A Comparison of Three Sudoku Solving Methods

Onokpasa Eva
Department of Computer Science,
University of Jos

Bakwa Dunka
Department of Computer Science,
University of Jos

ABSTRACT

The 9 X 9 board game of Sudoku is intriguing and brain tasking. There are various Sudoku solving methods. This research work is focused on comparing three Sudoku solvers: Pencil and paper method, backtracking and the method of alternating projections. This comparison is carried out by counting the number of iterations taken to solve 40 puzzles of various levels of difficulty, using php implementations of the solver algorithms.

Keywords

Iterations, sudoku, pencil and paper, backtracking & alternating projections

1. INTRODUCTION

In recent years the game of Sudoku has gained popularity and it appears in multiple media, including websites, newspaper, phones and books. More than just a game to entertain, engage and improve one's brain power, Sudoku has found relevance in Steganography, Artificial Intelligence, Aircraft routing, etc [1]. Generating and solving Sudoku puzzles thus, has useful real life applications. Sudoku puzzles are of different variants and different levels of difficulty. There are varying methods as well, to solve Sudoku problems This research is aimed at comparing the Pencil and paper, backtracking and alternating projections methods.

2. LITERATURE REVIEW

In this section a quick review is taken of the pencil and paper method, backtracking and alternating projection methods for solving Sudoku puzzles.

2.1 Pencil and Paper/Human Method/Rule Based Method

This algorithm tests a puzzle for certain rules that either fills in squares or eliminates candidate numbers. This algorithm is similar to the one human solvers use. Below are some of the rules [2]:

2.1.1 Naked Single

This means a cell has only one candidate* number.

2.1.2 Hidden Single

If a region# contains only one cell which can hold a specific number then that number must go into that cell.

2.1.3 Naked Pair

If a region contains two cells, in which each, has only two specific candidates. If such pair exists, then all occurrences of these two candidates may be removed from all other cells in that region. This concept can also be extended to three or more squares.

2.1.4 Hidden Pair

If a region contains only two cells which can hold two specific candidates, then those cells are a hidden pair. It is hidden because those cells might also include several other candidates. Since these cells must contain those two numbers, it follows that all other candidates in these two cells may be removed. Similar to naked pairs, this concept may also be extended to three or more cells.

2.1.5 Locked Candidate

Locked candidates are forced to be within a certain part of a row, column or block. Sometimes you can find a block where the only possible positions for a candidate are in one row or column within that block. Since the block must contain the candidate, the candidate must appear in that row or column within the block. This means that you can eliminate the candidate as a possibility in the intersection of that row or column with other blocks.

2.1.6 Guessing (Nishio)

The solver finds an empty cell and fills in one of the candidates for that cell. It then continues from there and sees if the guess leads to a solution or an invalid puzzle. If an invalid puzzle comes up the solver return to the point where it made its guess and makes another guess. This is similar to the backtracking algorithm.

2.2 Computer Algorithms

2.2.1 Backtracking

Also known as depth first search. When applied to Sudoku, each empty cell is scanned for possible candidates or solutions. Starting with cells with the least candidates one candidate is taken from this set placed in an empty string, if no violation occurs (i.e. the value for that cell maintains the properties of a Sudoku grid), the candidates for the next empty cell is considered, one of these candidates is added to the string, if no violation occurs, the process is repeated until all cells are traversed, to attain a solution. However, if a violation occurs, then the search backtracks to check the other candidates for the cell where the violation occurred. If all the candidates cause a violation, the search backtracks again to the previous cell and explores the next candidate for that cell, before furthering its search. The example shown next, taken from [3] illustrates backtracking algorithm clearly.

Consider the sample Sudoku puzzle in fig 1 with 34 clues. First a list of compatible digits for each empty cell is created, this is done by scanning through the row, column and sub grid of the empty cell as shown in fig 2.

