DOI: http://dx.doi.org/10.26483/ijarcs.v9i3.6096

ISSN No. 0976-5697

Volume 9, No. 3, May-June 2018

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

POLYNOMIAL 3-SAT REDUCTION OF SUDOKU PUZZLE

Deepika Rai
Assistant Professor
School of Computer Science and IT, DAVV
Indore (M.P), India

N.S. Chaudhari
Director and Professor
VNIT Nagpur and IIT Indore
Indore (M.P), India

Maya Ingle
Professor and Sr. System Analyst
SCSIT, DAVV
Indore (M.P), India

Abstract: 3-Satisfiability (3-SAT) reduction has always been remarkable asset in proving the NP-Completeness of other problems. 3-SAT problem is an NP-Complete problem used as a starting point to prove the hardness of other problems. Therefore, every NP-Complete problem can be reduced into 3-SAT that can be solved by a SAT solver. In this perspective, determining 3-SAT reduction from Sudoku Puzzle of size (n x n) is very helpful to obtain the solution of Sudoku Puzzle using SAT solver. Thus, we have obtained polynomial 3-SAT reduction of Sudoku Puzzle (n x n) as well as total number of 3-SAT clauses and new variables generated in 3-SAT reduction are 4 $[n^4 - 2n^2 + m]$ and 2 $[n^2\{n^2 + n - 6\} + m]$ respectively.

Keywords*:* 3-SAT, Sudoku Puzzle, NP-Complete Problem, SAT solver, Polynomial Time

I.INTRODUCTION

Satisfiability (SAT) is a growing area of research in computational complexity and theorem proving. It has been extensively used to prove the NP-Completeness of other problems. In this view, 3-SAT problem is first viable restriction of SAT, to prove the hardness of other problems. Expedition for development of efficient SAT solvers generates significant growth in this field. As a result, it turned to the fact that every NP-Complete problem can be reduced into 3-SAT, to obtain the solution of that problem using SAT solver [1][2]. Additionally, Sudoku Puzzle of size (n x n) is an NP-Complete problem that has been received noteworthy contribution in research area of computer science. Various applications of solving Sudoku Puzzle are witnessed in the fields of steganography, secret image sharing, encrypting SMS, digital watermarking, image authentication, image encryption and many others [3]. There exist various techniques for solving Sudoku Puzzles such as backtracking algorithms, stochastic search techniques, integer linear programming, genetic algorithms etc. Most of these existing techniques are primarily guess-based heuristic that require exponential time for large Sudoku Puzzle [4][5][6]. Sudoku Puzzle (n x n) can be reduced into 3-SAT in order to find solution using a SAT solver. There exist two SAT encodings for Sudoku Puzzle namely; extended encoding and minimal encoding. It has been noticed that extended encoding is better than minimal encoding as it has some redundant clauses that perform well in terms of resolution techniques [7]. On the other hand, many research efforts have been made on polynomial 3-SAT reduction for variety of NP-Complete problems such as graph k-colorability, determining vertex cover and independent set for a given graph, channel assignment problem in cellular network etc. [8][9][10]. However, there is a scope of 3-SAT reduction of Sudoku Puzzle (n x n) in polynomial time.

Polynomial time reduction from Sudoku Puzzle to 3-SAT formula is helpful to detect the satisfiability of the generated formula as well as to obtain the feasible solution. We discuss the basic details of 3-SAT and Sudoku Puzzle in Section 2. Interest in Sudoku Puzzle is expanding for its NP-Completeness as a result it is represented as 3-SAT to solve the puzzle. In Section 3, formulation of polynomial 3-SAT reduction of Sudoku Puzzle (n x n) is depicted. In Section 4, we proposed an algorithm for 3-SAT reduction of Sudoku Puzzle that generates 3-SAT clause in DIMACS form. Afterwards, we summarize the experimental results for Sudoku Puzzles of different size in Section 5. We have obtained satisfiable solution with the help of online Minisat solver. Finally, we conclude our work with suggesting future work in Section 6.

II. BACKGROUND

In this section, we have explored the background details of Sudoku Puzzle and 3-SAT problem.

