ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 3, Issue 2, pp: (687-691), Month: April - June 2015, Available at: www.researchpublish.com

Rubik's Solving Approach by Edge-Centric Matching In Artificial Intelligence

[1]Hareesh B. N, [2]Sangappa Kuragod, [3]Mahesh K. Kaluti

[1, 2] PG Student, Dept of CSE, AIET.
[3]Asst. Professor, Dept of CSE, AIET.

***Abstract*: the Rubik's Cube is perhaps the worlds most famous and iconic puzzle, well-known to have a rich underlying mathematical structure matrix. We show that the Rubik's Cube also has a rich underlying algorithmic structure. This paper tackled recruitment of computer to solve Rubik's cube 3×3×3 through a proposed algorithm by Edge-Centric matching that is moving edge center pieces towards their base colors by movements of faces as clockwise and anticlockwise directions. Dimension of the cube are Three layers, Upper-face (U) and Down face (D), Left-face (L), Right-face (R), Front-face (F). In case direction in anti-clockwise represent with faces 'i', Example Ri right-face rotates in anti-clockwise. The algorithm starts with Upper-face and top-layer, Next second layer, and third layer as well as bottom-layer will get matches. This method shows the new approach for solving Rubik's.**

***Keywords:* Rubik's, Edge-Centric, matches.**

1. INTRODUCTION

This document gives a detailed analysis of the procedure that was followed in the enhancement to an existing application titled 'Rubik's, Distributed'. The original application served to solve a 2 by 2 Rubik's Mini cube by distributing the solution over multiple processors. Its logical enhancement, 'Rubik's, Heuristics serves to extend various featuresby creating a user interface representing a 3 by 3 cube, designing a solution to solve the cube, implementing the solution using a heuristic approach and lastly distributing the solution over multiple processors.

1) Existing System:

The existing system had the following fully functional features:

a. A graphical interface which would display the Rubik's 2x2 cube in the following modes:

a. Two dimensional mode

b. Three dimensional mode

b. A mechanism to allow the user to attempt to solve the cube and facilitate easy interaction with the cube.

c. An algorithm to solve the Rubik's 2 by 2 cube.

d. Distribution of the algorithm over multiple processors.

e. A mechanism for evaluation of the efficiency of solution

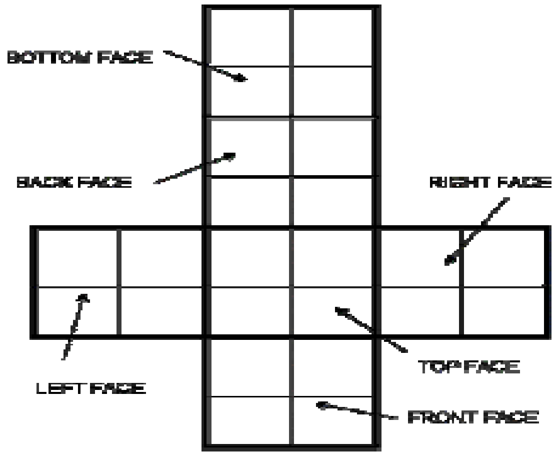

Fig. Two-Dimensional View of Rubik's Cube

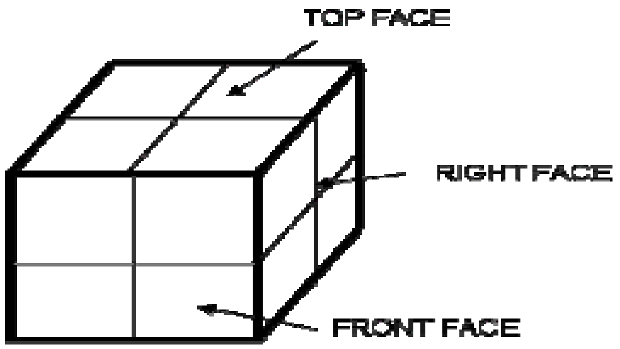

Fig 1.Three Dimensional View

2) Enhancements to existing system:

a. A graphical interface which would display the Rubik's **3X3** cube in the following modes:

a. Two dimensional mode

b. Three dimensional mode

b. A mechanism to allow the user to attempt to solve the cube and facilitate easy interaction with the cube.

