*IOSR Journal of Computer Engineering (IOSR-JCE)*
*e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 16, Issue 5, Ver. VI (Sep-Oct. 2014), PP 69-77*
*www.iosrjournals.org*

# A Comparative Study On The Performance CharacteristicsOf Sudoku Solving Algorithms

[1]Sankhadeep Chatterjee, [2]Saubhik Paladhi, [3]Raktim Chakraborty
[1, 2, 3]*Department of Computer Science & Engineering, Hooghly Engineering & Technology College*
*Vivekananda Road, Pipulpati, Hooghly-712103, West Bengal, India*

***Abstract:****Solving Sudoku, a NP-Complete combinatorial optimization problem has been carried out using the optimized Graph Referencing Algorithm (GRA),Genetic Algorithm (GA), Simulated Annealing (SA), Harmony Search (HS) and Brute Force algorithm. The present study is primarily aimed at finding out the fastest algorithm in terms of least time consumption in solving Sudoku. The performance characteristics of algorithms of interest are studied by deploying randomly selected puzzles with different difficulty levels. The comparative performance characteristics study reveals the superiority of the Graph Referencing algorithm over the other algorithms in taking least possible time to solve Sudoku.*

***Keywords:*** *Genetic Algorithm,Graph Referencing algorithm, Harmony Search, Simulated Annealing, Sudoku.*

## I. Introduction

Sudoku [1] puzzle, a NP – Complete constraint satisfaction problem [2] is a very well-known combinatorial optimization problem and has received considerable popularity over the past decade. Various exact implicit enumeration, heuristic and meta-heuristic[3] 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 problem. Research has revealed that it may be too costly in terms of time – complexity to solve some problems. Backtracking, on the other hand, has been found to be more promising in solving Sudoku problems [4] 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 the authors [5]. Genetic [6] algorithms (GA) have been revealed to be an effective approach in successfully handling Sudoku as a multi-objective optimization problem [7]. GA has further been studied and major improvements in accordance with Sudoku solving have been reported in literature [8]. A novel meta-heuristic algorithm for continuous optimization problem has been put forwarded by Lee et al. [9]. The optimization algorithm describes a new Harmony Search (HS) meta-heuristic algorithm based approach which is conceptualized using the musical process of searching of a perfect state of harmony. The HS algorithm has been reported to be a powerful search and optimization technique that is expected to yield better solution to combinatorial optimization problems. A successful adaptation of HS algorithm to solve Sudoku using meta-heuristic search technique has been reported by Geem[10]. The efficient modification over HS algorithm, implemented by Mandal et al. [11] is found to be more effective in terms of number of iterations to solve a given Sudoku. Another algorithm based on Simulated Annealing (SA) has been proposed by Henrik et al. [12].A Parallel Simulated Annealing based Sudoku solving strategy has been reported in literature [13]. Liu et al. [14] has proposed another GA based algorithm. Mantere et al. [15] studied the cultural algorithm based algorithms to solveand analyse Sudoku. An improved version of graph matching algorithm was developed by Cordella et al. [16] while Eppstein [17] has shown how graph can be used to solve Sudoku. Rodrigues Pereira et al.[18] applied constraint programming to solve Sudoku. Zhang [19] proposed an AutoCAD based Sudoku solving mechanism. Zhao et al. [20] studied Sudoku generating algorithms at a greater extent and commented on the difficulties which may be faced by Sudoku solving algorithms. Nicolau et al. [21] offered an evolutionary approach to tackle Sudoku problems. The wide involvement and research through the last decade insisted the authors to study the performance characteristics of major Sudoku solving algorithms which are primarily based on well-studied algorithmic approaches. The performance characteristics of different algorithms are compared by analysing the solution time taken to solve the same set of problems. The problems are chosen with different level of difficulties in order to provide a broader spectrum of analysis of performances. The various algorithm techniques considered by the authors are discussed in brief here:

### A. Brute-force method:-

This is the easiest approach as it avoids computational complexities. This algorithm visits all the empty cells in a specific order and fills it with a digit from available choices. If no choice is available for a cell then it backtracks and changes the digit of previous cell. Thus the algorithm continues until all the cells are filled with appropriate digits.

**B. Simulated Annealing:-**

Simulated annealing is a probabilistic optimization method. Sudoku is considered to be solved when all the cells are filled with appropriate choice. By simulated annealing a potential solution is created by filling empty cells with random values. When the entire board will be filled, the number of errors that has been counted and value of two cells of these boxes are swapped. The updated result is considered better if it contains fewer errors than the previous one and in this case the updated result becomes starting point of future iterations.

**C. Genetic Algorithm:-**

Genetic Algorithms are search methods which uses Principle of Natural Selection and Genetics. Good solution and bad solution should be distinguished by measuring the relative fitness of the solutions. Choosing appropriate population size, on which the evolution laws will act, is important as it will give more accurate result. To solve Sudoku all elements are taken within one array and the whole array is divided into couple of sub-arrays. Number of sub-arrays is equal to the number of sub-boards (e.g. 9 in case of 9×9 regular Sudoku). Crossover and mutation takes place over sub-arrays to reach the solution of the Sudoku. Another reference array can be used to avoid unsupported evolutions as initially the given elements cannot participate in Mutation.

**D. Harmony Seach Algorithm:-**

HS Algorithm originally came from the similarity between the music improvisation and optimization process[10].A natural musical performance process to find better state of harmony during musical composition has inspired Harmony search (HS) algorithm to a greater extent. The process of finding musically pleasing harmony can be imitated via seeking for a global solution as determined by an objective function. In music inventiveness, each player sounds any pitch within the possible range, together making one harmony vector. One variable's memory stores an experience if all the pitches make a good solution thereby increasing the possibility of getting a better solution next time. Several optimization operators, such as the harmony memory, the harmony memory considering rate, the harmony memory size (number of solution vectors in harmony memory), and the pitch adjusting rate have been inculcated in HS algorithm. In the HS algorithm, the harmony memory stores the feasible vectors, which are all in the feasible space. The harmony memory size determines how many vector to be stored. Generation of a new vector is done by randomly selecting the components of different vectors in the harmony memory.