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

Fig 1: Sample Sudoku puzzle with 34 givens

1	**5**	**7**	**6**	**4**	{2,3}	{3,9}	**8**	{2,3}
{6,8,9}	**4**	{6,8,9}	{2,7}	{1,3,5,8}	{2,3,7,8}	{3,5,6,7,9}	{3,5,6,9}	{2,3,5,6,7}
{6,8}	**3**	**2**	**9**	{5,8}	{7,8}	**1**	**4**	{5,6,7}
7	{6,9}	**4**	**1**	{3,9}	**5**	**2**	{3,6,9}	{3,6,8}
2	{1,9}	{1,9}	**8**	**6**	{3,9}	{3,5,9}	**7**	**4**
{3,6,8,9}	{6,9}	{3,6,8,9}	{2,4,6,9}	**7**	{2,3}	{3,5,6,9}	{3,5,6,9}	**1**
{3,4,5,6,9}	**8**	{3,5,6,9}	{7}	**2**	**1**	{3,4,5,6,7}	{3,5,6}	{3,5,6,7}
{5,6,9}	{2,6,7}	{5,6}	**3**	{8}	**4**	{5,6,7}	**1**	**9**
{3,4,9}	{1,9}	{1,3,4,9}	**5**	{9}	**6**	**8**	**2**	{3,7,9}

Fig 2: Same puzzle in fig 1, with the list of compatible digits for each empty cell shown in curly braces

References to cells are achieved using the ordered pair (i,j) where i represents the row, j the column and i and j values are from 1 to 9. L_{ij} will denote the list of compatible digits in the empty cell (i,j). Partial solutions are denoted by $s_{ij}s_{kl}s_{mn}\cdots$ which are character strings taken in dictionary order from the list of compatible digits in cells (i,j),(k,l),(m,n)…

Next, the ranking of the partial solutions based on the cardinality of their set of compatible digits from fig 5, starting from cells with cardinality of 1, then each of their values are added into a list as shown in fig 6.

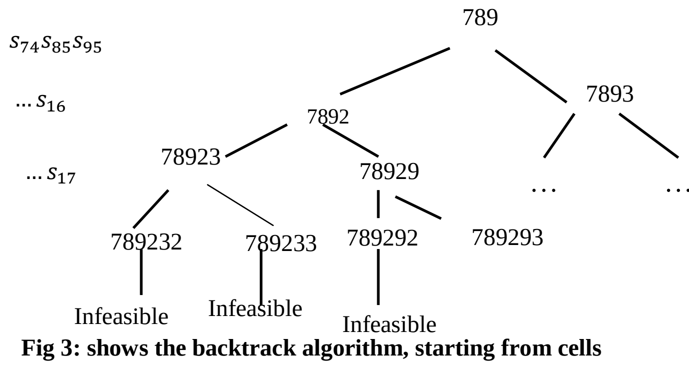

Fig 3: shows the backtrack algorithm, starting from cells with single digit, to double digits

$L_{74} = \{7\}, L_{85} = \{8\}, L_{95} = \{9\}$, the single values are pushed into the list to form

$s_{74}s_{85}s_{95}$= 789. Next, the double value lists. Scanning from the first row of the grid to obtain

$L_{16} = \{2,3\}$, $L_{17} = \{3,9\}$, $L_{19} = \{2,3\}$. Starting with L_{16} the first value 2, is added into the partial solution list which becomes $s_{74}s_{85}s_{95}s_{16} = 7892$, next is L_{17} the first value 3, is added in the partial solution and it gives $s_{74}s_{85}s_{95}s_{16}s_{17} = 78923$. Finally on that row is L_{19} with the first value 2, if this value is added to the list $s_{74}s_{85}s_{95}s_{16}s_{17}s_{19} = 789232$ is obtained, but this is a row violation (i.e. the digit 2 will be repeated on the same row) thus it is necessary to backtrack. The last value is removed and consider the next value in the list of compatible digits for L_{19} which is 3 is considered. Again if this value is placed in the list of partial solutions, $s_{74}s_{85}s_{95}s_{16}s_{17}s_{19} = 789233$ is obtained. Another row violation occurs, again a backtrack is made two steps, since the list of compatible digits for L_{19} has been exhausted. The last two digits in the list of partial solutions are removed. The next value in L_{17} which is 9, is considered. This value is placed into the list of partial solutions to arrive at $s_{74}s_{85}s_{95}s_{16}s_{17} = 78929$. This process continues until the partial solutions grows to complete all empty cells, without any violations and a complete solution of the puzzle is attained. Backtracking has the advantage of finding multiple solutions if they exist.

