VIT UNIVERSITY

COMPARITIVE STUDY OF DIFFERENT SUDOKU SOLVING ALGORITHMS

ANMOL JAIN
16BCE0313
anmoljain.2016@vitstudent.ac.in
VIT UNIVERSITY

HARSHIT SINGHAL
16BEC0470
harshit.singhal2016@vitstudent.ac.in
VIT UNIVERSITY

GURKARAN SAHNI
16BCB0051
gurkaran.sahni2016@vitstudent.ac.in
VIT UNIVERSITY

Faculty in Charge: Prof. Gayathri P.

VIT UNIVERSITY

COMPARITIVE STUDY OF DIFFERENT SUDOKU SOLVING ALGORITHMS

ANMOL JAIN
16BCE0313
anmoljain.2016@vitstudent.ac.in
VIT UNIVERSITY

HARSHIT SINGHAL
16BEC0470
harshit.singhal2016@vitstudent.ac.in
VIT UNIVERSITY

GURKARAN SAHNI
16BCB0051
gurkaran.sahni2016@vitstudent.ac.in
VIT UNIVERSITY

ABSTRACT

Sudoku is a logic-based, combinatorial number-placement puzzle. The objective is to fill a 9×9 grid with digits so that every column, every row, and every 3×3 sub grids that compose the grid contains all of the digits from 1 to 9.

The prime objective of this thesis is to comparatively study different Sudoku solving algorithms. Focus is going to be on the solving ability of the algorithm but other aspects such as time complexity etc. will also be taken into account. In this thesis three solving algorithms are focused and the above-mentioned comparisons will be done on these three techniques only.

The algorithms to be evaluated are **backtrack** and **brute force.** Results are going to be presented in the form of comparative graphs with considerable amount of input data for approximate results. All the results have been compiled in DEV-C++ 5.11.

KEYWORDS

Backtrack
Brute-force

1. INTRODUCTION

1.1 SUDOKU

Sudoku is a logic based number puzzle represented by a square grid. The most common variant of Sudoku has a grid consisting of 9 * 9 cells that are divided into boxes that consist of 3* 3 cells each. Numbers ranging from 1 to 9 shall be placed in the cells, filling the grid, so that every number occurs only once in each row, column and box. It exists different difficulty levels in Sudoku, where difficulty is based on how many clues are given from the beginning and partially on the layout of these. A Sudoku usually has at least of 17 clues and only one, distinct, solution. As previously mentioned there are 6.67 * 10^21 different Sudoku [1][2] but not all of them are playable by humans, at least not without guessing [3]. A common property of the unsolvable Sudoku is that they have multiple solutions, which means that not enough clues has been given to provide a distinct solution [5].
Sudoku became popular in Japan in the mid 80's but similar games had existed before that. It became widely popular around 2004 when it started to appear in newspapers where it became an appreciated alternative to the traditional crosswords. Today Sudoku does not only appear in newspapers but also as computer games and in international Sudoku championships. [6]

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

(a basic 9*9 sudoku)

5	3	4	6	7	8	9	1	2
6	7	2	1	9	5	3	4	8
1	9	8	3	4	2	5	6	7
8	5	9	7	6	1	4	2	3
4	2	6	8	5	3	7	9	1
7	1	3	9	2	4	8	5	6
9	6	1	5	3	7	2	8	4
2	8	7	4	1	9	6	3	5
3	4	5	2	8	6	1	7	9

(solution to the above problem sudoku)

1.2 SOLVING SUDOKUS

A Sudoku game begins with some of the cells already filled in with clues and the aim is to fill the leftover empty cells. Sudoku adapts only one rule [7]:
"Every row, column and box must end up containing all of the numbers from 1 to 9."
The above mentioned rule has an important outcome, which is the basis of all problem solving techniques.
"Each number can only appear for once in a row, column and box."
Therefore, a Sudoku game is over when there is a number in every cell and the one rule of Sudoku is satisfied.
There are many different ways to solve a Sudoku. Different algorithms and techniques work well in different occasions. In this section we will discuss some of the more common algorithms and

techniques that can be used when solving Sudokus with a focus on the techniques that we are studying in this thesis [5].

