Analysis of the Various Population Sizes in Genetic Algorithm for Solving N-Queen NP-Hard Problem

R.K.Singh [1], V.K.Panchal[2], B.K. Singh[3]
[1]*Department of Computer Science, Al-Falah University, Faridabad, Haryana, India*
[2]*Department of Computer Science, Al-Falah University, Faridabad, Haryana, India*
[3]*Department of Computer Science , Satyug Darshan Institute of Engineering & Technology, Faridabad, Haryana, India*
(E-mail: rajeshsingh22@gmail.com)

***Abstract*—** N-Queen is a well-known NP-Hard problem. It's a constraint satisfaction problem and problems in science and engineering can be solved using solutions of N-Queen problem. In N-Queen problem, N number of queens are to be placed on a chess board of size N*N such that no two queens can attack on each other. As the size of N increases, the complexity of the algorithm to solve N-Queen problem also increases. In this paper, Genetic Algorithm is applied to solve N-Queen problem. The GA is applied on many instances of N-Queen problem. The performance of GA varies on size of population. The effect of population size on the performance of GA to solve N-Queen NP-Hard problem is analyzed.

Keywords*—NP-Complete; N-Queen; Genetic Algorithm; Population Size*

I. INTRODUCTION

N-Queen problem is a NP-Hard problem. In N-Queen problem, positions of N queens are to be find such that no queen can attack on any other queen. In the game of chess, a queen can attack on other queen horizontally, vertically and diagonally. Figure 1 is showing a solution of eight queen problem. Eight Queens are placed on 8*8 chess board such that none of the queen is attacking on other queen.

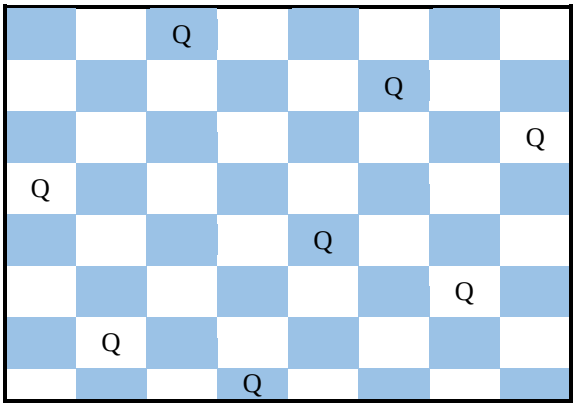

Figure 1 - 8-Queen Problem

As the number of queens increases, the size of the chess board also increases and the complexity of the algorithm to solve this problem also increases. N-Queen problem can be solved using backtracking algorithm. The backtracking algorithm also take so much take time when the size of N is very large i.e. in hundreds or in thousands. Genetic algorithm is also solved using heuristic algorithm such as Ant Colony Optimization ACO, Genetic Algorithm GA etc. In this paper N-Queen problem is solved using GA.

Genetic algorithm is an algorithm which is based on the Darwinian evolutionary theory - "survival of the fittest". The concept of GA is taken from natural science. The specifies make their population, perform meeting and generate new children. This process is going on. The concept is applied on some computational problems to solve. Genetic Algorithm have its operations such as encoding, initial population generation, selection, cross over, and mutation. The selection, cross over and mutation are carried out repeatedly until a satisfactory solution is not found. GA never guarantees to produce the best possible solution but it generates many solutions with different fitness values. The fitness of a solution determines its quality. Algorithm -1 is showing the steps of GA.

Algorithm-1 Genetic Algorithm

1. Encode the problem in genetic form
2. Generate Initial population
3. Repeat steps 4 to 7 until terminating criteria is not reached
4. Calculate fitness of the population
5. Perform selection
6. Perform cross over
7. Perform mutation

Terminating criteria is set to stop the iterations of the genetic algorithm. The terminating criteria may be number of iterations to perform or a required fitness value of the best solution etc. The performance of the genetic algorithm depends upon the size of the population. In this paper, the objective is to find and analyze the effect of population size on the performance of the genetic algorithm to solve N-Queen NP-Hard problem. The next section illustrates the survey of the recent work done by other authors in this field.

