UNIVERSIDAD AUTONOMA DE MADRID

Escuela Politécnica Superior

20|21

Trabajo fin de grado

Un estudio computacional del grupo de Rubik

Lucía Colmenarejo Pérez

Escuela Politécnica Superior
Universidad Autónoma de Madrid
C\Francisco Tomás y Valiente nº 11

Campus Internacional excelencia UAM CSIC+

www.uam.es

UNIVERSIDAD AUTÓNOMA DE MADRID
ESCUELA POLITÉCNICA SUPERIOR

Grado en Ingeniería Informática

TRABAJO FIN DE GRADO

Un estudio computacional del grupo de Rubik

Autor: Lucía Colmenarejo Pérez
Tutor: Simone Santini

mayo 2021

Lucía Colmenarejo Pérez
Un estudio computacional del grupo de Rubik

Lucía Colmenarejo Pérez
C\ Francisco Tomás y Valiente Nº 11

IMPRESO EN ESPAÑA – PRINTED IN SPAIN

Agradecimientos

En primer lugar, me gustaría expresar mi agradecimiento a toda mi familia por su comprensión y apoyo durante la realización del proyecto, así como durante estos cinco años de carrera, especialmente, a mi madre Esmeralda.

Agradezco el apoyo de mis compañeros de carrera, amigos para toda la vida, en especial, Emilio y Jesús, por sus palabras de ánimo y consuelo.

Y finalmente, expresar mi gratitud a mis profesores de la UAM; y con un afecto particular, a mi tutor Simone Santini, tanto a nivel académico como personal.

Resumen

En este trabajo se han abordado diferentes métodos de cara a la resolución de uno de los rompecabezas más conocidos mundialmente, el cubo de Rubik. Los algoritmos escogidos son muy distintos y por ello se realizará un estudio comparativo entre ambos. Previamente, se explica el origen del cubo de Rubik así como se explican las matemáticas que hay detrás de este rompecabezas. De igual manera, se emplearán dos representaciones en función del algoritmo, teniendo por un lado una representación basada en la interfaz gráfica y otra en los pilares matemáticos en los que se basa.

Por un lado, se ha estudiado el grupo de Rubik donde usamos las permutaciones y las nociones más importantes de grupo como base para uno de los métodos empleados, el algoritmo A* (A-estrella). Adicionalmente se hace una implementación especial de las configuraciones de los cubitos que componen al cubo de Rubik representadas por tuplas de dos y tres letras, que identifican las caras. Sobre estas permutaciones se aplican los movimientos del cubo de cara a encontrar la solución. Para el método de A* se han desarrollado unas heurísticas con el fin de valorar cuál de ellas es más eficiente. Adicionalmente, para poder comparar las heurísticas entre sí, se realiza un estudio en el que se ven la cantidad de nodos que ha sido necesario generar y explorar para llegar a la solución.

Por otro lado, se ha implementado el Algoritmo del Principiante el cual se desglosa en siete pasos distintos. En cada uno de estos pasos se tendrá en cuenta las distintas casuísticas en las cuales se puede encontrar el cubo para dar una solución parcial a cada uno de ellos. La representación del cubo para este algoritmo se basa en su forma geométrica en el plano tridimensional, la cual se emplea en la interfaz gráfica para mostrar la resolución del rompecabezas.

Por último, se realiza un estudio en el que se comparan el Algoritmo del Principiante con el A* usando cada una de las heurísticas para así poder estudiar su eficiencia. Finalmente, tras el estudio se concluye que el método de A* es mucho más eficiente en términos de número de movimientos que el Algoritmo del Principiante; sin embargo, consume una mayor cantidad de recursos, no pudiendo llegar a una solución en ciertas situaciones por tiempos de ejecución.

Palabras clave

Cubo de Rubik, grupo, permutaciones, A* (A-estrella), heurística, Algoritmo del Principiante

Abstract

In this work different methods for solving one of the best known puzzles in the world, the Rubik's cube, have been approached. The chosen algorithms are very different and therefore a comparative study will be made between them. Previously, the origin of the Rubik's cube is explained as well as the mathematics behind this puzzle. In the same way, two representations will be used depending on the algorithm, having on the one hand a representation based on the graphical interface and another one on the mathematical pillars on which it is based.

On the one hand, the Rubik's group has been studied where we use the permutations and the most important group notions as the basis for one of the methods used, the A* (A-star) algorithm. Additionally, a special implementation is made of the configurations of the cubes that compose the Rubik's cube represented by tuples of two and three letters, which identify the faces. On these permutations the cube movements are applied in order to find the solution. For the A* method, heuristics have been developed in order to evaluate which of them is more efficient. Additionally, in order to compare the heuristics with each other, a study is carried out in which the number of nodes that had to be generated and explored to reach the solution is shown.

On the other hand, the Beginner's Algorithm has been implemented, which is broken down into seven different steps. In each of these steps the different cases in which the cube can be found will be taken into account to give a partial solution to each of them. The representation of the cube for this algorithm is based on its geometric shape in the three-dimensional plane, which is used in the graphical interface to show the resolution of the puzzle.

Finally, a study is carried out in which the Beginner's Algorithm is compared with the A* using each of the heuristics in order to study their efficiency. Finally, after the study it is concluded that the A* method is much more efficient in terms of number of moves than the Beginner's Algorithm; however, it consumes a greater amount of resources, not being able to reach a solution in certain situations due to execution times.

Keywords

Rubik's Cube, group, permutations, A* (A-star), heuristic, Beginner's Algorithm

Índice

Listas

Lista de algoritmos

Lista de códigos

Lista de cuadros

Lista de ecuaciones

Lista de figuras

Lista de tablas

Lista de cuadros

1

Introducción

1.1. Historia

El rompecabezas del cubo de Rubik fue inventado en el año 1974 por el escultor y profesor de arquitectura húngaro Ernő Rubik [1]. El origen de este juguete o rompecabezas no fue otro que la obsesión de Ernő Rubik de encontrar una manera sencilla de modelar el movimiento tridimensional para poder mostrárselo a sus alumnos.

Ernő Ruik nació en Budapest (Hungría) el 13 de julio de 1944. Nació en un momento en el que Europa estaba sumergida en la Segunda Guerra Mundial. Su padre fue ingeniero aeroespacial, llegando a fundar su propia fábrica de planeadores, mientras que su madre era poeta. Ernő Rubik consiguió graduarse en 1967 en Ingeniería Civil en la Universidad de Budapest, momento a partir del cual empezó sus estudios en la Academia de Artes Aplicadas y Diseño. En los primeros años de su vida laboral estuvo trabajando como arquitecto (durante la época de los 70), pero finalmente regresó a la academia de arte de Budapest para presidir la facultad donde había estudiado años atrás ([2]).

Ernő Rubik siempre estuvo interesado en la geometría y el estudio de las formas tridimensionales. En los años donde estuvo investigando y que acabaron con la creación del cubo de Rubik, Ernő se encargaba de la dirección de una publicación húngara de juegos de ingenio. Después de pasar meses experimentando con bloques de madera y de papel, pegamento y clips, finalmente consiguió inventar la estructura del juguete más vendido de la historia y uno de los rompecabezas más complejos de resolver, el cual lleva su propio nombre, el cubo de Rubik. Dado que el origen de este rompecabezas era meramente educativo, al principio de los años 80 solo se difundió en algunos círculos de científicos húngaros y de manera muy limitada. En ese momento los únicos atraídos por el rompecabezas eran los científicos matemáticos, interesados en los problemas estadísticos y teóricos que este planteaba.

A principios de los 80 un matemático inglés publicó un artículo acerca del cubo de Rubik y gracias a él se extendió su fama a través de todo el mundo. Es así como en 1982 se habían vendido ya más de 100 millones de unidades del rompecabezas en Europa y América, bajo el nombre de 'Cubo Mágico'. Sería años después cuando recibiría el nombre de 'Cubo de Rubik' convirtiendo a Ernő Rubik en la persona más rica de Hungría. A día de hoy es el juguete más vendido del mundo, se estima que

alrededor de unas 350 millones de unidades han sido vendidas.

A pesar de la enorme popularidad lograda, originalmente Ernő Rubik no quería crear un rompecabezas, sino investigar el problema estructural de cómo se podían mover los bloques de forma independiente sin que se desmontara el cubo. El que se considera el juguete más vendido del mundo es también un reto matemático al que se han dedicado varios estudios científicos. En 2019, unos investigadores de la Universidad de California lograron crear un algoritmo que, a través del aprendizaje profundo o 'deep learning', fuera capaz de aprender por sí solo a resolver de forma eficiente el cubo, logrando hacerlo con el menor número de movimientos posibles en el 60 % de las ocasiones [3].

La mayor de las incógnitas fue averiguar cuál era el número máximo de movimientos necesarios para resolver el cubo de Rubik en cualquiera de las más de 43 trillones de posiciones posibles. A esta cifra se la conoce con el nombre de 'número de Dios'. El matemático británico Morwen Thistlethwaite probó a través de un complejo algoritmo que el cubo siempre podía resolverse en 52 movimientos o menos. A lo largo de tres décadas, varios matemáticos aceptaron el reto de analizar el problema y fueron utilizando nuevos algoritmos que reducían este número progresivamente: 42 en 1990, 29 en 1995, 23 en 2008 [4]. Finalmente, en el año 2010, un grupo de investigadores logró demostrar que no existía ninguna posición inicial que requiriera más de 20 movimientos, por lo que quedaba establecido que el número de Dios era 20 [5].

1.2. Objetivos

El objetivo de este trabajo es el de desarrollar una librería de funciones para el estudio computacional del cubo de Rubik. Se usará dicha librería para desarrollar heurísticas para la resolución del cubo. Además, se implementará de manera paralela un algoritmo de resolucion del cubo para poder hacer un estudio comparativo entre ambas opciones. En ese estudio comparativo se compararan los costes de ambas opciones para llegar a la resolución del cubo. Cuando hablamos de coste de la solución, a partir de ahora, nos referiremos al número de movimientos necesarios para llegar al cubo resuelto.

Por otro lado, se permitirá que el usuario pueda introducir la configuración de su cubo desordenado (ya sea en un fichero o a través de una interfaz, en este caso la terminal) y se le mostrarán cuáles son los movimientos que necesita aplicar en su cubo para llegar a la posición original de este. A su vez, el usuario podrá decidir el método de resolución que desea, ya sea mediante heurísticas (especificando entre las posibles) o usando el algoritmo mencionado previamente. La solución se guardará automáticamente en un fichero donde se muestra la secuencia de rotaciones necesaria o bien en un fichero la evolución del cubo a través de imágenes como resultado de aplicar los distintos pasos de la solución.

1.3. Metodología

La metodología que se ha llevado a cabo se basa en un diseño donde se separan la parte gráfica de la parte referente a los tipos de datos. De esta manera, la implementación de la representación gráfica es común a todos los métodos de resolución programados. Sin embargo, cada método de resolución está implementado de una manera distinta. Cabe destacar que se usan tipos de datos abstractos, con la utilización de clases de datos que nos permiten tener la información segmentada en las distintas partes del cubo.

Los métodos de resolución del cubo han sido programados de distinta manera. Por una parte tendremos el algoritmo A* a través del cual mediante la implementación de cuatro heurísticas tendremos nuestras primeras maneras de resolución. Por otro lado, se ha implementado el llamado 'algoritmo del principiante', el cual ha sido implementado como un algoritmo por pasos. Con esto nos referimos a que es necesario que se cumplan ciertas condiciones para pasar de un paso a otro, condiciones, que en nuestro caso, son comprobaciones parciales para ver si ciertos cubos están debidamente colocados. Estos métodos serán explicados más en profundidad en los siguientes apartados.

1.4. Estructura del documento

Este trabajo está organizado de la siguiente manera:

- En el apartado 2 se introducen otros juegos o rompecabezas similares al cubo de Rubik, así como un pequeño recorrido sobre las matemáticas que hay detrás y cuáles han sido las librerías empleadas en la programación tanto de la parte gráfica como de la relacionada con las estructuras de datos.
- En el apartado 3 se diferencian dos partes:
    - En la primera parte se explica la manera de representar el cubo gráficamente y cómo se tratan los cubos que forman el cubo principal para poder hacer las rotaciones.
    - En la segunda parte se explican los métodos empleados en la resolución del cubo de Rubik. En primer luegar se describe el algoritmo A* así como las heurísticas empleadas. En segundo lugar se analiza el 'Algoritmo del principiante', desglosando en los distintos pasos que lo componen y su implementación.
- En el apartado 4 se muestran los distintos resultados obtenidos para los dos métodos explicados. Se realiza una comparativa entre ambos para estudiar los costes obtenidos.
- En el último y quinto apartado se exponen cuáles han sido las conclusiones obtenidas y cuáles son las líneas de trabajo futuro así como las limitaciones del proyecto.

2

Estado del arte

En este apartado primero realizaremos una pequeña introducción al mundo de los rompecabezas, donde se explicarán algunos de los puzzles más famosos de la historia así como algunas variantes del propio cubo de Rubik.

Adicionalmente, se explicarán los dos pilares en los cuales se basa este proyecto. Por un lado, tendremos el pilar conceptual, a través del cual podemos ver el cubo de Rubik desde un punto de vista matemático y ver cuáles son los conceptos matemáticos más importantes que se esconden tras este rompecabezas. Por otro lado, se encuentra el pilar técnico, en el cual se detallarán cuáles son las librerías usadas en la realización del código elaborado para llegar a la resolución y representación del cubo de Rubik.

2.1. Otros rompecabezas

La historia de los rompecabezas se remonta a 1760. En torno a estas fechas fue cuando se desarrollaron los primeros rompecabezas comerciales y fueron creados por el cartógrafo y grabador británico John Spilsbury. Éste cogió un mapa de Europa de madera y con una sierra de marquetería cortó las fronteras de los países, creando lo que recibió el nombre de *'mapas diseccionados'*, los cuales serían usados como material educativo ([6]).

Tras este comienzo han sido muchos los distintos puzzles y rompecabezas que se han ido creando a lo largo de la historia, bien como mero entretenimiento, bien como una prueba de la destreza matemática y mental. El auge de los rompecabezas se vivió, sobre todo, después de la Gran depresión que azotó a Estados Unidos en 1929. El desencadenante de este hecho fue el empleo del cartón como material de confección, abaratando el precio de los puzzles, llegando así a más consumidores.

En esta sección veremos algunos de los rompecabezas más populares en la historia a parte del cubo de Rubik, junto con una pequeña descripción del juego. Además cabe mencionar que dada la situación mundial actual los rompecabezas han recuperado popularidad debido al coronavirus.

