International Journal of Modern Computer Science (IJMCS)
Volume 5, Issue 3, June, 2017
ISSN: 2320-7868 (Online)

Parametric Analysis of Genetic Algorithm for Light Control System

Priyanka Sambyal
Department of Computer Science & IT
University Jammu
J&K, India
E-mail: priyanka46778@gmail.com

Pawanesh Abrol
Department of Computer Science & IT
University of Jammu
J&K, India
E-mail: pawanesh.abrol@gmail.com

Parveen Lehana
Department of Electronics
University of Jammu
J&K, India
E-mail: pklehana@gmail.com

Abstract: Genetic Algorithm (GA) is a heuristic search technique that provides solution to complex optimization problems. The implementation of GA for solving optimization problem involves tuning of various GA parameters such as size of initial population, mutation probability and number of generations. There are complex interactions among GA parameters during working of GA. Many researches have been done to find out the optimum values of the GA parameters using different techniques. But due to very problem specific nature of GA, it is still an open ended question for researchers to find global values of the GA parameters. The efficiency of GA mainly depends upon the GA parameters and a good selection of GA parameters results in better convergence of the results. This paper analyses the effect of parameters such as population size, mutation probability and number of generations on Genetic Algorithm for optimization of light control system.

Keywords: Fitness function, Genetic Algorithm, Light control system, Mutation probability, Optimization, Population size

I. INTRODUCTION

GA (Genetic algorithm) is a stochastic search optimization technique that is based on the mechanics of natural genetics. The use of GA is highly suggested where an optimal solution is to be mined from large solution space [1]. The term was first coined by John Holland in 1960’s [2]. It is a nature inspired technique and is based on the principle of Darwin theory of natural evolution-“survival of the fittest” [3]. It is an iterative process that starts with some initial potential solutions and transforms the set of solutions to get optimum results [4], [5]. Each individual solution is associated with a fitness value that represents the efficiency of that solution. Individuals with high fitness value are selected and allowed to breed for next generation [6], [7]. The process is repeated until some stopping criterion is reached. Because of the simplicity of GA it is widely used in various optimization problems like digital signal processing, robotics, automation systems, object recognition, circuit designing etc. [8]. Although there are various other evolutionary optimization techniques exist such as PSO, simulated annealing etc., but GA has scored over them in the context that GA always maintains a population of solutions rather than a single point solution. GA does not guarantee global optimum solutions but find acceptable solutions to various optimization problems [9].

Genetic algorithm is governed by various parameters that affect the performance of the algorithm [10]. These parameters include population size, number of generations, mutation probability, crossover probability etc. When an optimization problem is addressed using genetic algorithm, it is required to fix these GA parameters. The working of genetic algorithm involves massive interaction of these parameters. Genetic algorithm begins with some initial population and proceeds to new solutions for better results. This main procedure remains same for every implementation and the GA parameters are varied for better performance. Any change in the values of these parameters highly effect the output obtained. Inappropriate setting of these parameters may results in unexpected behaviour of the algorithm. So, for better convergence of the algorithm tuning of GA parameters must be done to optimum values. Due to the application specific nature of GA it is very difficult to understand the behaviour of these parameters [11]. Different techniques have been implemented by various researchers to understand the behaviour of these parameters but still it is not fixed. By testing the algorithm for different parameter values some optimum values can be obtained.

Today the world is facing a big problem of energy crisis. In recent days, it had been seen that total energy consumption is growing faster than the current manufacturing capacities. This excessive utilization of light can be reduced by optimizing the light system [12]. If the light system is optimized in such a way that light is available only at the time of requirement than there will be reduction in wastage of light [13], [14]. Generation of different illumination patterns of lamps according to the requirement may results in optimization of light control system. In general, optimization is the process of finding the best solution under various given constraints. Optimization techniques results in increase of profit and reduction of cost. In context of the light control system, optimization refers to the optimized light system design that results in reduction of energy wastage and also provides sufficient energy according to need. In this paper, optimization of light control system is performed using Genetic algorithm. Because of simplicity and better performance, Genetic algorithm is used as an optimization technique. For experimentation different variables are identified that greatly affects the light control system. Based on the variables identified, fitness function is formulated that evaluates fitness of the solution. This fitness function is used by Genetic algorithm for estimating the fitness of solutions.