II. LITERATURE SURVEY

J. E. Aghazadeh Heris and M. A. Oskoei [1] proposed a modified genetic algorithm to solve N-Queen problem. A local search minimum conflict algorithm was used to improve the performance of genetic algorithm. The hybrid of minimum

conflict algorithm and genetic algorithm improved the performance of GA and GA converges fast to solve N-Queen problem. A. M. Turky and M. S. Ahmad [2] used genetic algorithm to solve N-Queen NP-Hard problem. Genetic algorithm was found a better solution to solve N-Queen problem as compared to other techniques. Y. Wang, H. Lin and L. Yang [3] applied Particle Swarm Optimization PSO to solve N-Queen problem. The PSO algorithm was applied with swarm refinement and the hybrid of swarm refinement and PSO solves the N-Queen problem better than the other approaches. I. Martinjak and M. Golub [4] compares different heuristic approaches to solve N-Queen problem. Three heuristic techniques, simulated annealing, tabu search, and genetic algorithm was used to solve N-Queen problem. The performance of these three algorithm was compared. It was observed that for some cases one algorithm was found better while for some other cases the other algorithm was found better. But none of these algorithms was found best to solve N-Queen problem. Many other authors [5-13] applied improved genetic algorithm to solve N-Queen or some other optimization problem using genetic algorithm. Vibhuti et al. [14] proposed a novel approach for distributed generator power optimization. N.M. Abroja et al. [15] uses genetic fuzzy system in hybridization. The fuzzy system is applied in GA parameters to perform cross over and mutation. It improves the results in hybridization. Gagandeep Kaur et al. [16] applied GA in hybrid code clone detection technique. By going through these recent approach it has been found that the performance of genetic algorithm matters a lot in solving different problems. The gap is found that there is need to improve the performance of genetic algorithm to solve NP-Hard problems. The performance of the genetic algorithm can also be improved by varying different genetic parameters such as population size, cross over rate and mutation rate. The next section illustrates how to found best values of GA parameters to solve N-Queen NP-hard problem.

III. PROPOSED METHODOLOGY

In this work, N-Queen problem is solved using genetic algorithm. Algorithm-1 is implemented to solve N-Queen problem. The value of the population size is selected and GA is applied with different set of these values.

Population Size – The size of population plays an important role in the performance of the genetic algorithm. It determines the number of chromosomes in the population. If population size is taken very large then it takes lot of computational time to generate the population but the chances to get solution in early iterations are improved. If the population size is kept very low, then the solution may not be found in early iterations. In this work the population size is kept from 20 chromosomes to 200 chromosomes in the steps of 20 and the performance of GA is evaluated.

Genetic algorithm is implemented in JAV using JDk1.7 and NetBeans 8.0.2 IDE. The next section illustrated the results found.

IV. RESULTS AND ANALYSIS

The genetic algorithm is implemented and its performance in terms of time taken is evaluated. Different values of population size are used and results are obtained.

Table-1 is illustrating the results found for solving 16-Queen problem. Different values of population size ranges from 20 chromosomes to 200 chromosomes are used and time taken in performing 100 iterations of the genetic algorithm is calculated. Table-1 is showing solution of 16 queen. It is showing population size, execution time, average clashes in the population and clashes between the queens in the best chromosomes after executing 100 iterations of Algorithm-1. The rate of cross over is kept 40% and rate of mutation is kept 3%.

TABLE I. ANALYSIS OF GA TO SOLVE 16-QUEEN PROBLEM BY VARYING POPULATION SIZE

Sr No	Queens	Population Size	Execution Time (ms)	Avg Clashes	Clashes in best chromosome
0	16	20	53.44	10.4	10
1	16	40	57.73	7.95	6
2	16	60	67.72	5.93	4
3	16	80	105.89	5.83	4
4	16	100	139.5	5.88	4
5	16	120	175.17	7.57	6
6	16	140	217.18	5.81	4
7	16	160	218.18	6.28	4
8	16	180	258.45	5.7	4
9	16	200	287.99	3.72	2

It is analyzed from the results obtained in Table-1 that the genetic algorithm performs better and provide good results with less number of clashes between the queens on large population size (200 chromosomes). However, GA takes more time (287.99 ms) to complete 100 iterations of 16-Queen problem. For small population size (20 chromosomes), GA only take 10.4 ms to complete 100 iterations. But here the quality of solution is not good and there are 10 clashes between the queens in the best solution. Thus it is analyzed that the large population size helps in providing quality results but need more computational time.

V. CONCLUSION AND FUTURE SCOPE

In this paper genetic algorithm is used to solve N-Queen NP-Hard problem. The performance of genetic algorithm is evaluated on different values of population size. It is concluded that GA with large population size take more time to complete its iterations. But the quality of solution is good with larger population size to solve N-Queen NP-Hard problem. In future, the value of other genetic parameters such as cross over rate, mutation rate can also be analyzed to solve N-Queen NP-hard problem. The analysis of GA parameters can also be done on

larger instances of N-Queen problem where number of queens are in hundreds or thousands.

REFERENCES

[1] J. E. Aghazadeh Heris and M. A. Oskoei, "Modified genetic algorithm for solving n-queens problem," *2014 Iranian Conference on Intelligent Systems (ICIS)*, Bam, 2014, pp. 1-5.

