Analysis of Sudoku Solving Algorithms

M.Thenmozhi [1], Palash Jain [2], Sai Anand R[3], Saketh Ram B[4]
Information Technology Department, SRM University, Chennai, Tamil Nadu, India
[1] thenmozhi.m@ktr.srmuniv.ac.in
[2] palashmjain@gmail.com
[3]saianand0427@gmail.com
[4]saketh.ram804@gmail.com

Abstract— **The idea of this paper is to analyse the performance of various Sudoku solving algorithms. A well laid out groundwork has been done to find out the efficient and fast approach to making the computer solve a Sudoku. After considering multiple options like Backtracking and Brute Force Algorithm, the research ended up with DANCING LINKS algorithm that was proposed to the world by DONALD KNUTH, which he claimed was the best approach to solving a Sudoku from a computational perspective. Also, a comparative analysis of the chosen algorithm with the more traditional approaches was performed to know the competence of our algorithm.**

Keyword - Dancing Links, Sudoku solver, Backtracking, Brute Force, Donald Knuth.

I. INTRODUCTION

Sudoku[1] is a puzzle made of a grid of little cells consisting nine squares which are further divided into nine small squares. The numeric values in cells;0-9, are entered in such a way that every number appears once in each horizontal line, vertical line and square. The phenomenon originated in 1980's in US, failed to gain popularity but Japan puzzle enthusiasts made it grow like a storm throughout the world. The storm resulted in the selling of 40000 sudoku title books in the US in a week, to make it the 3rd highest selling after the JK Rowling's "Harry Potter and the Half-Blood Prince" and Kevin Trudeau's "Natural Cures They Don't Want You to Know About." Enter Gould was the first one to develop a software for generating sudoku problem. He was about to earn millions just by building the software but now provides free puzzle to 120 newspapers across 36 countries.

In this computation world, an algorithm is a background process for every problem statement. An algorithm here takes an unsolved Sudoku grid as input and produces the solution through the application. After looking such enthusiasm of puzzle solver, we decided to work on the advanced and efficient solution. The speed of calculation was the real power of advanced Sudoku solution here. The algorithm which we analysed after a comparative study is DONALD KNUTH's Algorithm X using Dancing Links[2], holds real driving power of speed.

II. EXISTING ALGORITHMS

The puzzle who took the world by the storm has around 6 trillion possible solutions. There are many ways to solve a puzzle. It ranges from step to step pen and pencil method of solution finding to the algorithmic approach to have a progressive solution. Such algorithm helps in creating a software to solve and at the same time make us find the difficulty level of problem. There are many existing algorithms like heuristics and meta-heuristics[3] approaches to solving sudoku efficiently. The prominent methods considered for solving sudoku are Backtracking and Brute Force[4]. Out of this, Brute Force emerged to be more promising.

A. Backtracking Algorithm

Backtracking[5] is a progressive algorithm that considers every possible solution within defined constraints to get the solution. Backtracking Algorithm, in regards with Sudoku, is a simple method of filling one square and then moving to the other square, filling it with one number after counting 0-9, satisfying the situation. Once we are left with no choice to fill then roll back to the previous square and fill it will another number and then following the same.

B. Brute Force Algorithm

Brute Force Algorithm is considered as one of the easiest methods of solving a sudoku. Under this method, one visits every empty square and fill it with given options and continue till we find any dead end. Once the dead end is found, just backtrack to the old square and replace it with another number. It is simply relying on the muscle power keeping no technique in mind.

A sudoku solution is guaranteed, as long as the puzzle is valid, upon using this algorithm. The solving time is mostly unrelated to the degree of sudoku problem difficulty.

Once modelled after deductive methods, it may turn out to be slow as compared with the computer solution methods. The backtracking process after reaching the dead end costs higher in terms of speed and time.

The previous research[6] holds that Brute Force using backtracking for iterative processing uses backtracking and hence proved to be faster than the Backtracking algorithm.

III. PROPOSED IDEA

Sudoku, in the past, has proven to be a complex problem in both the field of mathematics and computer science. We are trying to accomplish a Sudoku solving application that makes use of the ALGORITHM X. Since any application demands both speed and accuracy the chosen algorithm must be able to meet both the requirements. Our options included the basic BRUTE FORCE, BACKTRACKING, DANCING LINKS (ALGORITHM X). As the chosen algorithm was a derivative of the Backtracking problem the analytical comparison included only Brute Force and Algorithm X. The chosen algorithm makes use of the Backtracking process including a concept called Dancing Links introduced by DONALD KNUTH.