2.2.2 *Alternating Projections*

The projection of a point y $\in \mathbb{R}^d$ to a set $C \subseteq \mathbb{R}^d$ is a point x* such that the Euclidean distance of x* to y ($\|x * -y\|$) is minimal. This is given by

d(x*,y) $= \|x^* - y\| \leq \|x - y\| \forall$ x ϵ C (C *is a closed and convex set*

Thus the projection operator that maps y to x* ϵ C is denoted by:

$$\boldsymbol{x^* = P_C(y)}$$

x* is unique if C is closed and convex [4]If there are two closed convex subsets of $\mathbb{R}^n$ say C_1 *and* C_2 with non empty intersection then the projection of y onto C_1 *and* C_2 can be found by alternating the projection to C_1 then to C_2 iteratively until it converges to a unique value x' found in $C_1 \cap C_2$.

The Sudoku problem is solved using alternating projections. First the Sudoku grid is represented with the 9X 9 matrix A(9,9) such that every cell in A is represented as x_{ij} i.e. the entry in the ith row and jth column, where $i, j \in (1,2, \dots 9)$ and $x_{ij} \in (1,2, \dots 9)$ and the grid must satisfy the following constraints, which are expressed in mathematical notation:

$C_1 := \forall\ x_{ij},\ x_{im}\ \in A\ \wedge\ j \neq m \iff x_{ij} \neq x_{im}$ i.e. Every row has unique entries from the integers (1,2, …, 9)

$C_2 := \forall\ x_{ij},\ x_{pj}\ \in A\ \wedge\ i \neq p \iff x_{ij} \neq x_{pj}$ i.e. Every column has unique entries from the integers (1,2, …, 9)

$C_3 := \forall\ x_{ij},\ x_{mn}\ \in A \mid \left[\frac{i-1}{3}\right] = \left[\frac{m-1}{3}\right] \wedge \left[\frac{j-1}{3}\right] = \left[\frac{n-1}{3}\right],\ \boldsymbol{i \neq m, j \neq n \iff x_{ij} \neq x_{mn}}$ i.e. Every sub grid has unique entries from the integers (1,2, …, 9) (note: the notation[x/p] returns the quotient of x divided by p with the decimals truncated. e.g. 2.6667 is truncated to 2)

Next, the Sudoku problem is projected from A(9,9) in 2 dimension, onto the 3 dimensional cube B(9,9,9) where for every element $y_{ijk} \in$ B, $y_{ijk} \in \{0,1\}, i, j, k \in (1,2, \dots 9)$.

Thus for every $x_{ij} \in A$, if $x_{ij} = l \in (1,2, \dots 9) \iff y_{ijl} = 1$, $y_{ijk} = 0\ \forall\ k \neq l$. And the constraints are redefined as follows using mathematical notations:

i) $C_1 := \forall\, y_{ijk} \in \mathrm{B},\ y_{imk} = 1 \iff y_{ijk} = 0\ \forall\, j \neq m$
$Every\ column\ vector\ is\ a\ unit\ vector$

ii) $C_2 := \forall\, y_{ijk} \in \mathrm{B},\ y_{njk} = 1 \iff y_{ijk} = 0\ \forall\, i \neq n$
$Every\ row\ vector\ is\ a\ unit\ vector$

$C_3 := \forall\, y_{ijk} \in \mathrm{B},\ y_{ijp} = 1 \iff y_{mnk} = 0\ \forall\, k = p\ \wedge\ m, n \in (1,2,\dots 9)\ |\ \left\lceil\frac{i-1}{3}\right\rceil = \left\lceil\frac{m-1}{3}\right\rceil \wedge \left\lceil\frac{j-1}{3}\right\rceil = \left\lceil\frac{n-1}{3}\right\rceil,\ \boldsymbol{i} \neq m, j \neq n$