The main contribution of the paper is threefold; firstly, identification of various variables affecting light control system and based on the identified variables a fitness function has been formulated. Secondly, Genetic algorithm is implemented on the designed cost function. Lastly, parametric analysis of GA parameters is performed for obtaining better results.

The rest of the paper is structured as follows. Section II highlights the basic working of Genetic algorithm and GA parameters. Section III gives the major insights of the proposed methodology. Experimentation and results are analysed in Section IV. Finally, the whole paper is concluded in Section V.

II. Genetic Algorithm and Parameters

Genetic algorithm is a search technique that generates population of solutions and allows them to evolve to reach final optimum solution. The basic steps of genetic algorithm are initialization, fitness evaluation, selection, crossover and mutation as shown in Figure 1.

Initialization: It is the first step of GA in which a set of population is initialized. The initial population represents some feasible solution to the problem. This process is random process in which a set of population is initialized in the search space.

Fitness evaluation: In order to select which individual will breed to produce next generation, the fitness of every solution is evaluated. The fitness of solutions is evaluated using a fitness function. Depending upon the optimization problem, a fitness function is formulated that evaluates fitness of solutions.

Selection: In selection step parents are selected from set of population to produce new generation [15]. The fitness values of each solution is analyzed and arranged in ascending order. The individuals with higher fitness values are selected as parents that are allowed to breed for next generation.

Crossover: The process of reproduction of selected parents to generate new generation is called crossover. The selected parents are allowed to produce new offspring by crossing over their genes.

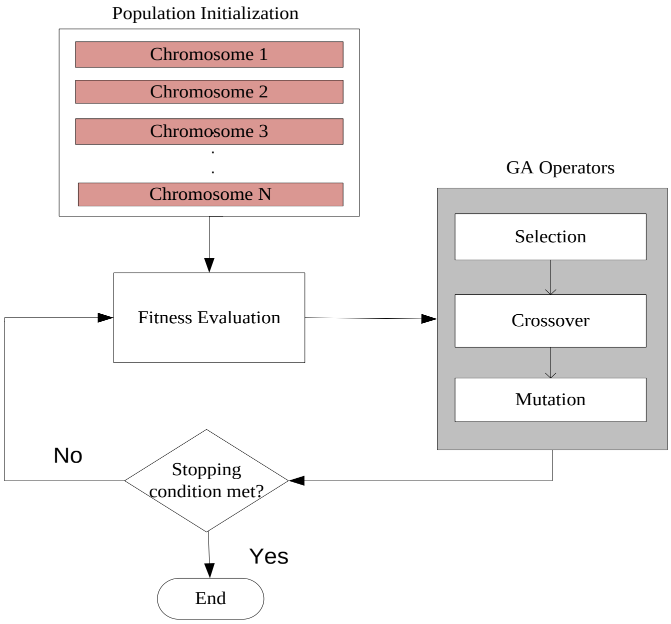


Figure1. Basic Steps of Genetic Algorithm

Mutation: In order to enable GA to explore new search space areas, mutation is performed [16]. Mutation also prevents GA solutions to fall in local maxima. Usually, mutation is incorporated in the solutions that have lowest fitness values.

The basic steps of GA remains same for every system, it is the values of parameters that changes. GA has various parameters that affect the performance of the algorithm. A different combination of these parameters leads to a drastic change in the results. GA parameters are divided into two types: structural and numerical parameters.

Structural Parameters: As the name implies, it is concerned with the structure of the algorithm. Any change in these parameters results in many changes in the algorithmic structure. These parameters include operator types, coding schemes and stopping criterion

Numerical Parameters: This class of parameters includes population size, number of generations and probability of mutation. Although these parameters do not affect the structure of the algorithm but has significant impact on the algorithm performance. Many researches have been done to find out the optimal values of these algorithm but no final conclusions are obtained.

III. Proposed Methodology

The major components of the proposed methodology are covered in this section. The methodology is divided into two components. Initially, different variables are identified for the light control system and depending upon these variables a fitness function is formulated to achieve the objective. Secondly, GA has been applied to obtain the optimized pattern.