2.1.1. Stomachion

El juego consiste en un rompecabezas en el que se parte de un cuadrado y se procede a su disección en 14 piezas distintas que lo componen. Lo interesante de este puzzle es que es el considerado el más antiguo del mundo. El objetivo final del juego es, con las 14 piezas procedentes de la disección del cuadrado, formar distintas figuras geométricas, símbolos o incluso imágenes de animales y personas.

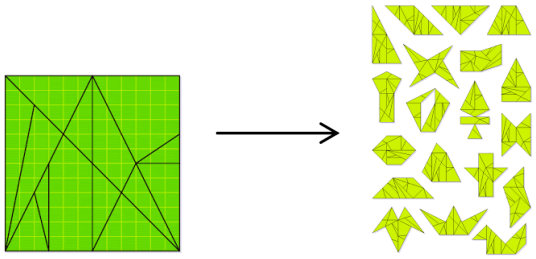


Figura 2.1: A la izquierda tenemos la configuración inicial del juego. A la derecha se ven las posibles soluciones.

2.1.2. Puzzle del quince

Este juego es un puzzle de permutaciones creado por Samuel Loyd (1841 - 1911) y en el cual se aprecian similitudes al rompecabezas del cubo de Rubik, el cual fue posterior. Consiste en una cuadrícula de 16 casillas, 4x4, en la que se numeran del 1 al 15 dejando una de las casillas vacía. La dinámica del juego consiste en mover los números mediante deslizamientos a la casilla vacía, para lo cual dichos números se han de encontrar en posiciones adyacentes al lugar de desplazamiento. Se parte de una posición en la que los números están desordenados. El objetivo final del juego es obtener la posición original de la cuadrícula, en la que los quinces números estén ordenados de menor a mayor dejando vacía la decimosexta casilla.

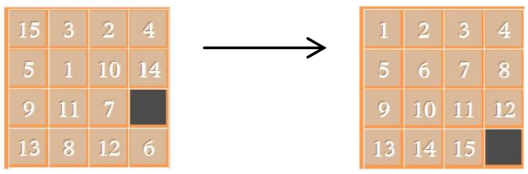


Figura 2.2: A la izquierda tenemos una posible configuración aleatoria incial del juego. A la derecha se observa cómo es la posición final a la que se ha de llegar.

2.1.3. Pyraminx

El Pyraminx es un rompecabezas muy similar al cubo de Rubik y que fue posterior a este. Tiene forma de tetraedro y fue inventado por Uwe Meffert en 1970. El rompecabezas al ser un tetraedro

está compuesto de 4 caras, cada una de las cuales está dividida en 9 triángulos de menor tamaño. El objetivo final no es otro que el mismo que en el cubo de Rubik, obtener las cuatro caras de un solo color. Los movimientos y rotaciones son distintos a los del cubo pero se pueden expresar como permutaciones y de la misma manera que se explicará en secciones futuras.

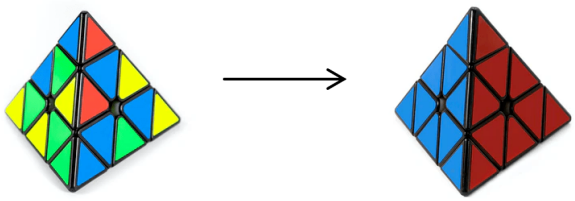


Figura 2.3: A la izquierda tenemos una posible configuración aleatoria incial del juego. A la derecha se observa cómo es la posición final a la que se ha de llegar.

2.1.4. Cubo de Rubik 2x2

Por último, vamos a explicar brevemente el cubo de Rubik 2x2, ya que nuestro proyecto se basa en la resolución del cubo 3x3. Al igual que el cubo 3x3, este cubo tiene 6 caras de disntintos colores, pero cada una de ellas en lugar de estar dividida en 9 cuadrados, lo está en 4, haciendo un total de 24 cuadrados. Otra de las similitudes es que los movimientos que se explicarán en la siguiente sección y que se encuentran disponibles en el Anexo A, son los mismos para el cubo de 3x3 que para el de 2x2.

La diferencia más importante de un cubo respecto a otro es que en el cubo 2x2 no existen lo que denominaremos como 'aristas' (cubito formado pos 2 caras de colores distintos). En este cubo sólo hay lo que denominamos como 'esquinas', es decir 8 cubitos donde cada uno de ellos tiene 3 caras de distintos colores.

Dado que está compuesto por 8 esquinas, existen $8!$ posibilidades de que estas ocupen distintas posiciones. Puesto que la colocación de 7 de estas esquinas supone la automática determinación de la octava, tenemos un factor de 3^7 posibilidades de orientación. Además, como cualquiera de las 6 caras puede ser la base en la que se apoya el cubo, se pueden presentar 4 caras distintas, lo que reduce las posiciones en un factor de 24. Así, obtenemos que las posibles configuraciones de este cubo son:

$$\frac{8! \cdot 3^7}{24} = 3{,}674{,}160$$

En la siguiente figura se muestra el cubo de Rubik 2x2, donde podemos ver que únicamente está compuesto de esquinas y que tiene 4 cuadrados por cada una de sus caras. Este sería el cubo original, el cual sería el objetivo a alcanzar desde una configuración aleatoria:

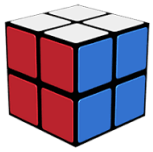


Figura 2.4: Cubo de Rubik 2x2.

2.2. Definición del juego

En esta sección describiremos la composición del cubo de Rubik así como sus posibles rotaciones y movimientos. Además de explicar cuál es el objetivo final del juego.

El objetivo del rompecabezas consiste en partiendo de una posición aleatoria de todas las posibiladades vistas anteriormente, encontrar los movimientos necesarios para llegar a que los 54 cuadrados estén colocados en su cara correspondiente de entre las 6 existentes, teniendo cada cara un único color.

2.2.1. Composición del cubo de Rubik

El cubo de Rubik está compuesto por tres tipos de piezas: las que ocupan las esquinas, las que están en las aristas y las que se encuentran en los centros. Tenemos 8 esquinas en el cubo, las cuales se pueden combinar entre sí de cualquier forma, ocupando las posiciones unas de otras, por lo que da lugar a $8!$ posibilidades. Por otro lado, con las aristas pasa de igual modo, por lo que tenemos $12!$ posibilidades. Sin embargo, la permutación de esquinas y de aristas ha de ser par, lo cual nos elimina la mitad de las posibilidades. Por otro lado, hemos de contemplar las distintas orientaciones de cada esquina. Para ello, sabemos que todas las esquinas pueden ser rotadas como queramos, menos la última que quedará determinada por la posición de las otras 7. Con todo ello, se obtienen 3^7 posibilidades de orientación de esquinas. De la misma manera ocurre con las aristas, teniendo así 2^{11} posibilidades debido a las 2 caras que componen cada una de las aristas y a las 11 aristas que tenemos para orientar (dado que la doceava queda determinada por el resto, no es necesario tenerla en cuenta en el cálculo). Es así como se puede deducir el número total de permutaciones, o lo que es lo mismo, el número total de posiciones en las que puede encontrarse el cubo de Rubik es:

$$\frac{8! \cdot 12! \cdot 3^7 \cdot 2^{11}}{2} = 43{,}252{,}003{,}274{,}489{,}856{,}000$$

Cabe mencionar, que si tenemos dos configuraciones que son equivalentes en el sentido matemático, las trataremos como la misma. Con esto nos queremos referir a que si una configuración se puede conseguir de la otra simplemente rotando todo el cubo, las trataremos igual. Es por ello, que las

rotaciones que implican todo el cubo no van a ser de nuestro interés, prestando atención únicamente al conjunto cociente por relación de equivalencia que definimos como C/R. Dicho conjunto cociente está definido por C como el conjunto de configuraciones y R como el grupo de rotaciones del cubo entero en los tres ejes.

De cara a la posterior implementación de la representación del Cubo de Rubik, tendremos en cuenta los siguientes aspectos de la composición del cubo de Rubik:

- El cubo está compuesto de 27 cubos de un tamaño menor (3 a lo alto x 3 a lo ancho x 3 a lo largo), a los cuales nos referiremos como cubitos a partir de ahora.
- Cada uno de los 27 cubitos mencionados se puede clasificar en:
  - Esquina: si tiene 3 cuadrados que lo componen con tres colores distintos.
  - Arista: si tiene 2 cuadraditos con dos colores distintos.
  - Centro: si tiene 1 cuadradito con un único color, las cuales guardan siempre la misma posición relativa entre ellas.
- El cubo tiene un total de 6 caras, que se pueden clasificar como 4 laterales y 2 topes (la que se encuentra en la parte superior y la de la parte inferior).
- En cada una de las caras tenemos 9 cuadrados (3 a lo alto x 3 a lo ancho) que tendrán un color asignado, por lo que en total habrá 54 cuadrados en todo el cubo (9 cuadrados x 6 caras).
- Hay un total de 6 colores distintos (rojo, amarillo, verde, azul, naranja y blanco), tantos como caras tiene el cubo de Rubik.

2.2.2. Notación y movimientos del cubo

Para llegar a la solución del cubo una de las cosas más importantes es definir una correcta notación empleada para referirnos a los posibles movimientos que se pueden realizar. En este trabajo se asociarán estas caras con unos colores desde el comienzo, tomando como referencia la notación empleada en [7]. Esta asociación se realiza para poder resolver el cubo de una manera más sencilla. Sin embargo, dicha asignación no afecta en nada a la inserción del cubo inicial el cual queremos resolver. Esta asociación de caras con colores es la siguiente:

- Cara frontal (Front): F $\rightarrow$ Rojo
- Cara trasera (Back): B $\rightarrow$ Naranja
- Cara derecha (Right): R $\rightarrow$ Azul
- Cara izquierda (Left): L $\rightarrow$ Verde
- Cara de arriba (Up): U $\rightarrow$ Blanco

- Cara de abajo (Down): D → Amarillo

En la siguientes figuras se pueden ver cómo es la asociación de estos colores y sus caras.

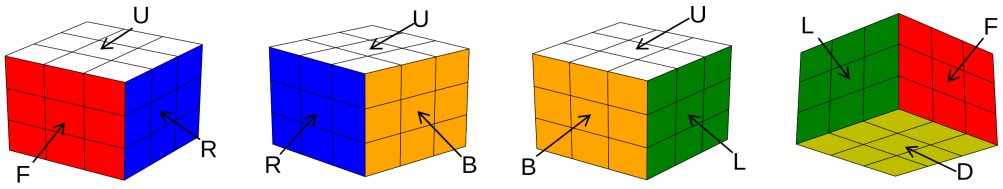


Figura 2.5: Asignación caras cubo

En cuanto a los movimientos, vamos a referirnos al movimiento de una cara con la misma letra a la cual nos referíamos a la cara implicada. Estas letras significarán un giro de un cuarto de vuelta, de 90 grados, en la dirección de las agujas del reloj. Por el contrario si esta letra va acompañada de un '-1' significa que el movimiento es inverso, por lo que será un giro de un cuarto de vuelta pero en la dirección contraria a las agujas del reloj (-90 grados). De esta manera, cuando veamos a lo largo del trabajo una secuencia de letras mayúsculas, nos estaremos refiriendo a una secuencia de movimientos que se realizarán de manera sucesiva sobre el mismo cubo. Se muestran con mayor detalle los distintos movimientos junto con sus respectivas inversas, gracias a unas representaciones gráficas, en el Anexo A. En él se pueden apreciar cómo son los movimientos normales, es decir, los que se realizan en sentido horario. Además, se adjuntan los mismos movimientos pero en sentido antihorario, es decir la inversa. Cabe señalar que si realizásemos el movimiento inverso después del normal, volveríamos al cubo original.

2.3. La matemáticas del cubo de Rubik

El cubo de Rubik también puede ser estudiado desde un punto de vista matemático, el cual, como se mencionó anteriormente, va a ser uno de los pilares de este trabajo. Para ello, se hace uso de una de las herramientas más importantes en el ámbito matemático. Esta herramienta en cuestión es la conocida como 'Teoría de grupos' y, más concretamente, el concepto de grupo, desarrollado por Evariste Galois [8]. A continuación, expondremos cómo se puede aplicar esta definición al cubo de Rubik, así como otras propiedades y curiosidades matemáticas acerca de él. Algunas de estas propiedades y características nos servirán de cara al futuro para poder representar nuestro cubo y poder realizar los movimientos y rotaciones correctamente.

2.3.1. El cubo de Rubik visto como un grupo

El expresar el cubo de Rubik como un grupo matemático nos permitirá simplificar la notación y poder aplicar propiedades que nos facilitarán el tratamiento de este. Antes de ver si el cubo de Rubik

puede considerarse un grupo, vamos a hacer una breve introducción a la definición de grupo y las propiedades básicas que este concepto implica ([9]).

¿Qué es un grupo?

Un grupo, (*G*, *), consta de un conjunto *G* y una operación * tal que es necesario que se cumplan los siguientes requisitos:

1.– *G* es cerrado bajo la operación *. Esto significa que si $a, b \in G$ entonces $a * b \in G$.

2.– La operación * es asociativa. Esto es, para cualquier $a, b, c \in G$, $a * (b * c) = (a * b) * c$.

3.– Hay un llamado 'elemento identidad', $e \in G$ el cual satisface lo siguiente: $g = e*g = g*e$ para todo $g \in G$.

4.– Existen los elementos inversos. Esto es, para cualquier $g \in G$, existe un elemento $h \in G$ tal que $g * h = h * g = e$. Es así como decimos que h es el inverso de g y viceversa.

Adicionalmente existe una propiedad la cual no es necesaria para ser un grupo, pero algunos grupos la presentan. Dicha propiedad se conoce como 'grupo abeliano'. Consideraremos que un grupo es abeliano si se cumple la propiedad conmutativa entre los elementos del grupo *G*.

Ahora la principal duda que surge una vez ya hemos visto cuál es la definición de grupo es si el conjunto de movimientos del cubo de Rubik se puede considerar un grupo. A continuación, veremos que sí.

A partir de ahora denotaremos al conjunto de movimientos posibles en el cubo de Rubik como *G*, siendo los elementos de este conjunto cada una de las posibles rotaciones explicadas anteriormente. Por otra parte, falta por definir cuál sería la operación de nuestro grupo. La operación del grupo quedará definida como: si M_1 y M_2 son dos movimientos, entonces $M_1 * M_2$ es el movimiento resultante de hacer primero M_1 y después M_2. Para comprobar que verdaderamente este conjunto *G* y esta opración, *, forman un grupo, hemos de comprobar las cuatro propiedades definidas previamente y que son de necesario cumplimiento para cualquier grupo. A continuación trataremos una por una.

