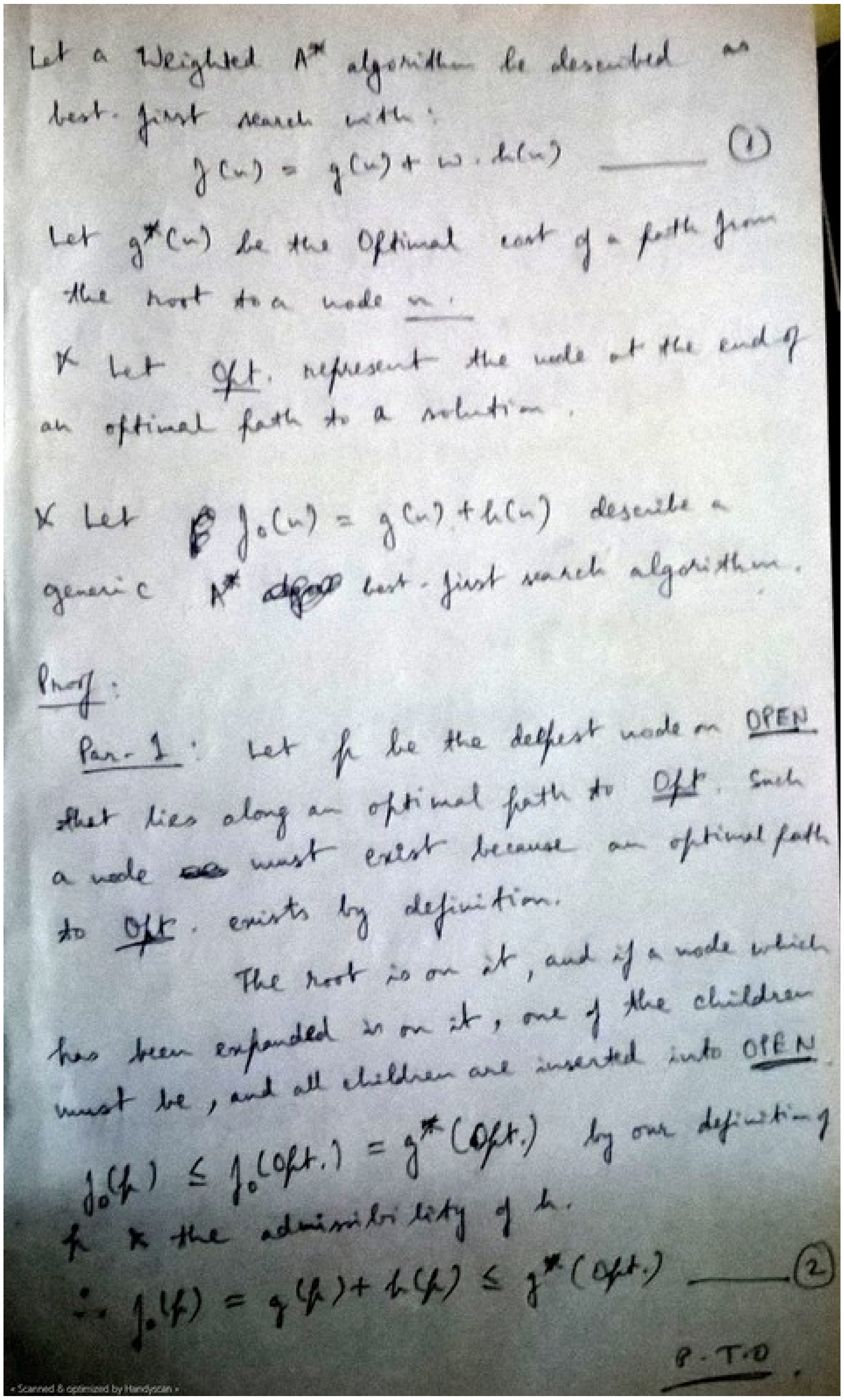
Artificial Intelligence – Assignment 2 (Theory)

Name: Sushovan Mandal **Roll Number: 10CS10048**

Ans 1.



Let a Weighted A* algorithm be described as best-first search with:

$f(n) = g(n) + w \cdot h(n)$ ——— (1)

Let $g^*(n)$ be the Optimal cost of a path from the root to a node n.

* Let Opt. represent the node at the end of an optimal path to a solution.

* Let $f_0(n) = g(n) + h(n)$ describe a generic A* ~~[illegible]~~ best-first search algorithm.

