Solving N-Queen Problem Using Genetic Algorithm And Analyzing The Effect Of GA Parameters On Its Performance

[1]R.K.Singh & [2]V.K.panchal & [3]B.K.Singh
[1,2,]Department of Computer Science, Al-Falah University, Faridabad, Haryana, India.
[3]Department of Computer Science , Satyug Darshan Institute of Engineering & Technology, Faridabad, Haryana, India.

Received: February 08, 2019 Accepted: March 25, 2019

ABSTRACT: *Genetic Algorithm GA is a metaheuristic technique which is based on theory of evolution. Genetic Algorithm is used to solve many computational problems of science and engineering. GA applied its genetic operators to solve a problem. The performance of genetic algorithm depends upon its genetic operators and parameter values set for population size, cross over rate and mutation rate. In this paper the performance of genetic algorithm is analyzed by changing values of genetic parameters for solving N-Queen problem. The GA is applied on N-Queen problem with 16 queens (N=16) and its performance is analyzed in terms of quality of solution achieved and time taken.*

Key Words: *Genetic Algorithm, N-Queen, NP-Hard*

Introduction

Genetic Algorithm is bases upon theory of evolution. 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. 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. In genetic algorithm different parameters such as size of initial population, cross over rate, mutation rate etc. are to be set. The performance of the genetic algorithm depends upon the value of genetic parameters. Many authors implemented GA to solve various problems using different values of genetic parameters. Figure-1 is showing 8-Queen problem.

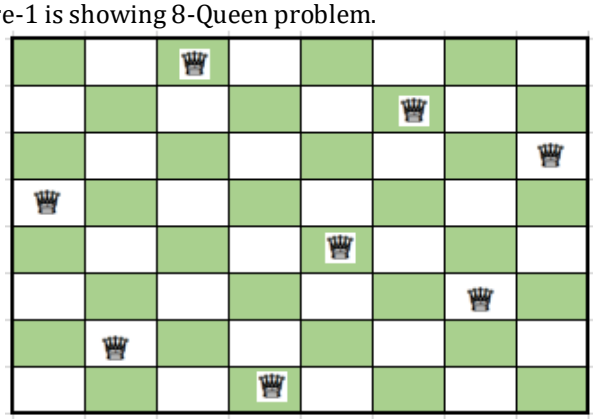

Fig. 1.8-Queen Problem

N-Queens have to be placed on a check board of size N*N in this problem. The constraint is that none of the queen should attack on any other queen. 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, the objective is to find best values of the genetic parameters to solve N-Queen NP-Hard problem. The next section illustrates the survey of the recent work done by other authors in this field.

Literature Survey

Hidayati Robbi et al. [1] applied modified genetic algorithm in resource allocation for cognitive radio networks. The genetic algorithm improves the network throughput for cognitive radio network. B. V. Ha et al. [2] applied compact genetic algorithm for synthesis of thinned arrays. The modified version of genetic algorithm performed better for synthesis of thinned arrays. J. E. Aghazadeh Heris and M. A. Oskoei [3] 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 et al. [4] used genetic algorithm to solve N-Queen NP-Hard problem. Genetic algorithm was found a better solution for N-Queen problem as compared to other techniques. Y. Wang, H. Lin and L. Yang [5] 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 [6] 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 [7-20] also modified the genetic algorithm to solve problems in science and engineering. 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.

3 Proposed Work

In this work, Genetic Algorithm is applied to solve N-Queen problem. The performance of N-Queen problem is analyzed on different values of three genetic parameters population size, cross over rate and mutation rate. The details of the parameters are shown below.

Population Size – A population is a group of candidate solutions of the problem. The quality of these solution varies by their fitness value. Population size determine the number of chromosomes in the population. Larger population size takes more time in performing genetic operations. In this work population size varies from 20 to 200 in steps of 20 and the performance of GA to solve N-Queen problem is analyzed.

Cross Over Rate – The rate of cross over determines how many chromosomes of the population will participate in the cross over. If the cross over rate is low, then the changes in the population are very less. But it takes very less time to perform cross over. If it taken high, then the changes in the population are very high and chances to get the solution in early generations also increases. In this work the cross over rate is taken from 20% to 100% and the performance of GA is evaluated.

Mutation Rate – Mutation introduces some changes in the population. After mutation, the fitness value of the chromosome may increase or decrease. Generally, mutation rate is kept very low. In this work the mutation rate is kept 1% to 10% and the performance of GA is evaluated to solve N-Queen problem. The next section illustrates the results of the experiments and their analysis.

4 Results and Analysis

The genetic algorithm is applied on N-Queen problem and its performance in terms of time taken is evaluated. Different values of population size, cross over rate and mutation rate are used and results are obtained.

Table 1 is showing the results of GA to solve N-Queen problem by varying population size and keeping other parameters fixed. Results are shown for executing 100 iterations of GA by taking cross over rate 40% and mutation rate 3%. Population size is varied from 20 to 200 in steps of 20. Table 2 is showing the results of GA to solve N-Queen problem by varying cross over rate and keeping other parameters fixed. Results are shown for executing 100 iterations of GA by taking population size 100 and mutation rate 3%. Cross over rate is varied from 20% to 100% in steps of 20%. Table 3 is showing the results of GA to solve N-Queen problem by varying mutation rate and keeping other parameters fixed. Results are shown for executing 100 iterations of GA by taking population size 100 and cross over rate 40%. Mutation rate is varied from 1% to 10% in steps of 1%.