**E. Graph Referencing Algorithm:-**

Graph Referencing method has been implemented over the existing backtracking algorithm to tackle combinatorial problems with greater ease. The optimized Graph Referencing method takes an important role in pruning, some unsuccessful searches over the search space, which is inevitable to improve the performance in terms of time complexity. The Reference Graph is considered by the algorithm in determining unsuccessful searches in any branch before it is explored. The reduction in time complexity is taken care of by the Reference Graph which significantly reduces the number of unsuccessful searches in the solution search space.

## II. Experimental Methodology

The experimental methodology takes an important role in deciding the fastest algorithm in terms of least time consumption for yielding solution to Sudoku. The algorithms (Brute Force, GA, SA, GRA, HS) discussed in the literature are implemented and test run in an IBM compatible PC with Dual Core Intel processor. The algorithms are fed with a set of 30 randomly chosen Sudoku puzzles from a collection of easy, medium and hard level puzzles [7]. The time taken by each algorithm to solve each puzzle is observed and recorded. The design of such testing environment is set up as the algorithms under consideration have both heuristic and non-heuristic techniques where theoretical analysis only would not serve the purpose of exact performance analysisof these algorithms. The set used to test the algorithms are found to be a good mix of all types of puzzles thereby indicating a good set to judge the ability of algorithms.

## III. Result & Discussion

The algorithms mentioned earlier have been tested using a randomly selected set of problems. The analysis has been done with a set of 30 randomly selected problems and solving them using the algorithms to judge their ingenuity in context to Sudoku problems solution techniques. Initially the set of problems has been sorted in ascending order of difficulty levels starting from easy to hard and are indexed. Next different algorithms are applied as a solution tool for cracking Sudoku problems. A study has been carried out to find out the solution time and based upon the least consumption in solution time the fastest algorithm is judged. Table 1 depicts the time taken by different algorithms to solve the same set of randomly selected puzzles.

Figure 1 depicts a plot of time vs. index of selected problems solved by Brute Force algorithm. The plot reveals that up to a moderate level of difficulty it consumes more or less same time though the time consumption shows a steep jump at index '21' and it follows a similar nature as the difficulty of solving the problem increases. Figure 2 depicts the same for Simulated Annealing based algorithm. The plot reveals almost flat increase in time consumption as the difficulty increases though most of the problems take more time than the previous algorithm. Figure 3 shows the plot for GA and reflects higher time consumption than previous algorithms. Figure 4 exhibits the same plot for Harmony search based algorithm to solve Sudoku problems. The study of Figure 4 reveals that the algorithm seems to be the most expensive among all that has been studied as its minimum time consumption at index '1' ( 3 s) is more than most of the problems solved by other algorithms even with higher level difficulty. The above comparative study establishes the ingenuity of Brute Force algorithm among the above mentioned algorithms.

Figure 5 depicts the plot of time vs. index of selected problem set which has been solved by GRA. The plot reflects that the time consumption is quite lesser than all the previously studied algorithms and even better than Brute Force algorithm in most of the cases. Different algorithm takes different amount of time to solve a particular problem. All together 30 randomly chosen problems have been used to check the time consumption of different algorithms. The average time taken to solve a problem among the set of problems is calculated and the average of those averages has been taken as quantity of time (Standard Time) to solve a random problem of this set by choosing an algorithm randomly. The Standard time is used as a parameter to judge the ability of the algorithms to solve a puzzle within minimum time.

Figure 6 depicts the plot of time vs. puzzle index where Standard time is always higher than the time taken by the Brute Force algorithm to solve any problem of the set thereby indicating an overall good performance. Similarly Figure 7 depicts the same for Simulated Annealing. The plot reveals a similar trend as that of the Brute Force algorithm. Figure 8 shows the same plot for Genetic Algorithm. The plot reveals that in most of the cases the algorithm performs well but problem indexed '25', '27', '28', '29', '30' has taken more time than the Standard time consumption indicating a moderate performance. The same analysis for Harmony Search algorithm reveals that most of the problems of the set in this experiment has failed to consume a time lesser than Standard time, which clearly indicates that Sudoku may not be efficiently solved by existing version of HS based algorithm (Figure 9). GRA, on the other hand, has shown remarkable results (Figure 10)in this experiment by taking consistently lesser time than Standard time to solve all the given problems.

Figure 11 depicts an overall comparison of the algorithms and shows that the performance of Brute Force, GA, SA and GRA is almost same up to 21st puzzle. Little variation is observed in GA after 21st puzzle but remaining three puzzles show almost same performance. Figure 12 reveals a more detailed and closer look at the performance characteristics of Brute Force, SA and GRA. The plot indicates that the consistency of Brute Force and GRA is very close but still GRA takes lesser time than Brute Force in solving many puzzles. The ingenuity of GRA is further more revealed by the Standard Deviation chart in Figure 13 which shows how standard deviation of time decreases and finally takes smallest value in case of GRA. Thus it can be concluded that GRA not only takes minimum time to solve problems but reflects consistency in solution time over a wider variation of difficulty level of given problems.