Proof:

Part-1: Let p be the deepest node on OPEN that lies along an optimal path to Opt. Such a node must exist because an optimal path to Opt. exists by definition.

The root is on it, and if a node which has been expanded is on it, one of the children must be, and all children are inserted into OPEN.

$f_0(p) \le f_0(Opt.) = g^*(Opt.)$ by our definition of p & the admissibility of h.

$\therefore f_0(p) = g(p) + h(p) \le g^*(Opt.)$ ——— (2)

P.T.O.

Part-2: From the above wonderful property of f_0, we can support a general result for Weighted A^*.

For any node n expanded by a best-first search guided by $f(n) = g(n) + w \cdot h(n)$,

- Consider an optimal path to Opt. If all nodes on this path have been expanded, we have the optimal solution $\times$

$$f(n) = g(n) + w \cdot h(n) = g^*(\text{opt.})$$
$$< w \cdot g^*(\text{opt.})$$

$\boxed{3'}$

[illegible] $(w > 1)$

- Otherwise, let p be the deepest node on OPEN that lies along the optimal path to Opt..

when we expand n, [illegible]

$f(n) \leq f(p)$ because n was selected for expansion before p.

$$f(p) = g(p) + w \cdot h(p) \leq w(g(p) + h(p))$$
$$= w \cdot f_0(p) \text{ by algebra.}$$

$\therefore \quad f(n) \leq w \cdot f_0(p)$

P.T.O

From result ②,

$$w \cdot f_0(p) \leq w \cdot f_0(\text{opt.}) = w \cdot g^*(\text{opt.}) \quad \text{———} \; (3'')$$

$\therefore$ Combining Results $(3')$ & $(3'')$, we get that for all cases in general,

For any node n expanded by a best-first search guided by $f(n) = g(n) + w \cdot h(n)$,

$$\underline{\underline{f(n) \leq w \cdot g^*(\text{opt.})}} \quad \text{———} \; ④$$

Part-3 :

For the solution s returned by weighted A^*_ε:

Because s is a goal node and h is admissible, $h(s) = 0$

$\therefore g(n) = f_0(n) = f(n)$ 4

by the definition of $f(n)$

& $f(n) \leq w \cdot g^*(\text{opt.})$ from Result ④

$$\therefore \underline{\underline{g(s) \leq w \cdot g^*(\text{opt.})}}$$

Ans 2.

The Rubik's cube can be seen as a collection of six faces, each made up of nine squares that can be coloured with one of six colours.

1.

In the Rubik’s Cube, each face can be represented by a unique colour. This is the colour of the center square, which never changes in relation to the others. Our colours are simply integer values from 0 to 5.

Locations on the cube are given as the pair (face, square), where the squares are a number from 0 to 8, as shown in figure 2.