A. 3-Satisfiability Problem

A 3-Satisfiability (3-SAT) problem i.e. the boolean satisfiability problem aims to check the given propositional formula is satisfiable or not. It is restricted form of a SAT, where each clause contains exactly 3 distinct literals. It consists of conjunction of clauses, where each clause consists of disjunction of exactly 3 distinct literals [11]. It is a special case of SAT problem that has been one of the Karp's 21 NP- complete problems as well as it is used as a starting point to prove the NP-Completeness of other problems [12].

Mathematically, let F be a 3-SAT formula consists of n clauses C_1 C_2, C_3, …, C_n and m literals l_1, l_2, l_3, …, l_m. An example of 3-SAT formula F, consists of 3 clauses and 4 literals is given by- $F = C_1 \wedge C_2 \wedge C_3$, where $C_1 = (a_1 \vee \neg a_2 \vee a_4)$, $C_2 = (\neg a_1 \vee a_3 \vee a_4)$, $C_3 = (a_2 \vee \neg a_4 \vee a_1)$ and, $l = \{a_1, a_2, a_3, a_4\}$ is the set of literals. This formula is to be satisfiable if it is true by assigning a suitable logical value to its literals.

B. Sudoku Puzzle (n x *n)*

Sudoku Puzzle of size (n x n) is a number placement puzzle that consists of n^2 cells and partially completed with digits 1 to n, where n is a perfect square. The aim of the solver is to complete the remaining cells with digits 1 to n in such a manner that each row, column, and ($\sqrt{n}$ x $\sqrt{n}$) subcell contains the digits from 1 to n exactly once [13][14]. For example, Fig. 1 depicts a Sudoku Puzzle (9 x 9) consists of 81 cells and partially completed with digits 1 to 9. On the other hand, Fig. 2 shows the solution of this puzzle that consists of 81 cells which are completed in such a way that each row, column, and subcells (3 x 3) contains the digits from 1 to 9 exactly once.

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

Fig. 1: Sudoku Puzzle (9 x 9)

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

Fig. 2: Solution of Sudoku Puzzle (9 x 9)

III. POLYNOMIAL 3-SAT REDUCTION OF SUDOKU PUZZLE

Sudoku Puzzle (n x n) is an NP-Complete problem; accordingly, it is represented as a SAT problem. We have used extended SAT encoding of Sudoku Puzzle (9 x 9). We generalize it for any value of n and generated the clauses in DIMACS format that stored in a file (say, sudo_SAT.txt). All the generated clauses for Sudoku Puzzle (n x n) are of length k (k = 1, 2, and n). As a result, SAT encoding of Sudoku Puzzle (n x n) consists of $4n^2$ clauses of length n, ($4n^2$ * ${}^nC_2$) clauses of length 2 and m clauses of length 1, where m is the number of preassigned numbers in given Sudoku Puzzle. Therefore, total number of clauses generated for Sudoku Puzzle (n x n) is given by:

$$|N| = 4n^2 + 4n^2 ({}^nC_2) + m \qquad \dots (1)$$

These generated clauses are converted to 3-SAT clauses using well-defined non-recursive method. To highlight this method, we consider C_1, C_2 and C_n as the clauses of length 1, 2 and n respectively. These clauses consist of literals $a_1, a_2, a_3, \dots, a_{n-1}, a_n$. By using some new variables $x_1, x_2, x_3, \dots, x_{n-3}$, these clauses are converted into 3-SAT clauses as C_x, C_y and C_z corresponding to C_1, C_2 and C_n.

$C_1 = (a_1)$, where k = 1

$$C_x = (a_1 \vee x_1 \vee x_2) \wedge (a_1 \vee \neg x_1 \vee x_2) \wedge (a_1 \vee x_1 \vee \neg x_2) \wedge (a_1 \vee \neg x_1 \vee \neg x_2) \qquad \dots (2)$$

$C_2 = (a_1 \vee a_2)$, where k = 2

$$C_y = (a_1 \vee a_2 \vee x_1) \wedge (a_1 \vee a_2 \vee \neg x_1) \qquad \dots (3)$$

$C_n = (a_1 \vee a_2 \vee a_3 \vee \dots \vee a_n)$, where k = n

$$C_z = (a_1 \vee a_2 \vee x_1) \wedge (\neg x_1 \vee a_3 \vee x_2) \wedge (\neg x_2 \vee a_4 \vee x_3) \wedge \dots \wedge (\neg x_{n-3} \vee a_{n-1} \vee a_n) \qquad \dots (4)$$

Equation (2), (3) and (4), generated all clauses of length 3. Thus, number of clauses and new variables required to transform a clause of length k (k = 1, 2 and n) into the clauses of length 3, are given by Table 1.

