Volume XXI 2018
ISSUE no.1
MBNA Publishing House Constanta 2018

Scientific Bulletin of Naval Academy

SBNA PAPER • OPEN ACCESS

Genetic algorithms - variable size populations of chromosomes. An adaptive approach

To cite this article: R Maniu and M V Manoliu, Scientific Bulletin of Naval Academy, Vol. XXI 2018, pg. 393-399.

Available online at www.anmb.ro

ISSN: 2392-8956; ISSN-L: 1454-864X

doi: 10.21279/1454-864X-18-I1-059

Genetic algorithms - variable size populations of chromosomes. An adaptive approach

R Maniu and M V Manoliu[1]

[1]Transilvania University, Brasov, Romania

E-mail: rares.maniu@yahoo.com, manoliu_mit@yahoo.com

Abstract. The size of the chromosome population is an essential parameter of genetic algorithms. A large population involves a large amount of calculations but provides a complete scroll of the search space and the increased probability of generating a global optimum. A small population size, through the small number of operations required, causes a quick run of the algorithm, with increasing the probability of detecting a local optimum to the detriment of the global one. This paper proposes the use of an adaptive, variable size of chromosome population. We will demonstrate that this approach leads to an acceleration of the algorithm operation, without having a negative impact on the quality of provided solutions.

1. Introduction

As the component of evolutionary computation, genetic algorithms are a class of adaptive techniques which search for the heuristic solutions of a given problem. The results are generated by refining a multitude of potential solutions grouped across generations with the help of genetic operators like selection, mutation, and crossover that ensure that the potential solutions space is scanned for identifying the set of optimal results.

Although genetic algorithms can solve a wide variety of provided problems, they must be encoded and have a function able to measure the quality of the solutions (fitness function). These algorithms also have a number of disadvantages. Generally, they are slower than the deterministic ones in solving the same problem, the scaling of the problem leads to an increased convergence time, and they have a dispersion of the solutions generated in several runs of the genetic algorithm on the same set of data. Because it is not known if the solution is global optimal, it is not possible to define exactly the time of stop for a genetic algorithm and the produced solutions may be local optimal and not necessarily global optimal.

However, genetic algorithms are used because of the benefits that they offer, making a good combination between exploring and exploiting the solutions space. They are a combination of a directed search and a random search, offering at any time a number of possible solutions and not only one at the end (as with the deterministic techniques) and they are easy to implement and use. The fitness function should not have some special features (continuity, derivability, convexity, etc.), and there is necessary just for a function to describe the problem.

The optimization of genetic algorithms involves both optimizing the quality of offered solutions and optimizing the running time. This paper proposes the optimization of running time by modifying the size of populations of potential solutions (chromosomes). Since the size of the population has a direct implication on running time, due to the number of mathematical operations required, the reduction in the number of chromosomes leads to a faster generation of solutions. To realize a complete search of solution space, a larger number of chromosomes is required at the first iterations of the algorithm, and as it converges, the number of chromosomes can be reduced because they will be placed in a limited area, around optimum.

2. Related work

Optimizing the functioning of genetic algorithms by using variable population populations has been approached in various ways.

In [1], the authors proposed the Genetic Algorithm with Variable Population Size (GAVaPS), which introduces the "age" and "lifetime" concepts for chromosomes. As a chromosome survives over generations, the "age" parameter is incremented. The "lifetime" parameter specifies the maximum time that the chromosome exists in the population. In this approach, there is no self-standing selection operator, each individual having the same chance of being chosen for reproduction, the actual selection being made according to the "lifetime" and "age" parameters. Well-adapted chromosomes will have a higher "lifetime". Through this mechanism, as the number of generations increases, the chromosome population will decrease, keeping the best solutions.

In [2] three independent genetic algorithms are used, each with its own populations, these being modified so as to optimize the intermediate population. At regular intervals (called epochs), the best fitness is used as a criterion for changing the population size based on predefined rules, searching to obtain an optimal number of chromosomes.

In [3] authors propose a type of genetic algorithm in which the population size changes according to the evolution of the best fitness in that population. The number of chromosomes will increase when the fitness decreases or when it does not change over many generations.

The paper [4] proposes an adaptive variation of population size depending on the difficulty of generating chromosomes that are better adapted than parents. The variable population is complemented by the proposed selection [5] to maximize fitness enhancement while preserving diversity. Only chromosomes with fitness higher than that of parents will be accepted in the offspring population and only if they contain sequences of alleles that no longer exist in the other members of the population.

In [6], authors developed a genetic algorithm with variable populations dependent on the fitness value of that population and the "lifetime" parameter that sets the maximum survival time of a chromosome.

3. Adaptive population size

Taking in consideration that the size of chromosomes population determines the running time of the algorithm, in this paper it is proposed to reduce it by using populations with a variable number of chromosomes. At initialization, the number of chromosomes of the genetic algorithm is maximal, to ensure a complete solution space search. As the algorithm evolves towards the convergence state, the chromosome population can be reduced, which determine the algorithm to perform faster.