1.3 THE RULES FOR SUDOKU

Sudoku is a number puzzle traditionally comprising of a 9 x 9 square grid of cells where each cell contains a single integer between 1 and 9 (inclusive). The grid is than further divided into boxes of 3 x 3 cells. The prime objective of the game is to fill the entire grid up so that a given number appears no more than once in any row, column or box. We are interested in the more general sudoku problem, where we are supplied with a grid of n x n cells. Each box is now a sub grid of $\sqrt{n}$ x $\sqrt{n}$ cells and each cell may contain the values 1 to n. The rules of the game remain the same. For the leftover discussion, we will refer to these rules as constraints. Formally, given an n x n sudoku grid i.e.. S, we define the sudoku constraint as,

1. **Cell Constraint :** All cells Cij ∈ C may contain no more than 1 value AND The value must be between 1 and n.
2. **Row Constraint :** All value in rows Ci ∈ C must be distinct, i.e.
   If Cij = x then it is necessary to have remaining elements in Ci not equal to x.
3. **Column Constraint :** All values in columns Cj ∈ C must be unique,
   i.e. If Cij = x then it is necessary to have remaining elements in Cj not equal to x.
4. **Box Constraint :** We define a box as being a $\sqrt{n}$ x $\sqrt{n}$ sub-grid of C such that the top left cell of the sub-grid is always Cij
   where $(i - 1)\%\sqrt{n} = 0$ and $(j - 1)\%\sqrt{n} = 0$.
   We denote a box with the top-left cell Cij as Bij . The box constraint thus says that every value in box Bij ∈ S must be unique. You can think that, even with these constraints there is a lot of valid sudoku grids (roughly 6.5×10^{21}, though only around 5.5×109, for the n = 9 case. [1]). And so, in an try to make each sudoku puzzle have only 1 valid solution they are always supplied with some of the cells already filled in. These cells, sometimes referred as "clues", which contain fixed numbers that cannot be changed by the player while they attempt to fill in the left empty cells sticking to the constraint listed above. A Puzzle can be restricted to a unique solution with no lesser than seventeen clues, but it is possible to have up to 77 clues & still don't have a distinct solution.

1.4 PROBLEM

As already mentioned, Sudoku is today a popular game throughout the world and it appears in various Medias, which includes websites, newspapers, and books, etc. As a result, it is of interest to find effective Sudoku solving and generating algorithms. For most purposes there already exists satisfactory algorithms, and it might be hard to see the benefit of studying Sudoku solving algorithms. There is, however, still some value in studying Sudoku solving algorithms as it might reveal how to deal with difficult variations of Sudoku, such as puzzles with 16x16 grids. 1 goal of this study is therefore to contribute to the discussion about how such puzzles can be dealt with.

1.5 METHODS USED TO SOLVE THE SUDOKU

The different algorithms that are used to solve Sudokus vary in complexity depending on approach. The brute-force algorithm fills in the missing numbers without any sense of logic and check afterwards if it was a valid placement or not. These algorithms use exhaustive search to solve Sudokus, which means a large amount of backtracking[15] and guessing; Hence not suitable to be used by a human. An advantage with this method is that it will always find a solution. However this also means that it will try all different solutions and since there are 6.67 * 10^21 possible Sudokus it could take a very long time in the worst-case scenario. [1][2] A human rule based algorithm uses a series of techniques that are based upon the given restrictions of Sudoku.

The techniques are then applied by the solver in order (from easiest to hardest) to find a possible number placement. One rule might be to check for any naked singles in a row, meaning searching for locations where one and only one number can be placed according to the Sudoku restrictions. A problem with this algorithm is that it does not guarantee that a solution will be found because the rules are not exhaustive, i.e. there are Sudokus that cannot be solved using only rules [3].

1.6 THE DIFFICULTY LEVEL OF SUDOKU PUZZLES

The difficulty level [16] of Sudoku puzzles depends on how the given number are placed in the Sudoku board and also how many numbers are given. In general, the most specific aspect of difficulty ratings of Sudoku puzzles is that which techniques are needed to solve the puzzles. In different words, it is important where the numbers are placed logically .Generally, a Sudoku puzzle needs at least 17-clues to solvable. It means that solving a Sudoku puzzles with 17 Clue is more difficult than a 30-clues.More given numbers, the easier and quicker the solution is. This statement may not always truth that if the number of clues becomes more the run-time of solving the puzzle would be shorter. For example, when solving the puzzle with 28 clues the solving time is increases quickly. The reason is that puzzle needs more methods to solve it or the algorithm needed to repeat as long as the solution is found. Explanation of levels:

1. **Extremely Easy**

- It will have more than fifty clues

2. **Easy**

- Sudoku will have 36 - 49 clues.
- Has more than one given in every box.
- In this level each digit from 1–9 appears as a given at least 3 times.
- It is very rapid to solve, less than 10 minutes, and requires no guesses.

