New Trends in Mathematical Sciences
http://dx.doi.org/10.20852/ntmsci.2016.111

Performance of population size on Knapsack problem

David Oyewola, Danladi Hakimi, Amuda Yusuph Yahaya and Gbolahan Bolarin

Department of Mathematics and Statistics, School of Physical and Applied Sciences, Federal University of Technology, Minna, Niger State, Nigeria

Received: 22 June 2016, Accepted: 7 October 2016
Published online: 21. November 2016.

Abstract: In this paper, an investigation of a performance of population size on the genetic algorithm (GA) for a knapsack problem is considered. Population sizes between 10 and 200 chromosomes in the population are tested.

In order to obtain meaningful information about the performance of the population size, a considerable number of independent runs of the GA are performed. Accurate model parameters values are obtained in reasonable computational time. Further increase of the population size, does not improve the solution accuracy. Moreover, the computational time is increased significantly.

Keywords: Population, parameter, chromosome, genetic algorithm, Knapsack problem.

1 Introduction

Genetic Algorithms (GA) have been studied to solve different problems since they are effective in solving problems that are non-linear and multiple objective problems. [8] applied Genetic Algorithms to a problem in assembly line balancing problem with parallel Workstation.

H.N.Al-Duwaish, [9] used a genetic algorithm to a linear dynamic systems with static nonlinearities. Genetic Algorithms approach was also used by [10] for the identification of time delay. Genetic Algorithm consists of different parameters which are population, crossover, mutation and generation. Different researchers has contributed immensely on the different parameters in genetic algorithms.

M.Arndt and B.Hitzmann, [11] on their paper titled ”feed forward feedback control of glucose concentration during cultivation of Esherichia coli” suggested that a small population size can lead the algorithms to a poor solutions.

C.R.Reeves, [1] use small population in genetic algorithm. He observed that small population could result to a local optima solution of the problem. V.K.Koumousis and C.P.Katsaras, [3] combined the effects of variable population size and reinitialization in order to enhance the performance of population in genetic algorithm.

O. Roeva [2], in his paper emphasized that population size influence the quality of the optimal solution. He then explained that more research should be done for the GA parameter(population).

The main purpose of this research is to investigate the performance of genetic algorithms on Knapsack problem. Knapsack problem is an NP hard problem and the decision problem.

* Corresponding author e-mail: davidakaprof01@yahoo.com

2 Preliminaries

In this section, the model and theoretical background of 0-1 knapsack problem and the genetic algorithms were introduced.

2.1 Knapsack problem

In the Knapsack Problem (KP), each item x_j has a profit p_j . Each x_j has a weight $w_j$that depends on the knapsack j . A selected object must be in all knapsacks.

The objective in KP is to find a subset of objects that maximize the total profit without exceeding the capacity of the knapsack. The equation (1) below can also be formulated as a Knapsack Problem(KP) as follows.

$$Max_x \sum_{j=1}^{n} p_j x_j \tag{1}$$

$$S.t \sum_{j=1}^{n} w_j x_j \leq C \tag{2}$$

where p =Profit, w =weight, n =number of items from which to choose, $x = 0,1,\ j = 1\ to\ n$.

2.2 Theoretical background of genetic algorithms(GA)

Genetic Algorithm (GA) is a model adaptation of genes in the body. It consists of representing x and y chromosomes as a binary strings (0 or 1) and also using crossover and mutation operation as being use in genetics. The following are the parameter used in genetic algorithms.

(i) Population: Is the combination of various chromosomes. In chromosomes, each gene controls a particular characteristic of the individual, The genes are represented as 0 or 1. In a chromosome, the genes are represented as in figure below.

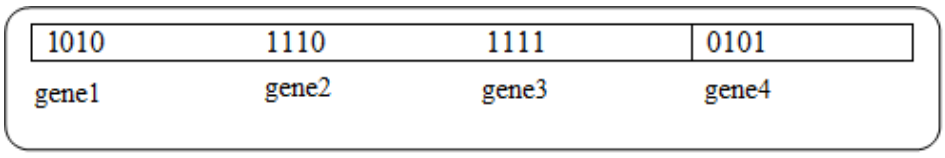

Fig. 1: Binary representation of chromosomes.

(ii) Crossover: Crossover operators is used to divide a pair of selected chromosomes into two or more parts. It consists of combining the chromosomes of two parents to produce a new offspring (child). Example of a single point crossover is given below

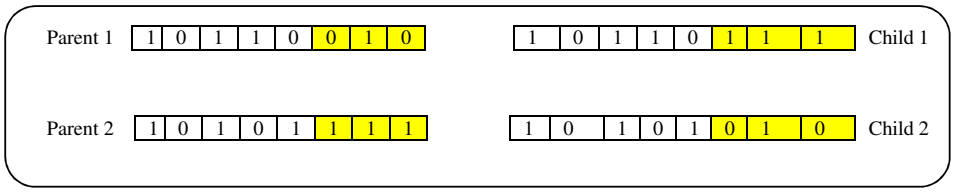

Fig. 2: Single Point Crossover.