1.– Está claro que el grupo *G* es cerrado, ya que si M_1 y M_2 son movimientos, $M_1 * M_2$ también es un movimiento.

2.– La operación * es asociativa. Cada movimiento está determinado por la posición y la orientación en la que coloca cada uno de los cubitos involucrados (recordemos que había 27 cubitos en el cubo). Así, si C es un cubito orientado escribiremos $M(C)$ para el cubito orientado resultante de aplicar M a C. Además, las caras deberán estar escritas en el mismo orden, es decir, la primera cara de C debe de terminar en la primera cara de $M(C)$ y así sucesivamente. Por ejemplo, el cubito que es la arista 'ur' (tiene una cara en la cara U y una en la cara R), depués de aplicar el movimiento R tenemos que acaba en la posición 'br'. En definitiva, R(ur) = br, donde la cara antes en U pasa a estar en B y la cara que se

encontraba en R se mantiene en esa cara.

Cabe mencionar que si M_1 y M_2 son dos movimientos, entonces $[M_1 * M_2](C)$ es el resultado de aplicar primero M_1 y después M_2. Por lo que, $[M_1 * M_2](C) = M_2(M_1(C))$.

Para mostrar que es asociativo, tenemos que mostrar que $(M_1 * M_2) * M_3 = M_1 * (M_2 * M_3)$ para cualquier movimiento M_1, M_2 y M_3. O lo que es lo mismo, $[(M_1 * M_2) * M_3](C) = [M_1 * (M_2 * M_3)](C)$ para cualquier cubito C. Aplicando lo visto en el párrafo anterior, sabemos que $[(M_1 * M_2) * M_3](C) = M_3(M_2(M_1(C)))$. Por otro lado, $[M_1 * (M_2 * M_3)](C) = (M_2 * M_3)(M_1(C)) = M_3(M_2(M_1(C)))$. Así que se cumple que $(M_1 * M_2) * M_3 = M_1 * (M_2 * M_3)$, concluyendo que * es asociativa.

3.– Si definimos el 'elemento identidad' como el movimiento vacío, es decir, un movimiento que consiste en dejar el cubo como está (o lo que es lo mismo, no hacer nada), entonces se cumplirá la propiedad siguiente: $M * e = e * M = M$. Con ello se concluye que (*G*, *) tiene un elemento identidad correcto.

4.– Comprobemos que se cumple la existencia de los elementos inversos. Si M es un movimiento, podemos deshacer este movimiento para volver a la posición previa, M^{-1}. Así, el movimiento $M * M^{-1}$ significa que primero se hace M y después se realiza M^{-1}, o lo que es lo mismo, deshace M. Así, podemos decir que M^{-1} es el elemento inverso de M, ya que $M * M^{-1} = e$. Concluímos, que todo elemento de *G* tiene un elemento inverso.

Una de las mayores ventajas de los grupos es que en vez de manejar el grupo entero del cubo de Rubik, ya que como vimos anteriormente había millones de cubos distintos, podemos manejar partes más pequeñas de este cojunto. Estas partes más pequeñas son lo que se denominan subgrupos de un grupo. Por lo tanto, si tenemos un subconjunto no vacío *H* del grupo (*G*, *) se llama **subgrupo** de *G* si (*H*, *) es un grupo.

2.3.2. Generadores del cubo de Rubik

Uno de las nociones más importantes para poder hablar de subgrupos dentro de grupos son los conocidos como **generadores del grupo**. Vamos a definir qué son estos generadores y cómo van a afectar al tema que nos ocupa, el cubo de Rubik.

Sea *G* un grupo y *S* un subconjunto de *G*. Decimos que *S* genera *G* o que *S* es un conjunto de generadores de *G* si $G = \langle S \rangle$, es decir, cada elemento de *G* puede escribirse como un producto finito (bajo la operación del grupo) de elementos de *S* y sus inversas.

El grupo de movimientos del cubo de Rubik, *G*, puede estar escrito como una secuencia finita de movimientos que son rotaciones del cubo de Rubik, por lo que $G = \langle D, U, L, R, F, B \rangle$, que como vemos son los movimientos básicos que definimos previamente para el cubo ([10]).

Ahora bien, sea $S = \{ D, U, L, R, F, B \} \subset G$. Entonces, cada $M \in S$ satisface la siguiente propiedad:

"*M* mantiene todo cubo central en su cubito". Si M_1, $M_2 \in G$ son tales que mantienen todos los cubos centrales en sus cubitos, luego M_1M_2 también mantiene todos los cubos centrales en sus respectivos cubitos. Esta propiedad es extremadamente útil para el cubo de Rubik, porque significa que con frecuencia solo necesitamos comprender las propiedades de los 6 movimientos básicos en lugar de los $5 \cdot 10^{20}$ posibles movimientos.

2.3.3. Cubo de Rubik visto con simetrías

Como mencionamos anteriormente, para encontrar el número de configuraciones posibles del cubo de Rubik, lo que usamos es el hecho de que cualquier movimiento envía cubitos que se encuentran en esquinas a otras esquinas. Así se determinaba que había 8 posibilidades para colocar los cubitos en las esquinas. En este apartado lo que haremos será explicar la base matemática necesaria para comprender este hecho.

En lugar de mirar solo las configuraciones de 8 cubitos, vamos a generalizarlo a situaciones donde tenemos n objetos (en este caso estos objetos son nuestros cubitos), a los cuales nos referiremos como de $1, 2, ..., n$. Es así como definimos la siguiente función:

$$\sigma : \{1, 2, ..., n\} \to \{1, 2, ..., n\}$$

donde tenemos que $\sigma(i)$ es el número que se encuentra en la posición i. Es importante mencionar que σ es una biyección, es decir, si suponemos que $x \neq y$ entonces $\sigma(x) \neq \sigma(y)$.

Por otra parte, tenemos el llamado *Grupo simétrico*, el cual es un conjunto de biyecciones de $\{1, 2, ..., n\}$ a $\{1, 2, ..., n\}$, con la operación de la composición. Este grupo se denota matemáticamente como S_n. Por ejemplo, sean $\sigma, \tau \in S_3$ definidas como:

$$\sigma(1) = 3 \qquad \sigma(2) = 1 \qquad \sigma(3) = 2$$
$$\tau(1) = 1 \qquad \tau(2) = 3 \qquad \tau(3) = 2$$

Entonces tendremos como resultado de operar con ambas lo siguiente:

$$(\sigma\tau)(1) = \tau(3) = 2 \qquad (\sigma\tau)(2) = \tau(1) = 1 \quad \text{y} \quad (\sigma\tau)(3) = \tau(2) = 3$$

Otro concepto importante acerca de la notación, es el concepto de *ciclo*. El ciclo $(i_1, i_2...i_k)$ es el elemento $\tau \in S_n$ definido por $\tau(i_1) = i_2$, $\tau(i_2) = i_3$, ..., $\tau(i_{k-1}) = i_k$, $\tau(i_k) = i_1$, y tenemos que $\tau(j) = j$ si $j \neq i_r$ para ningún r. Se dice además que dos ciclos son disjuntos si no tienen números en común, y entonces (y solo entonces) se cumple que $\sigma\tau = \tau\sigma$.

La pregunta que nos surge ahora es: ¿cómo podemos trasladar estas nociones al cubo de Rubik?

Nuestro objetivo es describir qué le sucede a cada uno de los cubitos orientados, es decir, quere-

mos describir dónde se mueve cada cubito y dónde queda cada una de las caras que lo componen. Por ejemplo, vamos a desplegar el cubo poniendo como cara principal la cara de abajo (Down). Si giramos 90º esta cara en el sentido de las agujas del reloj, es decir, aplicamos el movimiento D, entonces la cara inferior quedaría de la siguiente manera: Por ello, podemos aplicar los conocimientos que

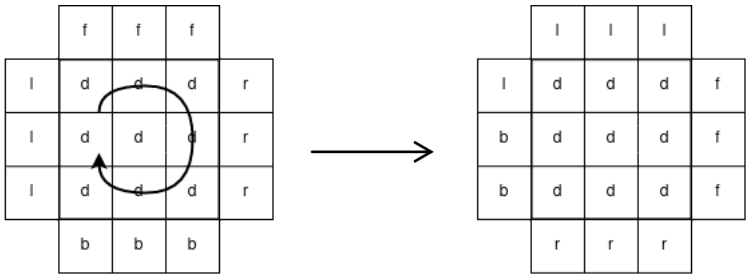


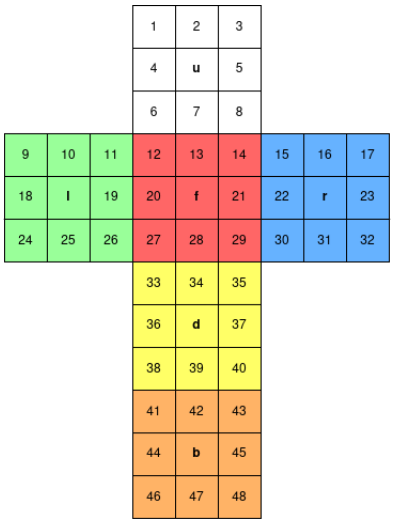
Figura 2.6: Movimiento D aplicado a permutaciones

hemos visto de los ciclos. Así, $D(dlf) = dfr$, porque el cuito dlf ahora se encuentra en la posición del cubito dfr. Esto significa que la cara del cubito que estaba en la cara de abajo (Down) permanece en esta cara; la cara que está en la cara izquierda (Left) pasa a estar en la cara frontal (Front); y, por último, la cara del cubito que estaba en la cara frontal (Front) pasa a estar en la cara derecha (Right). De manera similar, $D(dfr) = drb$, $D(drb) = dbl$ y $D(dlb) = dlf$. Si se aplica lo mismo a los cubitos que son aristas tenemos, en cojunto, lo siguiente:

$$U = (ulb\ ubr\ urf\ ufl)(ub\ ur\ uf\ ul)$$
$$D = (dlf\ dfr\ drb\ dbl)(df\ dr\ db\ dl)$$
$$L = (lbu\ luf\ lfd\ ldb)(lu\ lf\ ld\ lb)$$
$$R = (rfu\ rub\ rbd\ rdf)(ru\ rb\ rd\ rf)$$
$$F = (flu\ fur\ frd\ fld)(fu\ fr\ fd\ fl)$$
$$B = (bru\ bul\ bld\ bdr)(bu\ bl\ bd\ br)$$

Otra forma de ver el comportamiento de los cuadraditos en el cubo de Rubik mediante el empleo de ciclos es numerando cada uno de dichos cuadraditos (salvo el central dado que permanece siempre en el mismo lugar) tal y como se muestra en la imagen de abajo.

En la siguiente imagen se puede ver el cubo de Rubik desplegado en las distintas caras a modo de cruz y donde lo que antes tratábamos como las letras correspondientes a su cara ahora toman un único valor para poder seguir de una manera más clara el cambio producido por los distintos movimientos.



Así las rotacioens pueden ser expresadas de la siguiente manera:

$$U = (1\ 3\ 6\ 8)(2\ 5\ 7\ 4)(9\ 48\ 15\ 12)(10\ 47\ 16\ 13)(11\ 46\ 17\ 14)$$

$$D = (33\ 35\ 40\ 38)(34\ 37\ 39\ 36)(27\ 30\ 43\ 24)(28\ 31\ 42\ 25)(29\ 32\ 41\ 26)$$

$$L = (9\ 11\ 26\ 24)(10\ 19\ 25\ 18)(1\ 12\ 33\ 41)(4\ 20\ 36\ 44)(6\ 27\ 38\ 46)$$

$$R = (15\ 17\ 32\ 30)(16\ 23\ 31\ 22)(8\ 48\ 40\ 29)(5\ 45\ 37\ 21)(3\ 43\ 35\ 14)$$

$$F = (12\ 14\ 29\ 27)(13\ 21\ 28\ 20)(6\ 15\ 35\ 26)(7\ 22\ 34\ 19)(8\ 30\ 33\ 11)$$

$$B = (41\ 43\ 48\ 46)(42\ 45\ 47\ 44)(3\ 9\ 33\ 32)(2\ 18\ 39\ 23)(1\ 24\ 40\ 17)$$

Es así, como podemos extraer de estas nociones que el cubo de Rubik se puede escribir como un conjunto de permutaciones. Como el cubo en sí tiene 54 caras o facetas, podemos afirmar (usando el subgrupo que vimos anteriormente) que $G = \langle\ D, U, L, R, F, B\ \rangle \in S_{54}$.

2.3.4. Teoremas y propiedades de interés

Algunos de los siguientes teoremas y propiedades que veremos a continuación, nos serán de utilidad a la hora de implementar nuestro cubo de Rubik, tanto a la hora de los movimientos como de la representación gráfica. Por otro lado, otras de las nociones que aparecerán en esta sección son mencionadas dado su alto interés en el estudio del Grupo del Cubo de Rubik ([11]).

Antes de mencionar dichos teoremas, cabe señalar que se pueden realizar dos “clasificaciones” diferentes del Grupo del Cubo de Rubik: Grupo del Cubo de Rubik “Legal” y el Grupo del Cubo de Rubik “Ilegal”. La diferencia principal entre ambos radica en que el Grupo del Cubo de Rubik “Ilegal” permite al usuario que trata de resolverlo desarmar el cubo en su totalidad y volverlo a armar en una posición distinta, eso sí, sin cambiar los colores de cada cubito. Por el contrario, el Grupo del Cubo de Rubik “Legal” no permite realizar este desarme. Es por ello que, como veremos matemáticamente más

adelante, el Grupo del Cubo de Rubik “Legal” es un subgrupo del “Ilegal”.

Es importante destacar que no todas las permutaciones de S_{54} son situaciones posibles del cubo de Rubik, porque los cubos tienen que ir a posiciones que ocupan cubitos del mismo tipo. Además, el grupo $G = \langle$ *D, U, L, R, F, B* $\rangle$ no es isomórfico al grupo entero de permutaciones S_{54}, ya que para que lo fuera debería de tener el mismo número de elementos y no es así, ya que como mencionamos anteriormente el cubo de Rubik tiene alrededor de 43 trillones de posibilidades (43 trillones de elementos) mientras que S_{54} su orden es de $54!$ que es superior a la cifra anterior.