IV. CONSTRAINT PROGRAMMING

Sudoku is a constraint problem. In his paper Sudoku as a Constraint Problem, Helmut Simonis[7] describes many reasoning algorithms available in the form of constraints which can be applied to model and solve the problem. Some constraint solvers include an example how to the model and solve Sudoku problems. Once the constraint program modelling is used for sudoku solving it takes less than 100 lines of code in most solvers.

An exact cover problem is a problem where there are a given set of choices, and constraints and the user's challenge are to select the set of choices that will fill every constraint only once.

For example, consider the case of a magician who creates his show timeline. He holds a lot of tricks of disguise but always plans as such that no tricks get repeated again. Even though having a lot of tricks in his magic hat he chooses the ideal selection to cover all the tricks only once. In this example, the constraints are that he must perform every trick. The trick choices are the possible show timeline he could incorporate into his timeline.

A better way to represent such type of problems is to draw out a table where the constraints are columns and the choice are rows, and we have a big X in cells where a particular choice fulfils that constraint. As it turns out, on the given right constraints and choices, Sudoku can be described as an Exact Cover Problem.

We drew our inspiration from the solution that Donald Knuth suggested was the best approach to an exact cover problem. There have been many approaches when it comes to solving the Sudoku. It is one of the Intriguing problems and involves multiple complex operations.

A. Sudoku as a constraint problem

Sudoku can be transformed into an exact cover problem[8] using a technique called "Dancing Links" which was put forth by Donald Knuth. This allows both for an elegant description of the problem and an efficient solution using a backtracking algorithm. The usage of algorithms for exact cover, such as Dancing Links, typically solves a 9X9 sudoku problem with the computation time of the order of seconds. While exact cover does not guarantee efficient solution time for large grids.

In technical aspect, Algorithm X is a recursive[9], non-deterministic, depth-first[10] and backtracking algorithm. For solving such exact cover problem, Donald Knuth[11] used "Algorithm X" as the reference name for this trial and error approach in his paper called "Dancing Links".

V. MATERIALS AND METHODS

A. Methodology

The implementation was done in C# with our initial approach of including unity to package for android and iOS. But due to the technical difficulties, the approach has been changed to having a server in visual studio and creating a client in either of the devices.

B. How it works

As explained before the Algorithm X uses the technique Dancing Links to solve an exact cover problem or a Sudoku statement. The Dancing Links works in the following way.

1) Let us have a square matrix filled with 0's and 1's.
2) Either a column or a row with the least number of 1's is taken and corresponding columns or rows are selected which have the 1's and the rest are omitted.
3) After repeating this process for several times based on the size of the matrix. We get a matrix that has a 1 in all rows and columns.
4) This means that the dancing links gives the final result where there is a one in that row or a column, which is essential to solving a Sudoku problem.

Though the dancing links is an amazing technique to solve an exact cover problem it cannot by itself give a solution in real time. Hence Donald Knuth came up with the Algorithm X to reduce the number of processing statements and hence reduce the time complexity of the algorithm.

VI. ALGORITHM X

The below mentioned is the pseudo code of the Algorithm X:

if matrix a has no columns then
    the current partial solution is a valid solution;
    terminate successfully;
else
    otherwise choose column c (deterministically);
    choose a row r such that ar, c=1 (non- deterministically);
    include row r in the partial solution;
    for column j such that aj=1 do
        for row i such that ai, j=1 do
            delete row i from matrix a;
            delete column j from matrix a;
        end
    end
end
Repeat this algorithm recursively on the reduced matrix a;

