STM JOURNALS

Journal of Artificial Intelligence Research & Advances
ISSN: 2395-6720(online)
Volume 3, Issue 1
www.stmjournals.com

GA with Modifications for Efficiently Solving Number Puzzles

Anagha P. Khedkar*
Information Technology Department, Matoshri College of Engineering and Research Centre, Nashik, Maharashtra, India

Abstract

The previous research on the genetic algorithm (GA) applications especially for puzzle solving focuses on the GA capability for solving the puzzles of differing difficulty levels viz. magic square, jigsaw, sliding tile, crossword, path finding and of similar type. But this work attempts to solve logical number puzzles of exponential and factorial complexity using conventional GA with some important modifications depending on the puzzle type. The overall modifications done in GA are the advanced twin operator, order crossover and use of only mutation without crossover. GA is designed and applied with these modifications considering one at a time to solve the three number puzzles. The experimental results reflect the performance improvement in GA in terms of best and average convergence generation. The advanced twin operator enhances the overall performance of modified GA by 40% over the conventional GA. The modified GA is also tested for variation in population size which shows the faster convergence with sufficiently increased population size. Further to extend the current work, it is needed to find out the optimal population size depending on the puzzle problem.

Keywords: *GA, number puzzles, population size*

****Author for Correspondence*** E-mail: anagha_p2@yahoo.com

INTRODUCTION

Since ancient times, puzzles have attracted the world wide people of all ages because of their interesting and logical nature. Especially the interest and enthusiasm in puzzle solving enforces the logical development of solver and indirectly helps to solve large but complex real world problems in various fields, viz. computer networks, wireless networks, cognitive sciences, forensic engineering and VLSI systems. Among variety of puzzles, Sudoku puzzles with various difficulty levels, large jigsaw and number puzzles are very popular. From the computational complexity point of view, many puzzles have been proved to fall under the NP hard category. Number of techniques has been used to solve the puzzles and their efficiencies are measured. The techniques mainly comprise stochastic local search, simulated annealing, artificial intelligence, heuristics and genetic algorithm (GA). This work attempts to use GA for efficiently solving the number puzzles. As GA works on the principles of natural genetics, it is the suitable technique for efficiently finding the solutions of puzzles which have large and complex search spaces.

RELATED WORK

GA has been proved to be the efficient optimization technique after rigorously testing on standard benchmark test functions and huge number of applications in almost all facets of life. This empirical study focuses on the use of GA for the interesting application of puzzle solving. During last three decades, many researchers have worked on the application of GA for solving the problems of various types of puzzles viz. sliding tile, Sudoku, magic square, jigsaw, crossword, number magic. Here is a brief review of it.

In year 2012, Pillay has attempted to solve logical Sudoku puzzles using human heuristics and genetic programming [1]. The number of search and optimization methods viz. simulated annealing, genetic programming, particle swarm optimization, ant colony optimization, harmony search have been also tried to solve these puzzles [1]. In this research

work, 1800 puzzles of differing difficulty levels were successfully solved. They observed that the solution to many puzzles were found just using random moves without evolution, but sometimes evolution of one was needed to yield solution. Further they observed that the performance of genetic programming was comparable to other techniques and sometimes indicated better performance than others.

Simple evolutionary algorithms were applied to solve Sudoku problems of various difficulty levels but the success rate decreased as the difficulty level of puzzle increased [2]. Further the novel hybrid GA was designed and applied to solve Sudoku of various levels. In this, the operators of GA such as selection, crossover and mutation were changed as per the features of Sudoku puzzles. Researchers observed the improvement in convergence rate of GA [3]. The conventional GA was modified with mutation operation and tested on Sudoku puzzles [4].

Similarly typical conventional GA and GA with cultural learning were used to solve Sudoku of various levels [5–7]. Nicolau and Ryan applied GA to solve Sudoku of various levels but failed to found optimal solution for all levels of puzzles [8].

Moraglio *et al.* used different type of crossover called geometric crossover to solve Sudoku puzzles [9]. Sato applied proper genetic operations to preserve the building blocks in GA. He attempted to test such GA to solve Sudoku puzzles and further tried to reduce run times [10–12].

Besides Sudoku, many researchers have attempted to solve another interesting puzzle of magic square [13–16]. It is a rich combinatorial optimization problem and solved earlier using GA and other similar techniques. The performance of artificial intelligence minmax algorithm was comparatively higher than that of GA [15]. Toyama *et al.* have solved the jigsaw type puzzles and obtained good results for puzzles only up to 64 pieces [17].