3. **Medium**

- Sudoku have around 32–35 clues.
- It has a couple of boxes with only one given.
- This level has some digits may only appear twice, the rest will appear for at least 3 times.
- It take a bit longer to solve, somewhere between 10–20 minutes. No prediction are required.

4. **Hard**

- It has around 28–31 givens.
- It might be having couple of boxes with only 1 given
- This level has three or four digits might only appear twice as givens, and one digit may only appear once.
- It may take up to forty-five minutes to solve, & some trial and error may be needed for example, 1 box may have two candidates, and no way of finding which is correct.

5. **Very Hard**

- Sudoku may have several boxes with less than 22 - 27 givens.
- It may have some boxes with no givens at all
- This level has most digits appear only two or three times, as well as several single occurrences.
- It usually takes over an hour to solve, and require trial and error.

2. LITRATURE REVIEW

Several research have been made to solve Sudoku problems in a more efficient way. Various exact implicit enumeration ,heuristic and meta-heuristics [10] approaches have been made to solve Sudoku efficiently .The most primitive approach to solve Sudoku has been done through brute-force technique which guarantees a logical solution of any given problems Backtracking ,on other hand ,has been found to be more promising in solving Sudoku problem [10] by reducing the search for a solution to a greater extent .A new backtrack based enumerative algorithm using Graph referencing method (GRA) has been reported by author [11] . Genetic [12] algorithms(GA) have been revealed to be an effective approach in successfully handling Sudoku as multi – objective optimization problem [13].The optimization algorithm describes a new Harmony search (HS) algorithm to solve Sudoku using meta heuristic search technique has been reported by Geem [8]. Other algorithms based on Simulated Annealing (SA) has been suggested by Henirik et al [9].

3. PROPOSED MODEL

3.1 Brute-Force Algorithm

One of the better-known algorithms that are used when solving Sudokus on a computer is called brute-force. The brute-force algorithm is sometimes called an exhaustive search algorithm and it has become famous in computer science because of its striking characteristics: It always finds a solution because it will try all possible solutions hence it is not considered to be very effective [8]. A Sudoku solver that uses brute-force will visit all the empty cells and try to fill it with a number from the available choices. If no violations are found after an insertion it will continue to the next empty cell and start over. As soon as a violation to the Sudoku rules are found the algorithm will backtrack and increment the preceeding cell. [14]

Pseudo Code

```
boolean solve() :=
 x = 0, y = 0
 for x,y in grid:
if grid[x][y].value == 0
found = true
break
if !(found)
return valid()
candidates = boolean[10]
for i := 0 -> 9:
candidates[grid[x][i].value] = true
candidates[grid[i][y].value] = true
for (cells in same box as cell x,y):
candidates[cell.value] = true
for j := 1 -> 9:
if !(candidates[j]): grid[x][y].value = j
if solve() return true
grid[x][y].value = 0
return false
```

3.2 Backtracking

The backtrack algorithm can be viewed as guessing which numbers goes where. When a dead end is reached, the algorithm backtracks to an earlier guess and tries something else. This means that the backtrack algorithm does an exhaustive search to find a solution, which means that a solution is guaranteed to be found if enough

time is provided. Even though this algorithm runs in exponential time, it is plausible to try it since it is widely thought that no polynomial time algorithms exists for NP-complete problem, for example Sudoku. One way to tackle such problems is with brute-force algorithms provided that they are considerably fast. This method can also be used to determine if a solution is unique for a puzzle as the algorithm can easily be modified to continue searching after finding one solution. It tracks that the algorithm can be used to produce valid Sudoku puzzles (with unique solutions).

Pseudo Code

For template of (9x9) Sudoku. Start 2D array with 81 empty grids(nx=9,ny=9) now fill some numbers to make the problem and Make an original copy of the array.
Start from top left grid(nx=0,ny=0), check whether grid is empty
if (grid is empty)
assign the empty grid with values (i)
if (no numbers exists in same rows & same columns same as
(i) & 3x3 zone (i) is currently in)
fill in the number
if (numbers exists in same rows & same columns same as
(i) & 3x3 zone (i) is currently in)
reject (i) and repick other values (i++)
}else{
while (nx<9){
Proceed to next row grid(nx++,ny)
if (nx equal 9){
reset nx = 1
proceed to next column grid(nx,ny++)
if (any equal 9){
print solution}}
}

4. RESULTS AND DISCUSSION

The following testing result exposes the performance of algorithm which is better with respect to computing time with any levels of difficulties.