Table 1: Number of Clauses and Variables for a SAT Clause to 3-SAT Clauses

No. of Literals in a Clause (k)	*Total 3-SAT Clauses*	*Required New Variables*
1	4	2
2	2	1
n	n – 2	n – 3

Now, apply equation (2), (3) and (4) on generated DIMACS form (sudo_SAT.txt) of Sudoku Puzzle of size (n x n). By using equation (1) and table 1, we get the total number of 3-SAT clauses (say |C|) and new variables (say |V|) used in 3-SAT reduction of Sudoku Puzzle:

$$|C| = (n-2)(4n^2) + 2(4n^2)({}^nC_2) + 4m$$
$$\text{i.e. } |C| = 4[n^4 - 2n^2 + m] \qquad \dots (5)$$

and,

$$|V| = (n-3)(4n^2) + (1)(4n^2)({}^nC_2) + 2m$$
$$\text{i.e. } |V| = 2[n^2\{n^2 + n - 6\} + m] \qquad \dots (6)$$

Hence, it is clear from above formulation, we have obtained the polynomial 3-SAT reduction for Sudoku Puzzle of size (n x n). In next section, we have presented an algorithmic approach for 3-SAT reduction of Sudoku Puzzle.

IV. ALGORITHMIC APPROACH FOR 3-SAT REDUCTION

In this section, algorithm SUDO3SAT for 3-SAT reduction of Sudoku Puzzle (n x n) is presented.

Algorithm: SUDO3SAT

```
main ( )
{
STEP 1:  /*Input sudo_SAT.txt*/
            READ (sudo_SAT.txt);

STEP 2:  /* Initialize the file for storing 3-SAT clauses */
          Sudo_3SAT.txt = NULL;

STEP 3:  /* Count number of literals (say, k) of a clause,
          use some new variables {$x_1, x_2, x_3, \dots, x_{n-3}$} and
          develop the new 3-SAT clauses (say, New_clause)
          */

      if (k = 1)
          New_clause = $(a_1 \vee x_1 \vee x_2) \wedge (a_1 \vee \neg x_1 \vee x_2)$
                   $\wedge (a_1 \vee x_1 \vee \neg x_2) \wedge (a_1 \vee \neg x_1 \vee \neg x_2)$;
          Sudo_3SAT.txt  = New_clause;
    else
      if (k = 2)
          New_clause = $(a_1 \vee a_2 \vee x_1) \wedge (a_1 \vee a_2 \vee \neg x_1)$;
          Sudo_3SAT.txt  = New_clause;
      else
         if (k = 3)
            Sudo_3SAT.txt = Sudo_SAT;
         else
            if (k = n)
              New_clause =  $(a_1 \vee a_2 \vee x_1) \wedge$
                $(\neg x_1 \vee a_3 \vee x_2) \wedge (\neg x_2 \vee a_4 \vee x_3) \wedge$
                $\dots \wedge (\neg x_{n-3} \vee a_{n-1} \vee a_n)$;
              Sudo_3SAT.txt = New_clause;

STEP 4: /* Generate total clauses in Sudo_3SAT.txt */
         Repeat Step 3 until EOF;
 }  /* end of main ( ) */
```

V. RESULTS AND DISCUSSION

We have implemented the formulation of 3-SAT on various Sudoku Puzzle and obtained 3-SAT clauses in DIMACS format. 3-SAT clauses are passed to the online Minisat solver that produces the satisfiable values for this clauses. These satisfiable values provides the appropriate value for each cell of Sudoku Puzzle. Table 2 depicts the results of our implementation on different Sudoku Puzzles. The following are important results associated with 3-SAT reduction of Sudoku Puzzles (n x n):

- Total number of 3-SAT clauses for any Sudoku puzzle (n x n): $|C| = 4\,[n^4 - 2n^2 + m]$, where m is the number of preassigned numbers in given Sudoku Puzzle.
- Total number of new variables used for 3-SAT reduction of Sudoku puzzle (n x n): $|V| = 2\,[n^2\{n^2 + n - 6\} + m]$.
- Polynomial 3-SAT reduction of Sudoku puzzle (n x n) is obtained.
- On solving 3-SAT formula using SAT solver, n^2 variables (corresponding to each cell) are satisfiable for Sudoku puzzle of size (n x n).