Later Sholoman *et al.* attempted to solve the new type of two sided jigsaw puzzles of large size and increased complexity with efficient GA solver [18]. GA was started with the encoded chromosomes in the form of arrangement of puzzle pieces. They obtained significant improvement in the results and their GA solver provided the efficient solution for jigsaw puzzles of 22,834-piece size.

Recently in 2014, in my previous research work, the attempt was made to apply novel GA with advanced twin operator to solve the various number puzzles viz. a 10-digit self-describing number and last digits of n^2 [19–22]. The implementation of GA was carried out with the advanced twin operator and the conventional crossover and mutation operators. The results of this implementation showed the effectiveness of novel GA over conventional GA to generate the unique solution in terms of convergence time.

Further in the same year, Kazemi and Fatemi worked on GA modified by generating more informed chromosomes and applied it to solve Sudoku puzzles [23]. This modified GA was comparatively efficient than earlier approaches but failed to solve puzzle in minimum number of generations.

In this way the brief review of the literature highlights that original GA with some modifications has been used to solve the typical puzzles viz. Jigsaw of various sizes, Sodoku, magic square and number magic. The modifications in GA mainly include population re-initialization, geometric crossover, modified mutation and cultural learning. Considering the wide scope in this area, this research work has attempted to solve the type of number puzzles using conventional GA but with few modifications. The modifications in GA are done especially for crossover operation for some puzzles as the conventional crossover techniques viz. N-point crossover becomes invalid. The conventional crossover techniques generate invalid offspring for such puzzles. In the case of few puzzles, the efficiency of GA is improved by implementing GA with novel advanced twin operator along with its parameters. The design and implementation of advanced twin operator is based on the twin offspring formation in natural genetic systems [19–21].

THE PUZZLE PROBLEM AND DEFINITION

This work has applied modified GA to solve the following interesting number puzzles. The puzzles are defined as below:

1. Identify the 6-digit number below five lakh such that the sum of its all digits equals 43.
2. Create a number using only the digits 4, 4, 3, 3, 2, 2, 1 and 1. So it can only be eight digits and it should be such that the '1's are separated by one digit, the '2's are separated by two digits, the '3's are separated with three digits and the '4's are separated by four digits.
3. Create a number of 10 digits using only the digits 0 to 9 only once and find out unique 10 digit number such that first one digit should be divisible by 1, first two digits should be divisible by 2, first three digits should be divisible by 3 and so on till 10 digits should be divisible by 10.

The puzzle problem definitions are taken from the web source [24]. The puzzles are complex for the deterministic methods to find out solution as their complexity is in terms of exponential and factorial form. The search and optimization algorithm such as GA is suitable to find out the puzzle solution efficiently in terms of computational time. The significance of solving puzzles lies in getting solutions for the large number of real world problems especially in engineering field viz. VLSI, computer networks, wireless networks. In VLSI and networking field, it is the need to find out path or route from one node to other or from one cell to other, placement of cells. Thus GA based puzzle solving indirectly helps in finding out solutions for problems in other fields.

In this work, GA is used with few modifications viz. the use of order crossover, advanced twin operator and only mutation without crossover. At a time GA is applied with one modification to solve the puzzles. The novel operator called as advanced twin operator is used along with the other conventional GA operators to find out solution for puzzle 1. To solve puzzle 2, GA is used without conventional crossover but with only mutation operator. Thus GA requires few modifications in its conventional form to solve each puzzle efficiently. The GA with order crossover is used to generate valid offspring's in the case of solving puzzle 3.

GA Operators: Order Crossover (OX) and Advanced Twin Operator

Order crossover is a variation of partial mapped crossover (PMX) that uses different repairing technique to make valid offspring. In this, first substring is selected from $parent_1$ at random and then proto-child is created by copying the substring into the corresponding positions. After this, the substring genes are deleted from the $parent_2$ and the remaining genes from $parent_2$ are placed according to the order of sequence from left to right in the proto-child. This crossover is represented in Figure 1.

$Parent_1$: 1 2 3 4 5 6 7 8 9

$Parent_2$: 5 7 4 91 3 6 28

Proto-child: _ _ 3 4 5 6 _ _ _

Offspring: 7 9 3 4 5 6 1 2 8

Fig. 1: *Representation of OX Crossover.*

In this work, GA is designed with OX and its associated crossover probability p_c is set to 1 and it is used to solve puzzle 3.