Table of resolving time of the backtracking and brute-force method with respect to levels.

LEVELS:	Solving time on computer by algorithm(in seconds)	
	Backtracking	Brute-force
Easy	0.015991	0.013917
Medium	0.017014	0.016022
Hard	0.020996	0.019989

This study shows that brute-force algorithm is more feasible to solve any Sudoku puzzles, also an appropriate method to find a solution faster and more efficient in solving ability, representation, and performances with any level of difficulties. This study will also be helping hand in statistical tests and method to find some more results for comparing. Future work includes study of number of clues and run time of each difficulty level and comparing Brute Force and Backtracking algorithm based on each difficulty level.

5. CONCLUSION

This study has shown that the brute-force algorithm is a feasible method to solve any Sudoku puzzles. The algorithm is also an appropriate method to find a solution faster and more efficient compared to the backtracking algorithm. The proposed algorithm is able to solve such puzzles with any level of difficulties in a short period of time (less than one second). The algorithm seems to be a useful method to solve any Sudoku puzzles and it guarantee to find a solution. The algorithm does not adopt intelligent strategies to solve the puzzles. This algorithm checks all possible solutions to the puzzle until a valid solution is found which is a time consuming procedure resulting an inefficient solver. As it has already stated the main advantage of using the algorithm is the ability to solve any puzzles and a solution is certainly guaranteed. Further research needs to be carried out in order to optimize the rule based algorithm.

6. REFERENCES

1. Felgenhauer B, Jarvis F. Enumerating possible Sudoku grids. [Internet]. 2005 [cited 2013 Mar 21]. Available from: http://www.afjarvis.staff.shef.ac.uk/sudoku/
2. Felgenhauer B, Jarvis F. Mathematics of Sudoku I. [Internet]. 2006 [cited 2013 Mar 21]. Available from: http://www.afjarvis.staff.shef.ac.uk/sudoku/felgenhauer_jarvis_spec1.pdf
3. Mepham M. Solving Sudoku. [Internet]. 2005 [cited 2013 Apr 9]. Available from: http://www.sudoku.org.uk/PDF/Solving_Sudoku.pdf
4. Lewis,R,Meta heuristics, 2007, Can solve Sudoku puzzles .Journal of heuristics,13(4),387-401.

5. Stuart A. Sudoku Solver. [Internet]. 2005 [updated 2012 Oct 22; cited 2013 Mar 4]. Available from: http://www.sudokuwiki.org/sudoku.htm
6. Nationalencyklopedin. sudoku. [Internet]. c2013 [cited 2013 Apr 8]. Availible from: http://www.ne.se/lang/sudoku
7. Johnson A. Solving Sudoku. [Internet] c2005 [cited 2013 Apr 10]. Available from: http://angusj.com/sudoku/hints.php

8. Geem,Zong WOO, 2007, Harmony search algorithmpp371-378.

9. Viksten ,Henrik, 2013, Performance and scalability of Sudoku solvers,Bachelor's Theses at NADA,Sweden

10. Xu,J, 2009, Using backtracking method to solve Sudoku puzzle, computer programming skills & maintainence 5,pp 17-21.

11. Chakraborty ,r,Palidhi, S,Banerjee,, 2014, An optimized Algorithm for solving combinatorial probl;em using reference graph IOSR Journal of CS 16(3PP1-7)

12. Darwin , c, 1859, The origin of species Oxford University.

13. Mantere T Kolljonen solving rating & generating Sudoku with GA,Proc.of IEEE 1382-1389

14. Moler C. Cleve's Corner: Solving Sudoku with MATHLAB. [Internet]. 2009 [cited 2013 Apr 9]. Available from:http://www.mathworks.se/company/newsletters/news_notes/2009/clevescorner.html

15. E. Gurari, "Backtracking algorithms." http://www.cse.ohio-state.edu/~gurari/ course/cis680/cis680Ch19.html#QQ1-51-128.

16. Ch. X u , W. X u ,2009, The model and algorithm to Estimate The Difficulty Levels of Sudoku puzzles, Journals of Mathematicsv Research.