A continuación, vamos a describir cuáles son los grupos que describen las esquinas y aristas. Dado que cada esquina está compuesta por 3 caras que pueden estar en distintas posiciones, podemos representar cada esquina como el grupo C_3, grupo cíclico compuesto por 3 elementos. Como el número total de esquinas es de 8, tenemos que cualquier esquina estará en el conjunto C_3 x C_3 x C_3 x C_3 x C_3 x C_3 x C_3 x $C_3 = C_8$. Adicionalmente, también podemos usar el grupo S_8 para describir la posición de todas las facetas de las esquinas. De igual modo ocurre con las aristas, siendo C_2 x C_2 x C_2 x C_2 x C_2 x C_2 x C_2 x C_2 x C_2 x C_2 x C_2 x $C_2 = C_{12}$ o S_{12} para describir cualquier arista. Debido a esto es por lo que podemos expresar cada posición del cubo en términos de esquinas como una tupla de 8 elementos y en términos de aristas con una tupla de 12 elementos. Uniendo esto con la “clasificación” previa, podemos determinar que el Grupo del Cubo de Rubik “Ilegal” $= (C_2^{12}$ ó $S_{12})$ x $(C_3^8$ ó $S_8)$.

Para poder distinguir entre el Grupo de Cubo “Legal” e “Ilegal”, se necesitan los siguientes teoremas que vamos a enunciar: el Primer y Segundo Teoremas Fundamentales de la Teoría del Cubo.

El Primer Teorema Fundamental permite dar un “arreglo” solucionable al Cubo de Rubik, dado que el Grupo del Cubo de Rubik permite deshacer y montar de nuevo el cubo, pero no asegura que la nueva configuración tenga una nueva solución. Por ello enunciamos el siguiente teorema.

Primer Teorema Fundamental de la teoría del cubo. Sea $v \in C_3^8, r \in S_8, w \in C_2^{12}$ y $s \in S_{12}$.La tupla de 4 elementos (v, r, w, s) corresponde a una posible disposión (posición) del cubo si y solo si:

1.– signo(r) = singo(s) (paridad de una permutación)

2.– v_1 + v_2 + v_3 + ... + v_8 = 0 (mod 3) (conservación del número total de esquinas).

3.– w_1 + w_2 + w_3 + ... + w_8 = 0 (mod 2) (conservación del número total de aristas)

Por otro lado, tenemos el Segundo Teorema Fundamental de la teoría del cubo el cual aporta los criterios para los movimientos legales en el cubo de Rubik.

Segundo Teorema Fundamental de la teoría del cubo. Una operación del cubo de Rubik es posible si y solo si se cumple lo siguiente:

1.– El número total de ciclos de esquinas y aristas de longitud uniforme es uniforme.

2.– El número de ciclos esquina torcidos en la derecha ha de ser igual al número de ciclos

esquina de la izquierda (módulo 3).

3.– Hay un número par de ciclos de bordes de reorientación.

Si usamos ambos teoremas descritos anteriormente podemos reducir el Grupo del Cubo de Rubik "Ilegal" a $G_0 = \{(v, r, w, s) | v \in C_3^8, r \in S_8, w \in C_2^{12}$ y $s \in S_{12}\}$, donde G_0 cumple las propiedades descritas anterioremente.

Como dijimos anteriormente, el Grupo del Cubo de Rubik "Ilegal" también podía definirse como $I = (C_2^{12}$ ó $S_{12})$ x $(C_3^8$ ó $S_8)$, pero como las colocación de 11 aristas determina la posición de la doceava, así como la posición de las 7 esquinas determina la de la octava esto reduce el orden de nuestros subgrupos. Es así como podemos establecer el siguiente isomorfismo, donde reducimos C^8 a C^7 y C^{12} a C^{11}, siendo el siguiente:

$$G_0 \cong (C_3^7 \text{ ó } S_8) \text{ x } (C_2^{11} \text{ ó } S_{12})$$

Además se cumple: $|G_0| = |S_8||S_{12}||C_2^{11}||C_3^7| = 8! \cdot 12! \cdot 2^{11} \cdot 3^7$.

Finalmente, G se puede expresar como el grupo reducido G_0. El apartado 1 del Primer Teorema Fundamental de la teoría del cubo establece que el número de permutaciones pares e impares es igual, con lo que G_0 ha de reducirse un factor de C_2. Por lo que la expresión final para denotar al cubo de Rubik es la siguiente:

$$G = (C_3^7 \text{ ó } S_8) \text{ x } (C_2^{10} \text{ ó } S_{12})$$

Gracias a esta expresión conocemos un poco más cuál es la composición del cubo de Rubik, nociones que nos sirvieron de base de cara a la representación y manejo de los movimientos del cubo de Rubik, así como para llegar a su solución en uno de los métodos descritos posteriormente (A*).

2.4. Librerías de python empleadas

El proyecto ha sido implementado en su totalidad en el lenguaje de programación de python. Se ha optado por este lenguaje debido a las grandes ventajas que ofrece. Entre las más significativas cabe destacar que python cuenta con *frameworks* que permiten realizar desde páginas webs hasta el poder desarrollar algoritmos de cálculos avanzados. Otra de las ventajas más importantes y de la cual nos vamos a beneficiar es que python permite una programación orientada a objetos. De esta manera podremos orgranizar el código en función de clases y objetos.

Las principales librerías y paquetes que se han usado en este proyecto han sido las siguientes:

- **numpy**. Se trata de una librería especializada en el cálculo numérico y análisis de datos. Gracias a esta, tenemos una gran cantidad de funciones matemáticas de alto nivel, las cuales

nos permitirán, entre otras cosas, calcular y operar con las coordenadas del cubo de Rubik para poder hallar las coordenadas resultantes de realizar cualquier movimiento sobre este. Además nos va a permitir almacenar datos de una manera más eficiente y de más fácil acceso de cara a la posterior utilización de estos mediante el empleo de arrays, listas, matrices...

- **matplotlib.pyplot**. Es un cojunto de funciones procedente de la librería de *matplotlib* donde cada una de ellas permite realizar distintos tipos de cambios. Dicha librería será utilizada a la hora de representar gráficamente nuestro cubo de Rubik y poder mostrar los diferentes movimientos sobre este. Entre las funciones más usadas podemos resaltar: *draw*, que nos permite mostrar la figura que estamos creando; *pause*, que nos permite introducir pausas entre un movimiento y otro; o *savefig*, la cual nos permite guardar imágenes de nuestra figura cuando deseemos, lo cual será utilizado para guardar los pasos gráficos de nuestra solución.
- **mpl_toolkits.mplot3d** ([12]). Se trata de un conjunto de funciones que proporciona herramientas básicas a la hora de la representación en 3D. Quizás no se trata de la biblioteca 3D más completa y moderna que existe, pero está conectada con *matplotlib* lo cual nos facilita bastante ciertos puntos de la implementación. De dicho conjunto de funciones, se importarán *Axes3D* y *Poly3DCollection* con sus correspondientes funciones. La primera de ellas, *Axes3D*, permite crear un objeto que contendrá los ejes en 3D, lugar donde posteriomente añadiremos nuestro cubo gracias al otro tipo de objetos de esta librería. El segundo, *Poly3DCollection* contiene una colección de polígonos 3D junto con una gran variedad de funciones que permiten modificar el color de la figura, las coordenadas... Este conjunto de funciones será el que, principalmente, va a permitir que se pueda ver el cubo, así como sus distintos movimientos.

Gracias a estas librerías, junto con las funciones y objetos que engloban, así como a la orientación a objetos de python que nos permite emplear clases, hemos podido implementar un código que nos permite alcanzar la resolución del cubo de Rubik y representarlo gráficamente de una manera eficiente.

3

Representación y algoritmo

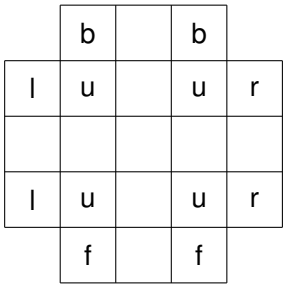
En esta sección vamos a explicar la representación empleada en el proyecto para mostrar el cubo de Rubik, así como para la realización de las rotaciones y movimientos llevados a cabo en él. Por otro lado, se procederá a la explicación de los algoritmos implementados para llegar a la solución de distintas maneras.

El propósito y objetivo final consiste en alcanzar la secuencia de movimientos, que devuelven el cubo a la posición original, de diferentes maneras para así poder hacer una posterior comparación en términos de coste. En este trabajo, mediremos el coste de alcanzar la solución contando el número de movimientos que son necesarios para que todas las caras del cubo sean del mismo color. Sin embargo, este tema se tratará más adelante, ya que en este capítulo daremos una explicación teórica a la manera de alcanzar dichas soluciones, así como su forma de representarlas.

3.1. Representación del cubo

En este apartado vamos a explicar la representación que se ha empleado para el cubo de Rubik en cada uno de los dos algoritmos programados, ya que por cuestiones de memoria e implementación no ha sido el mismo. Se decidió abordar de dos maneras distintas ya que en una se quería dar mayor peso a la interfaz gráfica mientras que en la otra se quería hacer uso de las nociones de permutaciones explicadas en apartados anteriores.

La representación del cubo en el algoritmo A* se hace tomando por separado las esquinas y las aristas. Una esquina del cubo (posición con respecto al cubo) se representa enumerando sus caras en el sentido de las agujas del reloj. Por ejemplo, si esta es la cara superior:



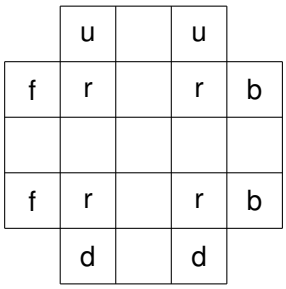
Las cuatro esquinas son ulb, ubr, urf, ufl. Obsérvese que ubr, rub, y bru representan el mismo cubo, pero en diferentes orientaciones. Esta situación se debe a que al haber 3 caras que componen cada esquina, podemos encontrar 3 posiciones distintas de ella y, por ello, en lo mencionado anteriormente hacemos uso de una de las propiedades de las permutaciones descritas previamente. La representación del cubo en su posición original simplemente enumera todas las esquinas y aristas.

$$Esquinas = (ufl \;\; urf \;\; ubr \;\; ulb \;\; dbl \;\; dlf \;\; dfr \;\; drb)$$
$$Aristas = (uf \;\; ur \;\; ub \;\; ul \;\; fl \;\; fr \;\; br \;\; bl \;\; df \;\; dr \;\; db \;\; dl)$$

Tenga en cuenta que la representación es una lista: el orden es importante. Todas las representaciones del cubo revuelto tendrán los cubos en la misma posición, y las orientaciones son relativas a ésta. Así, una representación de una configuración en la que el segundo elemento es *bru* significa que el cubo en posición *urf* (segundo elemento en la representación inicial) se ha desplazado a *bru*. Esto no implica un cambio de posición desde la posición frontal superior derecha a la superior derecha-posterior, sino una orientación precisa: las caras han sido desplazadas como

$$u \to b$$
$$r \to r$$
$$f \to u$$

Los movimientos se codifican por el cambio que provocan en la configuración estándar. Por ejemplo, tomar la cara derecha



El movimiento R es (los cubos que no se cambian no se mencionan aquí):

$$\text{urf} \rightarrow \text{bru}$$
$$\text{ubr} \rightarrow \text{bdr}$$
$$\text{drb} \rightarrow \text{frd}$$
$$\text{dfr} \rightarrow \text{fur}$$
$$\text{ur} \rightarrow \text{br}$$
$$\text{br} \rightarrow \text{dr}$$
$$\text{dr} \rightarrow \text{fr}$$
$$\text{fr} \rightarrow \text{ur}$$

La orientación importa ya que veamos por ejemplo los movimientos $R * R$. El movimiento R es el siguiente:

$$\text{urf} \rightarrow \text{bru}$$

Esta es la misma posición que ubr, pero en una orientación diferente, por lo que el segundo R no lo moverá como si estuviera en la orientación original. Para obtener ubr, debemos girar bru una vez en sentido antihorario, por lo que el resultado de la segunda R también se girará una vez en sentido contrario a las agujas del reloj:

$$\begin{array}{ccc} \text{ubr} & \rightarrow & \text{bdr} \\ & \Downarrow & \\ \text{bru} & \rightarrow & \text{drb} \end{array}$$

Así que el movmiento $R * R$ queda como sigue:

$$\text{urf} \rightarrow \text{bru} \rightarrow \text{drb}$$

Es decir, las caras se mueven de la siguiente manera:

$$u \rightarrow d$$
$$r \rightarrow r$$
$$f \rightarrow b$$

Lo mismo ocurre con el resto de los movimientos.

También hemos implementado el Algoritmo del Principiante [13] (Anexo B) donde hemos optado por utilizar la estructura de datos ya definida para la interfaz gráfica. Esto se debe a que no es necesario realizar una búsqueda en anchura ni ir probando distintos casos y caminos para llegar a la solución, ya que como se explicará más adelante es un algoritmo donde en función de la situación que esté se aplicará una secuencia de movimientos u otra. A la hora de usar este procedimiento de resolución no se

ha usado ningún concepto matemáitco, ya que se manejan las posiciones de cada uno de los cubitos en los ejes (x,y,z) así como sus colores y caras. Finalmente, se devuelve la secuencia de movimientos ya en su versión final para resolver el cubo. A diferencia del algoritmo anterior, aquí se parte del cubo deshecho para alcanzar el cubo resuleto, por lo que no es necesario cambiar el orden de la lista de movimientos ni invertir cada uno de ellos.

3.2. Algoritmos de resolución

En este apartado vamos a proceder a explicar cuáles han sido los dos algoritmos que se han desarrollado para la resolución del cubo de Rubik. Para ello, se aportarán pseudocódigos e imágenes que faciliten en cierta medida el seguimiento del algoritmo.

Se ha llevado a cabo la realización de dos algoritmos. Por una parte, se ha optado por el algoritmo de A* (A - estrella) [14] junto con cuatro heurísticas que nos dirán cuál es la manera de encontrar la solución y cuáles son las bases o principios que se están utilizando para seleccionar los movimientos que compondrán la solución final. Por otro lado, se ha tomado la decisión de implementar uno de los procedimientos más conocidos de cara a alcanzar la resolución del cubo. Este es uno de los algoritmos más básicos y, posiblemente, de los más fáciles de seguir para una persona que está aprendiendo y recibe el nombre de “Algoritmo del Principiante”.

En las dos subsecciones siguientes se procede a la explicación de cada uno de los algoritmos paso por paso y a las consideraciones que han sido tomadas durante la implementación.

3.2.1. Algoritmo A* y heurísticas

A continuación vamos a explicar el algoritmo A* y especificaremos las heurísticas empleadas para llegar a la solución. Este algoritmo ha sido implementado para realizar una comparativa con el algoritmo del principiante que será explicado posteriormente.