c. To design a suitable heuristic approach which would accurately measure the efficiency of a particular movement on the Rubik's Cube. A movement leading away from the solution would be given a negative heuristic value whereas a movement leading towards thefinal solved state would be given a high heuristic value.

d. To implement an algorithm to solve the Rubik's 3x3 cube.

e. To divide the algorithm in stages for ease of distribution of the algorithm over multiple processors.

f. To distribute the algorithm over multiple processors.

g. To extend the algorithm to give it the generalized capability to solve an NxN Rubik's Cube.

2. RUBIK'S 3X3 CUBE

1) Programming Language:

The choice of programming language for the project was JAVA, due to its extensive functionality and ease with which Remote Method Invocation (RMI) could be implemented. NetBeans IDE and Edit Plus were used as aids in the coding process while Windows XP was the choice of operating system.

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 3, Issue 2, pp: (687-691), Month: April - June 2015, Available at: www.researchpublish.com

2) Graphical User Interface (GUI):

Our aim was to create an accurate depiction of the Rubik's 2X2 cube and present the user with two views:

a. Two-dimensional view.

b. Three-dimensional view.

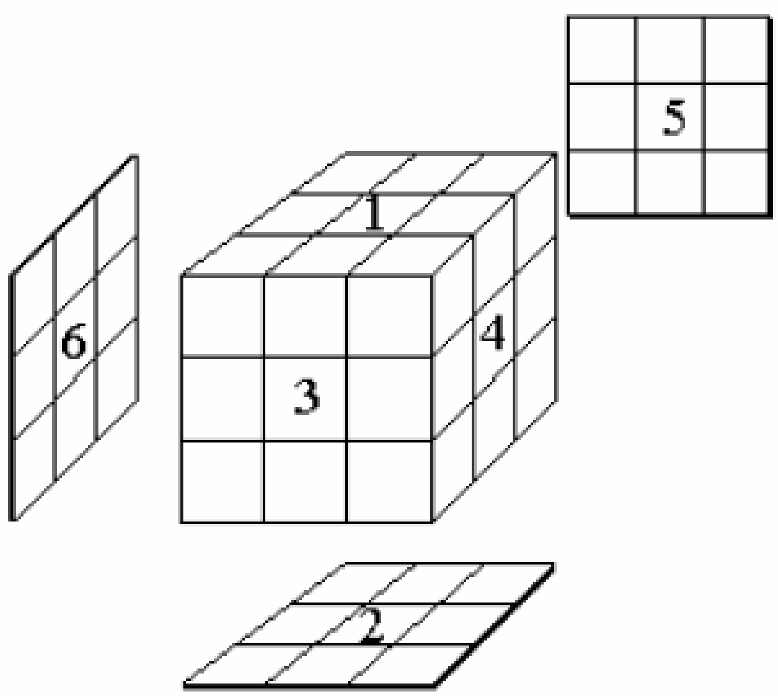

Fig.2. 3-D Representation of 3x3 Rubik's Cube

The various faces can be represented using the following notation:

I. F: Front face

II. T: Top face

III. B: Back face

IV. D: Bottom/Down face

V. R: Right face

VI. L: Left face

Rotation is represented using following terminology:

I. **R:** Rotate Right face 90 degrees clockwise.

II. **R':** Rotate Right face 90 degrees anticlockwise.

For example: **RU'L2**

The above notation implies the following:

a. Rotate Right face 90 degrees clockwise.

b. Rotate Top face 90 degrees counter clockwise.

c. Rotate Left face 180 degrees.

