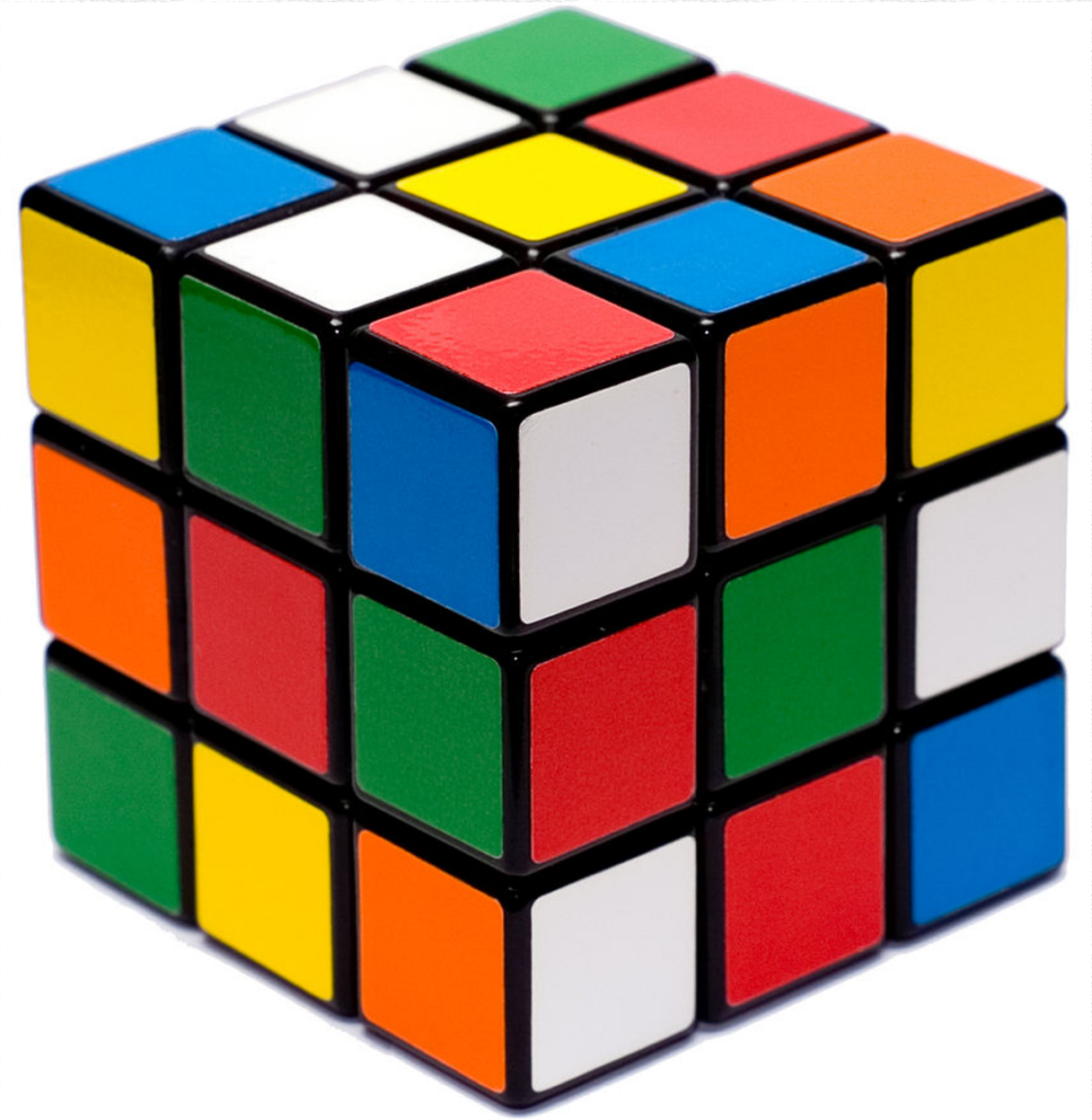
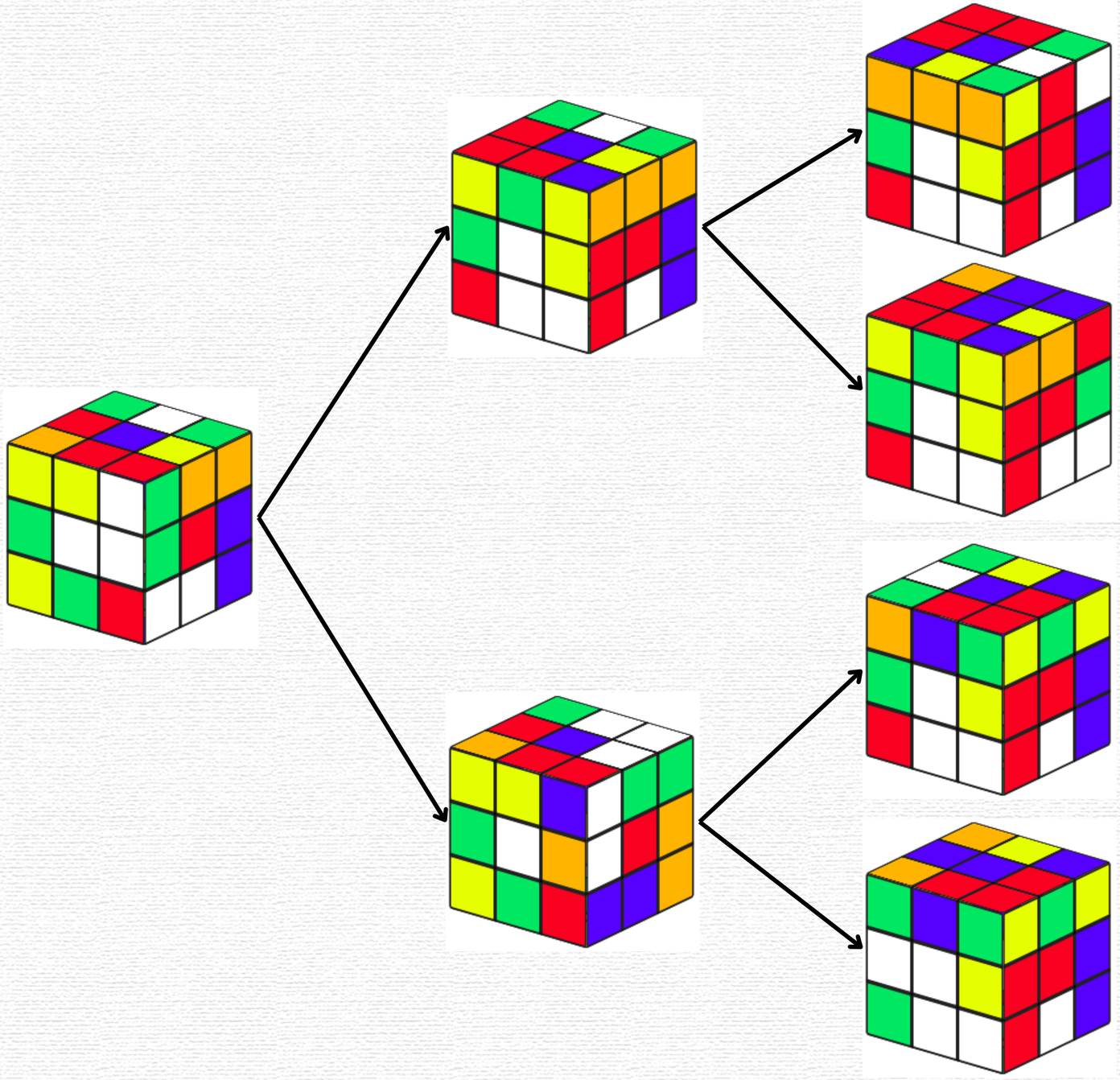
IPC 2023 - RUBIK'S CUBE DOMAIN

Integrantes:

Arthur José
Cauã Corrêa
Júlio César Costa
Pedro Sena

Cubo de Rubik (Cubo Mágico)





Solving the Rubik's Cube with a PDDL Planner

Bharath Muppasani, Vishal Pallagani, Kausik Lakkaraju,
Biplav Srivastava, Forest Agostinelli

AI Institute, University of South Carolina, SC, USA

AI INSTITUTE #AIISC
UNIVERSITY OF SOUTH CAROLINA

UNIVERSITY OF SOUTH CAROLINA

On Solving the Rubik's Cube with Domain-Independent Planners Using Standard Representations

Bharath Muppasani, Vishal Pallagani, Biplav Srivastava, Forest Agostinelli

AI Institute, University of South Carolina, Columbia, South Carolina, USA
{bharath@email., vishalp@email., biplav.s@, foresta@cse.}sc.edu

Abstract

Rubik's Cube (RC) is a well-known and computationally challenging puzzle that has motivated AI researchers to explore efficient alternative representations and problem-solving methods. The ideal situation for planning here is that a problem be solved optimally and efficiently represented in a standard notation using a general-purpose solver and heuristics. The fastest solver today for RC is DeepCubeA with a custom representation, and another approach is with Scorpion planner with State-Action-Space+ (SAS+) representation. In this paper, we present the first RC representation in the popular PDDL language so that the domain becomes more accessible to PDDL planners, competitions, and knowledge engineering tools, and is more human-readable. We then bridge across existing approaches and compare performance. We find that in one comparable experiment, DeepCubeA[1] solves all problems with varying complexities, albeit only 18% are optimal plans. For the same problem set, Scorpion with SAS+ representation and pattern database heuristics solves 61.50% problems, while FastDownward with PDDL representation and FF heuristic solves 56.50% problems, out of which all the plans generated were optimal. Our study provides valuable insights into the trade-offs between representational choice and plan optimality that can help researchers design future strategies for challenging domains combining general-purpose solving methods (planning, reinforcement learning), heuristics, and representations (standard or custom).

Introduction

The Rubik's Cube is a 3D puzzle game that has been widely popular since its invention in 1974. It has been a subject of interest for researchers in Artificial Intelligence (AI) due to its computational complexity and potential for developing efficient problem-solving algorithms. RC has motivated researchers to explore alternative representations that simplify the problem while preserving its complexity. Efficient algorithms have been developed to solve RC in the least number of moves, and they have been used in various applications, including robot manipulation, game theory, and machine learning. Therefore, in this paper, we aim to explore the different representations and algorithms to solve RC and evaluate their performance and effectiveness in solving this challenging puzzle.