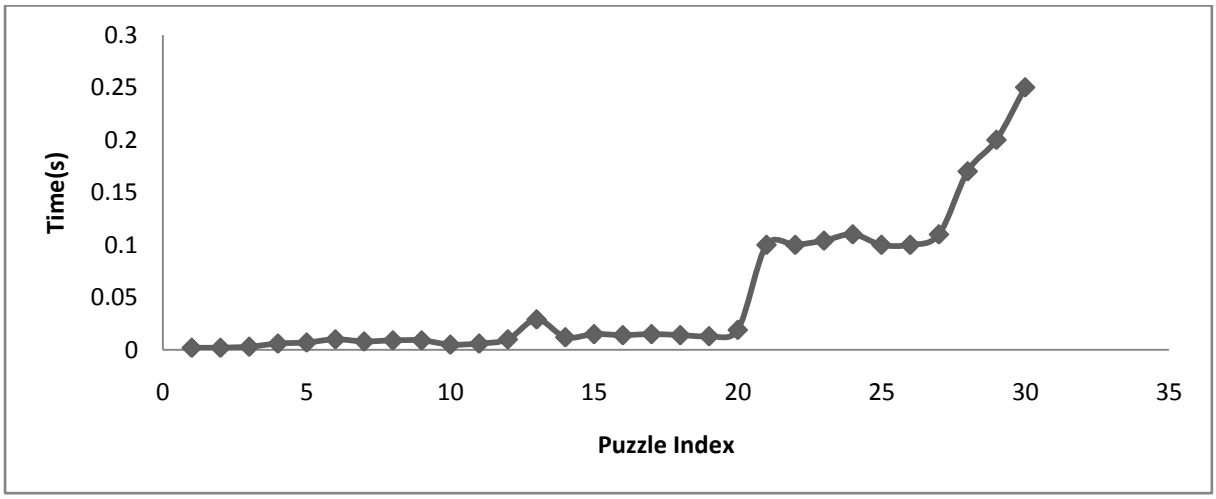


**Figure 1**: Graph depicting solving time of different puzzle by Brute Force method.

Table 1. Time taken by different algorithms to solve the set of randomly selected puzzles

| Index of Puzzle | Time taken to solve Sudoku problems using different algorithms (s) | | | | |
|---|---|---|---|---|---|
| | Brute Force | SA | GRA | GA | HS |
| 1 | 0.002 | 0.09 | 0.002 | 0.9 | 3 |
| 2 | 0.002 | 0.1 | 0.002 | 1.25 | 5.5 |
| 3 | 0.003 | 0.12 | 0.003 | 1.52 | 5.3 |
| 4 | 0.006 | 0.12 | 0.003 | 1 | 4.5 |
| 5 | 0.007 | 0.19 | 0.005 | 1.35 | 8 |
| 6 | 0.01 | 0.2 | 0.007 | 1.9 | 10 |
| 7 | 0.008 | 0.13 | 0.006 | 2.5 | 13 |
| 8 | 0.009 | 0.15 | 0.007 | 2.1 | 8.5 |
| 9 | 0.009 | 0.15 | 0.005 | 3.4 | 9 |
| 10 | 0.005 | 0.3 | 0.006 | 3.1 | 7 |
| 11 | 0.006 | 0.25 | 0.008 | 2.5 | 12 |
| 12 | 0.01 | 0.4 | 0.009 | 2.9 | 18 |
| 13 | 0.029 | 0.35 | 0.009 | 2.6 | 13 |
| 14 | 0.012 | 0.29 | 0.008 | 1.9 | 15 |
| 15 | 0.015 | 0.45 | 0.009 | 2.03 | 20 |
| 16 | 0.014 | 0.5 | 0.01 | 3.2 | 26.9 |
| 17 | 0.015 | 0.72 | 0.009 | 2.8 | 27.6 |
| 18 | 0.014 | 0.68 | 0.01 | 2.54 | 33.7 |
| 19 | 0.013 | 0.7 | 0.011 | 2.8 | 30.5 |
| 20 | 0.019 | 0.69 | 0.011 | 2.9 | 33.7 |
| 21 | 0.1 | 0.8 | 0.009 | 2.1 | 29 |
| 22 | 0.1 | 0.75 | 0.01 | 3.6 | 35 |
| 23 | 0.104 | 0.8 | 0.05 | 4.5 | 32.6 |
| 24 | 0.11 | 0.68 | 0.05 | 5.4 | 46 |
| 25 | 0.1 | 0.85 | 0.06 | 8.6 | 34 |
| 26 | 0.1 | 0.95 | 0.04 | 6.3 | 43.7 |
| 27 | 0.11 | 0.99 | 0.036 | 10.9 | 76.6 |
| 28 | 0.17 | 1.3 | 0.056 | 11.4 | 102.5 |
| 29 | 0.2 | 1 | 0.066 | 9.5 | 88.4 |
| 30 | 0.25 | 1.1 | 0.07 | 9 | 100.6 |


