Inteligência Artificial, 2016/2017
1º trabalho prático

Troika Cube

— Originalmente proposto pelo Prof. José Valente de Oliveira —

(As minhas desculpas pelo enunciado estar em inglês – Fernando Lobo)

1 Submission procedure and deadline

- Your code to: `http://mooshak.deei.fct.ualg.pt/~mooshak/` (contest IA1617, problem E — Troika Cube)
- Your report by email to fernando.lobo@gmail.com. The email subject must be: `IA troika G`*dd*. (Example: `IA troika G03`). The email must contain two and only two attachments.
  - a PDF file. Filename must be `troika-G`*dd*`.pdf`, where *dd* is your group number. (Example: `troika-G03.pdf`)
  - a ZIP file with the code mentioned in item 6 of Section 4. Filename must be: `troika-noHeuristics-G`*dd*`.zip`, where *dd* is your group number. (Example: `troika-noHeuristics-G03.zip`)
- Deadline: **01/Apr/2017**

2 Introduction

In 1983 Erno Rubik presented its *cube*; a (completely useless but exciting) puzzle. The Rubik's cube has $3 \times 3 \times 3$ coloured sliding tiles. In each cube face tiles can rotate clock or anti-clock wisely.

What makes this puzzle particularly exciting is the number of different possible cube configurations: 4.3252×10^{19}, a value that exceeds the age of the known universe. It is not surprising that no human is able to find the minimum number of moves between any given pair of such configurations; and any of them can be reached from any other.

Mathematicians and Computer Scientists love *cube* problems. These have been putting a considerable effort and computer power trying to prove theorems and to achieve efficient ways to find minimal number of movements between configurations. To add to the excitement, scientists have coined an algorithm that solves a cube in the minimum number of moves a *God's algorithm*.

In Portugal recently this panorama has changed considerably. Troika restricted computer power consumption in such a way that optimal solutions for the original $3 \times 3 \times 3$ Rubic's cube can no longer be experimented. Long known by their *desenrascanso* abilities Portuguese people quickly came out with an alternative, i.e., to study the $3 \times 3 \times 1$ cube, shown in Figure 1, who soon become known as the Troika's cube.

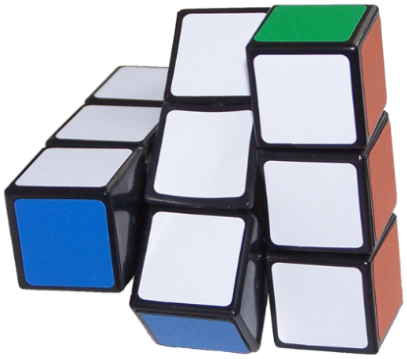

Figure 1: $3 \times 3 \times 1$ or Troika's cube.

3 Task

Write a program that reads a particular scrambled configuration of the Troika cube and output the minimum number of moves required for transforming it into the target configuration, the target configuration being every and each of the cube side with the same colour. For simplicity we may assume that the white face is at the front, green is at the top, blue at the bottom, orange at the right, yellow at the back, and red at the left. The adopted target configuration is represented by the cube planing given in Figure 2.

In the Troika cube a valid move corresponds to turning a cube "face" 180 degrees either clock or anti-clock wisely, a "face" being an ensemble of tiles.

In the representation of Figure 2 each cube side is represented by set of squares, each one representing a face of a tile. This same configuration can be represented by a sequence of chars, each one representing a square of a specific color. That is:

```
GGG
WWWOYYYR
WWWOYYYR
WWWOYYYR
BBB
```

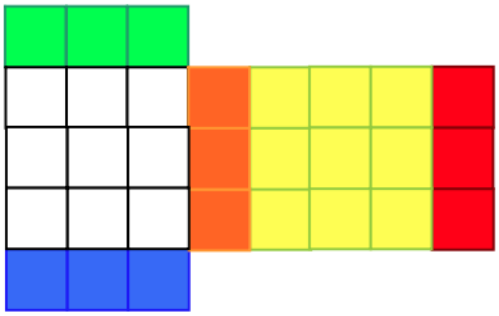

Figure 2: Planing of a Troika cube target configuration.

where G stands for a green tile's face, W for white, Y for yellow, R for red, O for orange, and B for blue.

4 Requirements

Your program should use an **informed search method**, i.e. use a heuristic in state-space search. In other words, use A* or IDA*. For the report you should address the following questions:

1. Explain how you represent a state of the problem.
2. Explain what are the operators (or moves) for your problem.
3. Explain your chosen heuristic in words.
4. Explain why your chosen heuristic is admissible.
5. For the instance shown in input example 3, compute the *effective branching factor* of your algorithm submitted to mooshak. Justify your result.
6. Solve the same problem without using heuristics (for example with Best-First Search, Depth-First Search or Iterative Depth-First Search) and again compute the *effective branching factor* for the instance shown in input example 3. Justify the result. **(The source code of this question in not submitted on mooshak, it should be in the ZIP file submitted by email).**

Mooshak Input

The input has 5 rows of characters. The first row has 3 characters. The second, third, and fourth rows have 8 characters. The fifth row has 3 characters.

The rows of the input represent a planing of a specific cube configuration, as explained above. For instance, Sample Input 1 has the planing shown in Figure 3, corresponding to move 180 degree the rightmost ensemble of $3 \times 3 \times 1$ tiles.

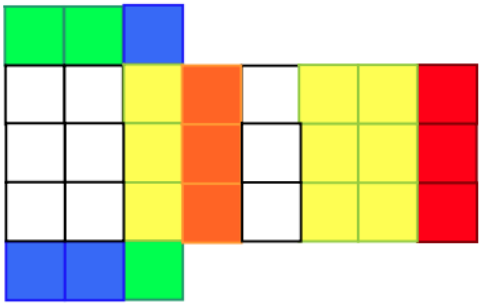

Figure 3: Planing of a Troika cube for Sample Input 1.

Mooshak Output

A natural number indicating the minimum number of moves required for transforming the given input into the target configuration, followed by a newline character.

Constraints

The possible values for each character are: G, W, Y, R, O, B

Input example 1

```
GGB
WWYOWYYR
WWYOWYYR
WWYOWYYR
BBG
```

Output example 1

```
1
```

Input example 2

```
GBG
WYWOYWYR
YWYRWYWO
WYWOYWYR
BGB
```

Output example 2

```
2
```

Input example 3

```
GGG
WYWOYWYR
YWYOWYWR
WYWOYWYR
BBB
```

Output example 3

```
6
```