$Every\ subgrid\ of\ each\ elevation\ has\ a\ magnitude\ of\ 1$

iii) $C_4 := \forall\, y_{ijk} \in \mathrm{B},\ y_{ijl} = 1 \iff y_{ijk} = 0\ \forall\, j \neq m\ \wedge\ i, k \in (1,2,\dots 9)$
$Every\ vertical\ vector\ is\ a\ unit\ vector\ conforming\ to\ the\ projection\ of\ A\ onto\ B$

This projection is also known as a Boolean representation [5].

Having this representation of the Sudoku puzzle C_1, C_2, C_3, C_4 are subsets of B(9,9,9), starting from a point y ∈ B, the method of alternating projections is used iteratively to project y to each of the subsets until it converges to a point which is the solution to the Sudoku puzzle in 3 dimension.

However it is important to state that C_1, C_2, C_3, C_4 are non-convex sets and the method of alternating projects is defined only for closed convex sets, but it is observed that if Elser's difference map is applied to the Sudoku problem, it converges quickly[6]**.**

Next, Elser's Difference Map is applied [7]

$$D = P_A(2P_B - Id) + (Id - P_B)$$

Where A and B are nonempty fixed convex and closed sets in X a real Hilbert space, and P_A, P_B are the associated projections onto the sets A and B and Id is the identity operator.

The Fienup's Hybrid Input-Output Algorithm and Douglas-Rachford Algorithm is described by$x_{n+1} = P_A(2P_B - Id) + (Id - P_B)x_n$ $x_n \in X, n = 1{,}2 x_0 \in \mathbb{R}^{9^3}$

Douglas-Rachford and Fienup HIO is used iteratively to produce a sequence $x_0, x_1, x_{2,\dots}$ if the Sudoku puzzle is well posed it converges to a point x_n which is the solution.

In this paper, the given Sudoku puzzle is used as the starting point, each clue or given is projected into $\mathbb{R}^{9^3}$, the empty cells are filled up by randomly selecting an integer from its list of compatible digits (as shown in figure 5), this is also projected into $\mathbb{R}^{9^3}$. With this value for x_0, the Douglas-Rachford and Fienup HIO are used iteratively until an integer N is attained such that

$x_N = x_{N+1} = x_{N+2} = \cdots$
Then the solution is x_N
It is important to state that in applying this iteration, Schaad's [6] php implementation was utilised with $x_0 = given\ sudoku\ puzzle.$

3. COMPARING THE THREE SUDOKU PUZZLE SOLVING METHODS

In this research work, the Pencil and paper, backtracking methods of solving Sudoku were coded in php, this code can be viewed on [8]. In the pencil and paper implementation only the naked single and hidden single rules were implemented. These were used along with the php implementation of the method of alternating projections produced by Schaad [6] to compare 40 sudoku puzzles from [9] and [10]. All three methods solve Sudoku problems iteratively. Thus the comparison is based on the number of iterations taken to solve each problem at different levels of difficulty. As shown in table 1, each puzzle is uniquely identified with the symbol S_a where a is an integer from 1 to 40. Puzzles with the ratings gentle, moderate, tough, diabolical and extreme were obtained from [www.sudoku.org/uk] along with the ratings. Those rated very difficult were obtained from [www.aisudoku.com]. Each puzzle was tested using the backtracking implementation[8] to ensure it had a unique solution. Table 1 gives a summary of the results. The puzzles which took infinite (∞) number of steps were unsolvable using that method.