**Figure 2**: Graph depicting solving time of different puzzle by Simulated Annealing method.

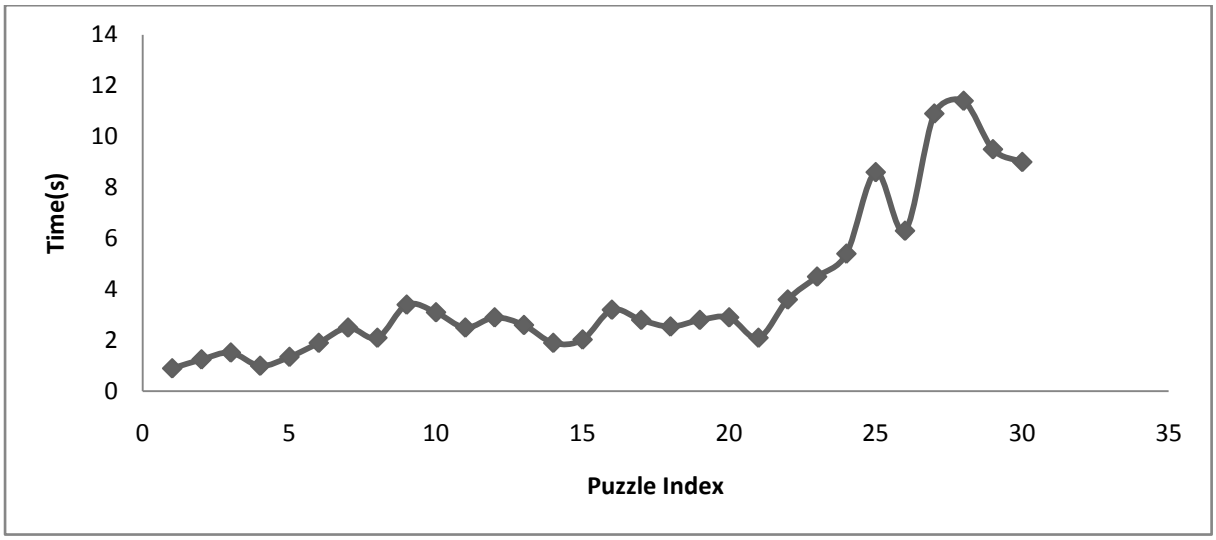


**Figure 3**: Graph depicting solving time of different puzzle by Genetic Algorithm method.

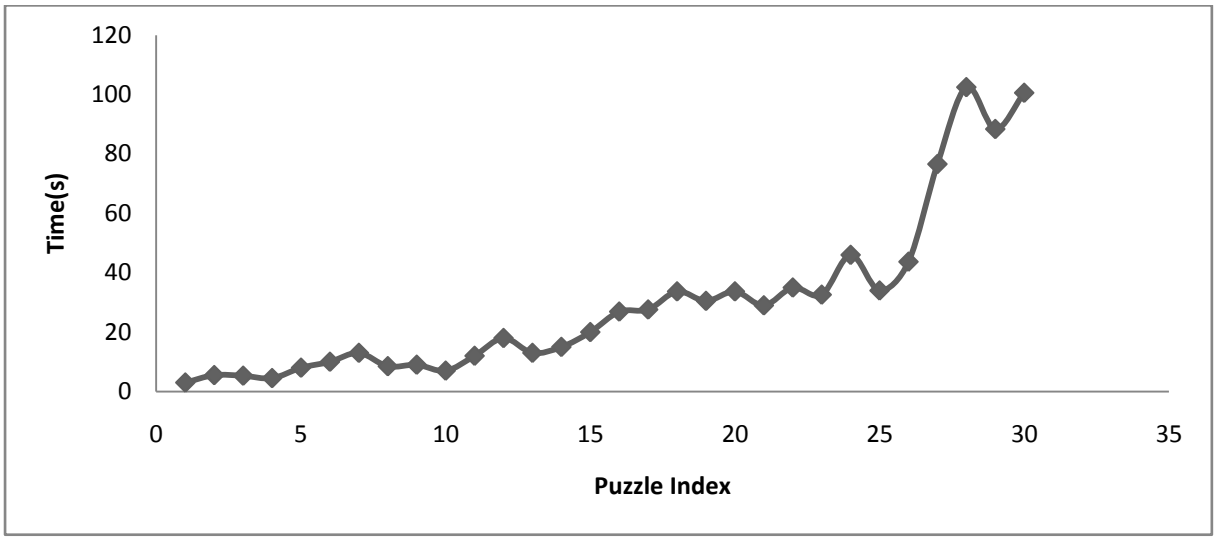


**Figure 4**: Graph depicting solving time of different puzzle by Harmony Search Algorithm.

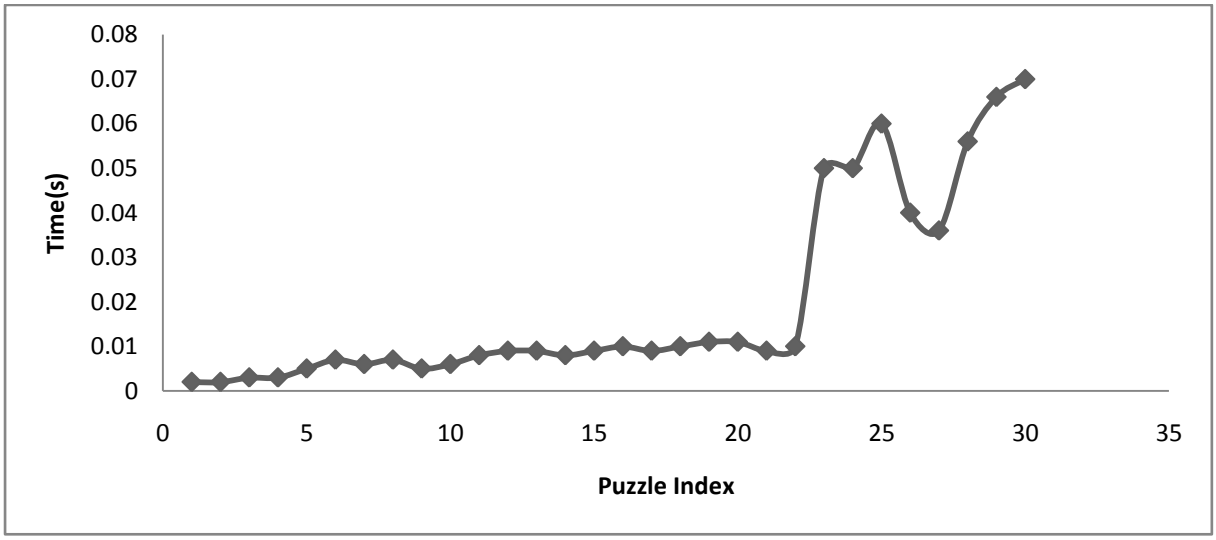


**Figure 5:** Graph depicting solving time of different puzzle by Graph Reference Algorithm.

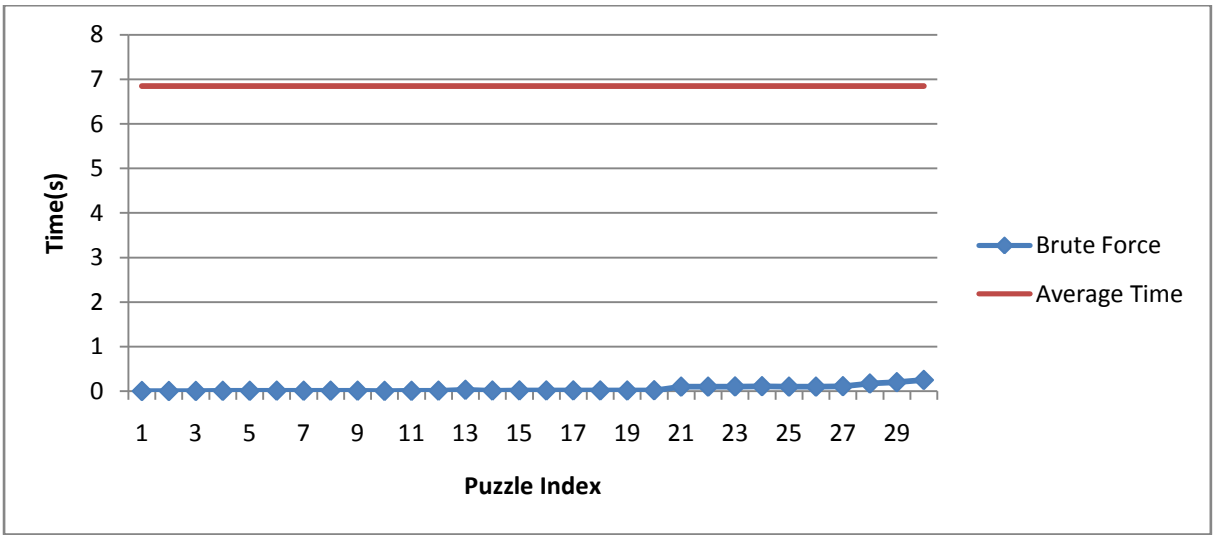


**Figure 6.** Plot of Time vs. Puzzle index of Brute Force algorithm and average time taken to solve a puzzle.

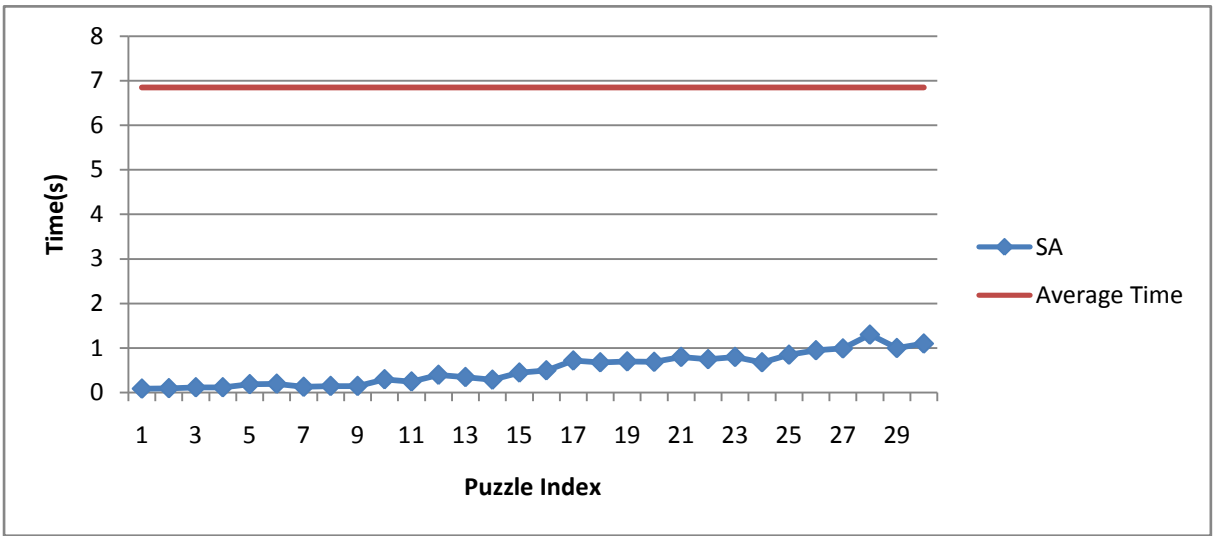


**Figure 7.** Plot of Time vs. Puzzle index of Simulated annealing algorithm and average time taken to solve a puzzle.

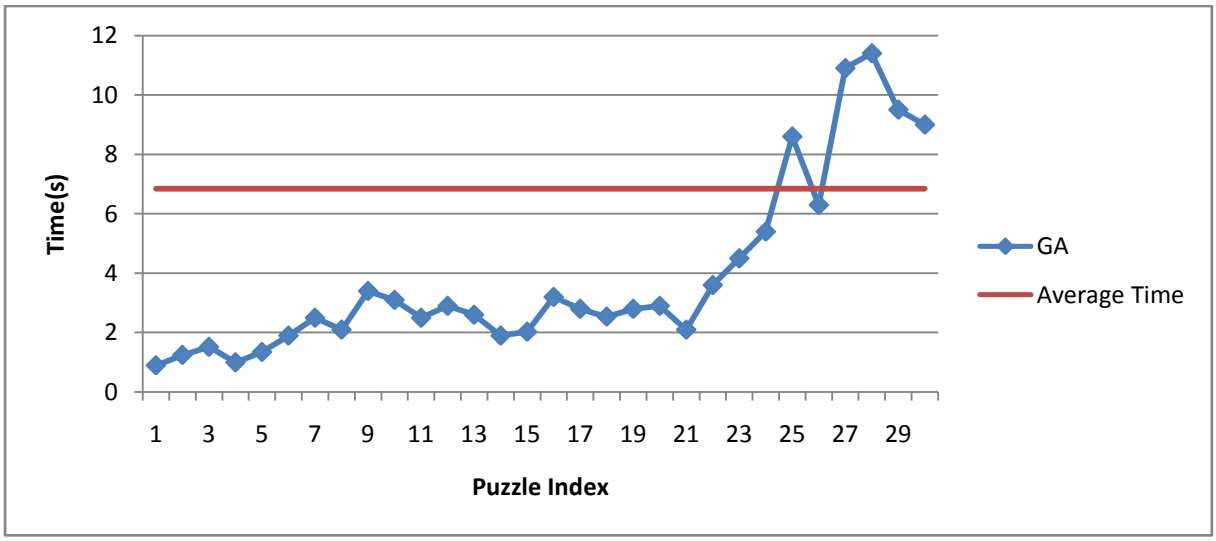


**Figure 8.** Plot of Time vs. Puzzle index of Genetic algorithm and average time taken to solve a puzzle.

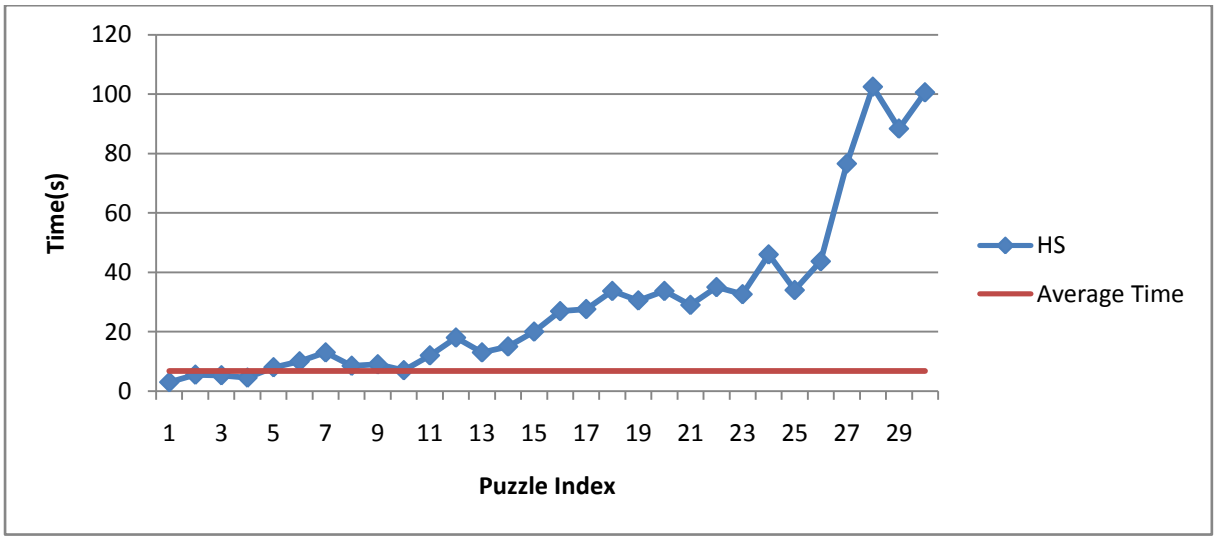


**Figure 9.** Plot of Time vs. Puzzle index of Harmony Search algorithm and average time taken to solve a puzzle.

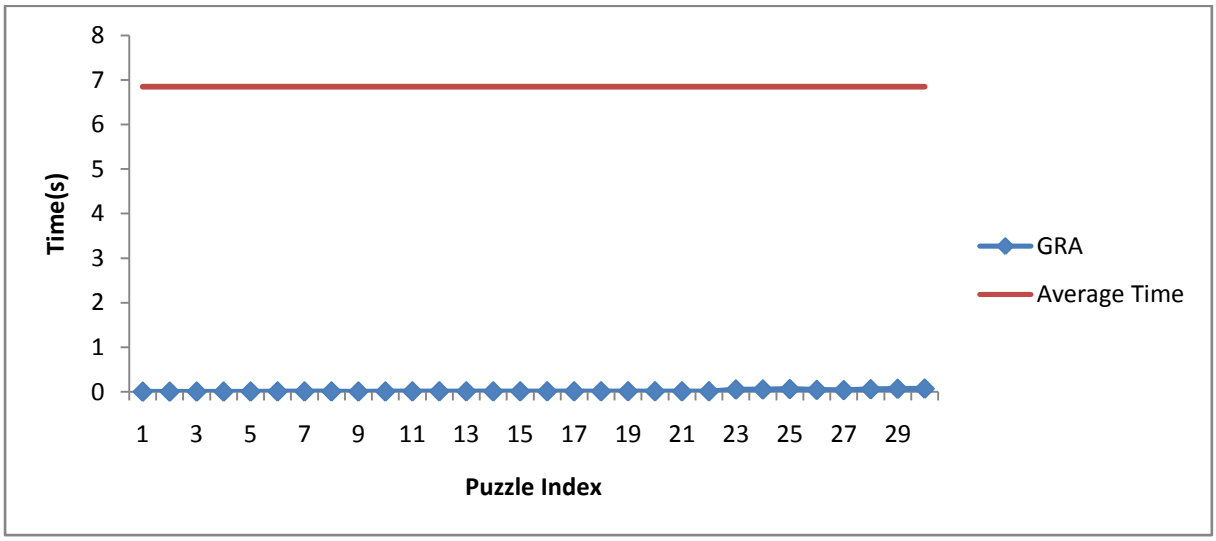


**Figure 10**. Plot of Time vs. Puzzle index of GRA algorithm and average time taken to solve a puzzle.

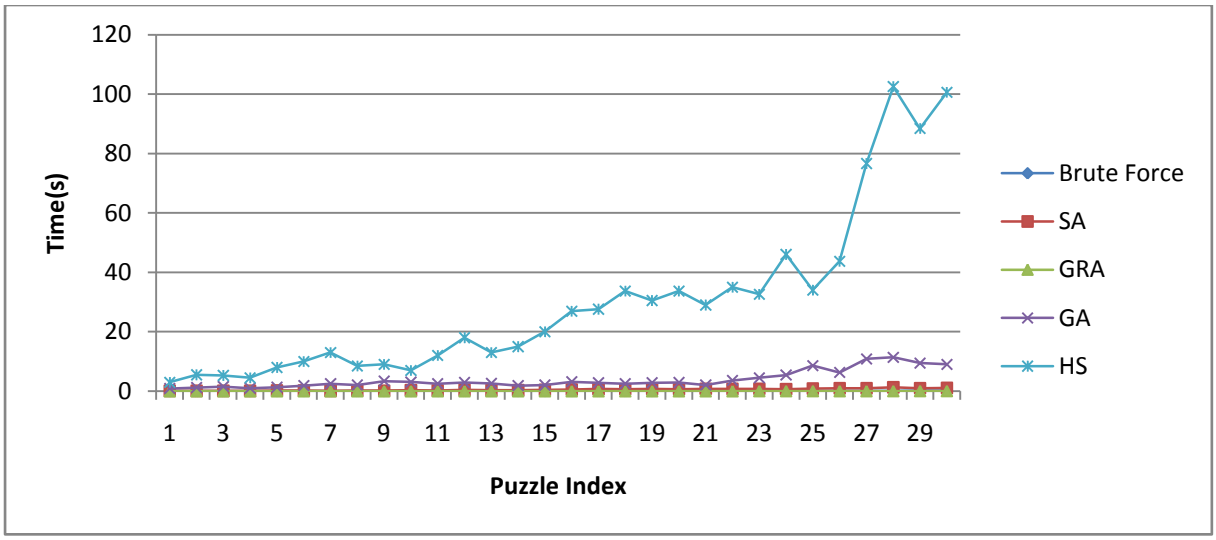