A. Identification of variables and formulation of fitness function

A fitness function has been formulated to evaluate the fitness of the solutions. Initially, different variables that affect the performance of light control system have been identified. Depending upon these variables a fitness function is formulated:

$$f = Max\left(\frac{\sum_{i=1}^{n} (A_i/TA)^2 * \sum_{i=1}^{n} PR_i}{\sum_{i=1}^{n} E_i * \sum_{i=1}^{n} NP_i} \right)$$

Where: f is the fitness function, A is the area illuminated by a pole, TA is the total area taken, PR is the priority given to pole, E is the energy consumed by a pole and NP is the total number of poles illuminated in a pattern. The objective function is of maximization type i.e. it tries to maximize the fitness value.

B. Implementation of Genetic Algorithm

The objective function that has been designed is implemented using GA. The sequence of the salient steps that are performed is shown in Figure 2. Solutions are represented as chromosomes and each chromosome consists of genes that represent the poles. The solutions are encoded in binary form i.e. solutions are the sequence of 0's and 1's. Following are the steps performed to achieve the objective:

1. Define variables and fitness function.
2. Initialize GA parameters:
   P: Population size
   N: Number of generations
   PM: Probability of Mutation
3. Randomly generates an initial population P of solutions.
4. For generations 1 to N do the following:
   4.1 Evaluate fitness for every DNA using Fitness function.
   4.2 Sort DNA in descending order of their fitness values.
   4.3 If the generation limit (N) has reached then exit main loop and return the best fit solution.
   4.4 Generate the next generation by following steps a) to c).
   a) Selection: Select the first DNA in the population which is having the best fitness value as Parent1 and randomly select the other DNA from the population as Parent2.
   b) Crossover: The two selected parents are then allowed to perform crossover. The crossover technique used is single point crossover. A random crossover point is generated about which crossover is performed. Two new offsprings are generated from the selected parents by exchanging the genes about crossover point.

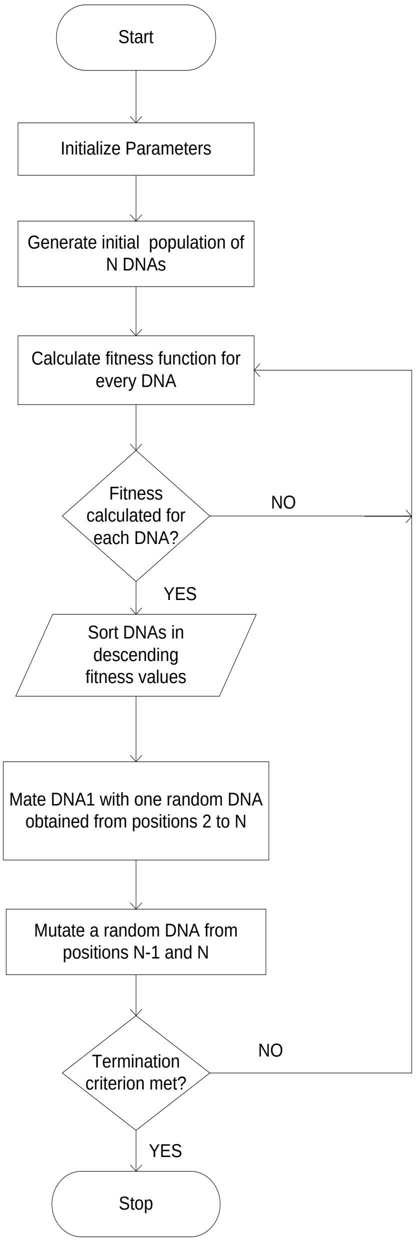


Figure2. Flowchart of Proposed Methodology

   c) Replacement: Replace the two lowest fit individuals with new generated offsprings to generate new generation. The next generation contains all the DNA of previous generation except the last two solutions is replaced by new offsprings.
   4.5 Mutation: Perform mutation on one of the two randomly generated offsprings in new population obtained at step 4.4 (b). A random position is generated and the gene value at that position is mutated.
   4.6 Repeat Step 3 with the new population generated.

In order to test and analyze the algorithm, data has been collected from location with 50 poles. Three different types of poles Type 1, Type 2 and Type 3 are considered with area of illumination as 10m, 20m and 30m respectively and wattage as 50W, 70W and 100W respectively.