Table 1: shows the number of iterations taken to solve each of the **40 sample puzzles for each of the three methods of solving Sudoku puzzles**

Puzzle	Difficulty Rating(S_1 to S_{31} from [9], S_{32} to S_{40} from [10])	No. of givens in clue or puzzle	Pencil and paper method (number of Iterations taken to solve the problem)	Backtracking (number of iterations taken to solve the problem)	Alternating projections (number of iterations taken to solve the problem)
S_1	Gentle	27	3	68630	124
S_2	Gentle	28	5	14288	330
S_3	Gentle	27	2	94444	170
S_4	Gentle	26	1	14934	285
S_5	Gentle	27	1	118785	85
S_6	Moderate	28	2	29273	162
S_7	Moderate	27	4	11680	116
S_8	Moderate	25	3	48923	90

S_9	Moderate	26	2	11016	256
S_{10}	Moderate	28	2	31394	39
S_{11}	Moderate	25	7	5499	89
S_{12}	Moderate	25	3	276829	387
S_{13}	Tough	30	2	369552	182
S_{14}	Tough	30	2	5661	311
S_{15}	Tough	30	2	130715	289
S_{16}	Tough	25	5	143246	337
S_{17}	Diabolical	25	5	20742	383
S_{18}	Diabolical	25	∞	27965	227
S_{19}	Diabolical	24	∞	478755	295
S_{20}	Diabolical	28	∞	3093	286
S_{21}	Diabolical	25	∞	240445	327
S_{22}	Diabolical	26	∞	21084	675
S_{23}	Diabolical	24	∞	42145	115
S_{24}	Extreme	27	∞	15221	667
S_{25}	Extreme	26	∞	40445	282
S_{26}	Extreme	29	∞	11275	471
S_{27}	Extreme	24	∞	372229	4009
S_{28}	Extreme	28	∞	1656	273
S_{29}	Extreme	26	∞	4150	324
S_{30}	Extreme	27	∞	26841	389
S_{31}	Extreme	31	∞	48785	512
S_{32}	Most Difficult (AI Escargot)	23	∞	31812	6627
S_{33}	Most Difficult (AI Killer)	21	∞	29524	174
S_{34}	Most Difficult (AI Lucky Diamond)	24	∞	46600	276
S_{35}	Most Difficult (AI Worm Hole)	22	∞	646742	4998
S_{36}	Most Difficult (AI Labyrinth)	24	∞	335156	12025
S_{37}	Most Difficult (AI Circles)	25	∞	399937	2410
S_{38}	Most	23	∞	434493	3252

	Difficult (AI Squadron)				
S$_{39}$	Most Difficult (AI Tweezers)	24	∞	240371	688
S$_{40}$	Most Difficult (AI Broken Brick)	23	∞	21384	1598

4. A DISCUSSION OF THE RESULT

Considering the results summarized in table 1, the pencil and paper (PnP) method solved the gentle, moderate, tough puzzles and 1 diabolical rated puzzle with the least number of iterations (between 1 to 10 number of iterations) compared to the other 2 methods, but was unable to solve the rest of the puzzles. The method of alternating projections (AP) solved all 40 puzzles. The gentle, moderate, tough and diabolical puzzles, in number of iterations falling between 30 and 700 iterations. The extreme and AI puzzles were solved in iterations falling between 150 and 12050. The backtracking (BT) method solved the puzzles in the most number of iterations. All 40 puzzles were solved in iterations falling between 1650 and 650,000 iterations.