Algoritmo A*

El algoritmo A* es uno de los más conocidos y se trata de un algoritmo de búsqueda en anchura. Dicho algoritmo es usado principalmente en búsquedas sobre grafos. Una búsqueda en anchura (BFS) es un algoritmo de búsqueda que recorre los nodos de un grafo, comenzando en el nodo inicial (o también conocido como nodo raíz), para luego explorar los nodos vecinos de este. Una vez explorados los nodos vecinos, se siguen explorando los nodos adyacentes así hasta recorrer todo el grafo. Sin embargo, si se llega al nodo final, o en nuestro caso, el nodo solución del cubo de Rubik, se detiene la exploración de otros caminos. Dicho método de búsqueda se usa para aquellos algoritmos donde resulta difícil elegir cuál es el mejor camino posible, y por ello, ha sido la búsqueda elegida para la

resolución del cubo de Rubik.

Si extrapolamos este algoritmo al tema que nos atañe que es la resolución del cubo de Rubik cada uno de los nodos del grafo representa los distintos estados del cubo de Rubik resultantes de hacer un movimiento sobre el estado del cual se parte.

El algoritmo A* realiza distintas rutas desde el punto inicial hasta el punto final, encontrando así diversas maneras de llegar a la solución final. Este algoritmo utiliza lo que se conoce como función de evaluación la cual se define tal y como sigue:

$$f(n) = g(n) + h^{'}(n)$$

Donde:

- $h^{'}(n)$: representa el valor de la heurística
- $g(n)$: representa el costo real del camino recorrido para llegar al nodo actual, n.
- $f(n)$: representa el valor final de la función de evaluación.

Se plantearon varias heurísticas para llegar a la solución y que expondremos más adelante. Una heurística consiste en añadir un valor numérico a cada uno de los nodos, con el fin de facilitar la elección del nodo que se va a visitar. Esto es así, porque en el A* se va a elegir como nodo a explorar y expandir aquel que tenga un menor valor de la función de evaluación. A la hora del tratamiento de los movimientos del cubo de Rubik, se considera que el coste de realizar cada uno de los movimientos será el mismo, de 1, es decir, se establece el mismo valor de $g(n)$ independientemente del nodo y movimiento. Es por ello, que la responsabilidad de la elección del nodo o estado del cubo recae en cómo definamos la función heurística.

Uno de los puntos más importantes del algoritmo A* son las dos estructuras auxiliares que son necesarias para la correcta implementación y funcionamiento. Estas estructuras de datos auxiliares son dos listas que podremos denotar como “generados” y “visitados”. La primera de estas listas se trata de una cola de prioridad ordenada en función del valor de la función de evaluación ($f(n)$) de cada nodo. Cabe señalar que se ha realizado un tratamiento especial de esta cola dado que en el cubo de Rubik podemos llegar al mismo estado mediante distintos movimientos. Es por ello que para evitar estados repetidos con diferentes valores y que suponga un gasto adicional de memoria innecesario, se ha hecho una comprobación de estados. Es así como si se encontraba dicho estado ya en la cola, su valor de la función de evaluación es reemplazado si es menor que el del estado que ya habia previamente en la lista, pero dicha sustitución solo se lleva a cabo si los estados del cubo de Rubik son completamente iguales. En segundo lugar, la lista de “visitados” es el lugar donde se va a almacenar la información referente a los nodos que ya han sido visitados y, por consiguiente, analizados.

En cada uno de los pasos del algoritmo, se va a expandir el nodo que esté primero en la lista de “generados”, es decir, aquel que tenga el menos valor de $f(n)$. En el caso de que no sea nuestro nodo

final (todos los cubitos que componen el cubo no están en su debida posición final), se expande dicho nodo. En nuestro caso, el factor de ramificación, es decir, la cantidad de nodos que se van a generar de un único nodo origen, va a ser de 12, ya que se genera uno por cada movimiento que se puede realizar (los movimientos explicados previamente en la sección “1.2.2 Notación y movimientos del cubo”). Una vez se han obtenido los 12 nuevos estados del cubo, eston son insertados en la lista de “generados” junto con su valor de la función de evaluación.

Una de las características a destacar en la implementación del algoritmo es que el cubo raíz es siempre el cubo ya resuelto. El propósito de esto es encontrar la combinación óptima de movimientos que nos conducen a la posición del cubo que queremos resolver. Así, una vez encontrada la secuencia simplemente habrá que invertir la lista de movimientos y cambiar cada uno de ellos por su inversa.

Al final del documento se va a exponer el pseudocódigo del algoritmo de A* adaptado a nuestro caso para facilitar una mayor comprensión. Cabe resaltar que de algunas de las funciones no se adjunta su pseudocódigo ya que son más sencillas y no son de una importancia relevante para el entendimiento del algoritmo. En este caso, vamos a explicar las funciones auxiliares que tratan la expansión de los nodos, dado que las funciones de las heurísticas se tratarán más adelante por separado.

Heurísticas

Las heurísticas son funciones que asignan a cada estado de un grafo un valor numérico y que solo depende de ese único estado. Las heurísticas surgieron con la intención de aportar estrategias, métodos o criterios que permitan resolver ciertos problemas de búsqueda de una manera más eficiente y rápida.

Como mencionamos anteriormente el papel que juega la función de heurística a la hora de encontrar la solución del cubo de Rubik es muy importante, dado que el coste de ir de un nodo a otro del grafo (de un estado a otro del cubo mediante un movimiento) tiene un coste de 1, por lo que su valor es determinante. Es por ello, que hemos realizado experimentos con 4 heurísticas distintas.

Podemos ver las heuríticas como una “ayuda” para guiar un proceso de búsqueda . El empleo de una heurística no siempre garantiza resultados óptimos ni que sean siempre la mejor solución, pero sí consigue resultados de calidad media en un tiempo acorde. Es por ello que hemos manejado heurísticas de distintas “calidades” y, a continuación, se enumeran y explican.

Las heurísticas abordadas en este proyecto son las siguientes:

1.– **Heurística 0 o BFS**. Es aquella en la cual le otorgamos a la función $h^{'}(n)$ el valor de 0 independientemente del nodo/estado en el que nos encontremos en el árbol de búsqueda. Este caso es el mismo que el conocido como BFS o búsqueda en anchura. Al otorgarle el valor de 0 se van a visitar todos los nodos en el orden de creación, dado que el valor de la $f(n)$ siempre será el mismo, con lo que encontrar la solución resulta prácticamente

imposible. Como ya mencionamos al comienzo del trabajo, hay $43{,}252{,}003{,}274{,}489{,}856{,}000$ posiciones distintas en las cuales se puede encontrar el cubo, con lo que visitar cada estado uno por uno hasta encontrar el camino idóneo resulta computacionalmente imposible y altamente costoso en términos de tiempo, por lo que los experimentos que se han realizado con dicha heurística implican un menor número de ejecuciones.

2.– **Heurística de esquinas colocadas**. Como su propio nombre indica es aquella donde se tienen en cuenta únicamente la colocación de las esquinas. Es por ello, que se comparan que las tres caras que componen una esquina estén en la posición (que no orientación) correcta. Si están en su posición se descuentan del total de las esquinas obteniendo así el valor de la heurística. En definitiva, la heurística refleja el número de caras de esquinas que no están correctamente posicionadas.

3.– **Heurística de aristas colocadas**. A diferencia del caso anterior, se presta atención a la colocación de las aristas. Se sigue el mismo mecanismo que el explicado en el caso anterior, pero observando las dos caras que componen cada una de las aristas.

4.– **Heurística de esquinas y aristas colocadas**. Por último, tenemos la heurística más completa e informada de las cuatro. Esta heurística combina las dos anteriores, es decir, se fijan en que todos los cubitos del cubo estén correctamente colocados (excepto los cubos centrales que siempre estarán en la posición correcta).

A la hora de realizar los experimentos debido a la gran cantidad de posibilidades que presenta el cubo de Rubik, se han tomado ciertas decisiones para realizar los experimentos en función de la heurística. Dichos resultados se abordarán en una sección posterior.

3.2.2. Algoritmo del Principiante

El algoritmo del Principiante es un proceso que permite llegar a la resolución del cubo de Rubik siguiendo una serie de pasos. En cada uno de esos pasos se definen una secuencia de movimientos a aplicar en función de la posición y la configuración que presente el cubo al comienzo de cada uno de los pasos. En este algoritmo vamos a diferenciar siete pasos, donde cada uno de ellos se ha de aplicar una vez completado el paso previo, ya que es un algoritmo secuencial. A parte de ser un algoritmo bastante intuitivo y de sencilla comprensión para el usuario, permite entender un poco más cuál es la estructura del cubo.

La identificación de pasos de este procedimiento se hace desde un punto de vista de secciones colocadas del cubo. Se procede a una resolución empezando por la parte superior del cubo (Paso 1) llegando a terminar con la parte inferior (Paso 7).

De cara a una mejor comprensión en la explicación de cada uno de los pasos del algoritmo, vamos a especificar cierta terminología relativa al cubo. Dicha notación se usará de cara a facilitar la explicación

de los movimientos a realizar, del porqué de estas secuencias y de las partes del cubo que se verán afectadas. De esta manera denotamos las siguientes partes del cubo como sigue:

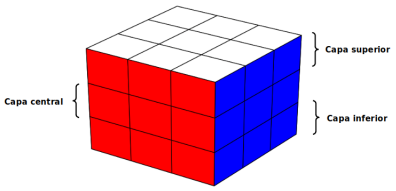


Figura 3.1: Distintas capas del Cubo de Rubik

- **Capa superior**. Nos referiremos a partir de ahora al tercio superior del cubo de Rubik. En nuestra explicación corresponderá con la cara superior asociada al color blanco y a los 3 primeros cuadraditos de las caras adyacentes.
- **Capa central**. Nos referiremos a partir de ahora al tercio situado en la parte media del cubo de Rubik. En nuestra explicación corresponderá con los 3 cuadraditos situados en el ecuador del rompecabezas, es decir, las caras afectadas serán la frontal, la derecha, la trasera y la izquierda (en nuestra explicación las de colores roja, azul, naranja y verde).
- **Capa inferior**. Nos referiremos a partir de ahora al tercio inferior del cubo de Rubik. En nuestra explicación corresponderá con la cara inferior asociada al color amarillo y a los 3 últimos cuadraditos de las caras adyacentes.

Una vez introducida esta notación, en el Anexo B se va a explicar cada uno de los pasos que componen este algoritmo. Para una mejor visualización de los pasos se adjuntarán unas imágenes generales donde se puedan apreciar cada una de las posibles casuísticas en cada uno. Además, cabe resaltar que las secuencias que se especifiquen en cada uno de los pasos son los movimientos que se han de realizar tomando la cara Frontal como referencia y para el caso expuesto (el resto de combinaciones se han tenido en cuenta en el código).

4

Experimentación

En esta sección se van a presentar los diferentes experimentos que se han llevado a cabo con los dos métodos descritos en la sección anterior. De igual manera, se va a realizar una comparativa entre ellos para poder obtener unas conclusiones acerca del trabajo realizado en este proyecto.

Con el objetivo de mostrar de una manera más visual cuáles han sido los resultados que se han obntenido, se adjunta una gráfica con cada uno de los experimentos, los cuales se describirán a continuación. Adicionalmente, se hará una pequeña introducción acerca de la metodología general empleada, así como un análisis de cada uno de los experimentos por separado.

4.1. Metodología general

Como mencionamos con anterioridad, en este proyecto se han desarrollado dos métodos distintos para la resolución del cubo de Rubik: el algorimto A* con distintas heurísticas y el Algoritmo del Principiante. Dado que se han desarrollado cuatro heurísticas distintas, se ha realizado un estudio comparativo de cada una de ellas con respecto al Algoritmo del Principiante. Como última instancia, también se realizado un experimento cuyo objetivo es llevar a cabo una comparativa entre las propias heurísticas, con el fin de analizar cuál de ellas es la más eficiente desde distintos puntos de vista.

A la hora de realizar los experimentos donde se comparan los resultados obtenidos con el A* y una heurística dada frente al Algoritmo del Principiante, se han llevado a cabo una serie de ejecuciones partiendo de los mismos cubos originales para que el estudio sea más objetivo. Para ello, se ha buscado la solución de cubos que han sido desordenados un número distinto de veces con el objetivo de estudiar la evolución que se sigue en función del caso en el que nos encontremos. Como consecuencia de los tiempos de ejecución y del gran número de posibilidades en las cuales se puede encontrar el cubo de Rubik, se ha decidido tomar las mediciones con cubos que han sido desordenados con 5, 6, 7, 8 y 10 movimientos. Estos experimentos han sido llevados a cabo para cada una de las heurísticas así como para el Algoritmo del Principiante.

De cara a obtener unos resultados más fiables, se han efectuado 5 ejecuciones con cada uno de estos números para desordenar el cubo. Posteriormente, se obtuvo la media de movimientos necesa-

rios para resolver el cubo con cada una de las maneras de resolución, así como una representación de la varianza para medir la variabilidad de estos datos. En las gráficas que se adjuntan para cada uno de los experimentos, podemos apreciar que la línea continua une los valores de las medias para cada uno de los casos. Además, se podrá ver una barra vertical de color rojo que describe la varianza obtenida para cada conjunto de ejecuciones.

Como último punto, es importante recalcar que estas ejecuciones han sido aleatorias, por lo que los resultados obtenidos son orientativos dado que el factor de aleatoriedad del cubo de Rubik es muy elevado (por el hecho de que tiene alrededor de 43 trillones de posiciones posibles). Además, debido a los tiempos de ejecución no se han realizado experimentos desordenando el cubo de Rubik con más de 10 movimientos.