Various solution approaches have been proposed RC including Reinforcement Learning (RL) and search. For instance, DeepCubeA (Agostinelli et al. 2019a) uses RL to learn policies for solving RC, where the cube state is represented by an array of numerical features. Although DeepCubeA is a domain-independent puzzle solver, it employs a custom representation for RC. On the other hand, Büchner et al. (2022) utilized SAS+ representation to model the RC problem in a finite domain representation, which enables standard general-purpose solvers like Scorpion to be used on the RC problem. Despite the success of these approaches, no prior work has explored the use of Planning Domain Definition Language (PDDL) to encode a 3x3x3 RC problem. While a previous study[2] has encoded a 2x2x2 RC problem using PDDL and solved it with a Fast-Forward planner, there exists no PDDL encoding for a 3x3x3 RC problem.

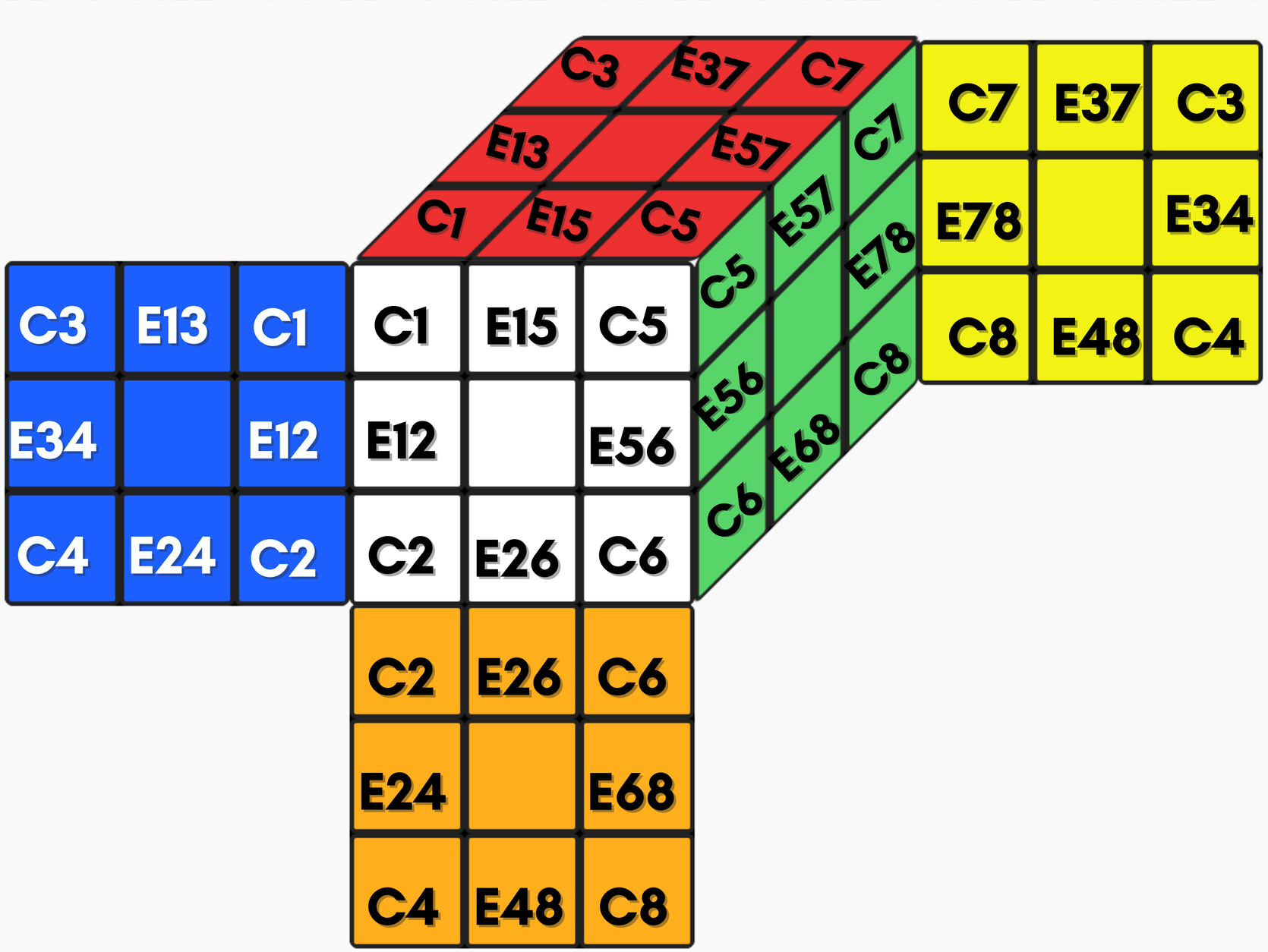
In this paper, we introduce a novel approach for representing RC in PDDL. We encode the initial state and goal state using a set of predicates, each of which specifies the color of a sticker on a particular cube piece or edge piece. We then define the actions that can be taken to manipulate the cube pieces and edges. Our PDDL representation enables us to model RC as a classical planning problem, which can be solved using off-the-shelf planning tools. To the best of our knowledge, this is the first attempt to represent RC formally using PDDL. We also evaluate the effectiveness of our approach by comparing it with other state-of-the-art representations in terms of the efficiency and effectiveness of problem-solving. Our major contributions are:

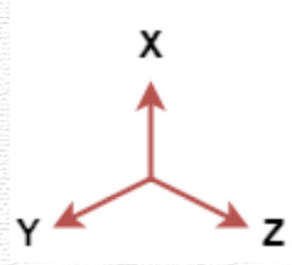
- We develop the first PDDL formulation for the 3x3x3 Rubik's Cube, which is a novel and significant contribution to the existing literature. This PDDL formulation will enable the use of standard PDDL planners for solving Rubik's Cube problems, which was not previously possible.
- We bridge across hither-to incomparable RC solving approaches, compare their performance and draw insights from results to facilitate new research.

[1]DeepCubeA trained with 12 RC actions

[2]https://wu-kan.cn/2019/11/21/Planning-and-Uncertainty/

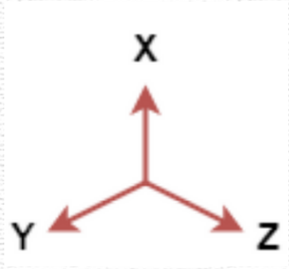
Predicados


C3 E37 C7
E13 E57
C1 E15 C5
C7 E37 C3
E78 E34
C8 E48 C4
C5 E57 C7
E56 E78
C6 E68 C8
C3 E13 C1
E34 E12
C4 E24 C2
C1 E15 C5
E12 E56
C2 E26 C6
C2 E26 C6
E24 E68
C4 E48 C8


X
Y
Z

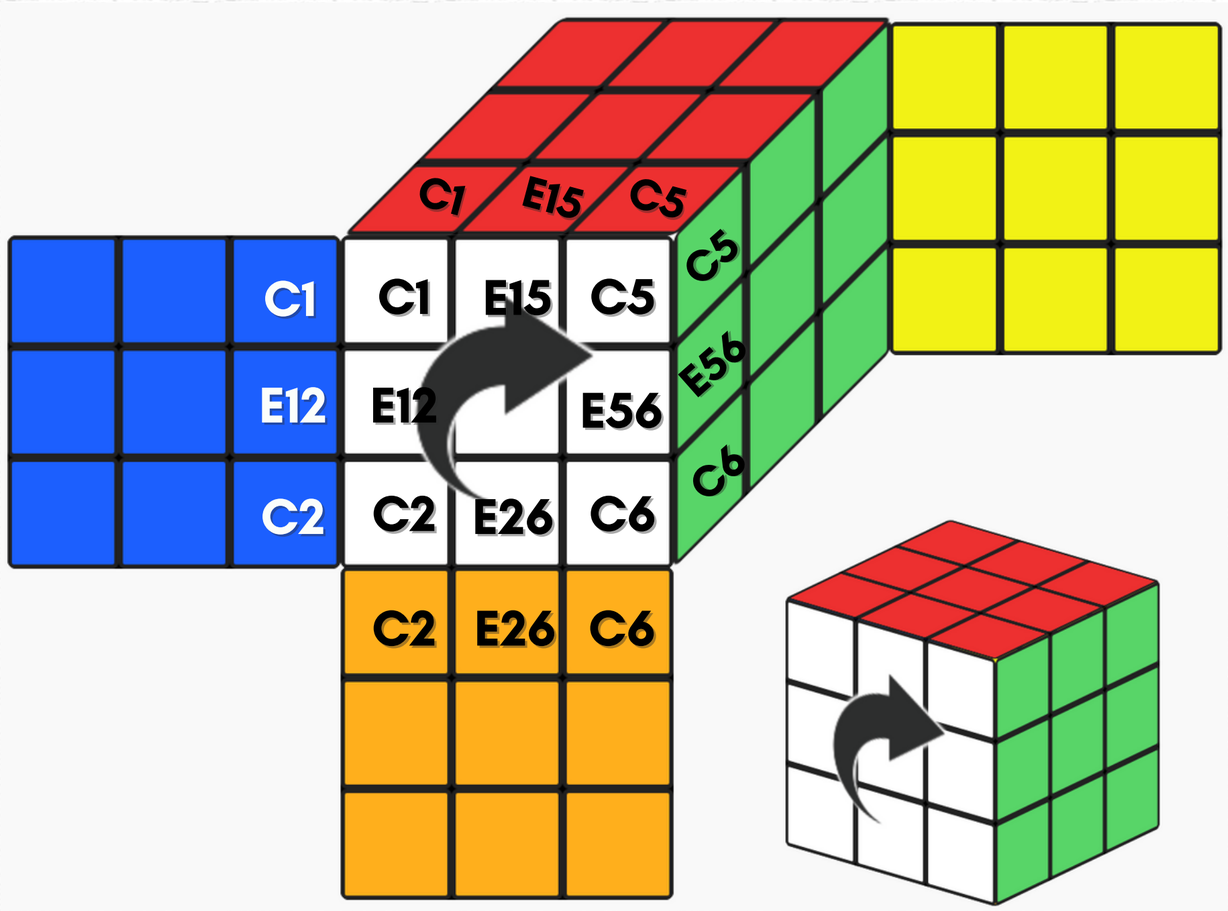
Predicados

```
;; 12-action variant
(define
    (domain rubiks-cube)
    (:requirements :adl)
    (:predicates
        (cube1 ?x ?y ?z)
        (cube2 ?x ?y ?z)
        (cube3 ?x ?y ?z)
        (cube4 ?x ?y ?z)
        (cube5 ?x ?y ?z)
        (cube6 ?x ?y ?z)
        (cube7 ?x ?y ?z)
        (cube8 ?x ?y ?z)
        (edge12 ?y ?z)
        (edge13 ?x ?z)
        (edge15 ?x ?y)
        (edge26 ?x ?y)
        (edge24 ?x ?z)
        (edge48 ?x ?y)
        (edge34 ?y ?z)
        (edge37 ?x ?y)
        (edge56 ?y ?z)
        (edge57 ?x ?z)
        (edge68 ?x ?z)
        (edge78 ?y ?z)
    )
```