3) Cube Rotation:

The virtual cube, an exact simulation of the Rubik's cube was given the ability of rotating about its central axis in the following ways:

a. Front Clockwise: Rotating the front face in the clockwise direction (F)

b. Front Anticlockwise: Rotating the front face in the anticlockwise direction (Fi)

c. Right Clockwise: Rotating the right face in the clockwise direction. (R)

d. Right Anticlockwise: Rotating the right face in the anticlockwise direction. (Ri)

e. Left Clockwise: Rotating the left face in the clockwise direction (L)

f. Left Anticlockwise: Rotating the left face in the anticlockwise direction(Li)

g. Back Clockwise: Rotating the back face in the clockwise direction (B)

h. Back Anticlockwise: Rotating the back face in the anticlockwise direction.(Bi)

i. Down Clockwise: Rotating the back face in the clockwise direction (D)

j. Down Anticlockwise: Rotating the back face in the anticlockwise direction.(Di)

k. Upper Clockwise: Rotating the back face in the clockwise direction (U)

l. Upper Anticlockwise: Rotating the back face in the anticlockwise direction.(Ui)

3. THE ALGORITHM

1. Start
2. Initialize the Rubik's Cube.
3. Scramble the Rubik's Cube
4. Select face color which is already done or
5. Select any base color start rotation of Edge-center color towards base.
6. If base color and Edge-center color not matched then Fi U Li Ui.
7. Next for corner pieces Ri Di R D until Upper face completes
8. At this position upper face and first layer get solved
9. Flip the cube upper to down and match second layer Edge-center
   (1) From upper edge-center to second layer left Edge-center Ui Li U L U F Ui Fi
   (2) From upper edge-center to second layer right Edge-center U R Ui Fi U F until second layer completes.
10. After go for upper face match again Edge-center with 'L' shape then with front face that one F R Ri Ui Fi until matches.
11. For each base color and Edge-center color matches R U Ri U R U U Ri
12. Next for corner pieces U R Ui Li U Ri Ui L until they are in right places then Ri Di R D magically Rubik has been solves.

4. DISTRIBUTING THE ALGORITHM

1) Remote Method Invocation (RMI):

If a client has access to an object on a different machine, then using this mechanism, the client can invoke remote methods of that object. It is RMI that handles shipping of the remote parameters to the client machine.

2) Distributed Computing:

The cube is randomly generated at the server and sent to three or more client machines. Using Remote Method Invocation the clients attempt to solve the cube by invoking procedures step1, step2, step3 … of the algorithm.

At each stage, the client machine upon fastest completion of the intermediate step would send the result to the server. The other clients would stop processing upon notification of result having been acquired. The server would then send the newly acquired cube to the clients which would then perform next step.

This procedure would be repeated for all three steps until final solution was obtained. As a result, the fastest of the three solutions is adopted to solve the puzzle, thus proving wrong the adage: *Too many cooks spoil the broth*.

5. CONCLUSION

A Edge-Centric approach for solving the Rubik's cube was analyzed and successfully designed. An algorithm for the solution of the Rubik's 3x3 cube was developed and a strategy for the successful deployment of the algorithm over multiple processors was devised.

It was observed that the algorithm for solving the 3x3 cube was not much different to that of the Rubik's Mini Cube. An interesting observation to note was the presence of a central element on each and every face on the Rubik's 3x3 cube that remained constant. Thus, the color of the faces of the 3x3 cube was known in advance which lent a certain degree of simplicity to the cube.

ACKNOWLEDGMENT

I would like to take this opportunity to thank the Computer Department at Alva's Institute of Engineering Collage for providing us with the necessary resources needed for the successful implementation of the project. I would also like to thank Mr. Mahesh Kaluti in helping me for his guidance through the completion of this technical publication.

REFERENCES

[1] David Lee and Winston Miller, Using the Best Fast Search Algorithm and Profile Tables to solve the Rubik's Cube.

[2] Wilkipedia, Distributed Computing

[3] Hart, P. E., Nilsson, N. J.; Raphael, B. (1968). "A Formal Basis for the Heuristic Determination of Minimum Cost Paths".IEEE Transactions on Systems Science and Cybernetic.

[4] Hart, P. E., Nilsson, N. J.; Raphael, B. (1972). "Correction to "A Formal Basis for the Heuristic Determination of Minimum Cost Paths"".

[5] Nilsson, N. J. (1980). Principles of Artificial Intelligence. Palo Alto, California: Tioga Publishing Company.

[6] Russell, S. J.; Norvig, P. (2003). Artificial Intelligence: A Modern Approach,