VII. RESULTS AND DISCUSSIONS

A. BRUTE FORCE ALGORITHM

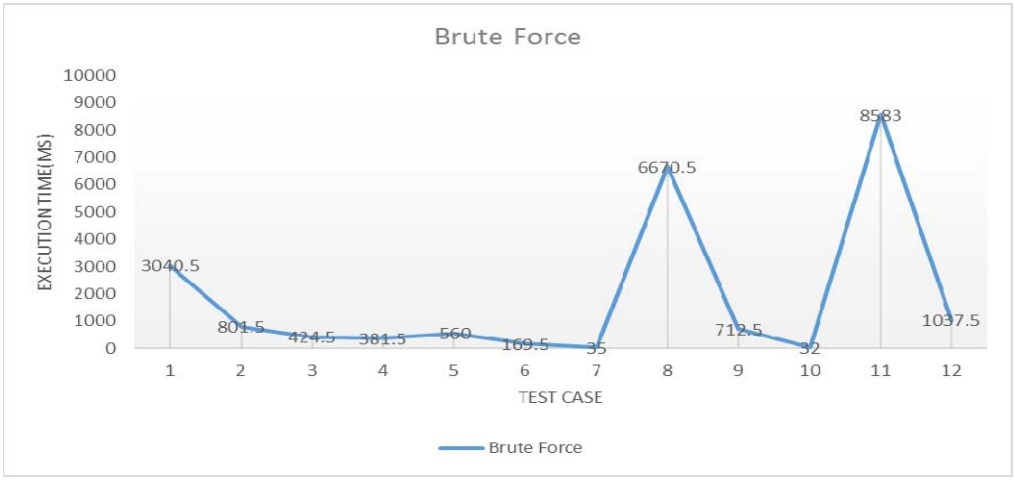

Fig. 1. Test Cases versus Execution Time(milliseconds) for Brute Force Algorithm

B. ALGORITHM X

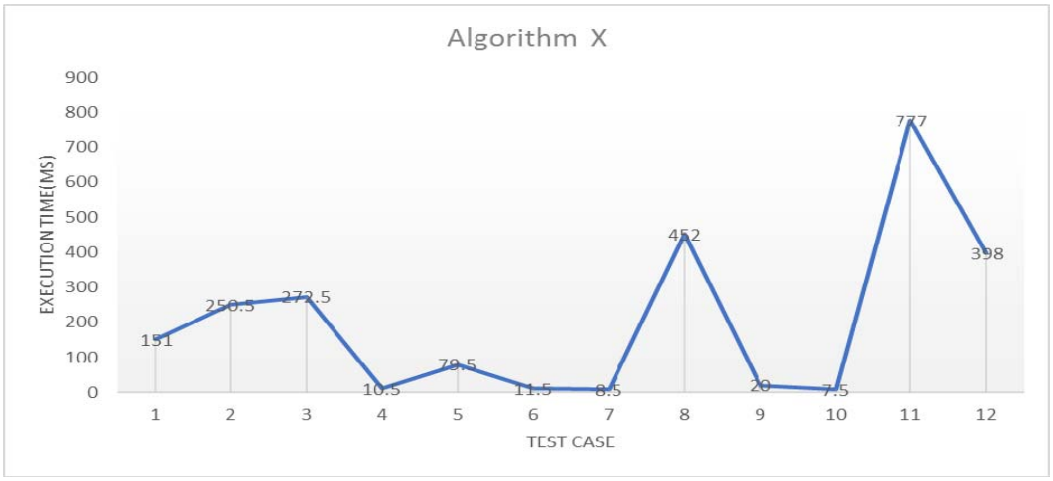

Fig. 2. Test Cases versus Execution Time(milliseconds) for Algorithm X.

This is a simple iterative approach to the same rather than using the traditional recursive approach.

Theoretically, the computation time of brute force algorithm is **O(M *N^3)** while Algorithm X has **O(N^3).** where M are the number of zeroes present in the grid and N equals to 9.

T(Algorithm X solution) <= T(K*brute force solution)

Where K= 1/M

while they both were clearly distinctive on paper, practical efficiency was key for the successful working of the application.

TABLE I. Algorithm X versus Brute Force Algorithm

Test Case	Algorithm X*	Brute Force*
1	151	3040.5
2	250.5	801.5
3	272.5	424.5
4	10.5	381.5
5	79.5	560
6	11.5	169.5
7	8.5	35
8	452	6670.5
9	20	712.5
10	7.5	32
11	777	8583
12	398	1037.5

*All execution time values are in milliseconds.

C. ALGORITHM X VS BRUTE FORCE

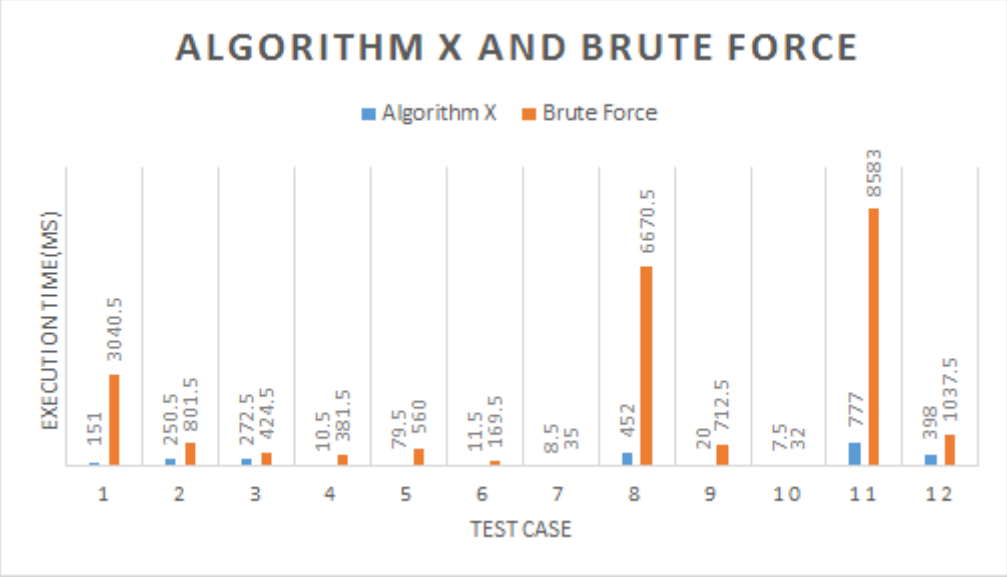

Fig. 3. Test Cases versus Execution Time(milliseconds) comparison between Algorithm X and Brute Force Algorithm.

