1[j] == 2) {
      rotR(x[j]);
      e1[j] = 0;
      j++;
    }
    if (j < H) rotR(x[j]);
    e1[j]++;
  } while (j < H);
}
```

Here `e1` is the sequence of exponents e. The exponents e are bounded by $e < 3$ because R^3 is the identity. The procedure is carefully created in such a way that the final value of column x is equal to the initial value. In this way, copying of columns is avoided. The body of the outer loop of the procedure first filters the current column with Acc and searches its refinements, and subsequently computes its successor. The columns accepted form the set S_1, and are subjected to

```
void search2(char x[][]) {
  int j, e2[] = {0, ...};
  do {
    if (pred(0, x) && pred(2, x))
      search3(x);
    j = 0; // determine successor
    while (e2[j] == 3) {
      rotL(x[j]);
      e2[j] = 0;
      j++;
    }
    if (j < H - 1) rotL(x[j]);
    e2[j]++;
  } while (j < H - 1);
}
```

Note that L^4 is the identity. Therefore, the exponents e are bounded by $e < 4$. On the other hand, index j remains $< H - 1$, so that the topmost cube is not rotated. The columns accepted by P_0 and P_2 form the set S_2, and are subjected to

```
void search3(char x[][]) {
  int j, e3[] = {0, ...};
  do {
    if (pred(1, x)) printstack(x);
    j = 0; // determine successor
    while (e3[j] == 1) {
      rotM(x[j]);
      e3[j] = 0;
      j++;
    }
    if (j < H - 1) rotM(x[j]);
    e3[j]++;
  } while (j < H - 1);
}
```

As rotation K^2 is the identity, the exponents e are now bounded by $e < 2$. As in `search2`, the index j remains bounded by $j < H - 1$ and the topmost cube is not rotated. The columns accepted by P_1 form the set S_3 of the principal solutions. Therefore, they are printed.

Example. For $H = 5$, the initial column

$$(AEDDEA, BBCBDC, ADBDCA, BDABAE, EEEBCD)$$

has the unique solution

$$(AEEDAD, DCCBBB, CBAADD, BADEAB\ , EDBCEE).$$

Remark. One can replace the test `accept(x)` by `true` in `search1` if one replaces the test `pred(1, x)` in `search3` by the conjunction of `pred(1, x)` and `pred(3, x)`. This gives a marginal improvement of the worst-case time complexity, but it is bad for the average time complexity. One can extend the program so that it also says how to rotate the cubes.