**Figure 11.** Plot of Time vs. Puzzle index for different algorithms

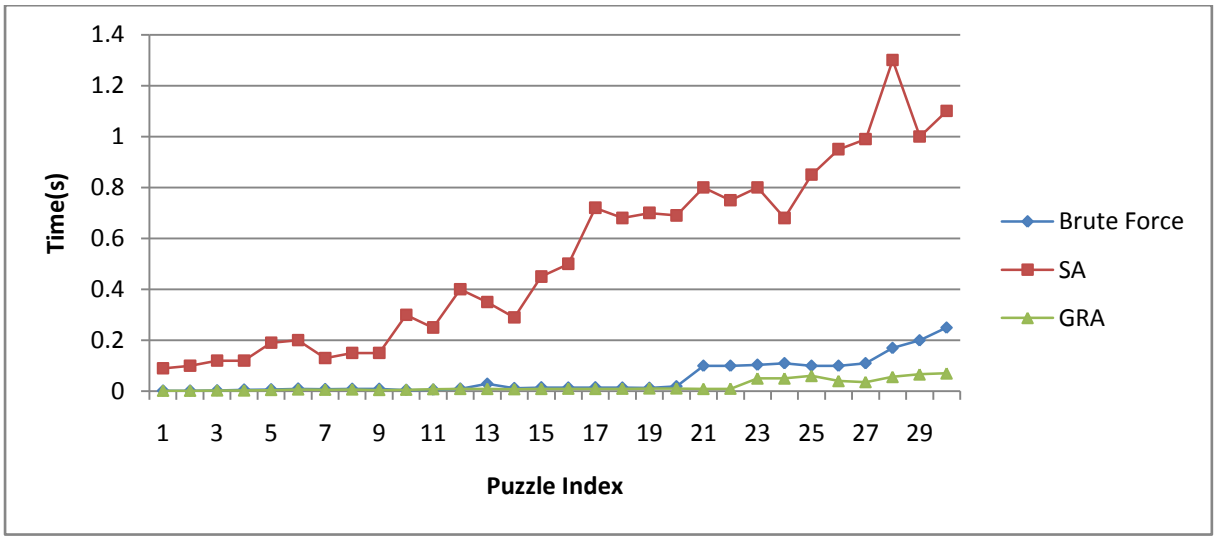


**Figure 12.** Plot of Time vs. Puzzle index of GRA, Brute Force, SA algorithm and average time taken to solve a puzzle

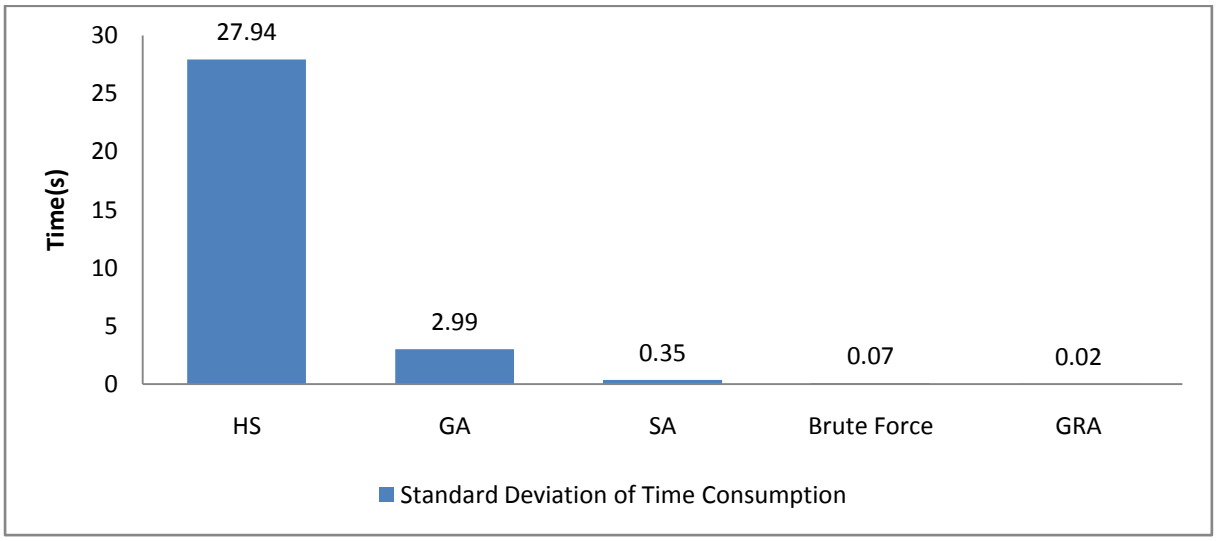


**Figure 13.** Standard Deviation of Time taken by different algorithms.

## IV. Conclusion

The performance analysis has been executed to reveal the fastest existing algorithm which is based on well- studied algorithmic approaches to tackle Sudoku in the authors' best knowledge. The comparative analysis has shown the performance of enumerative algorithms and non-heuristic algorithms to be the best. Though heuristic approaches like Simulated Annealing has reflected good performance, the performance is moderate comparable to that of GRA and Brute Force. Genetic Algorithm might have found extensive application in handling other optimization problems but fails to give satisfactory results in solving Sudoku. Harmony Search algorithms reflected serious concern on its performance and can be concluded to be the weakest attempt ever made to solve Sudoku problems. Thus the comparative performance characteristics study reveals that in terms of least possible time to solve Sudoku problems, GRA establishes