4.2. Experimento 1

En este primer experimento se ha realizado la comparativa entre el algoritmo A* con la heurística 0 o también conocida como BFS (Búsqueda en Anchura) y el Algoritmo del Principiante. Recordemos que el algoritmo de búsqueda en achura tenía la característica de que se van a visitar todos los nodos que se generen (y en orden), dado que el valor de la función de evaluación es el mismo en todos. Esta heurística tiene un coste computacional muy elevado, por ello se han reducido los experimentos, desordenando el cubo utilizando 5, 6 y 7 movimientos. La gráfica resultante de realizar estas ejecuciones, obteniendo el número de movimientos para resolver el cubo en función del número de movimientos para deshacerlo es la siguiente:

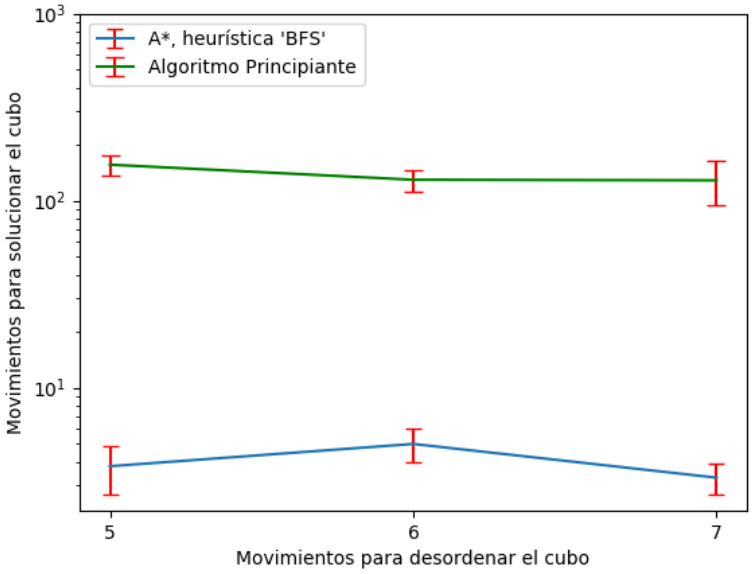


Figura 4.1: Comparativa BFS vs. Algoritmo del Principiante

Como podemos observar en la gráfica adjunta el número de movimientos para resolver el cubo usando el algoritmo A* con BFS es mucho menor que el empleado por el Algoritmo del Principiante. Podemos apreciar también que la varianza es mucho menor en BFS que en el Algoritmo del Principiante, es decir, la variablidad en el número de movimientos para resolver el cubo de Rubik es despreciable. La explicación de estos datos que se han obtenido reside en que el Algoritmo del Principiante sigue unos pasos donde su objetivo es colocar ciertos cubitos independientemente de la situación del resto. Es por ello, que no se busca cuál es el camino más corto para alcanzar la solución, sino que se siguen unos pasos y se aplican unas secuencias de movimientos dadas. En términos de movimientos (que es el factor que más nos interesa en este trabajo) es mucho más costoso el Algoritmo del Pricipiante, sin embargo, si hablamos de coste de tiempo el algoritmo A* usando BFS es mucho más costoso porque recorrerá infinidad de nodos para alcanzar la solución.

4.3. Experimento 2

En el segundo experimento se ha realizado la comparativa entre el algoritmo A* con la heurística de esquinas y el Algoritmo del Principiante. Dicha heurística solo tiene en cuenta la colocación de las 8 esquinas que componen el cubo. En este experimento se han llevado a cabo las ejecuciones que se mencionaron en el apartado previo de metodología, dado que al no recorrer todos los nodos como en el experimento anterior, los tiempos de ejecución son abordables. Los resultados obtenidos son los siguientes:

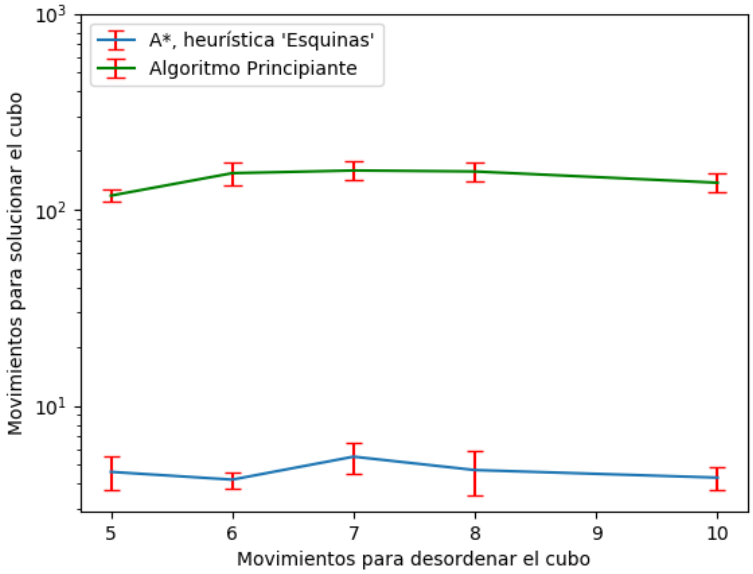


Figura 4.2: Comparativa A* con heurística “Esquinas” vs. Algoritmo del Principiante

Como podemos observar en la gráfica adjunta el número de movimientos para resolver el cubo

usando el algoritmo A* con la heurística de “Esquinas” es mucho menor que el empleado por el Algoritmo del Principiante. Podemos apreciar también que la varianza es mucho menor para el primer algoritmo que para el Algoritmo del Principiante. La explicación de estos datos que se han obtenido es similar a la del anterior experimento. En términos de movimientos (que es el factor que más nos interesa en este trabajo) es mucho más costoso el Algoritmo del Pricipiante. En lo referente a tiempos de ejecución el algoritmo de A* con la heurística de “Esquinas” es mucho menos costoso que el A* con BFS, dado que ahora el valor de la función de evaluación no es el mismo para todos los nodos.

4.4. Experimento 3

El tercer experimento es similar al anterior solo que cambiando la heurística empleada por el algoritmo de A*, siendo ahora la heurística de “Aristas”. Los resultados obtenidos son muy similares (en términos de movimientos) a los obtenidos con la heurística de “Esquinas” del pasado experimento, con la diferencia que ahora nos fijamos en las aristas que están colocadas y no en las esquinas, es decir, tenemos 12 cubitos en cuenta.

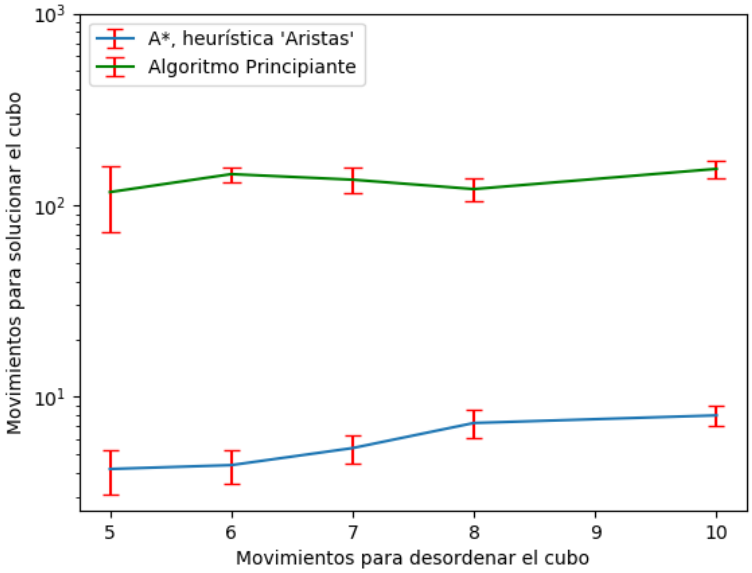


Figura 4.3: Comparativa A* con heurística “Aristas” vs. Algoritmo del Principiante

La gráfica muestra cóimo los resultados obtenidos son muy similares a los de anteriores experimentos. Además se aprecia que el número de movimientos para resolver el cubo usando la heurística es práctimente constante. Sin embargo, el número de movimientos del Algoritmo del Principiante muestra rangos de valores mucho más amplios, dado que dependerá, en su totalidad, del cubo que introduzcamos para realizar el experimento.

4.5. Experimento 4

Por último, en el cuarto experimento se compara la última de las heurísticas con el Algoritmo del Principiante. Esta heurística es una mezcla de las dos que se han empleado en experimentos anteriores. Se trata de la heurística de “Esquinas y aristas”, donde se comprueba que tanto las 8 esquinas como las 12 aristas estén correctamente posicionadas obteniendo lo siguiente:

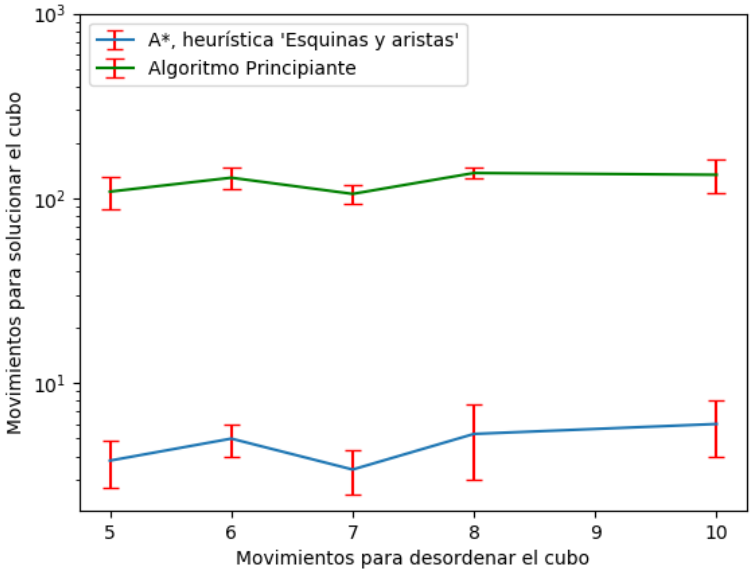


Figura 4.4: Comparativa A* con heurística “Esquinas y aristas” vs. Algoritmo del Principiante

En cuanto a movimientos para alcanzar la solución del cubo con la heurístca, vemos que es similar a las heurísticas anteriores y que por otra parte el Algoritmo del Principiante muestra un comportamiento similar al mostrado en experimentos anteriores.

Dado que en los experimentos mostrados anteriormente las diferencias entre las heurísticas, en cuanto a número de movimientos para resolver el cubo, es prácticamente inexistente, hemos realizado un último experimento donde se comparan el coste de búsqueda de cada uno de ellas.

4.6. Experimento 5

La motivación de realizar este experimento es que en los experimentos previos los resultados obtenidos no mostraban diferencias a la hora de usar una heurística en lugar de otra. Este hecho, llevó a realizar una comparación entre las tres heurísticas, quitando la BFS, de nodos que han sido “generados” y “visitados”. Se excluyó de este experimiento a la heurística 0 o BFS debido a los tiempos de ejecución y al alto coste que supondría lanzar el mismo número de experimentos que para el resto. Es

por ello que de comparar las tres heurísticas se obtienen los siguientes resultados en cuanto a nodos "generados" y nodos "visitados":

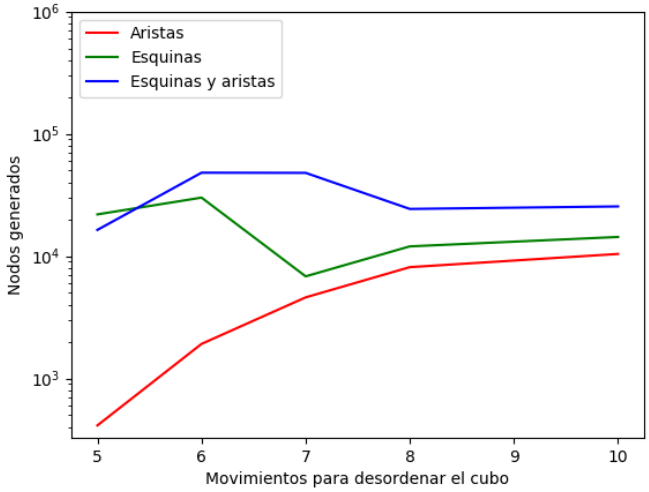


Figura 4.5: Comparativa de nodos generados con las tres heurísticas

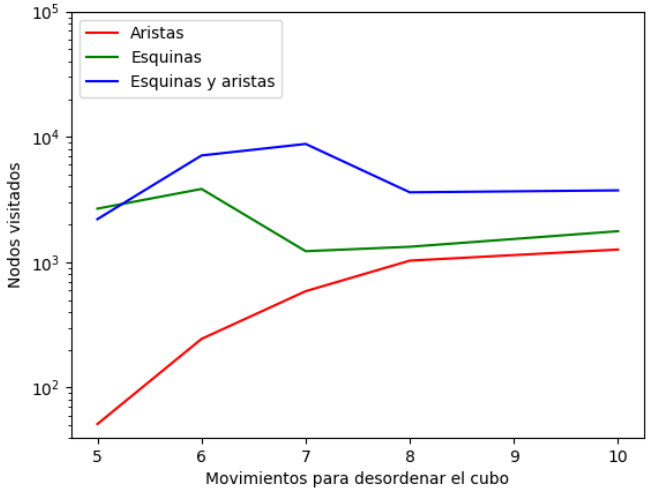


Figura 4.6: Comparativa de nodos visitados con las tres heurísticas

Como podemos apreciar en los gráficos adjuntos el número de nodos tanto "generados" como "visitados" es mayor para la heurística de "Esquinas y aristas" que para el resto, siendo la heurística de "Aristas" la que tiene un número menor. Podemos ver que el número de ambos nodos es mayor para la heurística de "Esquinas" que para la de "Esquinas y aristas", pero este hecho se puede deber a la aleatoriedad de las ejecuciones. Para la colocación de las líneas, correspondientes a las distintas heurísticas, la explicación encontrada es la restricción de cada una de ellas a la hora de la comprobación. En las heurísticas de "Aristas" y "Esquinas" se comprueban el estado de 12 aristas (cada una de dos caras) y 8 esquinas (cada una de 3 caras) respectivamente. Sin embargo, en la heurística de

“Esquinas y aristas” nos fijamos tanto en las 12 aristas como en las 8 esquinas, por lo que la función de evaluación toma valores en un rango mayor, habiendo mayor posibilidad de cubrir distintos caminos que con las otras heurísticas, de ahí que la diferencia sea tan significativa.

5

Conclusiones y limitaciones

En este apartado vamos a proceder a explicar las conclusiones finales que se han obtenido del trabajo y de los experimentos que han sido llevados a cabo y que se han explicado anteriormente. Así mismo, se hará un análisis de cuáles son las limitaciones que se han encontrado durante el transcurso del proyecto.

5.1. Conclusiones

En este trabajo hemos intentado alcanzar la secuencia de movimientos necesaria para la resolución del cubo de Rubik a través de dos métodos distintos. Mediante ambos métodos se han encontrado buenas soluciones, es decir, soluciones válidas.

Con el método de A* y viendo las distintas heurísticas planteadas, hemos podido comprobar que los resultados obtenidos por este algoritmo garantizan una ruta óptima. Sin embargo, también hemos podido comprobar que el coste en términos de tiempo y de recursos es mucho superior. Para ello, se realizó el experimento en el que se comparaban los nodos “generados” y “visitados”, para poder tener una visualización de la magnitud de estas búsquedas. Podemos afirmar de esta manera que dicho método nos proporciona la solución óptima al problema ya que en los experimentos realizados el número de movimientos necesarios para resolver el cubo nunca ha sido superior al número de movimientos invertidos para deshacerlo.