Table 1. . Results of GA for varying population size

Sr No	Queens	Population Size	Execution Time	Avg Clashes	Clashes in best chromosome
1	16	20	95	9.4	8
2	16	40	56	7.7	6
3	16	60	69	6.47	4
4	16	80	105	5.68	4
5	16	100	132	5.86	4
6	16	120	157	3.83	2
7	16	140	201	3.66	2
8	16	160	221	5.41	4
9	16	180	257	5.67	4
10	16	200	281	4.45	2

Table 2. . Results of GA for varying cross over rate

Sr No	Queens	Population Size	Execution Time	Avg Clashes	Clashes in best chromosome
1	16	0.2	163.04	7.62	6
2	16	0.4	120.15	6.1	4
3	16	0.6	216.88	5.8	4
4	16	0.8	150.73	5.76	4
5	16	1.0	162.88	3.76	2

Table 3. . Results of GA for varying mutation rate

Sr No	Queens	Population Size	Execution Time	Avg Clashes	Clashes in best chromosome
1	16	1%	179	5.82	4
2	16	2%	175	6.14	4
3	16	3%	128	5.62	4
4	16	4%	134	7.86	6
5	16	5%	123	5.74	4
6	16	6%	135	7.46	6
7	16	7%	136	7.82	6
8	16	8%	132	7.16	4
9	16	9%	126	5.62	4
10	16	10%	122	3.8	2

Table-1, Table-2 and Table-3 are showing the results of the experiments. Next section discusses the conclusion and future scope of the paper. From Table-1 is observed that with small population, GA takes less time in execution (95 ms for population size 20) as compared to large population size (281 ms for population size 200). However, the average fitness of the population is better as there are only 4.45 average classes in the population with population size 200 and 9.4 average clashes for population size 20. From Table-2 is observed that with small cross over rate (20%) the average clashes in the population are 7.62 whereas with high value of cross over rate (100%), the average clashes in the population are 3.76. So results with large cross over rate are better as compared to small cross over rate. From Table-3 is observed that with less mutation rate (1%) the average clashes in the population are 5.82 whereas with high value of mutation rate (100%), the average clashes in the population are 3.8. So results with large mutation rate are better as compared to small mutation rate.

Conclusion and Future Scope

In this paper Genetic Algorithm is applied to solve N-Queen problem. The performance of GA is analyzed by varying population size, cross over rate and mutation rate. It is concluded from the results that

the performance of GA improves for high value of population size, cross over rate and mutation rate. However, the algorithm takes more time for high value of these parameters. In future the performance of GA can be analyzed for larger instances of N-Queen problem. The performance of GA can also be checked for solving other NP-Hard problem and GA can be compared with other hubristic techniques such as ACO, PSO and SA.

References

1. Robbi, Niki Min Hidayati, I. Wayan Mustika and Widyawan Widyawan. "A Modified Genetic Algorithm for Resource Allocation in Cognitive Radio Networks." 2018 4th International Conference on Science and Technology (ICST) (2018): 1-5.
2. B. V. Ha, M. Mussetta, P. Pirinoli and R. E. Zich, "Modified Compact Genetic Algorithm for Thinned Array Synthesis," in IEEE Antennas and Wireless Propagation Letters, vol. 15, pp. 1105-1108, 2016.
3. Heris, Jalal Eddin Aghazadeh and Mohammadreza Asgari Oskoei. "Modified genetic algorithm for solving n-queens problem." 2014 Iranian Conference on Intelligent Systems (ICIS) (2014): 1-5.
4. Turky, Ayad Mashaan and Mohd Sharifuddin Ahmad. "Using genetic algorithm for solving N-Queens problem." 2010 International Symposium on Information Technology2 (2010): 745-747.
5. Wang, Yuh-Rau, Hsieh-Liang Lin and Ling Yang. "Swarm Refinement PSO for Solving N-queens Problem." 2012 Third International Conference on Innovations in Bio-Inspired Computing and Applications (2012): 29-33.
6. Martinjak, Ivica and Marin Golub. "Comparison of Heuristic Algorithms for the N-Queen Problem." 2007 29th International Conference on Information Technology Interfaces (2007): 759-764.
7. Li, Tian and Sihai 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) (2017): 122-125.
8. Pan, Shing-Tai and Ching-Fa Chen. "Design of Stable IIR Filter by Improved Genetic Algorithm." 2012 Third International Conference on Innovations in Bio-Inspired Computing and Applications (2012): 122-126.
9. 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.
10. 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.
11. 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.
12. 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.
13. 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.
14. 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.
15. 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.
16. P. Roy and A. Chakrabarti, "Implementation of genetic algorithm and modified shuffled frog leaping algorithm for transmission loss minimum re-scheduling," 2011 Annual IEEE India Conference, Hyderabad, 2011, pp. 1-4.
17. M. Xu, S. Wang, J. Tao and G. Liang, "Research on Cooperative Task Allocation for Multiple UCAVs Based on Modified Co-evolutionary Genetic Algorithm," 2013 International Conference on Computational and Information Sciences, Shiyang, 2013, pp. 125-128.
18. C. Zeng, Q. Zhang and X. Wei, "Robotic Global Path-Planning Based Modified Genetic Algorithm and A* Algorithm," 2011 Third International Conference on Measuring Technology and Mechatronics Automation, Shangshai, 2011, pp. 167-170.
19. Yanqin Ma, "The Modified Hybrid Adaptive genetic algorithm for 0–1 knapsack problem," 2012 24th Chinese Control and Decision Conference (CCDC), Taiyuan, 2012, pp. 326-329.
20. Xu Yaoqun and Wei Wenjian, "Modified Adaptive Genetic Algorithm for Flexible Job-shop Scheduling Problem," 2010 International Conference on Future Information Technology and Management Engineering, Changzhou, 2010, pp. 52-55.