Below are graphs produced by plotting the 40 puzzles against the number of iterations taken to solve each puzzle using the given methods. The PnP graph is shown in fig 4 with the line graph halting at S17, the average number of iterations taken to solve puzzles S1 to S17 is 3 and the standard deviation 1.658. The graph of the method of AP (fig5) interestingly, crawls close to the x-axis from S1 to S26 and from S27 to S40 makes some major leaps at S26, S32 and S36. It is also worthy of note that the puzzles rated gentle, moderate, tough and diabolical which lie between S1 to S23 have a mean number of iterations of 242 and a standard deviation of 141, implying that with the given data, most puzzles in that rating (gentle, moderate, tough and diabolical) will be solved using the AP method within 101 to 383 iterations. The mean iterations for puzzles S24 to S40 (Extreme and AI puzzles) with this method is 2293, a huge difference when compared from the average taken for S1 to S23. The overall average of the number of iterations taken to solve all 40 puzzles using the AP method is 1,113. Analyzing the graph of the BT method fig 6, the graph from S1 to S11 has relatively small humps, and seems to be bounded above at 200,000 iterations but from S12 to S40 high peaks are observed. The overall average of the number of iterations taken to solve the 40 puzzles using the BT method is 122,893, a relatively enormous value when compared with the average taken using the AP method. Fig 7 shows the number of iterations taken to solve each puzzle using all three methods. The graphs for the pencil and paper method, in fig 7 lies close to the x-axis and stops at puzzle S17 while the graph for AP lies close to the x-axis and for puzzles S1 to S32 and rises slightly above the x-axis for puzzles S33 to S40. The backtracking method makes huge leaps away from the x-axis with enormous number of iterations for puzzles S13, S19, S27, S35, S37, S38

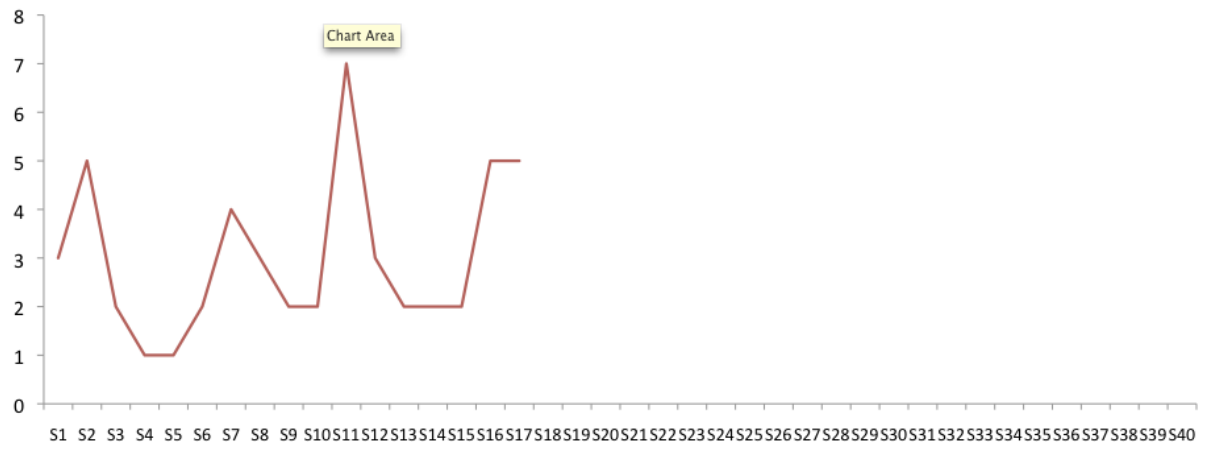

Fig 4: Graph shows the number of iterations taken to solve puzzles s1 to s40 using the Pencil and paper method