IV. RESULTS AND DISCUSSION

Different experiments have been conducted for analyzing the behavior of GA parameters- number of generation, population size and mutation probability. Initially all the basic steps of GA are implemented by setting number of generations to 10000, initial population size is fixed to 50 and mutation probability is 0.03. The value of the objective function is calculated using fitness function and recorded for 1, 50, 100, 500, 750, 1000, 2500, 5000, 8000 and 10000 generations. Genetic algorithm stabilizes the fitness value after some iteration. From Table1 it can be seen that the fitness value stabilizes after 5000 generations. The algorithm is converged to optimum solution at 5000 generation.

Table1. Variation in Fitness Value at different Iterations

Iteration No.	Fitness value	Area (%)	No. of poles illuminated	Energy Efficiency (%)	Priority
1	0.49	67	30	36	183
50	0.61	78	35	26	219
100	0.63	76	33	29	217
500	0.63	74	32	30	216
750	0.65	71	30	34	213
1000	0.66	70	29	36	212
2500	0.71	60	26	40	215
5000	0.73	70	28	37	230
8000	0.73	70	28	37	230
10000	0.73	70	28	37	230

The variation in the value of variables at different iterations is also shown in Table1. For iteration 1 the percentage of area illuminated is 67 percent with 30 poles and the energy efficiency is 36%. For iteration 50, 78% of area is illuminated with 35 poles and energy efficiency is 36%. Similarly, moving from iteration 50 to 5000 the value of the fitness function is maximizing and the values of the variables are varying accordingly. The value of the fitness function is converged at 5000 generation. After 5000 generations the value of the objective function is stabilized. So, the final values of variables are: 70% of area is illuminated with 28 poles and the energy efficiency is 37%.

To improve the performance of the algorithm, different parameter values are tested. Firstly, the effect of mutation on convergence of solution for different population size is analyzed. The number of generations is fixed to 10000 generations and the fitness values are calculated for iteration number 1, 50, 100, 300, 500, 750, 1000, 2500, 5000, 8000 and 10000. Investigations are made for population size of 50, 100, 250, 500, 750 and 1000. Each of the population size is tested for mutation probability of 0.03, 0.05, 0.08, 0.3, 0.5 and 0.8.

Genetic algorithm converge the solution after some number of generations. In Table II the effect of mutation probability is analyzed for convergence of the solution. It can be seen that when population size is 50, for probability of mutation 0.03, 0.05 and 0.08 the solution converges after 5000 generations. For mutation probability as 0.3 and 0.5 the solution converges after 750 generations and for 0.8 the solution converges after 500 generations. Similarly, for population size 100, the number of generations required to converge solution for probability of mutation 0.03, 0.05 and 0.08 is 5000. For 0.3 the solution converges after 1000 generations and for 0.5 the solution converges after 750 generations. It can be clearly seen that a high value of mutation probability converges the solution for less number of generations. For mutation probabilities 0.03, 0.05 and 0.08 the number of generations required to converge the solution is nearly 5000 which is a high value. Thus, it can be seen that a high mutation probability leads to faster convergence of the solution. Figure3 graphically shows the effect of mutation probability on convergence of the solutions.