The concept behind the development of advanced twin operator is explained here. The concept of natural twin mechanism is extracted to develop the novel twin operator for GA. The twin operator proposed for GA produces the twin offspring of good characteristics in the simulated environment of GA. These twin offspring are further expected to explore the search space efficiently after the crossover operations in the forthcoming generations. In my earlier work, the twin operator is designed and developed in two phases. In the first phase, the primary or the first version of the twin operator is proposed. It is empirically tested on the optimization test function by implementing micro-GA. In the second phase, the advanced twin operator or the second version of this operator is proposed and developed. GA with this advanced twin

operator is tested for efficiency and performance on the standard benchmark optimization functions [20].

The design of the advanced twin operator resembles the twin's formation due to single ovulation process in natural genetics. The process of generating twin individual strings involves the selection of fit parents for reproduction, crossover of these parents and then application of the advanced twin operator. The selection operator of GA is responsible for selecting the fit parents for reproduction. Similar to natural system, one of the fit parents is equivalent to single ovum of the natural system. The elitist strategy of GA requires that the best individual or the individual of first rank of the current generation should be forwarded as it is to the next generation. Therefore the second rank individual or individual next to the best individual should be selected for reproduction as the one of the parents resembling the single ovum in single ovulation. This parent is referred to as p_1. The other parent should be randomly selected from the current generation. This parent is referred to as p_2. The two parents are crossed using any conventional crossover method to generate two offspring say $child_1$ and $child_2$ respectively. After this, the advanced twin operator is designed and applied as follows: After reproduction, the hamming distance of each child from both the parents is calculated. Hamming distance indicates the unequal genes of the child from the respective parent. Consider that H_1 and H_2 are denoted as the hamming distances of $child_1$ from p_1 and p_2 respectively. The proposed advanced twin operator should randomly select exactly half the number of unequal genes from H_1 as well as H_2 and change only the values of these genes by keeping all other genes same as that of $child_1$. This creates the first twin mate child say $child_3$. In this way the advanced twin operator generates the first twin pair $child_1$:$child_3$. The selection of unequal genes from H_1 and H_2 has considerable effect on the decoded value of the twin mate child. Similar process is repeated with $child_2$ to generate its twin mate $child_4$.

The design parameter of twin operation is the probability of twin operator. It should be low that resembles the frequency of twining in natural genetics. The other design parameter is number of unequal genes randomly selected from H_1 and H_2. It affects the twin separability parameter that increases in proportion to the number of unequal genes. This number should be equal 50 to 25% of the total number of unequal genes represented in hamming distances.

In this work, GA is designed with the advanced twin operator along with the setting of conventional parameters and further applied to solve puzzle 1.

Application of GA for Puzzles

The design and implementation of GA comprises the setting and tuning of its parameters and operators. The parameters involve encoding of GA individual, population size, type of crossover, crossover probability, mutation probability, twin probability and the termination criteria. The genetic operators, viz. selection, crossover and mutation, advanced twin operator and its technique need to be selected according to problem. The setting of GA parameters differs as per the puzzle problem to be solved. The design of GA according to puzzle problem is specified below:

Puzzle 1

For this puzzle, the GA individual is encoded as six decimal digits length in which all five digits except MSB digit can vary from 0 to 9. The MSB digit can vary only from 0 to 4. The appropriate population size is selected by setting standard values and taking execution trials. After taking number of execution trials, the population size was set to 20. The tournament selection operator is selected for parents' selection. The termination criterion is set as the maximum number of generations. The single point crossover technique with crossover probability p_c of 1 is chosen to completely explore the search space. The mutation probability p_m is set to 0.01 after executing number of repeated run trials. The twin probability is set to 0.05 and twin separability to 50% as mentioned to get best performance [21]. In this way, simple GA called as SGA and GA with advanced twin operator called ATGA are designed and implemented to find out the best solution for this puzzle. At the same time the effects of advanced twin operator are also recorded to

check the quality of the solution in terms of computational time.

Puzzle 2

GA individual is encoded as length of eight digits in which any two digits should be same with value from 1 to 4. The population size is varied from 5 to 20. When population size is five, then GA is called as micro-GA. The selection operator is chosen as binary tournament selection operator with replacement. For this puzzle, GA is implemented without any type of crossover but with only mutation technique. The swap mutation with probability of 0.07 is used. The swap mutation technique yields the valid individuals whereas others generate invalid individuals. The value of p_m is set after taking lot of trials of GA execution. The termination criterion is set as the maximum number of generations.