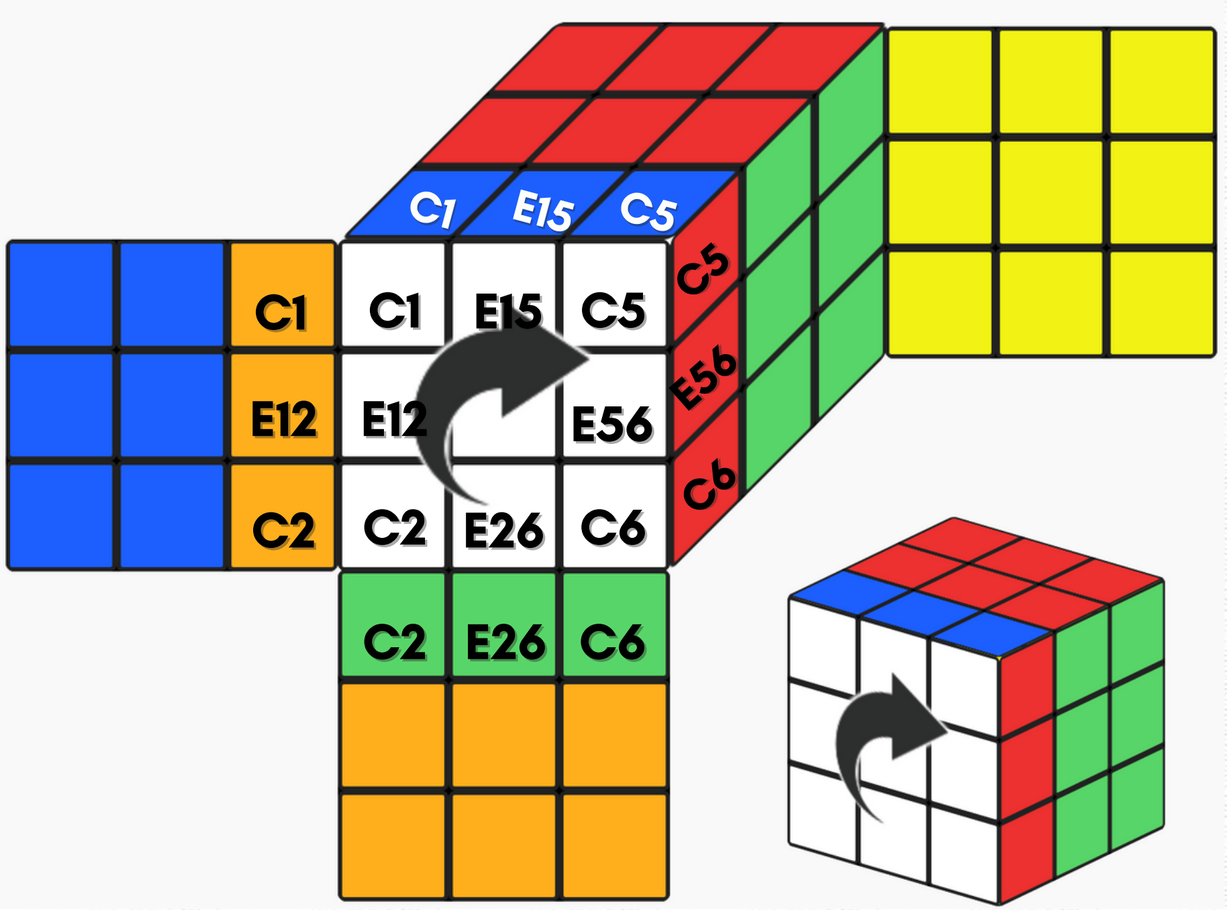


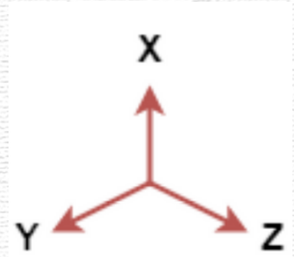
Ação Front

Antes

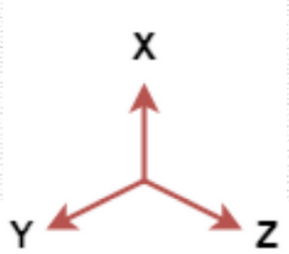


Depois



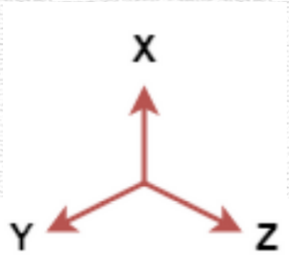


Ação Front




```
175         (:action F
176          :parameters ()
177          :precondition (and)
178          :effect
179             (and
180                 (forall (?x ?y ?z) (when (cube1 ?x ?y ?z) (and (not (cube1 ?x ?y ?z)) (cube5 ?z ?y ?x) )))
181                 (forall (?x ?y ?z) (when (cube5 ?x ?y ?z) (and (not (cube5 ?x ?y ?z)) (cube6 ?z ?y ?x) )))
182                 (forall (?x ?y ?z) (when (cube6 ?x ?y ?z) (and (not (cube6 ?x ?y ?z)) (cube2 ?z ?y ?x) )))
183                 (forall (?x ?y ?z) (when (cube2 ?x ?y ?z) (and (not (cube2 ?x ?y ?z)) (cube1 ?z ?y ?x) )))
184
185                 (forall (?x ?y) (when (edge15 ?x ?y) (and (not (edge15 ?x ?y)) (edge56 ?y ?x))))
186                 (forall (?y ?z) (when (edge56 ?y ?z) (and (not (edge56 ?y ?z)) (edge26 ?z ?y))))
187                 (forall (?x ?y) (when (edge26 ?x ?y) (and (not (edge26 ?x ?y)) (edge12 ?y ?x))))
188                 (forall (?y ?z) (when (edge12 ?y ?z) (and (not (edge12 ?y ?z)) (edge15 ?z ?y))))
189             )
190         )
```