(iii) Mutation: Is the changing of chromosomes. A commonly used method for mutation is shown below.The shaded region changes from 0 to 1 or vice versa

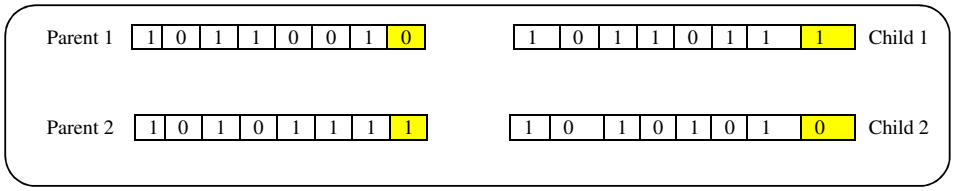

Fig. 3: Single Point Mutation.

(iv) Fitness evaluation of population: fitness function quantifies optimality of chromosome.

(v) Ranking of Chromosome: It consists of a ranking of the best chromosome from the population.sum of ranks is computed and then selection probability of each individual is computed as below:

$$rsum_i = \sum_{i=1}^{ngen} r_{i,j} \tag{3}$$

$$PRANK_i = \frac{r_{i,j}}{rsum_i} \tag{4}$$

where i varies from 1 to ngen, $rsum_i$ =sum of ranks in the generation, $r_(i,j)$ =rank of jth individual in ith generation for rank selection and $ngen$ =number of generation.

(vi) Survival Selection: Is the selection of the higher fitness value.

3 Results

Computational results are performed using Intel(R) Pentium(R) Dual CPU T3200 @ 2.00GHz, 3067MB Memory and MATLAB 7.9.

A series of numerical experiments are performed on the population size in GA using different population size range from 10 to 200. We also use a mathematical model of equations 1 and 2 which consists of 20 dimension problem [12] and varies the population size on it. The number of generation (iteration) is fixed at 200. The GA parameters used in this paper are in section 2.2. The GA algorithms were run ten times.

Table 1 below shows the algorithms run of the objective function, population size, average and standard deviation while

Table 2 shows the computational times for each run of the population size, mean and standard deviation. We observed from table 1 that increase the size of the population from 10 to 100 chromosomes improves the objective function value from 987.8-1023.8 but further increases of the population does not increase the value of the objective function. From Table 2 there was a significant increase in computational time without improving the value of the Objective function from 100-200 chromosomes.

Table 1: Performance of objective function.

Population Size	Objective function	
	Average(Avg)	Standard Deviation(Std)
10	987.8	23.0497
30	1004.2	15.6901
50	1006.7	14.3453
60	1014.8	10.4754
80	1018.5	5.1908
90	1017.1	8.4123
100	1023.8	0.6325
120	1023.7	0.6749
150	1023.9	0.3162
200	1024	0.0000

Table 2: Performance of computational time

Population Size	Computational time	
	Average(Avg)	Standard Deviation(Std)
10	0.1814	0.0092
30	0.2845	0.1462
50	0.3316	0.1134
60	0.3190	0.0247
80	0.4012	0.0568
90	0.4695	0.0635
100	0.4285	0.0279
120	0.5240	0.1479
150	0.5775	0.0334
200	0.6324	0.0324

For a better illustration of the numerical results, the Figure 4 shows the algorithm run, population size and Objective function while Figure 5 is the population size, algorithm run and computational time.The Figure 4 illustrated that increase in the population size result in an acceleration of the objective function from the population size of 10 to 100 and further increases in the population size from 100 to 200 does not increase the optimal solution. Moreover, in Figure 5 increase in the population size result in an accelerating increase in the computational time.

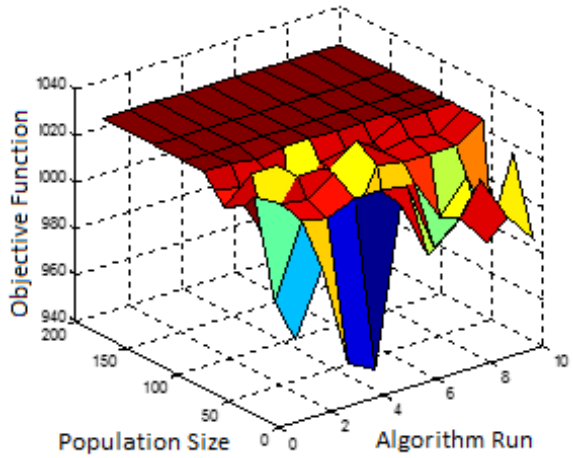

Fig. 4: 3D graph of objective function.