[2] A. M. Turky and M. S. Ahmad, "Using genetic algorithm for solving N-Queens problem," *2010 International Symposium on Information Technology*, Kuala Lumpur, 2010, pp. 745-747.

[3] Y. Wang, H. Lin and L. Yang, "Swarm Refinement PSO for Solving N-queens Problem," *2012 Third International Conference on Innovations in Bio-Inspired Computing and Applications*, Kaohsiung, 2012, pp. 29-33.

[4] I. Martinjak and M. Golub, "Comparison of Heuristic Algorithms for the N-Queen Problem," *2007 29th International Conference on Information Technology Interfaces*, Cavtat, 2007, pp. 759-764.

[5] T. Li and S. Guo, "Research on Two-Dimensional Entropy Threshold Method Based on Improved Genetic Algorithm," *2017 International Conference on Industrial Informatics - Computing Technology, Intelligent Technology, Industrial Information Integration (ICIICII)*, Wuhan, 2017, pp. 122-125.

[6] S. Pan and C. Chen, "Design of Stable IIR Filter by Improved Genetic Algorithm," *2012 Third International Conference on Innovations in Bio-Inspired Computing and Applications*, Kaohsiung, 2012, pp. 122-126.

[7] J. Liu, C. Huang and P. Li, "Optimal scheduling of wind farm with storage and forecasting based on improved genetic algorithms," *The 26th Chinese Control and Decision Conference (2014 CCDC)*, Changsha, 2014, pp. 80-85.

[8] Y. Long, Y. Su, H. Zhang and M. Li, "Application of Improved Genetic Algorithm to Unmanned Surface Vehicle Path Planning," *2018 IEEE 7th Data Driven Control and Learning Systems Conference (DDCLS)*, Enshi, 2018, pp. 209-212.

[9] J. Xiao-Ting, X. Hai-Bin, Z. Li and J. Sheng-De, "Flight Path Planning Based on an Improved Genetic Algorithm," *2013 Third International Conference on Intelligent System Design and Engineering Applications*, Hong Kong, 2013, pp. 775-778.

[10] S. Cui and J. Dong, "Detecting Robots Path Planning Based on Improved Genetic Algorithm," *2013 Third International Conference on Instrumentation, Measurement, Computer, Communication and Control*, Shenyang, 2013, pp. 204-207.

[11] D. XiaoTang, Z. Hui, W. ZhenXing and Z. WenWen, "Back analysis on dynamic parameters of pump roon structure based on improved genetic algorithm," *2015 7th International Conference on Modelling, Identification and Control (ICMIC)*, Sousse, 2015, pp. 1-5.

[12] Y. Gao and J. Ye, "An Improved Genetic Algorithm Based on Normal Distribution for Solving the Traveling Salesman Problem," *2018 International Conference on Virtual Reality and Intelligent Systems (ICVRIS)*, Changsha, 2018, pp. 360-362.

[13] N. Jun, "An Improved Genetic Algorithm for Intelligent Test Paper Generation," 2014 7th International Conference on Intelligent Computation Technology and Automation, Changsha, 2014, pp. 72-75.

[14] Vibhuti et al. "Novel approach for DG Power optimization by Multi objective Genetic algorithm." International journal of research in electronics and computer engineering, IJRECE VOL. 5 ISSUE 3 JULY. -SEPT. 2017.

[15] N.M.Abroja et al. "Genetic Fuzzy System in Hybridization.", International journal of research in electronics and computer engineering, IJRECE VOL. 6 ISSUE 2 APR.-JUNE. 2018

[16] Gagandeep Kaur et al. "The Hybrid Code Clone Detection Technique Using Genetic Algorithm and Artificial Intelligence (FFNN)", International journal of research in electronics and computer engineering, IJRECE VOL. 5 ISSUE 1 JAN.-MAR. 2017

Mr Rajesh Singh completed his M.Tech from YMCA University of science and technology Faridabad, Haryana, India. He is a research scholar of Al-Falah University Faridabad, Haryana, India. He has published twelve papers in various conferences and journals till date. His research area include Algorithms, Compiler Design and Genetic Algorithm.

Prof. (Dr.) Bhupesh Kumar Singh received B.E. in (Computer Science & Engineering) from Government College of Engineering. Tirunelveli, Anna University, India in the year of 1999, and Ph. D in Sketch Recognition from BIT Mesra, Ranchi in the year 2013. Currently working as Professor & Director at Satyug Darshan Institute of Engineering & Technology, Faridabad, India. His research interest includes theoretical computer science, image processing and pattern recognition.