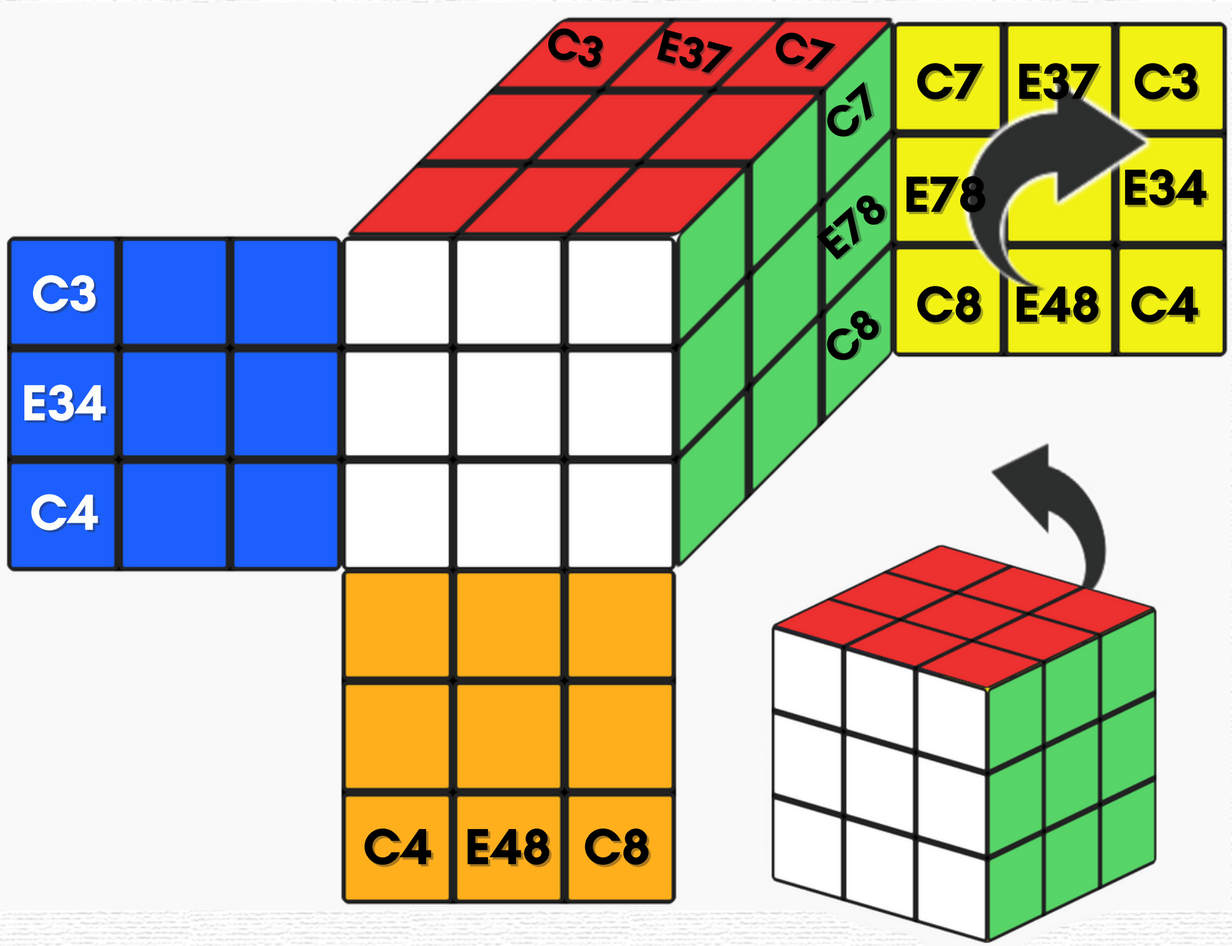
Ação Front Reverse



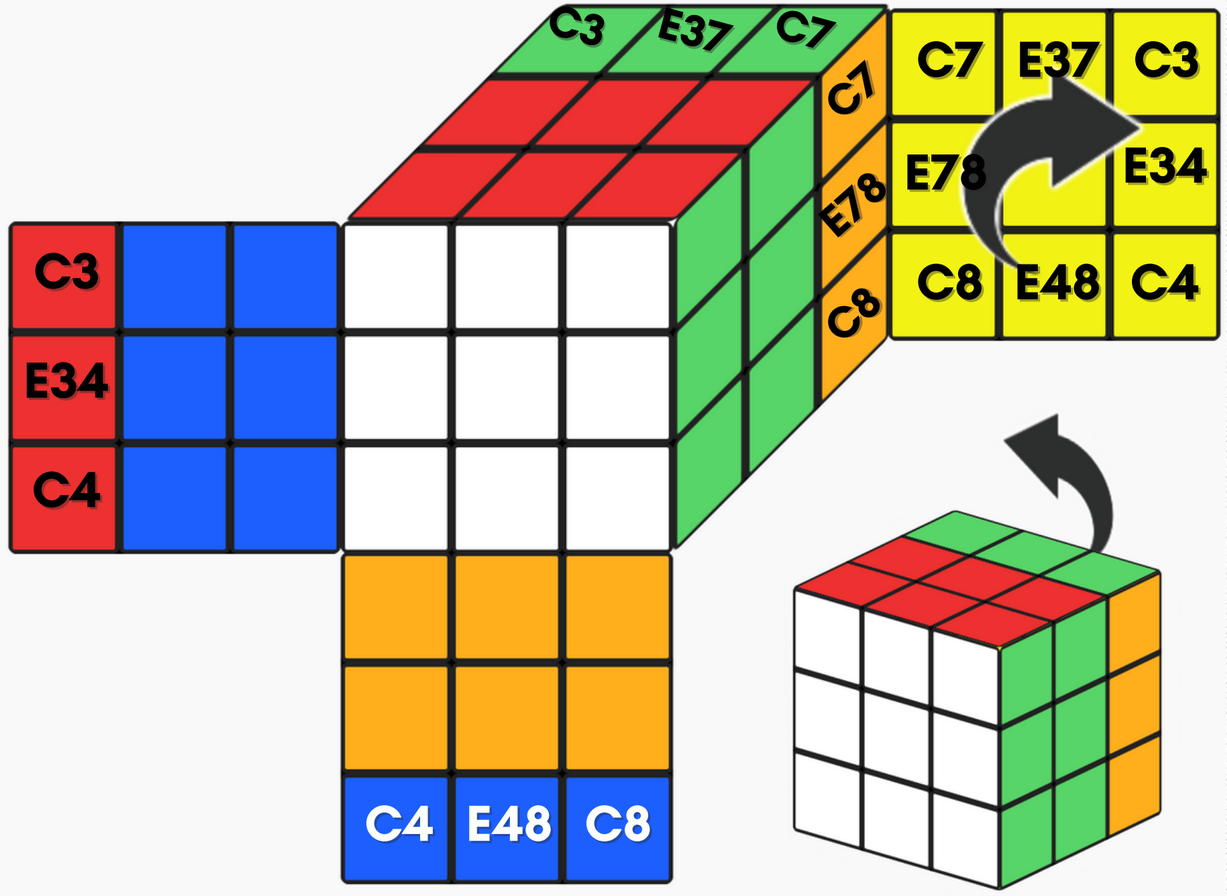

```
192         (:action Frev
193          :parameters ()
194          :precondition (and)
195          :effect
196             (and
197                 (forall (?x ?y ?z) (when (cube1 ?x ?y ?z) (and (not (cube1 ?x ?y ?z)) (cube2 ?z ?y ?x) )))
198                 (forall (?x ?y ?z) (when (cube2 ?x ?y ?z) (and (not (cube2 ?x ?y ?z)) (cube6 ?z ?y ?x) )))
199                 (forall (?x ?y ?z) (when (cube6 ?x ?y ?z) (and (not (cube6 ?x ?y ?z)) (cube5 ?z ?y ?x) )))
200                 (forall (?x ?y ?z) (when (cube5 ?x ?y ?z) (and (not (cube5 ?x ?y ?z)) (cube1 ?z ?y ?x) )))
201
202                 (forall (?x ?y) (when (edge15 ?x ?y) (and (not (edge15 ?x ?y)) (edge12 ?y ?x))))
203                 (forall (?y ?z) (when (edge56 ?y ?z) (and (not (edge56 ?y ?z)) (edge15 ?z ?y))))
204                 (forall (?x ?y) (when (edge26 ?x ?y) (and (not (edge26 ?x ?y)) (edge56 ?y ?x))))
205                 (forall (?y ?z) (when (edge12 ?y ?z) (and (not (edge12 ?y ?z)) (edge26 ?z ?y))))
206             )
207         )
```

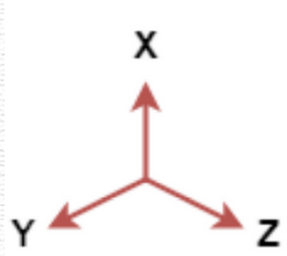
Ação Bottom

Antes

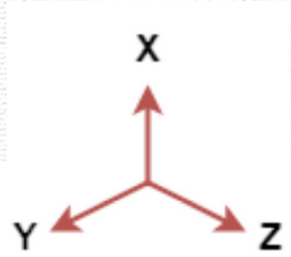


Depois



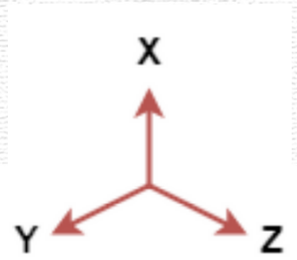


Ação Bottom




```
209         (:action B
210          :parameters ()
211          :precondition (and)
212          :effect
213             (and
214                 (forall (?x ?y ?z) (when (cube3 ?x ?y ?z) (and (not (cube3 ?x ?y ?z)) (cube4 ?z ?y ?x) )))
215                 (forall (?x ?y ?z) (when (cube7 ?x ?y ?z) (and (not (cube7 ?x ?y ?z)) (cube3 ?z ?y ?x) )))
216                 (forall (?x ?y ?z) (when (cube8 ?x ?y ?z) (and (not (cube8 ?x ?y ?z)) (cube7 ?z ?y ?x) )))
217                 (forall (?x ?y ?z) (when (cube4 ?x ?y ?z) (and (not (cube4 ?x ?y ?z)) (cube8 ?z ?y ?x) )))
218
219                 (forall (?x ?y) (when (edge37 ?x ?y) (and (not (edge37 ?x ?y)) (edge34 ?y ?x))))
220                 (forall (?y ?z) (when (edge78 ?y ?z) (and (not (edge78 ?y ?z)) (edge37 ?z ?y))))
221                 (forall (?x ?y) (when (edge48 ?x ?y) (and (not (edge48 ?x ?y)) (edge78 ?y ?x))))
222                 (forall (?y ?z) (when (edge34 ?y ?z) (and (not (edge34 ?y ?z)) (edge48 ?z ?y))))
223             )
224         )
```