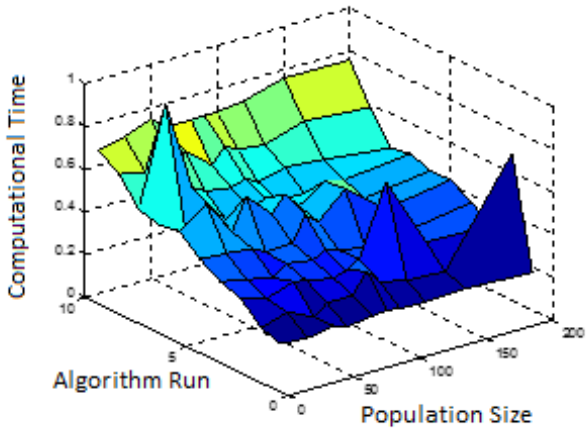

Fig. 5: 3D graph of computational time.

4 Conclusion

Genetic Algorithms is a meta-heuristics approach to solve Knapsack Problem. In this paper, the performance of one of key GA parameters (population size) on the GA performance, is studied. As a test problem, the Knapsack problem model is considered. Different population sizes(10 to 200) of the GA are explored. Thus, accurate model parameters values are obtained with reasonable computational efforts. We observed that the use of smaller populations resulted in a lower accuracy of the solution, obtained for a smaller computational time. The further increase of the population size increases the accuracy of the solution. The use of larger populations does not improve the solution accuracy but only increase the needed computational resources.

Competing interests

The authors declare that they have no competing interests.

Authors' contributions

All authors have contributed to all parts of the article. All authors read and approved the final manuscript.

References

[1] C. R. Reeves, *Using Genetic Algorithms With Small Populations*,In Proceedings of the Fifth International Conference on Genetic Algorithms, (1993), pp. 92-99.

[2] O. Roeva and Ts. Slavov, *Fed-batch Cultivation Control based on Genetic Algorithm PID Controller Tuning*, Lecture Notes on Computer Science, Springer-Verlag Berlin Heidelberg, Vol. 6046, (2011), pp. 289-296.

[3] V. K. Koumousis and C. P. Katsaras, *A sawtooth genetic algorithm combining the effects of variable population size and reinitialization to enhance performance*, IEEE Transactions on Evolutionary Computation,Vol. 10, No. 1, (2006), pp. 19–28.

[4] M. Pelikan, D. E. Goldberg, and E. Cantu-Paz, *Bayesian optimization algorithm, population sizing, and time to convergence, Illinois Genetic Algorithms Laboratory*, University of Illinois, Tech. Rep., (2000).

[5] A. Piszcz and T. Soule, *Genetic programming: Optimal population sizes for varying complexity problems*, In Proceedings of the Genetic and Evolutionary Computation Conference, (2006), pp. 953–954

[6] F. G. Lobo and D. E. Goldberg, *The parameterless genetic algorithm in practice, Information Sciences Informatics and Computer Science*, Vol. 167, No. 1-4, (2004), pp. 217–232.

[7] F. G. Lobo and C. F. Lima, *A review of adaptive population sizing schemes in genetic algorithms, In Proceedings of the Genetic and Evolutionary*, Computation Conference, (2005), pp. 228–234.

[8] S.Akpinar and G.M. Bayhan, *A Hybrid Genetic Algorithm for mixed model assembly line balancing problem with parallel workstations and zoning constraints*, Engineering Applications of Artificial intellligence. Vol. 24, No 3 (2011), pp. 449-457

[9] H.N.Al-Duwaish, *A Genetic Approach to the Identification of Linear Dynamic Systems with Static Nonlinearities. International Journal of Systems Science*, Vol. 31, No. 3, 2000, pp. 307-313.

[10] J.P.Paplinksi, *The Genetic Algorithm with Simplex Crossover for Identification of Time Delays, Intelligent Information Systems*, 2010, pp. 337-346.

[11] M. Arndt and B.Hitzmann, *Feed Forward/Feed back Control of Glucose Concentration during Cultivation of Escherichia coli*, 8^{th} IFAC Int. Conf. on Comp. Appl. in Biotechn, Canada, 2001, pp. 425-429.

[12] Kaushik Kumar Bhattacharjee, S.P.Sarmah, *Shuffled frog leaping algorithm and its application to 0/1 Knapsack problem*, Applied Soft Computing, 19(2014) 252-263

[13] Vasquez, M. & Hao, J. K. *A logic-constrained knapsack formulation and a tabu algorithm for the daily photograph scheduling of an earth observation satellite*.Comput. Optim. Appl.,20, 137–157, (2001).

[14] Vincent, Boyer, Didier, El Baz and Moussa Elkihel, *Solution of multidimensional knapsack problems via cooperation of dynamic programming and branch and bound*, European J. Industrial Engineering, Vol. 4, No. 4,(2010).

[15] Wagner, H.M., Principles of Operations Research, Prentic-Hall, Inc,(1969).

[16] Yamada, T. & Futakawa, M. *Heuristic and reduction algorithms for the knapsack sharing problem*. Comput. Oper. Res.,24, 961–967,(1997).