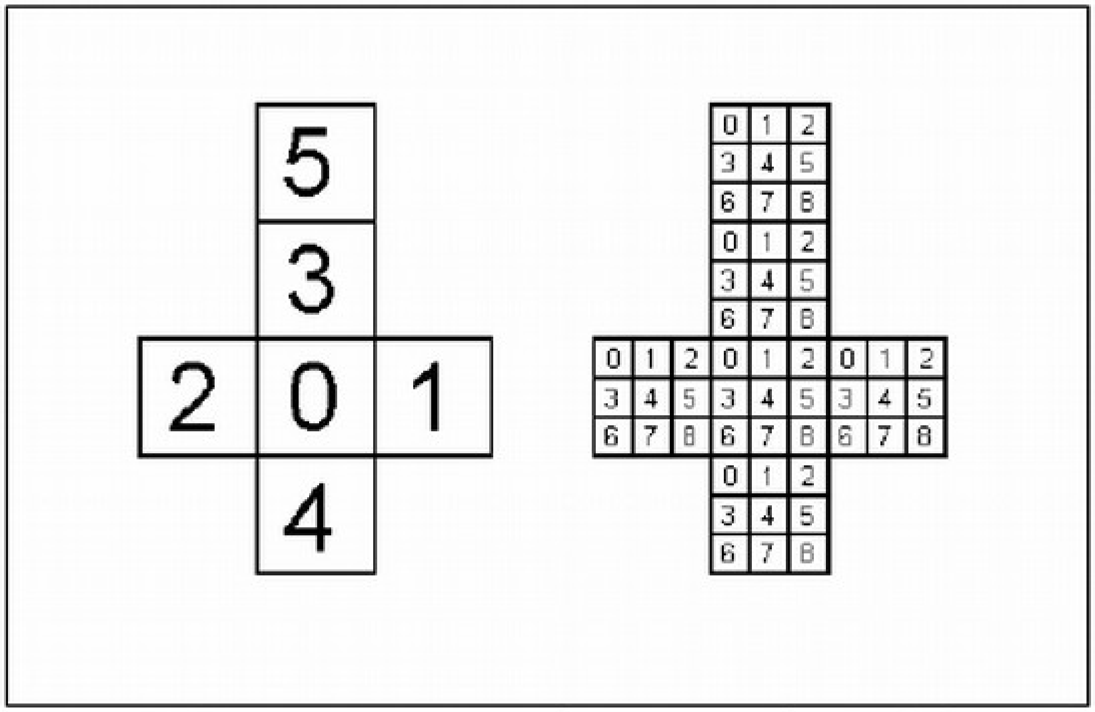


Figure 2: Representation of squares of a Rubik’s Cube

So for each cubit on the surface the cube is represented as:

((1,4), 0) → which reads as face 1 (out of faces 0-5), square no. 4 on face no. 1, coloured 0.

So there will be 3^3 = 27 such cubits in each state.

The original (3×3×3) Rubik's Cube has eight corners and twelve edges. There are 8! (40,320) ways to arrange the corner cubes. Seven can be oriented independently, and the orientation of the eighth depends on the preceding seven, giving 3^7 (2,187) possibilities. There are 12!/2 (239,500,800) ways to arrange the edges, since an even permutation of the corners implies an even permutation of the edges as well. (When arrangements of centres are also permitted, as described below, the rule is that the combined arrangement of corners, edges, and centres must be an even permutation.) Eleven edges can be flipped independently, with the flip of the twelfth depending on the preceding ones, giving 2^{11} (2,048) possibilities.

$$8! \times 3^7 \times (12!/2) \times 2^{11} = 43{,}252{,}003{,}274{,}489{,}856{,}000$$

which is approximately 43 quintillion states !

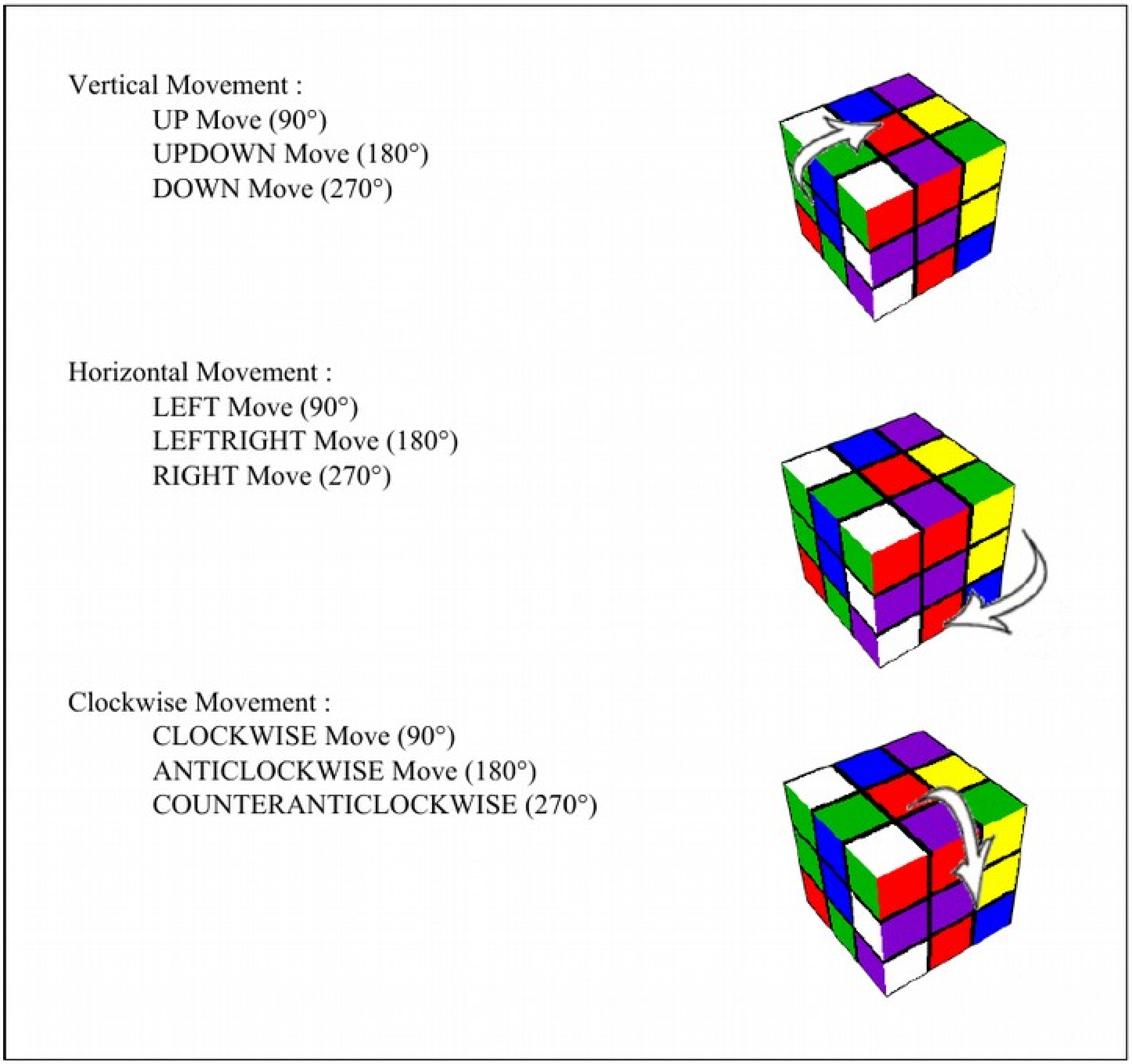
2.