Puzzle 3

For this, GA individual is encoded as a string of ten decimal digits in which only MSB digit can vary from 1 to 9. The population size is set to 100 after taking large number of execution trials. The maximum number of generations is set to 40. The selection operator is chosen as binary tournament selection operator with replacement. The order crossover technique with crossover probability p_c of 1 is used. The conventional crossover techniques viz. one point, two point and N-point cannot be used as they create invalid offspring. The swap mutation technique with probability p_m=0.07 is used.

The objective function for each puzzle is defined on the basis of the respective puzzle logic. The fitness function is derived from the corresponding objective function. The large numbers of experimental simulations are carried out for setting GA parameters. The population size is varied in distinct simulations and its effect is observed on the performance of GA. The brief experimental set up for all puzzles is displayed in Table 1.

Table 1: *Experimental Setup of GA Parameters.*

Puzzles	Encoded String Length	Population Size	Crossover (pc=1)	Fixed p_m	Max. Gens
Puzzle 1	6 digits	6, 10, 20	Single point along with advanced twin operator	0.01, 0.03, 0.05	20
Puzzle 2	8 digits	20, 40	No crossover	0.05, 0.08	20
Puzzle 3	10 digits	100, 200, 300	Order (OX)	0.07	40

RESULTS AND DISCUSSIONS

The simulations of GA are carried out for repeated number of run trials to find the best solution for three puzzles. For most simulations, the number of run trials executed is 25. The performance parameters of GA viz. best convergence generation, average convergence generation, number of computations that produce the puzzle solution are recorded and further analyzed to observe the effects of GA parameters on its performance.

Table 2: *Effect of Advanced Twin Operator on GA convergence.*

Puzzle No.	Puzzle Solution	SGA		ATGA		Total Run Trials
		Best Convergence Generation	Average Convergence Generation	Best Convergence Generation	Average Convergence Generation	
Puzzle 1	499849	6	10	3	6	15
Puzzle 1	499849	7	12	4	10	25

Table 3: *Effect of Variable Population Size on GA Convergence for puzzle 2 Problem.*

Puzzle No.	Puzzle Solution	Population Size	SGA–Best Convergence Generation	SGA–Average Convergence Generation	Total Run Trials
Puzzle 2	41312432	20	4	10	25
Puzzle 2	41312432	40	4	8	25
Puzzle 2	41312432	60	3	6	25

Table 4: *Effect of Variable Population Size on GA Convergence for puzzle 3 Proble.*

Puzzles No.	Puzzle Solution	Population Size	SGA–Best Convergence Generation	SGA–Average Convergence Generation	Average Convergence Run Trial	Total Run Trials
Puzzle 3	3816547290	100	8	22	18	25
Puzzle 3	3816547290	200	7	20	8	25
Puzzle 3	3816547290	300	6	16	5	25

From Table 2, it is observed that ATGA provides solution for puzzle 1 more efficiently than that of SGA. This is reflected from the values of best convergence and average convergence generation. Additionally it is found that ATGA produced the multiple solutions for this puzzle which gives out the sum to be 43. But the Table 2 displays one sample solution displayed on the website mentioned [24].

The results displayed in Table 3 highlight the effects of variable population size on the puzzle 2 solution. It clearly indicates that as population size increases, the GA converges relatively faster to generate solution for puzzle 2. The large number of execution trials of GA has resulted in multiple valid solutions for puzzle 2. But Table 2 displays only one sample solution. For puzzle 3, it has been highlighted from Table 4 that population size needs to be increased to get the solution. As the population size increases, the chances of getting solution also increase and GA converges faster.

CONCLUSION AND FUTURE SCOPE

The efficiency of GA is reflected using the performance parameters viz. best convergence generation, average convergence generation and population size variation. This work clearly shows that the efficiency of ATGA is improved because of the implementation of advanced twin operator. From the population size variation, it is concluded that the GA converges relatively faster when population size increases. But when the population size increases beyond certain limit, then it becomes difficult for GA to generate solution in optimum number of computations. Therefore it is necessary to find out optimal population size required for the particular puzzle problem. In future work, the work will focus on finding out optimal population size which will produce result in reasonable number for computations. Also the work will be tested on the puzzles of increased complexity.

COMPETING INTERESTS

The authors declare that they have no competing interests.