its superiority over the others.The current study and experimental results will be immensely helpful for researchers worldwide in tackling not only Sudoku but other combinatorial optimization problems as well.

## References

[1]. Wikipedia: Sudoku. Available via www: http://en.wikipedia.org/wiki/Sudoku (16.08.2014)
[2]. Yato, T. and T. Seta.: Complexity and Completeness of Finding Another Solution and Its Application to Puzzles, IEICE Trans. Fundamentals, Vol. E86-A, No. 5, pp. 1052-1060 (2003)
[3]. Lewis, R., Metaheuristics can solve sudoku puzzles. Journal of heuristics, 13(4), 387-401, (2007)
[4]. Xu, J.: Using backtracking method to solve Sudoku puzzle, Computer Programming Skills & Maintenance, 5, pp. 17-21 (2009).
[5]. Chakraborty, R., Paladhi, S., Chatterjee, S., Banerjee, S.: An Optimized Algorithm for Solving Combinatorial Problems using Reference Graph, IOSR Journal of Computer Engineering, 16(3), pp. 1-7 (2014)
[6]. Darwin, C.: The Origin of Species: By Means of Natural Selection orThe Preservation of Favoured Races in the Struggle for Life, OxfordUniversity Press, London, 1859, A reprint of the 6th edition (1968)

[7]. Mantere, T., Koljonen, J.: Solving, rating and generating Sudoku puzzles with GA, Proc. of IEEE Congress on Evolutionary Computation (CEC 2008), IEEE, New York, pp. 1382-1389 (2007).