Table2. Effect of Mutation on convergence of solution

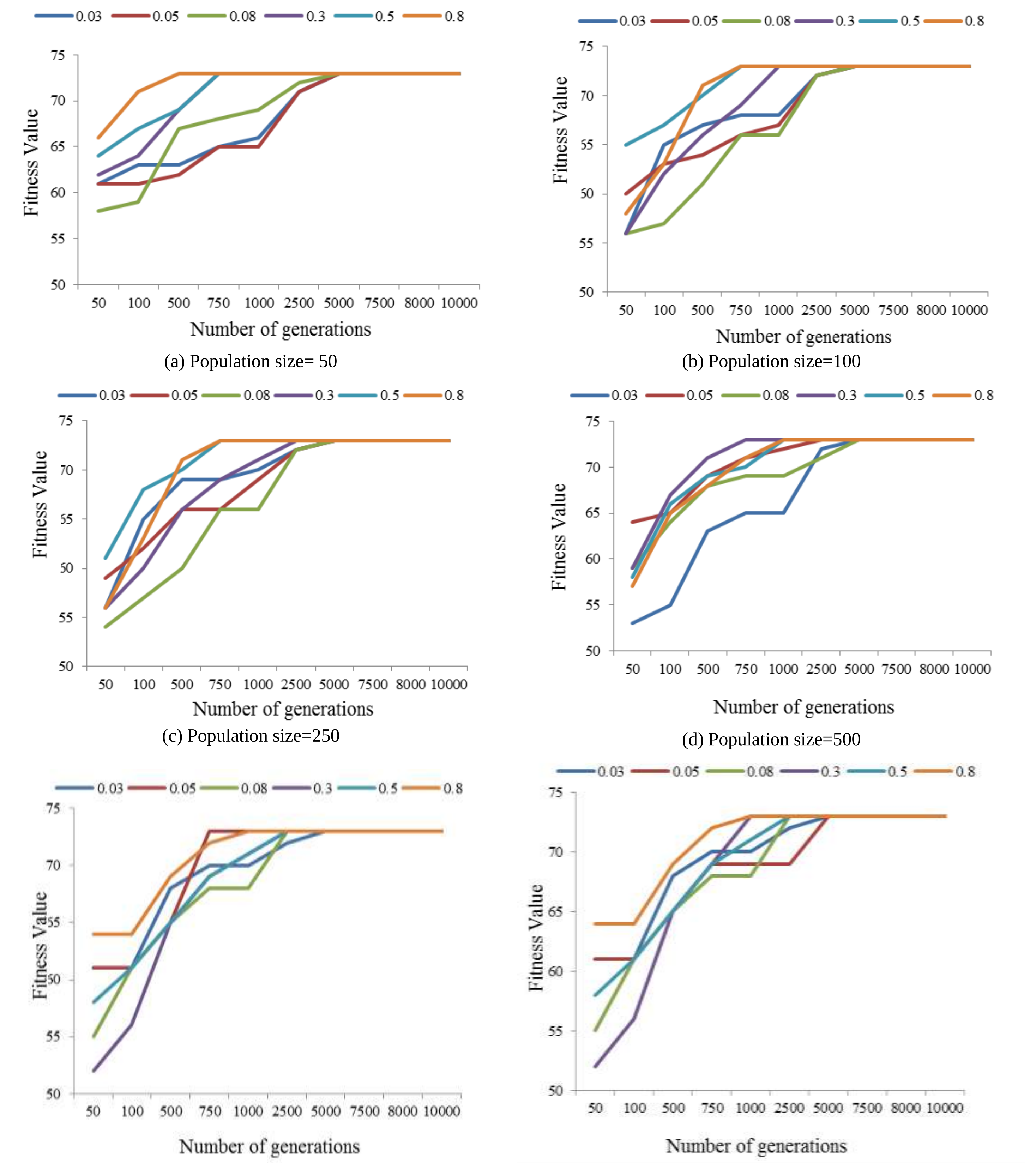
Population Size	Mutation Probabilities					
	0.03	0.05	0.08	0.3	0.5	0.8
50	5000	5000	5000	750	750	500
100	5000	5000	5000	1000	750	750
250	5000	5000	2500	2500	750	750
500	5000	2500	5000	750	1000	1000
750	5000	750	2500	2500	2500	1000
1000	5000	5000	2500	1000	2500	1000

The effect of population size on CPU time is also investigated. Large population size increases the computation of the solution so the time taken by algorithm may increase. Table3 shows the effect of population size on CPU time. The CPU time for population size 50, 100, 250, 500, 750 and 1000 have been calculated. It can be seen that population size of 50, 100 and 250 takes less CPU time i.e. less than minute. While as population size is increased from 500 generation to 1000 CPU time increases.

Table3. Effect of Population size on CPU Time

Population size	50	100	250	500	750	1000
Time(Sec)	12	20	50	92	120	200

A high value of population size increases the execution time. From figure3 and Table3 it can be analysed that population size greater than 250 have almost same impact of mutation probabilities as for value less than 250. Moreover, a high value of population size just increases the execution time. So, in context of CPU time low population size may result in better convergence of result in less time.