REFERENCES

1. Pillay N. Finding Solutions to Sudoku Puzzles Using Human Intuitive Heuristics. *SACJ*. 2012; (49): 25–34p.
2. Aid T. *Evolutionary Sudoku Solver*. 2011. http://www.scienceandatheism.com/wpcontent
3. Deng XQ, Li YD. A Novel Hybrid Genetic Algorithm for Solving Sudoku Puzzles. *Optim Lett*. 2011; 1–17p.
4. Hamnes DO, Julstrom BA. Iterated Mutation in an Evolutionary Algorithm for Sudoku. *Proceedings of 26th IASTED International Conference on Artificial Intelligence and Applications*. CA, USA: ACTA Press Anaheim; 2008; 272–7p.
5. Mantere T, Koljonen J. Solving and Rating Sudoku Puzzles with Genetic Algorithms. *Proceedings of the 12th Finnish Artificial Intelligence Conference STeP*. 2006; 86–92p.
6. Mantere T, Koljonen J. Solving, Rating and Generating Sudoku Puzzles with GA. *Proceedings of the 2007 IEEE Congress on Evolutionary Computation*. 2007; 1382–9p.
7. Mantere T, Koljonen, J. Solving and Analyzing Sudokus with Cultural Learning. *Proceedings of the 2008 IEEE Congress on Evolutionary Computation*; 1–6 Jun 2008; Hong Kong. 2008; 4053–60p.
8. Nicolau M, Ryan C. Solving Sudoku with the GAuGE System. *Proceedings of 9th European Conference on Genetic Programming*; 2006. 213-24p.

9. Moraglio A, Togelius J, Lucas S. Product Geometric Crossover for the Sudoku Puzzle. *Proceedings of the IEEE Congress on Evolutionary Computing (CEC 2006)*. 2006; 476–83p.
10. Sato Y. Solving Sudoku with Genetic Operations that Preserve Building Blocks. *Proceedings of the 2010 IEEE Conference on Computational Intelligence and Games (CIG '10)*. 2010; 23–9p.
11. Sato Y, Hasegawa N, Sato M. GPU Acceleration for Sudoku Solution with Genetic Operations. *Proceedings of the IEEE Congress on Evolutionary Computation (CEC)*. Jun 2011; 296–303p.
12. SatoY, Hasegawa N, Sato M. Acceleration of Genetic Algorithms for Sudoku Solution on Many-Core Processors. *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation*. ACM Press; 2011; 407–14p.
13. Ardel DH. *Genetic Algorithms and Genetic Programming at Stanford 1994*. TOPE and Magic Squares, a Simple GA Approach to Combinatorial Optimization. In: Koza JR, editors. *Genetic Algorithms in Stanford*. Stanford, CA: Stanford Bookstore; 1994.
14. Alander JT, Mantere T, Pyylampi T. *Digital Halftoning Optimization via Genetic Algorithms for Ink Jet Machine*. In: Topping BHV, editors. DCMHPC; 1999; 211–6p.
15. Fonseca Jose B. *The Magic Square as a Benchmark: Comparing MIP to improved GA and to a High Performance Minimax AI Algorithm.*
16. Sandip B. MSc. Second Year Project. Dept. of Mathematics. IIT Bombay. (Technical Report).
17. Toyama F, Fujiki Y, Shoji K, *et al*. Assembly of Puzzles Using a Genetic Algorithm. *IEEE International Conference on Pattern Recognition*. 2002; 389–92p.
18. Dror S, Omid D, Nathan S. A Genetic Algorithm-Based Solver for Very Large Jigsaw Puzzles. *IEEE Conference on Computer Vision and Pattern Recognition*. 2013; 1767–74p.
19. Khedkar AP, Subbaraman S. Novel Twin Operator for Genetic Algorithm. *Proceedings of 3rd International Conference on Data Management and Advances in Computer Science and Engineering*. 2010; 111–17p.
20. Khedkar AP, Subbaraman S. Effect of Advanced Twin Operator on the performance of Genetic Algorithm. *IJERT*. 2010; 3: 721–31p.
21. Khedkar AP, Subbaraman S. The Novel Approach of Adaptive Twin Probability for Genetic Algorithm. *IJASCSE*. 2013; 2(2): 31–7p.
22. Khedkar AP, Subbaraman S. Effectiveness of Novel GA with Advanced Twin Operator for Solving the Number Puzzles. *IJARCET*. 2014; 3(8): 2752–5p.
23. Kazemi SM, Fatemi B. A Retrievable Genetic Algorithm for Efficient Solving of Sudoku Puzzles. *IJCISE*. 2014; 8(5): 681–5p.
24. http://gpuzzles.com/quiz/hard-math-puzzles-with-answers/4

Cite this Article
Khedkar Anagha P. GA with modifications for efficiently solving number puzzles. *Journal of Artificial Intelligence Research & Advances*. 2016; 3(1): 1–7p.