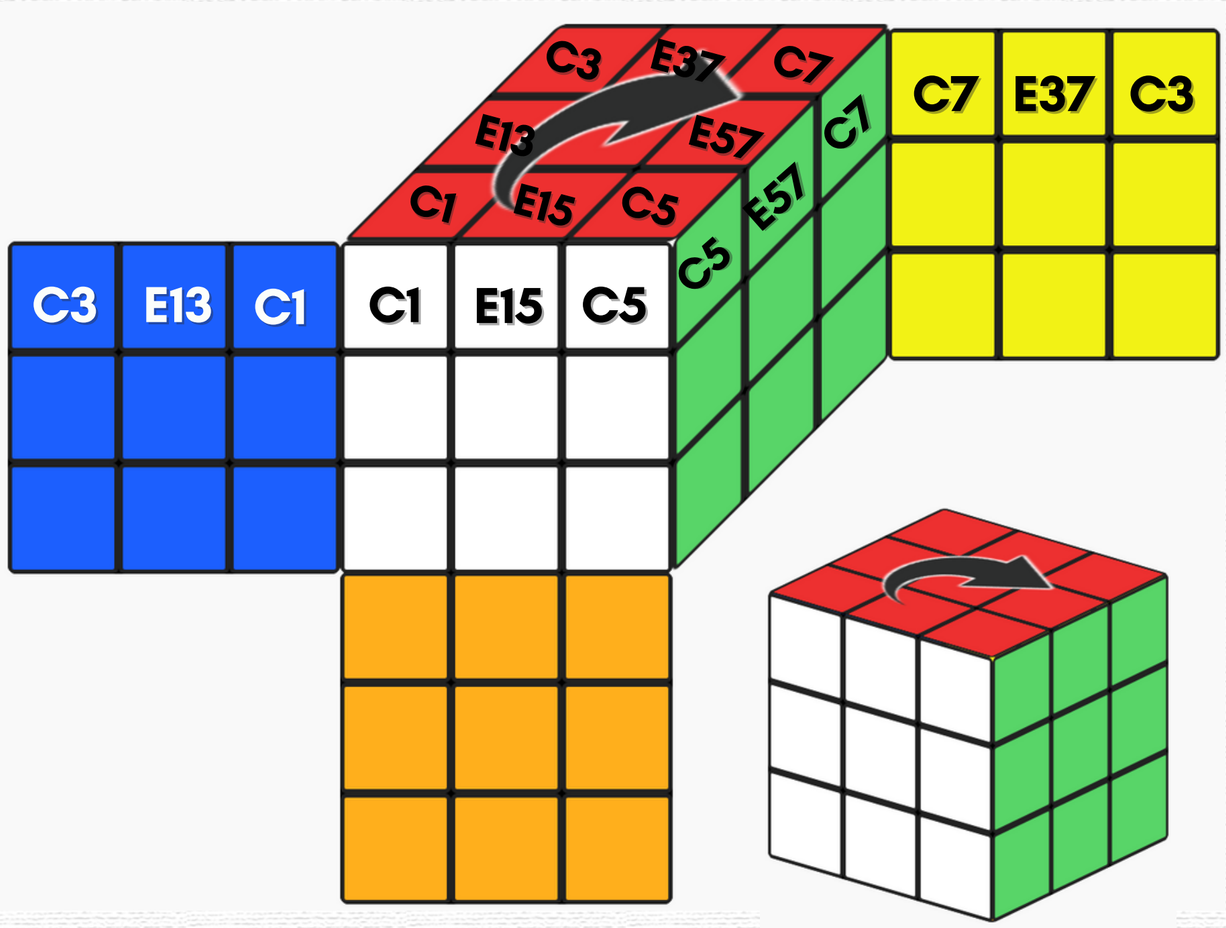
Ação Bottom Reverse



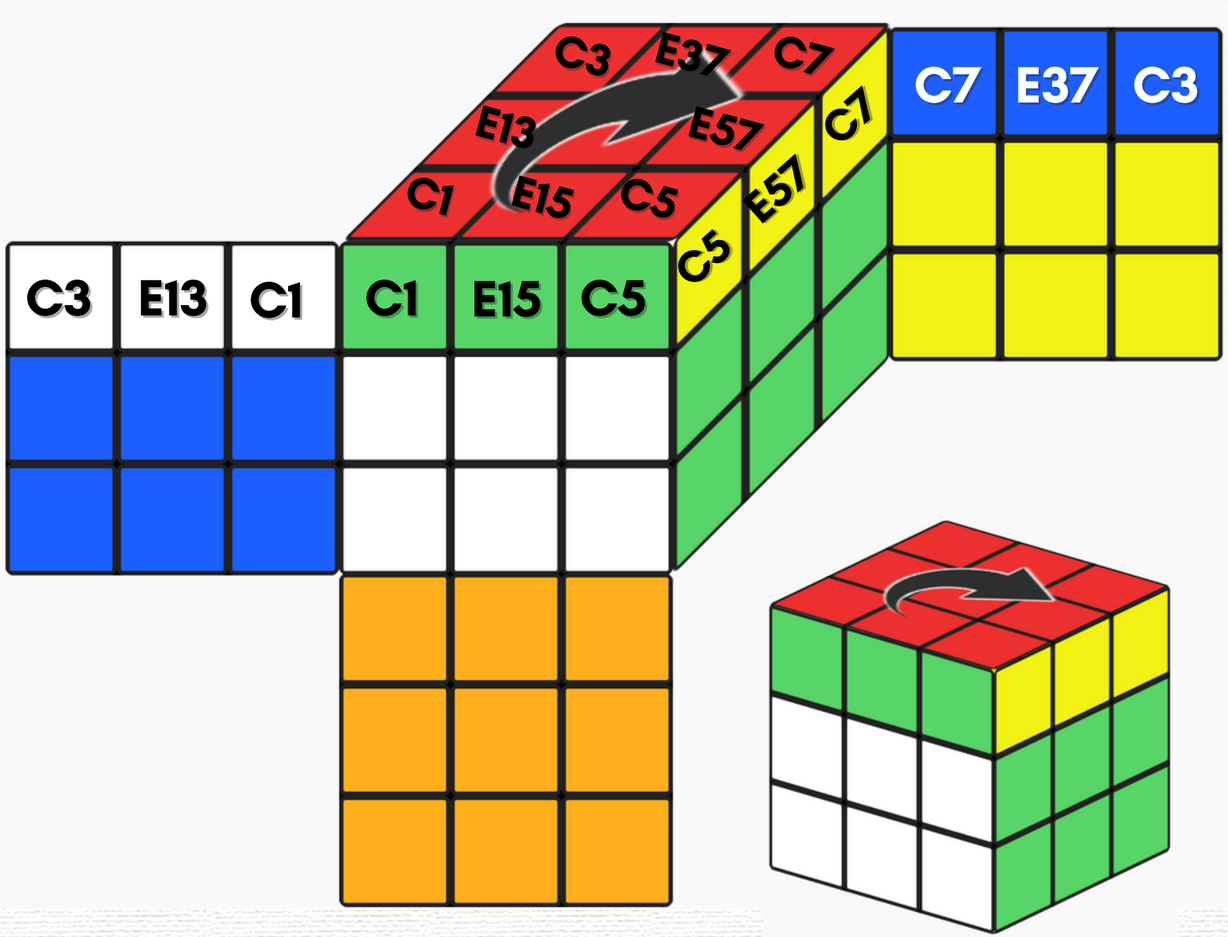

```
226         (:action Brev
227          :parameters ()
228          :precondition (and)
229          :effect
230             (and
231                 (forall (?x ?y ?z) (when (cube3 ?x ?y ?z) (and (not (cube3 ?x ?y ?z)) (cube7 ?z ?y ?x) )))
232                 (forall (?x ?y ?z) (when (cube7 ?x ?y ?z) (and (not (cube7 ?x ?y ?z)) (cube8 ?z ?y ?x) )))
233                 (forall (?x ?y ?z) (when (cube8 ?x ?y ?z) (and (not (cube8 ?x ?y ?z)) (cube4 ?z ?y ?x) )))
234                 (forall (?x ?y ?z) (when (cube4 ?x ?y ?z) (and (not (cube4 ?x ?y ?z)) (cube3 ?z ?y ?x) )))
235
236                 (forall (?x ?y) (when (edge37 ?x ?y) (and (not (edge37 ?x ?y)) (edge78 ?y ?x))))
237                 (forall (?y ?z) (when (edge78 ?y ?z) (and (not (edge78 ?y ?z)) (edge48 ?z ?y))))
238                 (forall (?x ?y) (when (edge48 ?x ?y) (and (not (edge48 ?x ?y)) (edge34 ?y ?x))))
239                 (forall (?y ?z) (when (edge34 ?y ?z) (and (not (edge34 ?y ?z)) (edge37 ?z ?y))))
240             )
241         )
```

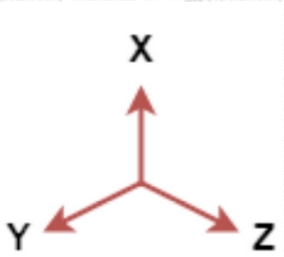
Ação Up

Antes

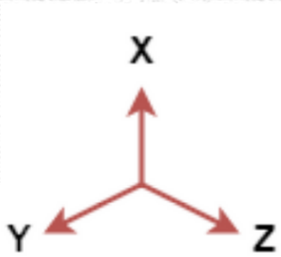


Depois



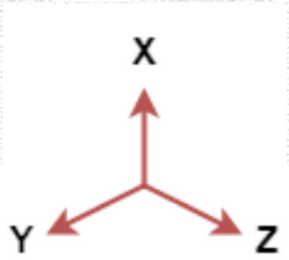


Ação Up




```
141         (:action U
142          :parameters ()
143          :precondition (and)
144          :effect
145             (and
146                 (forall (?x ?y ?z) (when (cube1 ?x ?y ?z) (and (not (cube1 ?x ?y ?z)) (cube3 ?x ?z ?y))))
147                 (forall (?x ?y ?z) (when (cube5 ?x ?y ?z) (and (not (cube5 ?x ?y ?z)) (cube1 ?x ?z ?y))))
148                 (forall (?x ?y ?z) (when (cube7 ?x ?y ?z) (and (not (cube7 ?x ?y ?z)) (cube5 ?x ?z ?y))))
149                 (forall (?x ?y ?z) (when (cube3 ?x ?y ?z) (and (not (cube3 ?x ?y ?z)) (cube7 ?x ?z ?y))))
150
151                 (forall (?x ?y) (when (edge15 ?x ?y) (and (not (edge15 ?x ?y)) (edge13 ?x ?y))))
152                 (forall (?x ?z) (when (edge57 ?x ?z) (and (not (edge57 ?x ?z)) (edge15 ?x ?z))))
153                 (forall (?x ?y) (when (edge37 ?x ?y) (and (not (edge37 ?x ?y)) (edge57 ?x ?y))))
154                 (forall (?x ?z) (when (edge13 ?x ?z) (and (not (edge13 ?x ?z)) (edge37 ?x ?z))))
155             )
156         )
```