[8]. Deng, Xiu Qin., Li, Yong Da .: A novel hybrid genetic algorithm for solving Sudoku puzzles, Optimization Letters, 7(**2**), pp 241-257, February (2013)

[9]. Lee, S, K.., Geem, W, Z.: A new meta-heuristic algorithm for continuous engineering Optimization, Comput. Methods Appl. Mech. Engrg, 194, pp.3902–3933 (2005).

[10]. Geem, Zong Woo .: Harmony Search Algorithm for Solving Sudoku, Knowledge-Based Intelligent Information and Engineering Systems, Vol. 4692, pp 371-378, (2007)

[11]. Mandal, SatyendraNath., Sadhu, Soumi.: An Efficient Approach to Solve Sudoku Problem by Harmony Search Algorithm, An International Journal of Engineering Sciences ISSN: 2229-6913 Issue September 2011, Vol. 4, pp. 312-323 (2011)

[12]. Vikstén, Henrik., Mattsson, Viktor.: Performance and Scalability of Sudoku Solvers, Bachelor's Thesis at NADA, Royal Institute of Technology, University in Stockholm, Sweden (2013)

[13]. Volume 6146, 2010, pp 461-467. Sudoku Using Parallel Simulated Annealing, Zahra Karimi-Dehkordi, Kamran Zamanifar, Ahmad Baraani-Dastjerdi, Nasser Ghasem-Aghaee, First International Conference, ICSI 2010, Beijing, China, Proceedings, Part II, June 12-15, (2010).

[14]. Liu, Y., Liu, S.: Algorithm based on genetic algorithm for Sudoku puzzles, Computer Science, 37(3), pp. 225-226 (2010).