In order to properly measure the potential and analyse the algorithm x, many complex and hard level Sudoku problems were taken and both were run in two different algorithms.

One was the Brute Force algorithm and the other was the employed Algorithm X, the brute force runs every probable option for all the grids until an acceptable answer is reached and the Algorithm X eliminates possibilities to increase efficiency and accuracy.

There were 12 different samples that were run to make a comparative analysis of the algorithms. After they were executed the difference between the runtimes was phenomenal, the Algorithm X was faster than the Brute Force algorithm by a lot, by a whole second in few cases. This kind of combination of accuracy and speed was essential for the efficiency of the application which made the employed algorithm the best choice without question.

VIII. CONCLUSION

After a complete analysis of the potential bore by the algorithm X and the efficiency it offers the application through the concept of dancing links, we can infer that the algorithm X is far advanced and superior compared to traditional algorithms in an approach to solve a Sudoku. The statistical analysis clearly states that higher the difficulty level of sudoku problem, the higher is the calculation time difference between the Algorithm X and Brute Force.

IX. FUTURE SCOPE

The speed of Algorithm X execution against Brute Force Algorithm has done all the talking. But experiencing the speed with naked eyes gives more satisfaction. To get it to that level we can use Augmented Reality, a new boom in the technical world. The fetching of physical Sudoku on the computer screen though the camera and then providing the solution using the Algorithm X within milliseconds back on the screen will make anyone feel the speed against existing slow algorithm.

REFERENCES

[1] various: Sudoku Wikipedia entry. http://en.wikipedia.org/wiki/Sudoku (2005)
[2] Knuth, Donald E. (2000), "Dancing links", in Davies, Jim; Roscoe, Bill; Woodcock, Jim, Millennial Perspectives in Computer Science: Proceedings of the 1999 Oxford-Microsoft Symposium in Honour of Sir Tony Hoare, Palgrave, pp. 187–214, arXiv:cs/0011047, ISBN 978-0-333-92230-9.
[3] Lewis,R,Meta heuristics, 2007, Can solve Sudoku puzzles .Journal of heuristics,13(4),387-401.
[4] Xu,J, 2009, Using backtracking method to solve Sudoku puzzle, computer programming skills & maintainence 5,pp 17-21.
[5] various: https://en.wikipedia.org/wiki/Sudoku_solving_algorithms
[6] Akta Agrawal,Padma Bonde, July 2015, Study and Performance Characteristics of Sudoku Solving Algorithms, International Journal of Computer Applications (0975 – 8887) Volume 122-No.1
[7] Simonis, Helmut (2005). "Sudoku as a Constraint Problem" . H Simonis homepage. Cork Constraint Computation Centre at University College Cork: Helmut Simonis. Retrieved 8 December 2016. paper presented at the Eleventh International Conference on Principles and Practice of Constraint Programming"
[8] Rob Beezer,October 15, 2010, Solving Sudoku with Dancing Links, Department of Mathematics and Computer Science, University of Puget Sound,Tacoma, Washington USA.
[9] Graham, Ronald; Donald Knuth; Oren Patashnik (1990). Concrete Mathematics. Chapter 1: Recurrent Problems.
[10] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. Introduction to Algorithms, Second Edition. MIT Press and McGraw-Hill, 2001. ISBN 0-262-03293-7. Section 22.3: Depth-first search, pp. 540–549.
[11] various:Knuth's Algorithm-X wikipedia entry: https://www.revolvy.com/main/index.php?s=Knuth%27s%20Algorithm%20X&item_type=topto

AUTHOR PROFILES

Dr.M. Thenmozhi[1] is an Assistant Professor and has received her Ph.D from SRM university and M.E. degree from Anna University. Her research interests are data structures, algorithms and database. She has 18 years of teaching experience in engineering education. She is teaching for computer science students and her favourite subjects are design and analysis of algorithms, Database management systems and real time systems.

Palash Jain[2], an undergraduate student with Information Technology as major from SRM University Chennai, India. His research interest is in the area of algorithm optimization that improves the efficiency of the various problem. He holds interest in networking field as well and undergoing a CCNA certification course.

Sai Anand R[3], Saketh Ram[4], are undergraduate students with Information Technology as major from SRM University Chennai.