There is only 1 goal state which is the state in which all cubits on any one face of the cube are all of the same colour.

The start state is the muddled up cube, any of the non-goal states can be a legal start state for the Rubik's cube problem.

3.

There are three basic movements available to someone trying to solve a Rubik’s cube. These moves can then be further broken into 3 sub moves for each movement, defined as follows:



Vertical Movement :

UP Move (90°)
UPDOWN Move (180°)
DOWN Move (270°)

Horizontal Movement :

LEFT Move (90°)
LEFTRIGHT Move (180°)
RIGHT Move (270°)

Clockwise Movement :

CLOCKWISE Move (90°)
ANTICLOCKWISE Move (180°)
COUNTERANTICLOCKWISE (270°)

Figure 3: Possible Moves for a Rubik's Cube

Taking all possible moves on each of the three rows gives us 27 possible moves in each state, but the middle row is not used as a move, so finally we have **18 moves** from each state.

We can use the "Singmaster notation" to encode these moves into successor function.

Its relative nature allows algorithms to be written in such a way that they can be applied regardless of which side is designated the top or how the colours are organized on a particular cube.

- *F* (Front): the side currently facing the solver
- *B* (Back): the side opposite the front
- *U* (Up): the side above or on top of the front side
- *D* (Down): the side opposite the top, underneath the Cube
- *L* (Left): the side directly to the left of the front
- *R* (Right): the side directly to the right of the front
- *f* (Front two layers): the side facing the solver and the corresponding middle layer
- *b* (Back two layers): the side opposite the front and the corresponding middle layer
- *u* (Up two layers) : the top side and the corresponding middle layer
- *d* (Down two layers) : the bottom layer and the corresponding middle layer
- *l* (Left two layers) : the side to the left of the front and the corresponding middle layer
- *r* (Right two layers) : the side to the right of the front and the corresponding middle layer
- *x* (rotate): rotate the entire Cube on *R*
- *y* (rotate): rotate the entire Cube on *U*
- *z* (rotate): rotate the entire Cube on *F*

When a prime symbol (′) follows a letter, it denotes a face turn counter-clockwise, while a letter without a prime symbol denotes a clockwise turn. A letter followed by a 2 (occasionally a superscript ^{2}) denotes two turns, or a 180-degree turn. *R* is right side clockwise, but *R′* is right side counter-clockwise. The letters *x*, *y*, and *z* are used to indicate that the entire Cube should be turned about one of its axes, corresponding to R, U, and F turns respectively. When *x*, *y* or *z* are primed, it is an indication that the cube must be rotated in the opposite direction. When they are squared, the cube must be rotated 180 degrees.

4.

One admissible heuristics for the Rubik's cube problem is the **Center-Corner Heuristic**. It compiles a hash table that stores the minimum length solution to solve every pattern of corner cubies, another hash table also stores up to six of the edge cubies.

This heuristic is a pattern database consisting of 3 parts: Corner Cubie Pattern Database, 6-Edge Cubie Pattern Database, and Remaining 6-Edge Cubie Database.

During problem solving, for each state, the different sets of cubies are used to compute indices into their pattern databases.

The overall heuristic value is the maximum of the three different database values.