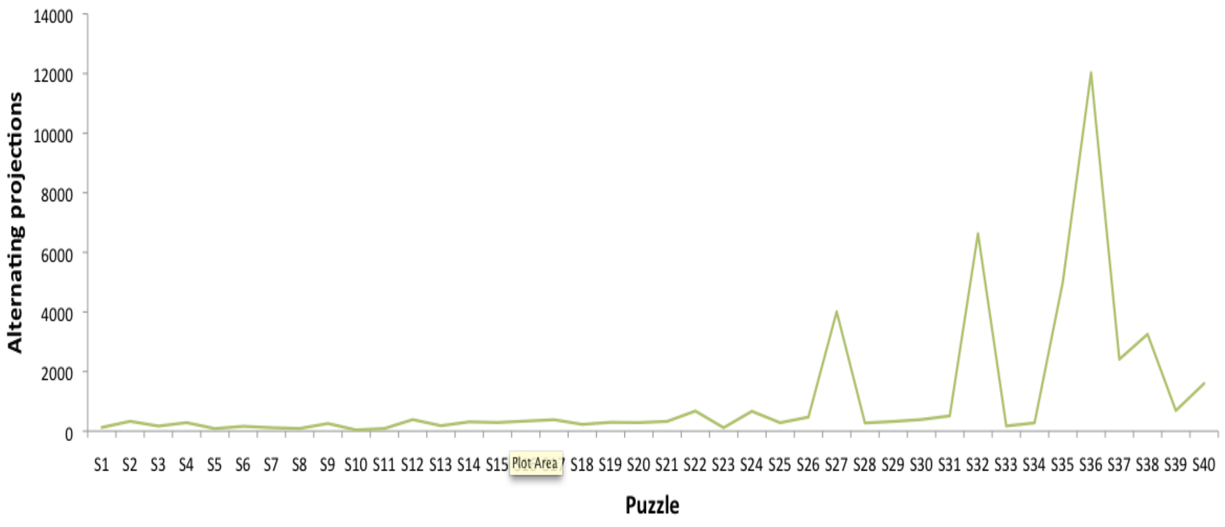

Fig 5: Graph shows the number of iterations taken to solve puzzles s1 to s40 using the method of Alternating projections

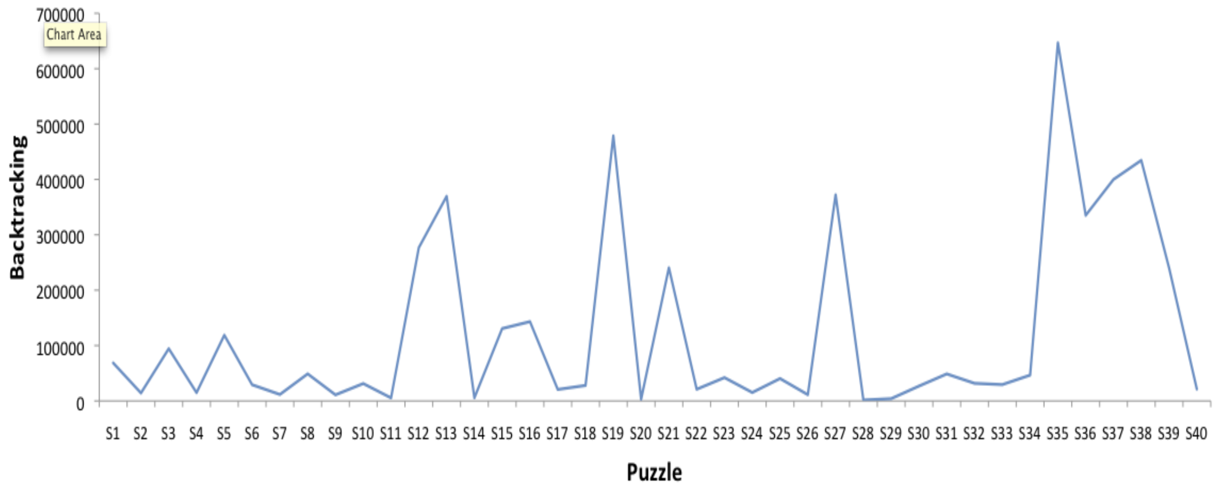

Fig 6: Graph shows the number of iterations taken to solve puzzles s1 to s40 using the Backtracking method