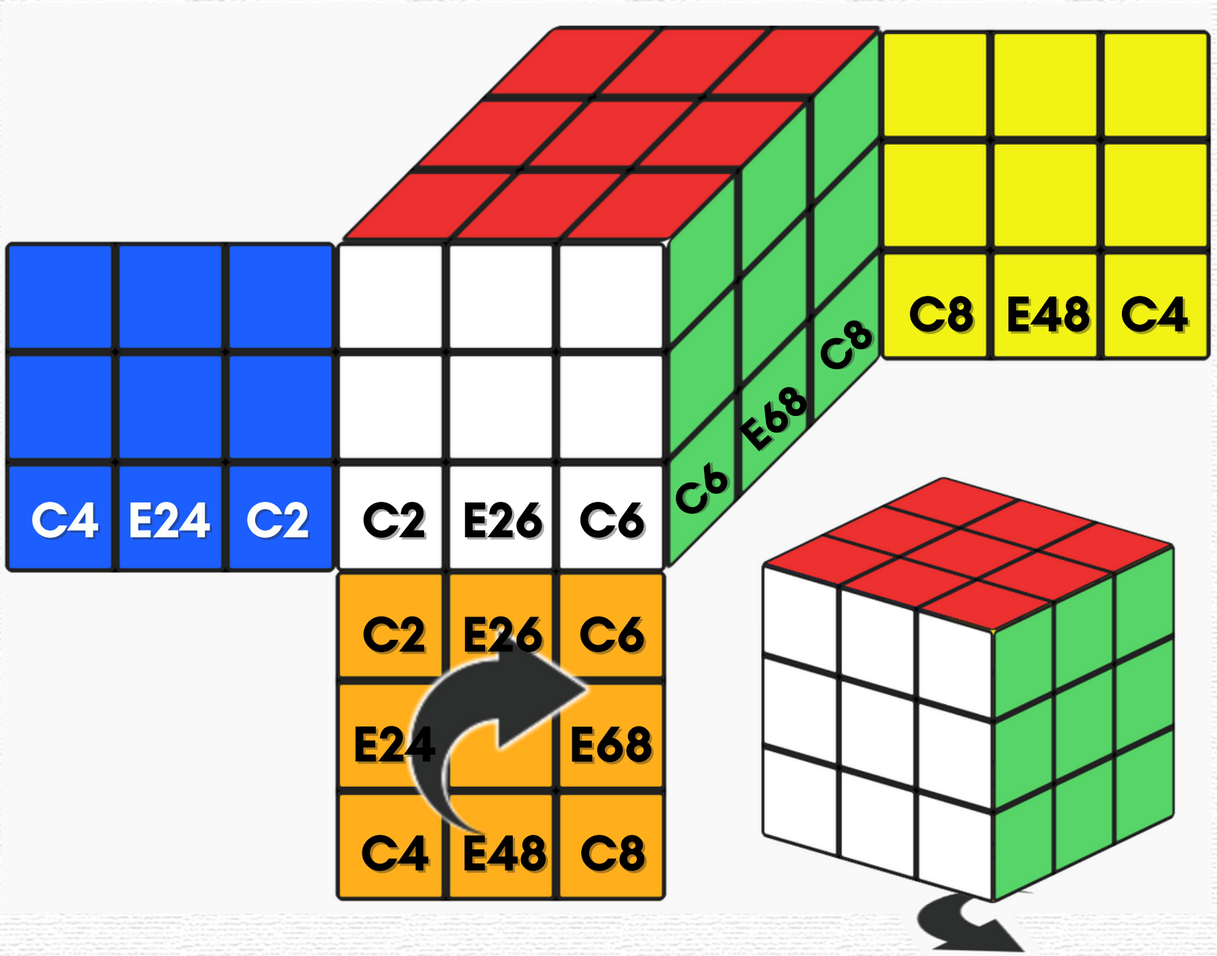
Ação Up Reverse



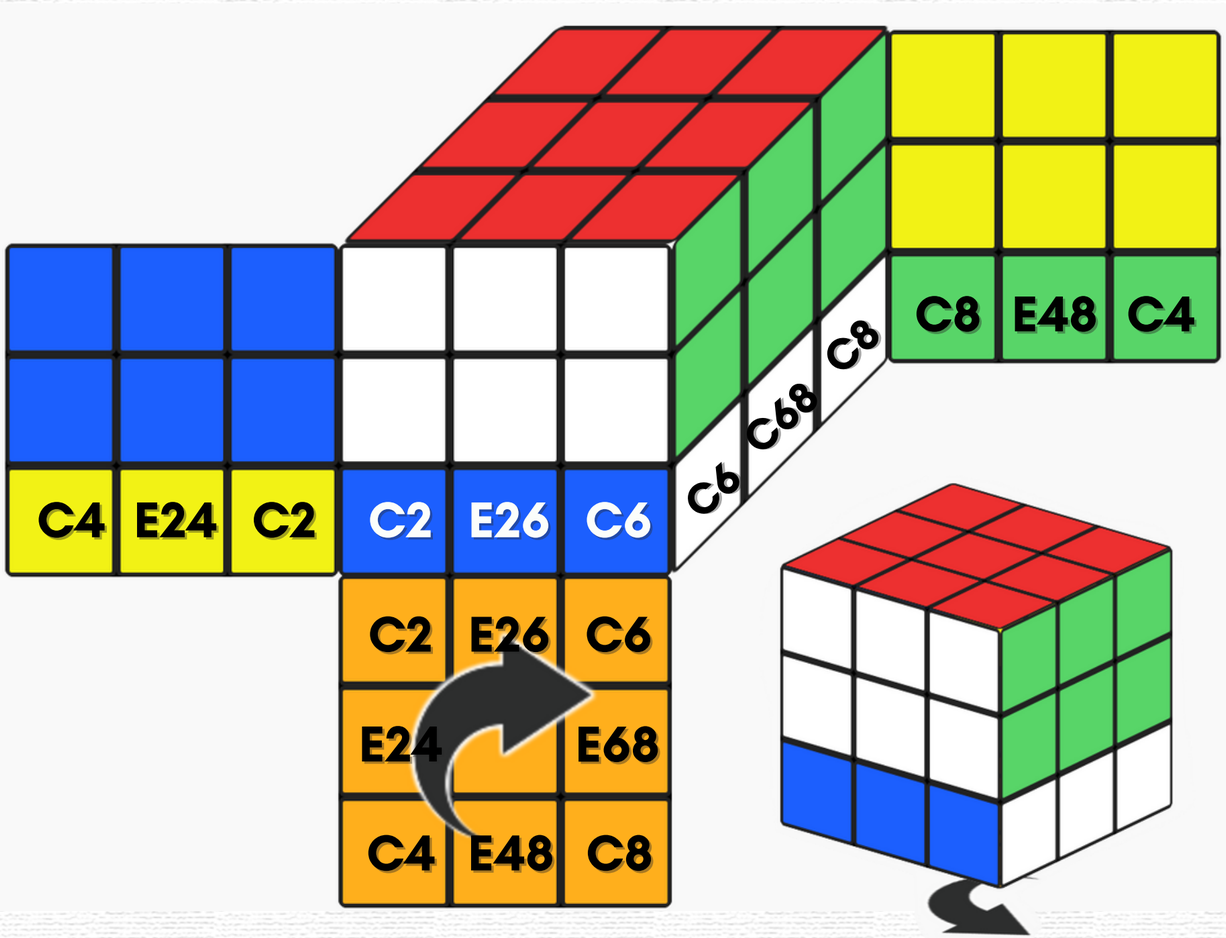

```
158         (:action Urev
159          :parameters ()
160          :precondition (and)
161          :effect
162             (and
163                 (forall (?x ?y ?z) (when (cube1 ?x ?y ?z) (and (not (cube1 ?x ?y ?z)) (cube5 ?x ?z ?y))))
164                 (forall (?x ?y ?z) (when (cube5 ?x ?y ?z) (and (not (cube5 ?x ?y ?z)) (cube7 ?x ?z ?y))))
165                 (forall (?x ?y ?z) (when (cube7 ?x ?y ?z) (and (not (cube7 ?x ?y ?z)) (cube3 ?x ?z ?y))))
166                 (forall (?x ?y ?z) (when (cube3 ?x ?y ?z) (and (not (cube3 ?x ?y ?z)) (cube1 ?x ?z ?y))))
167
168                 (forall (?x ?y) (when (edge15 ?x ?y) (and (not (edge15 ?x ?y)) (edge57 ?x ?y))))
169                 (forall (?x ?z) (when (edge57 ?x ?z) (and (not (edge57 ?x ?z)) (edge37 ?x ?z))))
170                 (forall (?x ?y) (when (edge37 ?x ?y) (and (not (edge37 ?x ?y)) (edge13 ?x ?y))))
171                 (forall (?x ?z) (when (edge13 ?x ?z) (and (not (edge13 ?x ?z)) (edge15 ?x ?z))))
172             )
173         )
```

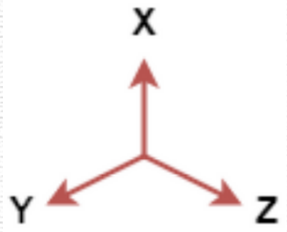
Ação Down

Antes

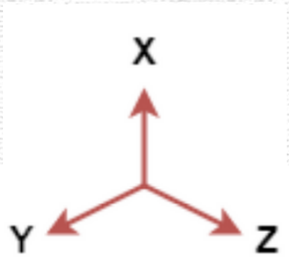


Depois



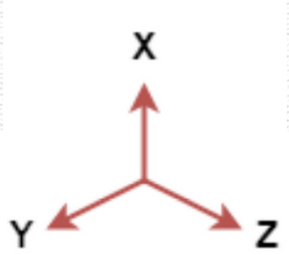


Ação Down




```
107         (:action D
108          :parameters ()
109          :precondition (and)
110          :effect
111             (and
112                 (forall (?x ?y ?z) (when (cube2 ?x ?y ?z) (and (not (cube2 ?x ?y ?z)) (cube6 ?x ?z ?y))))
113                 (forall (?x ?y ?z) (when (cube6 ?x ?y ?z) (and (not (cube6 ?x ?y ?z)) (cube8 ?x ?z ?y))))
114                 (forall (?x ?y ?z) (when (cube8 ?x ?y ?z) (and (not (cube8 ?x ?y ?z)) (cube4 ?x ?z ?y))))
115                 (forall (?x ?y ?z) (when (cube4 ?x ?y ?z) (and (not (cube4 ?x ?y ?z)) (cube2 ?x ?z ?y))))
116
117                 (forall (?x ?y) (when (edge26 ?x ?y) (and (not (edge26 ?x ?y)) (edge68 ?x ?y))))
118                 (forall (?x ?z) (when (edge68 ?x ?z) (and (not (edge68 ?x ?z)) (edge48 ?x ?z))))
119                 (forall (?x ?y) (when (edge48 ?x ?y) (and (not (edge48 ?x ?y)) (edge24 ?x ?y))))
120                 (forall (?x ?z) (when (edge24 ?x ?z) (and (not (edge24 ?x ?z)) (edge26 ?x ?z))))
121             )
122         )
```