Por otro lado, con el Algoritmo del Principiante podemos afirmar que siempre llegaremos a la solución. Esta afirmación se debe a que se han descrito todas las posibles casuísticas en las que se puede encontrar el cubo y se ha aportado una secuencia de movimientos para resolverla. En cuanto a términos de coste, el coste temporal de resolverlo es mucho menor, ya que no se realiza una búsqueda, sino que se aborda la situación que se encuentra en cada uno de los pasos, es decir, no se plantea nunca “dar marcha atrás”. Sin embargo, cabe destacar que aunque el coste temporal y de recursos sea menor (no se necesitan las estructuras auxiliares de nodos “generados” y visitados”) el coste en movimientos es mucho más elevado como hemos podido comprobar. De este modo, garantizamos que siempre se encontrará una solución en un tiempo relativamente pequeño, pero esta solución no será

ni la óptima ni la más corta.

Es así como este estudio nos ha permitido realizar una comparativa entre dos métodos de resolución del cubo de Rubik. En estas se usaron diferentes maneras de representar al cubo para llegar a su solución: una representación basada en la visualización gráfica y una representación matemática mediante permutaciones. Así pudimos abordar dos maneras distintas de tratar al cubo de Rubik, porque este rompecabezas, en el fondo no deja de ser un problema relacionado con el mundo de las matemáticas.

En definitva, hemos podido ver que aunque el cubo de Rubik sea considerado uno de los rompecabezas más comunes y habituales en los últimos años, no es un puzzle cuya solución se alcance de una manera trivial. Hay muchos métodos y maneras de resolverlo, pero no siempre la manera más rápida de encontrar la solución es la más eficiente computacionalmente, así como tampoco la forma que nos da la solución lo antes posible es la más óptima.

5.2. Limitaciones

Durante el desarrollo de este proyecto se han encontrado ciertas limitaciones que se expondrán a continuación. Cabe señalar que encontrar un método que resuelva de manera eficiente el cubo de Rubik en términos de tiempo y de movmientos es algo que se sigue estudiando en la actualidad. Como mencionamos anteriormente, se demostró que el cubo de Rubik siempre se puede resolver en menos de 20 movimientos, pero llegar a esta solución no es una tarea que requiera poco tiempo ni recursos.

A la hora de realizar los dos métodos, se encontraron limitaciones al usar el algoritmo de A* y no en el Algoritmo del Principiante. Este hecho se debe a que en el Algoritmo del Principiante se aplican las secuencias de movimientos necesarias para resolver las distintas posiciones que puede presentar el cubo en cada uno de los pasos. Sin embargo, en el método de A* con las distintas heurísticas vimos más limitadas las capacidades para encontrar una solución de manera “eficiente”.

El algoritmo de A* es un procedimiento de búsqueda en anchura basado en heurísticas que consume una gran cantidad de recursos si el factor de ramificación es muy amplio y en nuestro caso ese factor era de 12 (uno por cada movimiento). Es por ello que se barajaban unas cantidades de nodos a explorar y de nodos a generar que hacían que llegar a la meta propuesta fuera muy costoso. Cuanto más se aumenta el número de movimientos empleados para deshacer el cubo mayor es el tiempo requerido en las ejecuciones. Es por este motivo que nuestros experimentos se vieron capados a, como máximo, 10 movimientos, dado que usar una cantidad superior nos llevaba a alcanzar tiempos de ejecución desorbitados. Además, se obtuvo la conclusión de que para cubos con un orden superior a 10 movimientos existe una importante limitación en cuanto a términos de memoria, haciendo prácticamente imposible alcanzar su resolución mediante este algoritmo.

5.3. Trabajo Futuro

En la actualidad se siguen investigando formas para resolver el cubo de Rubik. Sin embargo, en este proyecto se han alcanzado los resultados que se esperaban, ya que se tenía un previo conocimiento del alto costo en memoria y tiempo del A*.

Una de las ramas que se pensó abordar pero que no dio tiempo, fue realizar podas a la hora de buscar la solución. Este pensamiento se tuvo a raíz de que la eficiencia de nuestra solución depende de manera directa de la calidad de la heurística propuesta. Es por ello, que si se descubre una forma nueva de saber cuál es el camino idóneo para alcanzar la solución puede que se reduzca el tiempo de ejecución. Este hecho combinado con ciertas podas que se puedan proponer en función de la nueva heurística sería una línea muy interesante de trabajo futuro.

Finalmente, se pueden investigar y estudiar otras alternativas para llegar a la solución del cubo, como puede ser el uso de algoritmos genéticos o usar otros tipos de búsqueda con las heurísticas propuestas, como puede ser IDA* (Iterative Deepening A*). Por otro lado, también se podría hacer uso de otros lenguajes que agilicen la búsqueda de la solución, como puede ser el lenguaje de C++.

Bibliografía

[1] O. Pekonen, "Cubed: The puzzle of us all by ernő rubik," 2021.

[2] "Erno rubik." https://www.biografias.es/famosos/erno-rubik.html.

[3] F. Agostinelli, S. McAleer, A. Shmakov, and P. Baldi, "Solving the rubik's cube with deep reinforcement learning and search," *Nature Machine Intelligence*, vol. 1, no. 8, pp. 356–363, 2019.

[4] J. Palmer, "Cracking the last mystery of the rubik's cube," *New Scientist*, vol. 199, no. 2668, pp. 40–43, 2008.

[5] D. Joyner, "The man who found god's number," *The College Mathematics Journal*, vol. 45, no. 4, pp. 258–266, 2014.

[6] "Historia de los rompecabezas." https://mymodernmet.com/es/historia-rompecabezas/.

[7] M. E. Larsen, "Rubik's revenge: the group theoretical solution," *The American Mathematical Monthly*, vol. 92, no. 6, pp. 381–390, 1985.

[8] W. R. Scott, *Group theory*. Courier Corporation, 2012.

[9] J. Chen, "Group theory and the rubik's cube," 2004.

[10] T. Rokicki, H. Kociemba, M. Davidson, and J. Dethridge, "The diameter of the rubik's cube group is twenty," *siam REVIEW*, vol. 56, no. 4, pp. 645–670, 2014.

[11] L. Daniels, "Group theory and the rubik's cube," *Lakehead University, Kanada*, 2014.

[12] "mplot3d tutorial." https://matplotlib.org/2.0.2/mpl_toolkits/mplot3d/tutorial.html.

[13] N. S. Shuhaimi and N. F. Mustari, *Comparative study of methods for solving the Rubik's Cube*. PhD thesis, Universiti Teknologi MARA, 2019.

[14] A. Norvig and P. Russel, *Artificial Intelligence: A modern approach*. Prentice Hall Upper Saddle River, NJ, USA:, 2002.

Apéndices

A

Movimientos sobre el cubo

A continuación se muestran los posibles movimientos que se pueden realizar sobre el cubo de Rubik, así como sus inversas.

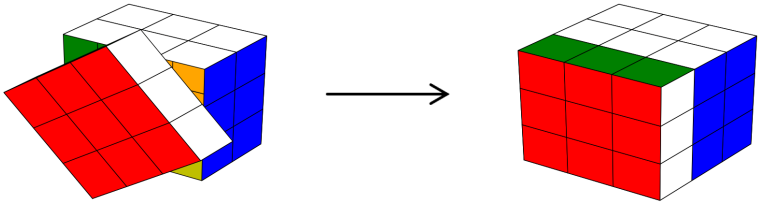


Figura A.1: Movimiento Front horario y su resultado

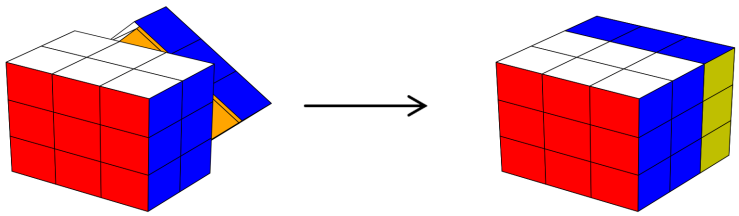


Figura A.2: Movimiento Back horario y su resultado

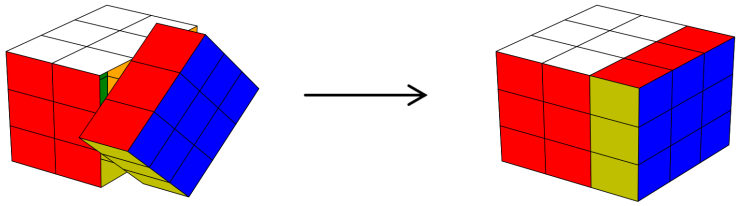


Figura A.3: Movimiento Right horario y su resultado

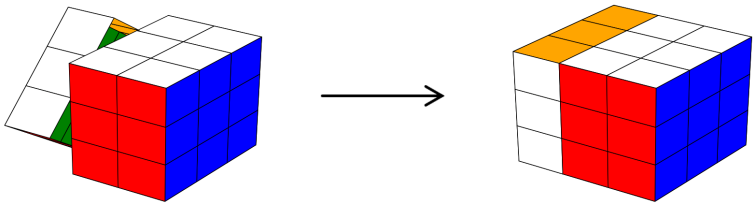


Figura A.4: Movimiento Left horario y su resultado

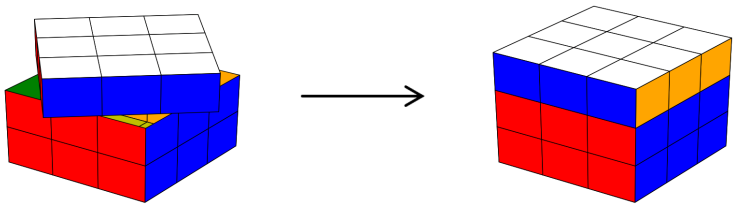


Figura A.5: Movimiento Up horario y su resultado

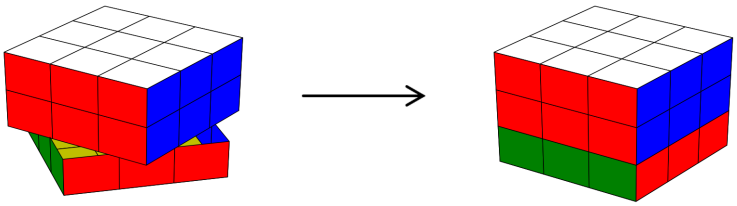


Figura A.6: Movimiento Down horario y su resultado

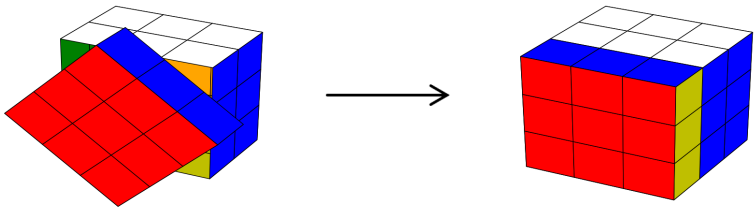


Figura A.7: Movimiento Front antihorario y su resultado

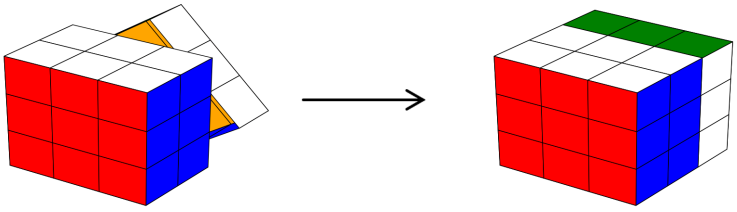


Figura A.8: Movimiento Back antihorario y su resultado

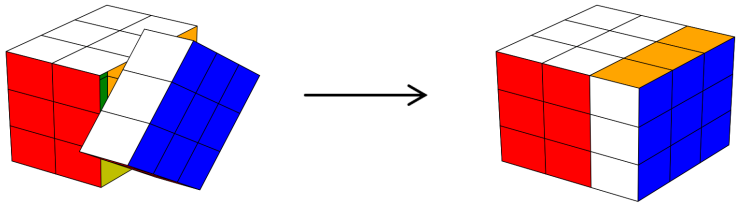


Figura A.9: Movimiento Right antihorario y su resultado

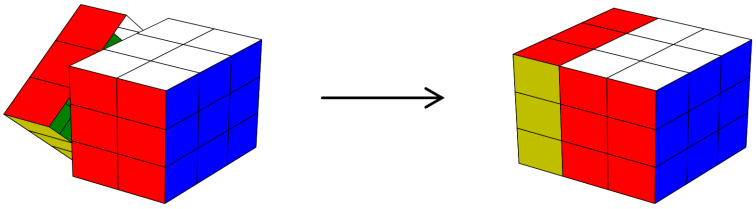


Figura A.10: Movimiento Left antihorario y su resultado

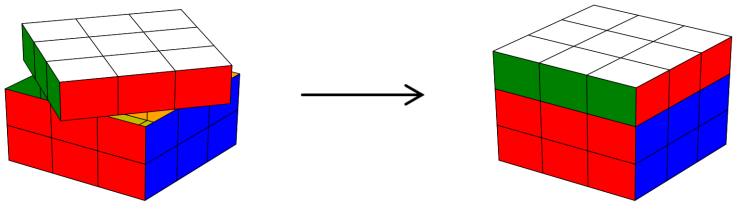


Figura A.11: Movimiento Up antihorario y su resultado

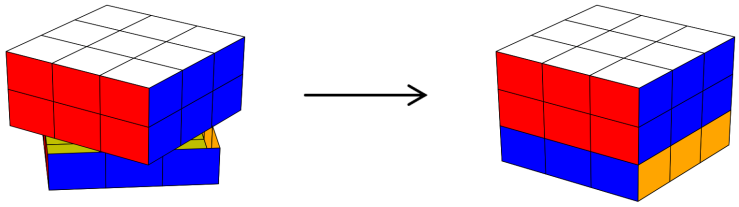


Figura A.12: Movimiento Down antihorario y su resultado

B

Algoritmo del Principiante

B.1. Paso 1. Cruz de la cara superior

En este primer paso el objetivo consiste en realizar la cruz de la cara superior, es decir, la cruz de la cara blanca. Además, se ha de realizar de manera que las aristas tienen que tener sus dos cuadraditos en las caras que les corresponden, es decir, han de estar bien orientados los cuatro cuadraditos que se pretenden colocar. El objetivo es el siguiente:

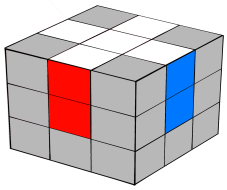


Figura B.1: Cruz cara superior

Diferenciaremos entre 3 posiciones donde se aplicarán las secuencias que especificaremos a continuación para colocar la arista y que esté correctamente orientada. Esto significa que tenemos que conseguir previamente que alguna de las cuatro aristas a colocar estén en una de las siguientes posibilidades para poder aplicar las secuencias adjuntadas.