[15]. Cordella, P, L., Foggia, P., Sansone, C., Vento, M.: An improved algorithm for matching large graphs. Proc. of the 3rd IAPR-TC-15 International Workshop on Graph based Representations, , Ischia, Italy, pp. 149–159 (2001).

[16]. Mantere, T., Koljonen, J.: Solving and analyzingSudokus with cultural algorithms, Proc. of IEEE Congress on Evolutionary Computation (CEC 2008), IEEE, New York, pp. 4053-4060 (2008).

[17]. Eppstein, David. "Nonrepetitive paths and cycles in graphs with application to Sudoku." arXiv preprint cs/0507053 (2005).

[18]. Pereira, M. R., Vargas, P. K., França, F. M., de Castro, M. C. S., & de Castro Dutra, I. Applying Scheduling by Edge Reversal to constraint partitioning. In Computer Architecture and High Performance Computing, 2003. Proceedings. 15th Symposium on (pp. 134-141). IEEE, (2003, November).

[19]. Zhang, Z.: Solve & generate Sudoku puzzle by programming in AutoCAD, Computer Programming Skills & Maintenance, 17, pp. 16-18 (2008).

[20]. Zhao, Z., Guo, J., Yang, L.: Study of algorithm about Sudoku generation, Neijiang Science and Technology, 7, pp. 22-23 (2008).

[21]. Nicolau, Miguel.,Conor, Ryan.: Solving sudoku with the gAuGE system, Genetic Programming. Springer Berlin Heidelberg, pp. 213-224, (2006).