We intend to realize the adaptability of the number of chromosomes depending on their position in the search space, using the notion of central chromosome described in [7]. This chromosome is defined as an average of all chromosomes in a population and is not necessarily a valid solution to the problem addressed by genetic techniques, being defined as:

$$m_i = [m_i(0), m_i(1), \ldots, m_i(\mathrm{n})] \qquad [7] \tag{1}$$

where

$$m_i(k) = \sum_{j=0}^{N} x_j(k) / N \qquad [7] \tag{2}$$

N is the number of chromosomes in generation,

x(k) is the k-th allele.

When the algorithm is convergent, the distance between this central chromosome and the real chromosomes is lower.

The distance between two chromosomes is calculated using this formula:

$$dist(x_k, y_k) = \sum_{i-0}^{N} |x_k(n) - y_k(n)| \qquad [7] \qquad (3)$$

In which $x_k(n), y_k(n)$ are k-generation chromosomes and N parameter is the length of chromosomes.

Diversity, defined in relation to this central chromosome, is calculated as the difference between it and the chromosomes of the population. At convergence, this diversity has a lower value, because almost all chromosomes are placed in a narrow area of search space. After initialization of the algorithm, when the initial population of chromosomes are distributed in the entire of search space, the diversity has the maximum value. As the genetic algorithm moves towards convergence, diversity decrease. Considering this aspect, the variation in the number of proposed chromosomes is dependent on the ratio between the diversity of the current population and the maximum diversity, being of:

$$N' = N(1 - {d_{current}}/{d_{initial}}) \qquad (4)$$

Where N' is new population size, N – initial population size, $d_{current}$ is current diversity and $d_{initial}$ represents initial diversity.

4. Experimental results

To study the effects of population chromosome variation based on their position in search space, three variants of genetic algorithms have been implemented: a classical genetic algorithm (fixed population size, random mutation, fitness-based selection, crossover at the half of the chromosome), a genetic algorithm with adaptive population size, according to the evolution of fitness, in line with the formula:

$$N' = N(1 - {f_{initial}}/{f_{current}}) \qquad (5)$$

(Where N' is new population size, N – initial population size, $f_{current}$ is current population fitness and $f_{initial}$ represents initial population fitness) and a genetic algorithm with adaptive population size based on diversity, according to formula (4).

The problem is to determine the path with minimum cost in a graph.

The tests were performed for the initial generations of 25 and 35 chromosomes, the same graph, the same percentage of mutation, and the same crossover and selection operators were used. Here is measured the evolution of fitness and running time.

For populations of 25 chromosomes, the evolution of fitness is represented in Figure.1. the evolution of the number of members of the populations in Figure.2., the evolution of running time in Figure.3. and the minimum fitness average obtained in the 10 runs of the algorithms in Figure.4..