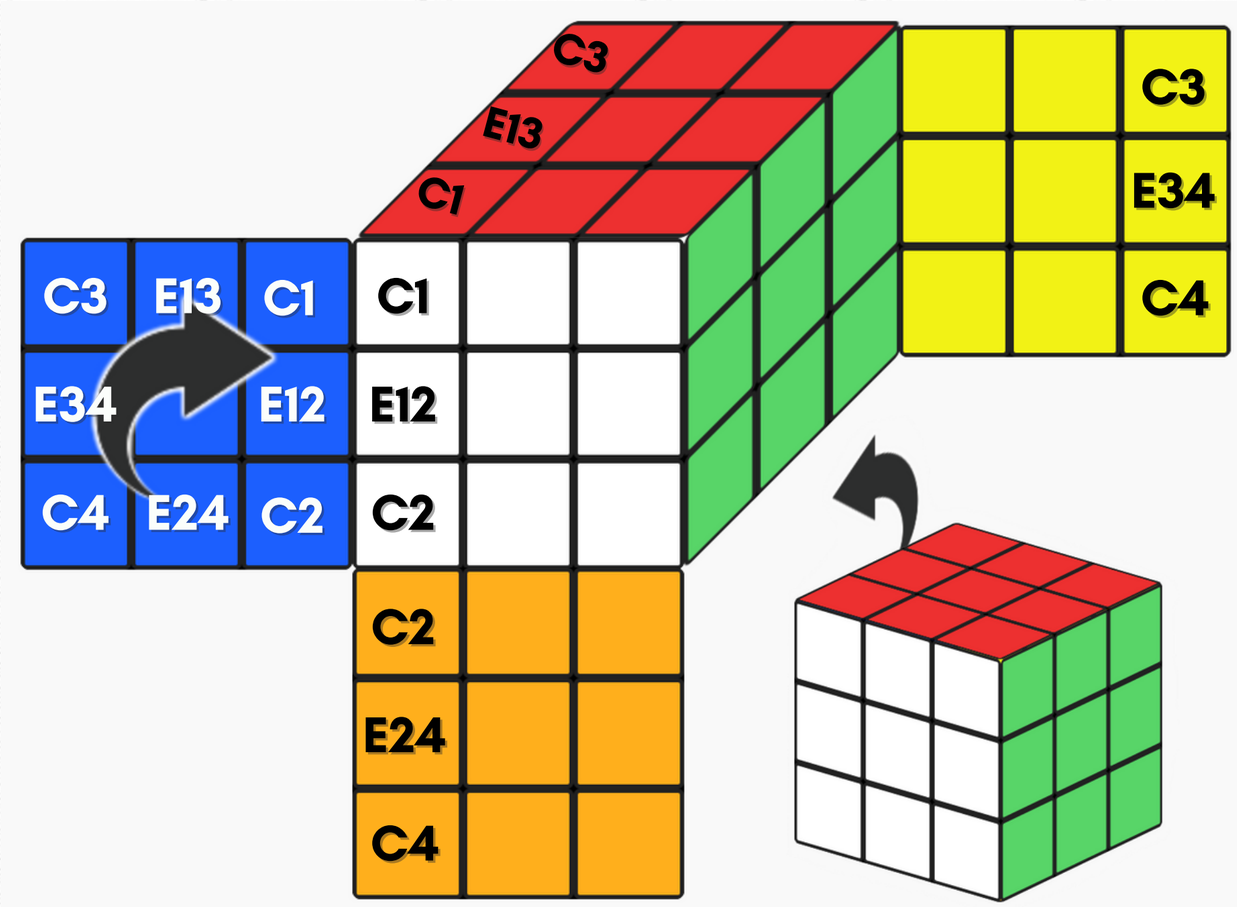
Ação Down Reverse



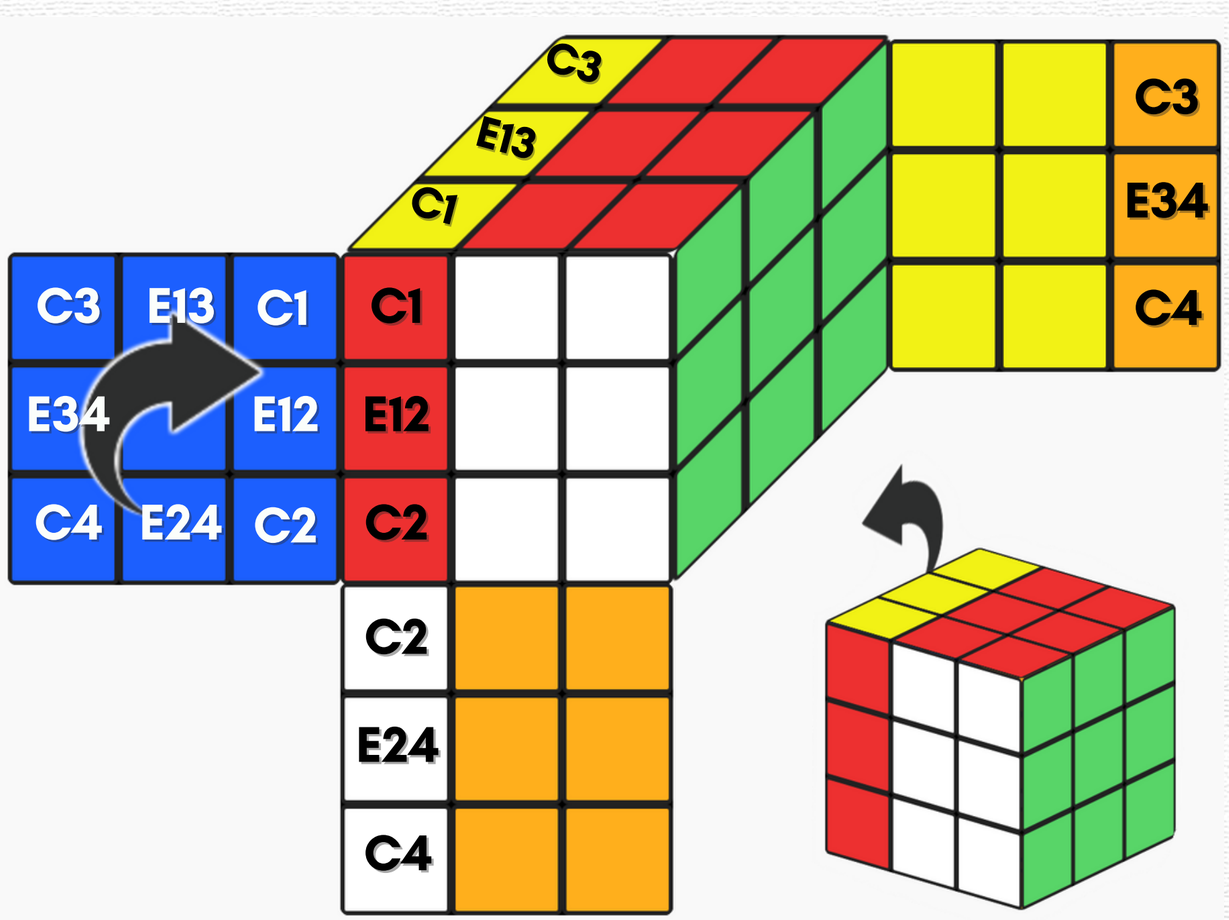

```
124         (:action Drev
125          :parameters ()
126          :precondition (and)
127          :effect
128             (and
129                 (forall (?x ?y ?z) (when (cube2 ?x ?y ?z) (and (not (cube2 ?x ?y ?z)) (cube4 ?x ?z ?y))))
130                 (forall (?x ?y ?z) (when (cube4 ?x ?y ?z) (and (not (cube4 ?x ?y ?z)) (cube8 ?x ?z ?y))))
131                 (forall (?x ?y ?z) (when (cube8 ?x ?y ?z) (and (not (cube8 ?x ?y ?z)) (cube6 ?x ?z ?y))))
132                 (forall (?x ?y ?z) (when (cube6 ?x ?y ?z) (and (not (cube6 ?x ?y ?z)) (cube2 ?x ?z ?y))))
133
134                 (forall (?x ?y) (when (edge26 ?x ?y) (and (not (edge26 ?x ?y)) (edge24 ?x ?y))))
135                 (forall (?x ?z) (when (edge68 ?x ?z) (and (not (edge68 ?x ?z)) (edge26 ?x ?z))))
136                 (forall (?x ?y) (when (edge48 ?x ?y) (and (not (edge48 ?x ?y)) (edge68 ?x ?y))))
137                 (forall (?x ?z) (when (edge24 ?x ?z) (and (not (edge24 ?x ?z)) (edge48 ?x ?z))))
138             )
139         )
```

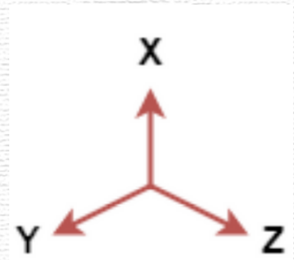
Ação Left

Antes

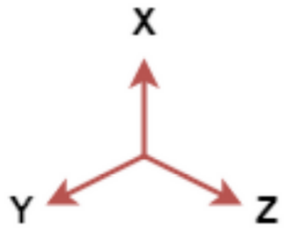


Depois



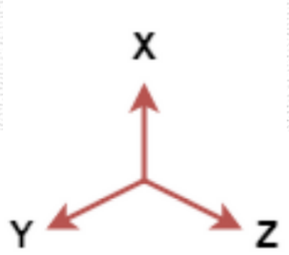


Ação Left




```
64        (:action L
65         :parameters ()
66         :precondition (and)
67         :effect
68            (and
69                (forall (?x ?y ?z) (when (cube1 ?x ?y ?z)
70                    (and (not (cube1 ?x ?y ?z)) (cube2 ?y ?x ?z))))
71                (forall (?x ?y ?z) (when (cube3 ?x ?y ?z)
72                    (and (not (cube3 ?x ?y ?z)) (cube1 ?y ?x ?z))))
73                (forall (?x ?y ?z) (when (cube4 ?x ?y ?z)
74                    (and (not (cube4 ?x ?y ?z)) (cube3 ?y ?x ?z))))
75                (forall (?x ?y ?z) (when (cube2 ?x ?y ?z)
76                    (and (not (cube2 ?x ?y ?z)) (cube4 ?y ?x ?z))))
77
78                (forall (?x ?z) (when (edge13 ?x ?z)
79                    (and (not (edge13 ?x ?z)) (edge12 ?x ?z))))
80                (forall (?y ?z) (when (edge34 ?y ?z)
81                    (and (not (edge34 ?y ?z)) (edge13 ?y ?z))))
82                (forall (?x ?z) (when (edge24 ?x ?z)
83                    (and (not (edge24 ?x ?z)) (edge34 ?x ?z))))
84                (forall (?y ?z) (when (edge12 ?y ?z)
85                    (and (not (edge12 ?y ?z)) (edge24 ?y ?z))))
86
87            )
88        )
```