(a) Population size= 50

(b) Population size=100

(c) Population size=250

(d) Population size=500

(e) Population size=750

(f) Population size=1000

Figure3. Effect of Mutation Probability on number of generations required to converge solution for different population sizes

V. Conclusion

Genetic algorithm is a powerful optimization technique that provides solutions to complex optimization problems. In this paper optimization of light control system is performed using genetic algorithm. Different variables affecting the design of optimized light control system has been identified and a fitness function is formulated to calculate the fitness of the solution. To enhance the performance of GA an analysis of different parameters of GA has been performed. Selection of suitable value of GA parameters results in better convergence of the results. Due to the application specific nature of GA, it is very difficult to draw global values of GA parameters. So, investigations are carried out for better convergence of the solution by testing different values of mutation probability, population size and number of generations. It can be observed that increasing the probability of mutation results in better convergence of results. Also, the effect of population size on CPU time is analysed. The results obtained illustrate that increase in population size results in increase of CPU time.

REFERENCES

[1] Chaahat, M. Bahl, P. Lehana, and S. Kumari, "Image brightness enhancement of natural and unnatural images using continuous genetic algorithm," International Journal of Advanced Research in Computer Science and Software Engineering, Vol. 3, Issue 9, September 2013.

[2] P.K. Yadav and Dr. N.L. Prajapati, "An overview of genetic algorithm and modelling," International Journal of Scientific and Research Publications, Vol. 2, Issue 9, September 2012.

[3] R. Garg, and S. Mittal, "Optimization by genetic algorithm," International Journal of Advanced Research in Computer Science and Software Engineering, Vol. 4, Issue 4, April 2014.

[4] R. Neethu and S. Vydehi, "Overview of different text similarity methods in research document," International journal of modern computer science, Vol.4, Issue 4, August 2016.

[5] P. Thakur and M. Mahajan, "Different scheduling algorithm in cloud computing: a survey," International Journal of modern computer science, Vol. 5, Issue 1, February 2017.

[6] H. Kour, P. Sharma, and P. Abrol, "Analysis of fitness function in genetic algorithms," International Journal of Scientific and Technical Advancements, Vol. 1, Issue 3, pp. 87-89, 2015.

[7] M. Saraswat, and A.K. Sharma, "Genetic algorithm for optimization using matlab," International Journal of Advanced Research in Computer Science, Vol. 4, March 2013.

[8] P. Sambyal, P. Abrol, and P. Lehana, "Optimization of light switching pattern on large scale using genetic algorithm," International journal of Scientific and Technical Advancements, Vol. 3, Issue 1, March 2017.

[9] A. Sharma, and Sanjay, "Study of genetic algorithm and implementation of various test function using crossover operator," International Journal for Research in Applied science & Engineering Technology, Vol. 5, Issue V, May 2017.

[10] R. Kumar, S. Gill and A. Kaushik, "A crossover probability distribution in genetic algorithm under process scheduling problem," International Journal of Advanced Research in Computer Science, Vol.1, November-December 2010.

[11] M. Kaur, "An analysis of genetic parameters and operators for knapsack problem," International Journal of Advanced Research in Computer Science and Software Engineering, Vol. 6, Issue 1, January 2016.

[12] A. W. Levy, "Lighting controls, patterns of lighting consumption, and energy conservation," IEEE Transactions on Industry Applications, Vol. IA.16, No. 3, June 1980,pp. 419-427.

[13] G. F.M. Lima, I. S.Peretta, and A. Cardoso, "Optimization of lightning design using genetic algorithm," IEEE Xplore, December 2010.

[14] A. Mattoo, P. Abrol, and P. Lehana, "Server hardware design and LUA based implementation using wi-fi based risc processor," International Journal of Scientific and Technical Advancements, Vol.3, Issue 1, March 2017.

[15] M. Srinivas, and L. M. Patnaik, "Adaptive probabilities of crossover and mutation in genetic algorithms", IEEE Transactions on systems, man and cybernetics, Vol. 24, No.4, April 1994.

[16] K. F. Man, K. S. Tang, and S. Kwong, "Genetic algorithms: concepts and applications," IEEE Transactions on Industrial Electronics, Vol. 43, No. 5, October 1996.