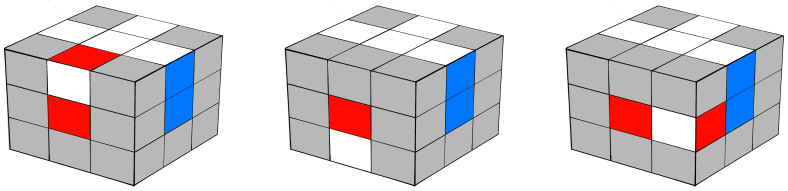


Figura B.2: Diferentes casuísticas del paso 1

B.2. Paso 2. Esquinas de la cara superior

Una vez tenemos ya la cruz de la cara superior, procedemos a completar la capa superior del cubo con la correcta colocación (incluida orientación) de las esquinas superiores. El objetivo a alcanzar sería el siguiente:

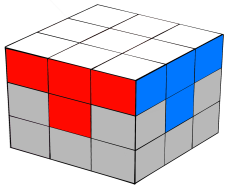


Figura B.3: Esquinas cara superior

Para poder aplicar la secuencia de los distintos casos que vamos a ver a continuación, es necesario que la esquina a colocar esté alineada con la posición en la que tiene que estar, es decir, esté en las mismas coordenadas “x” e “y” (variando solo la altura “z”). Con esta información, diferenciamos las siguientes posibilidades:

1.– **Caso 1**. La esquina a colocar está en la capa inferior, con la cara blanca en la cara roja, es decir, la cara que debería de ir en la cara superior (U) se encuentra en la cara frontal (F).

2.– **Caso 2**. En este caso, la cara de color blanco se encuentra en la cara que corresponde al color amarillo, es decir, la cara que debería de ir en la cara superior (U) se encuentra en la cara inferior (D).

3.– **Caso 3**. En este caso la esquina está en la posición correcta, pero mal orientada. La cara blanca se encuentra en el lugar de la cara roja, es decir, la cara que debería de ir en la cara superior (U) se encuentra en la cara frontal (F).

4.– **Caso 4**. En este caso la esquina está en la posición correcta, pero mal orientada. La cara blanca se encuentra en el lugar de la cara azul, es decir, la cara que debería de ir en la cara superior (U) se encuentra en la cara derecha (R).

5.– **Caso 5**. La esquina a colocar está en la capa inferior, con la cara blanca en la cara azul, es decir, la cara que debería de ir en la cara superior (U) se encuentra en la cara derecha (R).

Las secuencias de movimientos necesarias para colocar correctamente esta esquina, y que en el código están extrapoladas al resto de esquinas, son las siguietes en función de los casos descritos anteriormente:

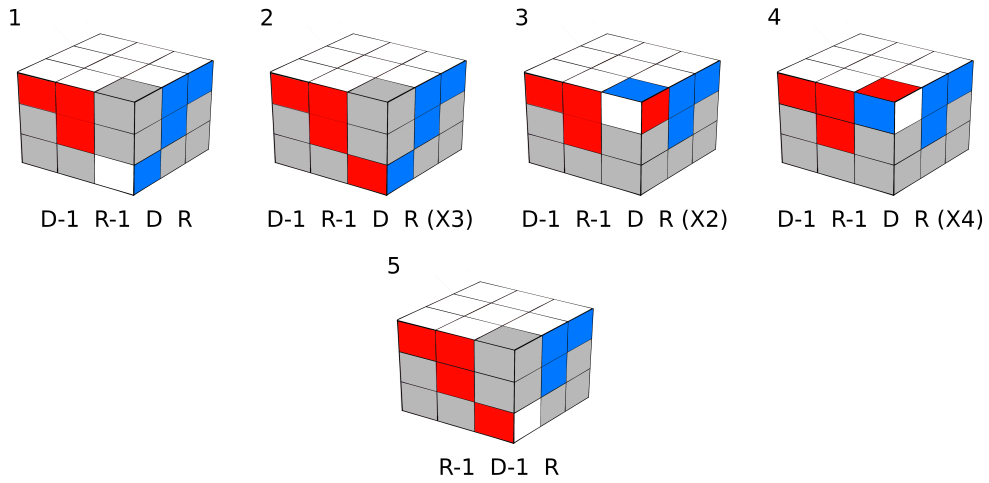


Figura B.4: Diferentes casuísticas del paso 2

B.3. Paso 3. Capa central

Una vez tenemos la capa superior ya completa, pasaremos a realizar la capa central. Nuestro objetivo va a ser colocar en la posición y orientación correctas las cuatro aristas que la componen.

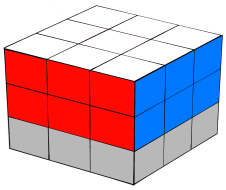


Figura B.5: Capa central

Para llegar a este estado, nos podemos encontrar ante dos posibles escenarios. Cabe destacar que existe un tercer posible estado, donde será necesario aplicar una de las dos secuencias descritas en los casos 1 y 2 para poder encontrarse en una de las dos casuísticas anteriores.

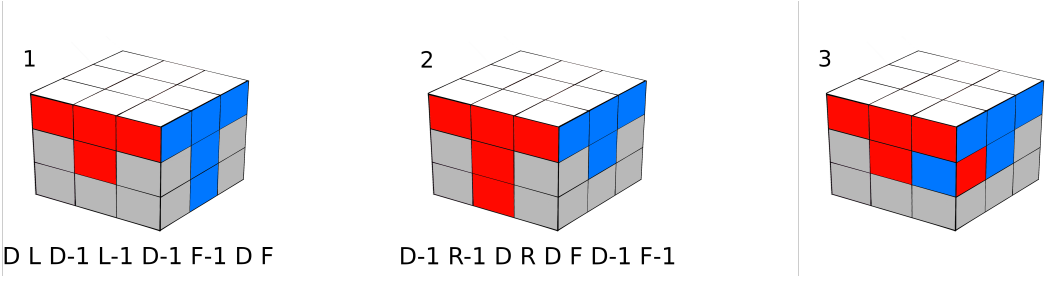


Figura B.6: Casuísticas del paso 3

B.4. Paso 4. Cruz de la cara inferior no orientada

En este paso vamos a proceder a rotar el cubo 180º para poder ver la cara inferior del cubo de Rubik, dado que ahora los cambios restantes afectarán a esta cara. En este paso tenemos que formar la cruz en la cara inferior (D), es decir, en la cara amarilla. Sin embargo, en este paso no vamos a comprobar que las aristas que componen la cruz están correctamente orientadas, solo nos fijaremos en su posición, buscando como resultado final:

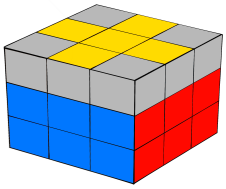
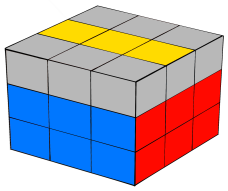
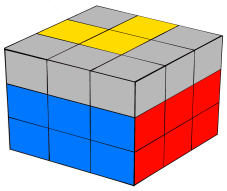


Figura B.7: Cruz de la cara inferior no orientada

Vamos a buscar que se cumplan una de las dos casuísticas siguientes para aplicar las secuencias de movimientos (tomando como referencia la cara frontal, pese a la orientación del cubo en las imágenes). En caso de que no se cumplan ninguna de las dos posiciones expuestas, será necesario aplicar alguna de las secuencias para poder tener una de las dos casuísticas. Las dos opciones son:



L-1 F-1 D-1 F



L-1 D-1 F-1 D F L

Figura B.8: Casuísticas del paso 4

B.5. Paso 5. Cruz de la cara inferior orientada

Ahora el cometido va a ser orientar las aristas de la cara inferior que en el paso anterior fueron colocadas en su correcta posición. El resultado a obtener es:

Vamos a tener dos posibles estados en los que se encuentren nuestras aristas. A la hora de fijarnos en los dos casos, vamos a prestar atención en dónde se encuentran las aristas que queremos orientar y justo la arista que correspondería a la de la cara que se encuentra a la derecha desde un punto de vista frontal (en las imágenes nos fijamos en la arista rojo-amarillo y en la siguiente que sería la

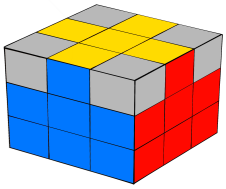


Figura B.9: Cruz de la cara inferior orientada

azul-amarillo, que si giramos el cubo 180º sería la de la derecha desde el punto de vistal frontal). Estos casos son:

1.– **Caso 1**. La arista a colocar (roja-amarilla) y la de la "derecha", es decir, la azul-amarilla (aunque al tener el cubo invertido sea la de la izquierda) estén en posiciones invertidas. La secuencia de movimientos a aplicar se hará con el propósito de cambiar la posición entre ambas aristas, además de colocar las otras que no se ven en la imagen (verde-amarilla y naranja-amarilla).

2.– **Caso 2**.En este segundo caso, la arista roja-amarilla está correctamente posicionada y orientada, pero la arista de su derecha, es decir, la azul-amarilla, no está en ninguna de las posiciones adyacentes, encontrándose en la cara opuesta. Por este motivo, hay que aplicar la misma secuencia de movimientos a la anterior, pero dos veces.

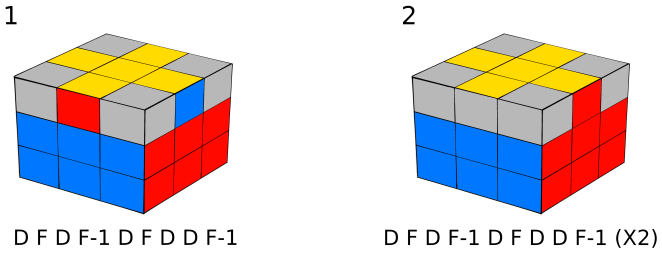


Figura B.10: Cubo del paso 5

B.6. Paso 6. Esquinas de la cara inferior no orientadas

Una vez llegados a este punto nos quedan solo cuatro cubitos por colocar para llegar a la solución final del cubo. Estos que quedan por colocar son las esquinas de la capa inferior. El objetivo en este paso consiste en que cada esquina esté en su correcta posición, sin estar necesariamente orientadas. Uno de los posibles resultados sería el siguiente, ya que puede haber alguna esquina orientada correctamente o de otra manera, pero siempre en esa posición:

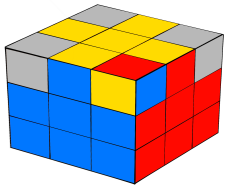
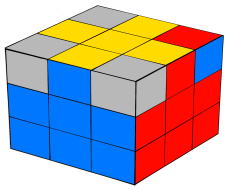


Figura B.11: Esquinas de la cara inferior no orientadas

Para poder aplicar la secuencia que vamos a indicar a continuación es necesario que al menos una de las cuatro esquinas que tenemos que colocar esté correctamente posicionada (que no orientada). En el caso en el que no hubiera ninguna esquina colocada, sería necesario aplicar la misma secuencia (2 ó 3 veces, las veces que sean necesarias) para conseguir que al menos una esquina esté colocada para aplicarla una vez más y colocar así el resto. El caso que estamos describiendo junto con su secuencia de movimientos es el siguiente:



L D-1 R-1 D L-1 D-1 R D

Figura B.12: Cubo del paso 6

B.7. Paso 7. Esquinas de la cara inferior orientadas

Nos encontramos en el último paso del Algoritmo del Principiante, donde solamente quedan por orientar las esquinas de la capa inferior que ya están correctamente posicionadas del paso anterior. El objetivo consiste en terminar de hacer la capa inferior del cubo y con ello completar el rompecabezas.

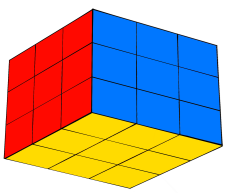


Figura B.13: Esquinas de la cara inferior orientadas

Los dos casos que nos podemos encontrar son los siguientes:

1.– **Caso 1**. La cara correspondiente a la cara inferior (amarilla) se encuentre en la cara frontal.

2.– **Caso 2**. La cara correspondiente a la cara inferior (amarilla) se encuentre en la cara de la derecha.

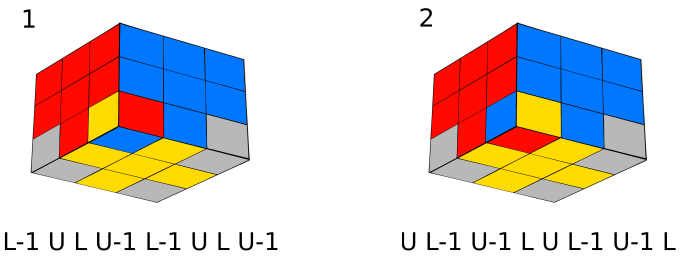


Figura B.14: Cubo del paso 7

Una vez coloquemos una de las esquinas, hacemos el movimiento de D (down) y volvemos a aplicar el mismo algoritmo de las secuencias anteriores.

C

Pseudocódigos

```
input : esq_orig, ari_orig, esq_fin, ari_fin, movs_pos: Estados de los cubos inicial y final así como los posibles
        movimientos del cubo
output: sol: secuencia de movimientos que llevan al cubo resuelto a nuestro cubo que queremos resolver
1  begin
2      generados = []
3      visitados = []
4      pos_ini ← posicion( esq_orig, ari_orig )
5      pos_fin ← posicion( esq_fin, ari_fin )
6      insertar( pos_ini, generados )
7      while no_vacío( generados ) do
8          pos_nueva ← extraer_menor( generados )
9          if pos_nueva ≠ pos_fin then
10             return pos_nueva
11         end
12         if es_expandible( pos_nueva, generados ) then
13             lista_pos ← expandir_todo( pos_nueva, movs_pos, esq_fin, ari_fin )
14             visitados ← visitados + [pos_nueva]
15             foreach p ∈ lista_pos do
16                 elem ← reemplazar_o_insertar( p.f, p, compara_pos )
17             end
18         end
19     end
20     return
21 end

22 function expandir_todo( pos_nueva, movs_pos, esq_fin, ari_fin ) :
23     foreach i ∈ movs_pos do
24         l ← expandir_uno( pos_nueva, i, esq_orig, ari_orig )
25     end
26     return l

27 function expandir_uno( pos_nueva, mov, esq_orig, ari_orig ) :
28     e, a ← mover_cubo( pos_nueva.esquinas, pos_nueva.aristas )
29     pos_nueva.h ← heuristica( e, a, esq_orig, ari_orig )
30     pos_nueva.f = p.coste + pos_nueva.h
31     return pos_nueva
```

Algoritmo C.1: Pseudocódigo principal del algoritmo A* para el cubo de Rubik


UAM
UNIVERSIDAD AUTONOMA
DE MADRID