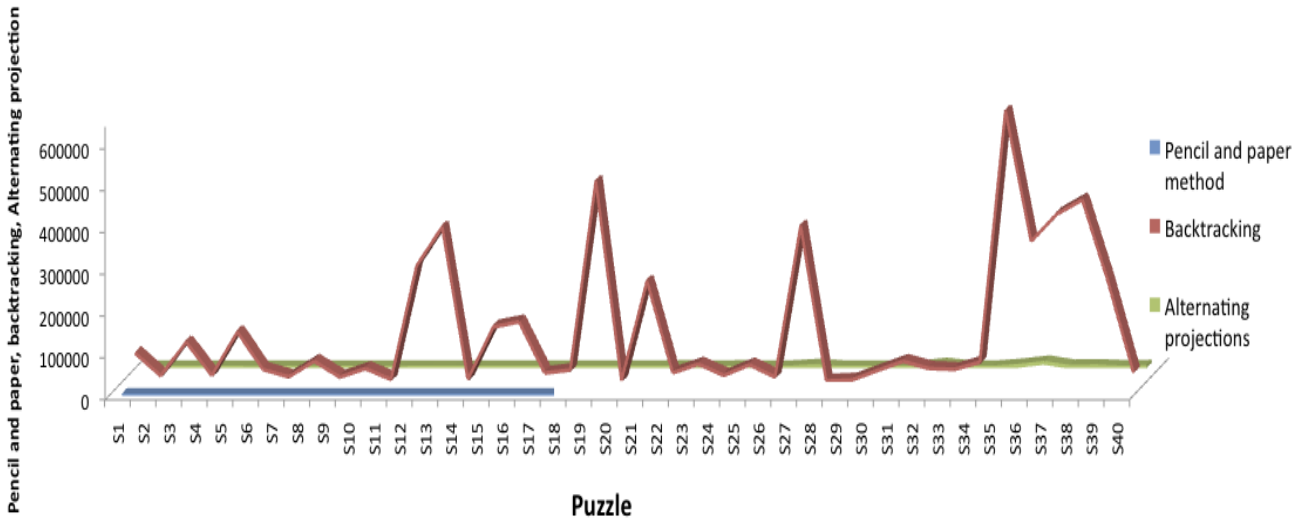

Fig 7: All three methods and the number of iterations taken to solve the puzzle

5. CONCLUSION AND FUTURE WORK

It is clear that for simpler puzzles rated gentle, moderate, tough, the pencil and paper method was the most efficient method in solving those puzzles as it took the least number of iterations, however it proved ineffective for harder puzzles as it was unable to solve them. The method of alternating projections took more iterations in solving the gentle, moderate and tough puzzles than did the pencil and paper method, but solved the harder puzzles in finite number of iterations. The backtracking method took the most number of iterations to solve all the puzzles, thus being the least efficient of all three methods. Can a hybrid solver which optimally combines these three methods or at least two of them be produced to provide a guaranteed solution, with the least number of iterations? Is there a relationship between the number of clues or givens with the level of difficulty? These questions are left to be answered in the future.

6. REFERENCES

[1] Shetty, B.R., Rohith, J., Mukund, V., Honwade, R., & Rangaswamy, S. (2009). Steganography Using Sudoku Puzzle. 2009 International Conference on Advances in Recent Technologies in Communication and Computing, 623-626.

[2] Learn-Sudoku.com (2008)https://www.learn-sudoku.com/ [Accessed 20/09/18]

[3] .Chi E. & Lange K. (2012), Techniques for Solving Sudoku Puzzles, arXiv:1203.2295 [math.OC]

[4] .Lange K., Optimization, Springer-Verlag, New York, 2004.

[5] .Ercsey-Ravasz M. & Zolta´n Toroczkai (2010)The Chaos Within Sudoku SCIENTIFIC REPORTS | 2 : 725 | DOI: 10.1038/srep00725

[6] Schaad J. (2010)Modeling the 8-Queens Problem and Sudoku using an Algorithm based on Projections onto Nonconvex Sets, Master's thesis, The University of British Columbia. Pages 23, 93-95Available from:

[7] Elser V., Rankenburg I., & Thibault P. (2007)Searching with iterated maps. Proc. Natl. Acad. Sci. USA,104(2):418{423, 2007.Available from: http://dx.doi.org/10.1073/pnas.0606359104. pages 1, 26Google(2018)https://drive.google.com/open?id=1uKvWCO2j_8SHzTapelRICRN3567GBtWk

[8] www.sudoku.org.uk (2018)http://www.sudoku.org.uk/Daily.asp[Accessed:20/11/2018].

[9] www.aisudoku.comhttp://www.aisudoku.com/index_en.html[Accessed:20/11/2018]