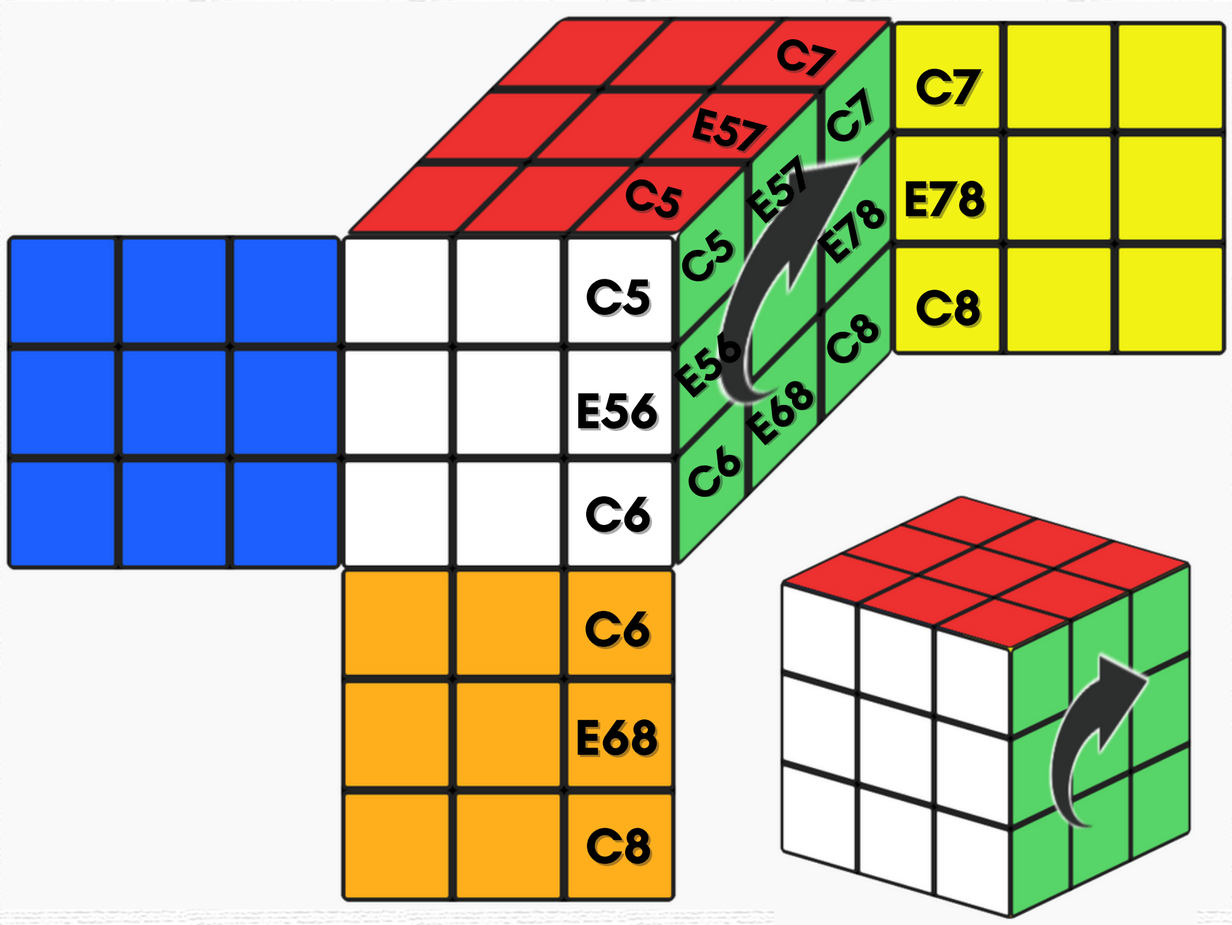
Ação Left Reverse



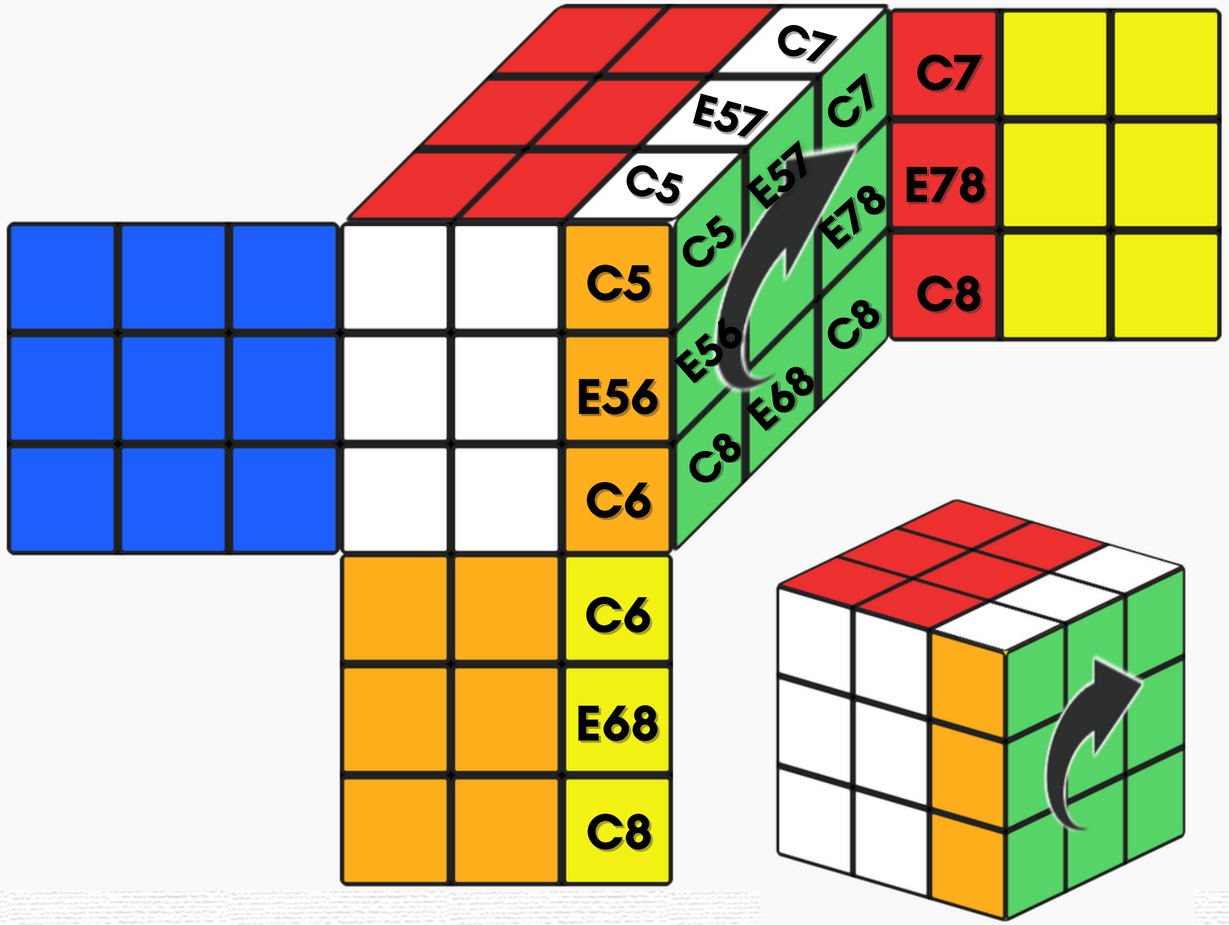

```
90          (:action Lrev
91           :parameters ()
92           :precondition (and)
93           :effect
94              (and
95                  (forall (?x ?y ?z) (when (cube1 ?x ?y ?z) (and (not (cube1 ?x ?y ?z)) (cube3 ?y ?x ?z))))
96                  (forall (?x ?y ?z) (when (cube3 ?x ?y ?z) (and (not (cube3 ?x ?y ?z)) (cube4 ?y ?x ?z))))
97                  (forall (?x ?y ?z) (when (cube4 ?x ?y ?z) (and (not (cube4 ?x ?y ?z)) (cube2 ?y ?x ?z))))
98                  (forall (?x ?y ?z) (when (cube2 ?x ?y ?z) (and (not (cube2 ?x ?y ?z)) (cube1 ?y ?x ?z))))
99
100                 (forall (?x ?z) (when (edge13 ?x ?z) (and (not (edge13 ?x ?z)) (edge34 ?x ?z))))
101                 (forall (?y ?z) (when (edge34 ?y ?z) (and (not (edge34 ?y ?z)) (edge24 ?y ?z))))
102                 (forall (?x ?z) (when (edge24 ?x ?z) (and (not (edge24 ?x ?z)) (edge12 ?x ?z))))
103                 (forall (?y ?z) (when (edge12 ?y ?z) (and (not (edge12 ?y ?z)) (edge13 ?y ?z))))
104             )
105         )
```

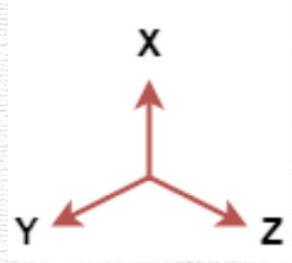
Ação Right

Antes

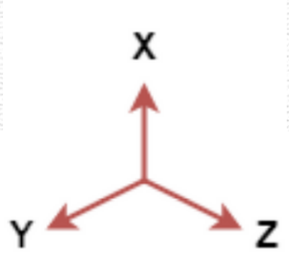


Depois



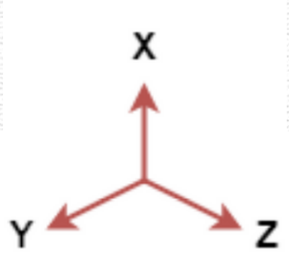


Ação Right




```
28          (:action R
29           :parameters ()
30           :precondition (and)
31           :effect
32              (and
33                  (forall (?x ?y ?z) (when (cube5 ?x ?y ?z) (and (not (cube5 ?x ?y ?z)) (cube7 ?y ?x ?z))))
34                  (forall (?x ?y ?z) (when (cube7 ?x ?y ?z) (and (not (cube7 ?x ?y ?z)) (cube8 ?y ?x ?z))))
35                  (forall (?x ?y ?z) (when (cube8 ?x ?y ?z) (and (not (cube8 ?x ?y ?z)) (cube6 ?y ?x ?z))))
36                  (forall (?x ?y ?z) (when (cube6 ?x ?y ?z) (and (not (cube6 ?x ?y ?z)) (cube5 ?y ?x ?z))))
37
38                  (forall (?x ?z) (when (edge57 ?x ?z) (and (not (edge57 ?x ?z)) (edge78 ?x ?z))))
39                  (forall (?y ?z) (when (edge78 ?y ?z) (and (not (edge78 ?y ?z)) (edge68 ?y ?z))))
40                  (forall (?x ?z) (when (edge68 ?x ?z) (and (not (edge68 ?x ?z)) (edge56 ?x ?z))))
41                  (forall (?y ?z) (when (edge56 ?y ?z) (and (not (edge56 ?y ?z)) (edge57 ?y ?z))))
42
43              )
```

Ação Right Reverse




```
46          (:action Rrev
47           :parameters ()
48           :precondition (and)
49           :effect
50              (and
51                  (forall (?x ?y ?z) (when (cube5 ?x ?y ?z) (and (not (cube5 ?x ?y ?z)) (cube6 ?y ?x ?z))))
52                  (forall (?x ?y ?z) (when (cube6 ?x ?y ?z) (and (not (cube6 ?x ?y ?z)) (cube8 ?y ?x ?z))))
53                  (forall (?x ?y ?z) (when (cube8 ?x ?y ?z) (and (not (cube8 ?x ?y ?z)) (cube7 ?y ?x ?z))))
54                  (forall (?x ?y ?z) (when (cube7 ?x ?y ?z) (and (not (cube7 ?x ?y ?z)) (cube5 ?y ?x ?z))))
55
56                  (forall (?x ?z) (when (edge57 ?x ?z) (and (not (edge57 ?x ?z)) (edge56 ?x ?z))))
57                  (forall (?y ?z) (when (edge78 ?y ?z) (and (not (edge78 ?y ?z)) (edge57 ?y ?z))))
58                  (forall (?x ?z) (when (edge68 ?x ?z) (and (not (edge68 ?x ?z)) (edge78 ?x ?z))))
59                  (forall (?y ?z) (when (edge56 ?y ?z) (and (not (edge56 ?y ?z)) (edge68 ?y ?z))))
60
61              )
```

Referências

RUBIKS Cube Solver. 2023. Disponível em: https://rubiks-cube-solver.com/. Acesso em: 18 set. 2023

Optimal solutions for the Rubik's Cube. 2023. https://en.wikipedia.org/wiki/Optimal_solutions_for_the_Rubik%27s_Cube. Acesso em: 20 set. 2023_

MUPPASANI, Bharath. SRIVASTANA, Biplav. “Rubik’s Cube PDDL Domain”. 2023. Disponível em: https://github.com/ipc2023-classical/domain-rubiks-cube. Acesso em: 20 set. 2023

MUPPASANI, Bharath. SRIVASTANA, Biplav. “On Solving the Rubik’s Cube with Domain Independent Planners Using Standard Representations”. 2023. Disponível em: https://arxiv.org/pdf/2307.13552.pdf. Acesso em: 20 set. 2023