Table 2: Execution of 3-SAT Formula for Various Sudoku Puzzle

Size of Puzzle	*No.of Preassigned Values (m)*	*Total SAT Clauses*	*Total 3SAT Clauses*	*Total New Variables in 3SAT*	*Total Satisfiable Values*
4 X 4	6	454	920	460	16
	8	456	928	464	
	9	457	932	466	
9 X 9	22	12010	25684	13652	81
	32	12020	25724	13672	
	35	12023	25736	13678	
16 X 16	90	123994	260456	136372	256
	101	124005	260500	136394	
	110	124014	260536	136412	

VI. CONCLUSION

Proposed approach transforms the Sudoku Puzzle (n x n) into 3-SAT formula for any value of n. Online Minisat solver provides the satisfiable values for this 3-SAT formula. This is helpful to obtain the feasible solution of Sudoku Puzzle using SAT solver. We conclude that our formulation for 3-SAT reduction of Sudoku Puzzle (n x n) is polynomial. We have obtained the total number of 3-SAT clauses and new variables required for this transformation. A direction for future work is to optimize the obtained number of 3-SAT clauses and new variables used in 3-SAT clauses.

VII. REFERENCES

[1] Marques Silva J., Malik S., “Propositional SAT Solving”, In: Clarke E., Henzinger T., Veith H., Bloem R. (eds) Handbook of Model Checking, Springer, Cham, pp. 247-275, 2018.

[2] Claessen, Koen, N. Een, M. Sheeran, N. Sorensson, “SAT-Solving in Practice”, 9th IEEE International Conference on Discrete Event Systems, pp. 61-67, 2008.

[3] A.K. Maji, R.K. Pal, “Sudoku Solver using Minigrid Based Backtracking”, IEEE International Conference on Advance Computing (IACC), pp. 36-44, 2014.

[4] Perez, Meir, TshilidziMarwala, “Stochastic Optimization Approaches for Solving Sudoku”, Archives of Neural and Evolutionary Computing, Cornell University Library, 2008.

[5] Deng, Xiu Qin, Yong Da Li., “A Novel Hybrid Genetic Algorithm for Solving Sudoku Puzzles”, Springer Journal of Optimization Letters, Vol. 7, No. 2, pp. 241-257, 2013.

[6] Bartlett, Andrew C., Amy N. Langville, “An Integer Programming Model for the Sudoku Problem”, Journal of

Online Mathematics and its Applications, Vol. 8, Article ID 1798, 2008.

[7] Ines Lynce, Joel Ouaknine, "Sudoku as a SAT Problem", Proceedings of 9th International Symposium on Artificial Intelligence and Mathematics, 2006.

[8] Prakash C.Sharma and Narendra S. Chaudhari, "A Graph Coloring Approach for Channel Assignment in Cellular Network via Propositional Satisfiability", International Conference on Emerging Trends in Networks and Computer Communications (ETNCC) at Udaipur, pp. 23-26, 2011.

[9] N.Dahale, N.S.Chaudhari, M. Ingle, "Determining Vertex Cover Using Polynomial Encoding of 3sat", VNSGU Journal of Science and Technology, Vol. 4, No. 1, pp. 197-204, 2015.

[10] Prakash C. Sharma and Narendra S. Chaudhari, "Polynomial 3-SAT Encoding for K-Colorability of Graph", Evolution in Networks and Computer Communications-A Special Issue from IJCA, Article 4, No. 1, pp. 19-24, 2011.

[11] Tovey, Craig A., "A simplified NP-complete satisfiability problem",Discrete Applied Mathematics, Elsevier, Vol. 8, No. 1,pp. 85-89, 1984.

[12] N. Chandrasekaran, K.L.P Mishra, "Theory of Computer Science", PHI Learning publishing, 3rd edition, 2011.

[13] Felgenhauer, Bertram, Frazer Jarvis, "Mathematics of Sudoku-I", Mathematical Spectrum, Vol. 39, No. 1, pp. 15-22, 2006.

[14] G. Kendall, A.J. Parkes, K. Spoerer, "A Survey of NP-Complete Puzzles", International Computer Games Association (ICGA) Journal, Vol.31, No. 1, pp. 13-34, 2008.