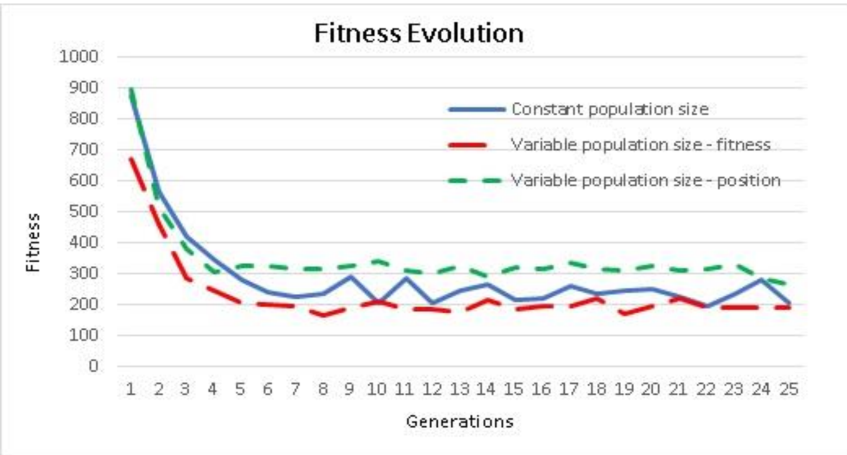

Figure.1. Fitness evolution

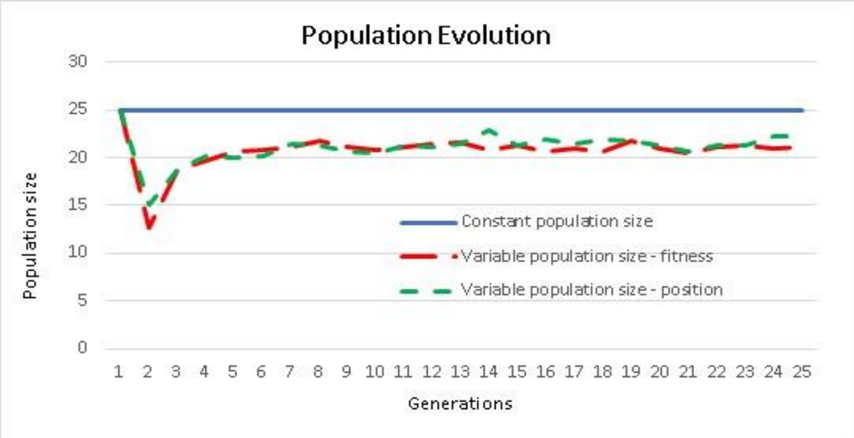

Figure.2.Population evolution

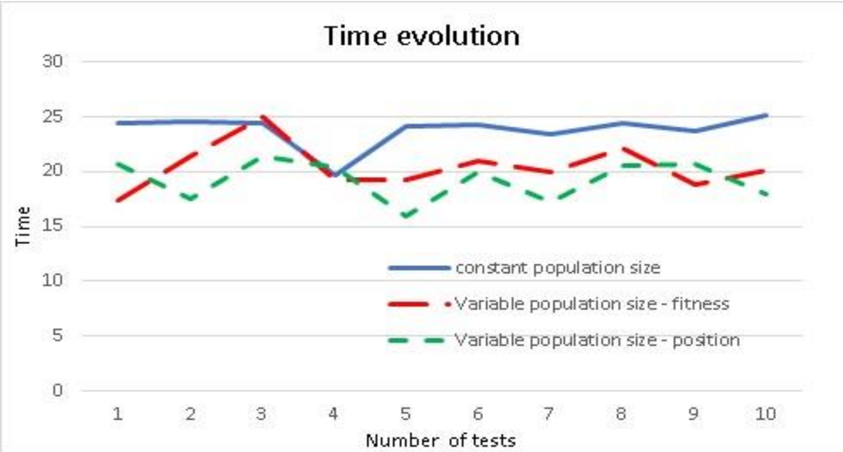

Figure.3. Time evolution

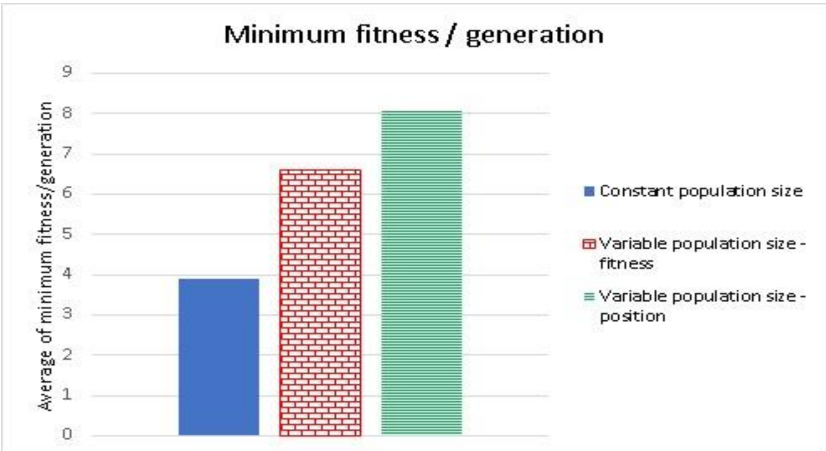

Figure.4. Medium fitness

For populations of 35 chromosomes, the evolution of fitness is represented in Figure.5, the evolution of the chromosomes populations in Figure .6., the evolution of running time in Figure.7. and the minimum fitness average obtained in the 10 runs of the algorithms in Figure.8.

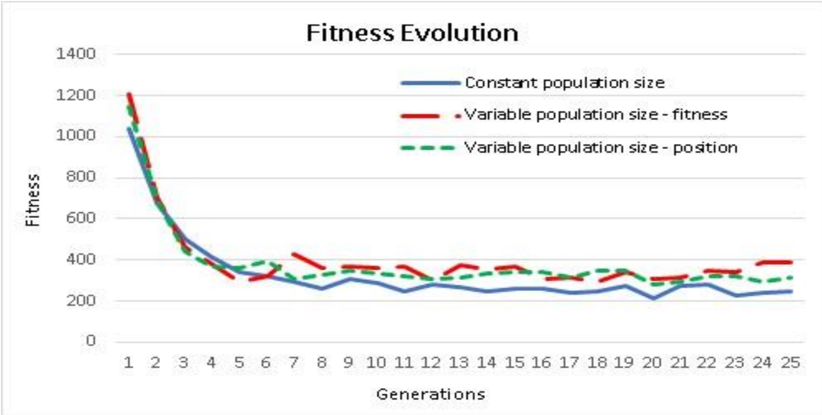

Figure.5 Fitness evolution

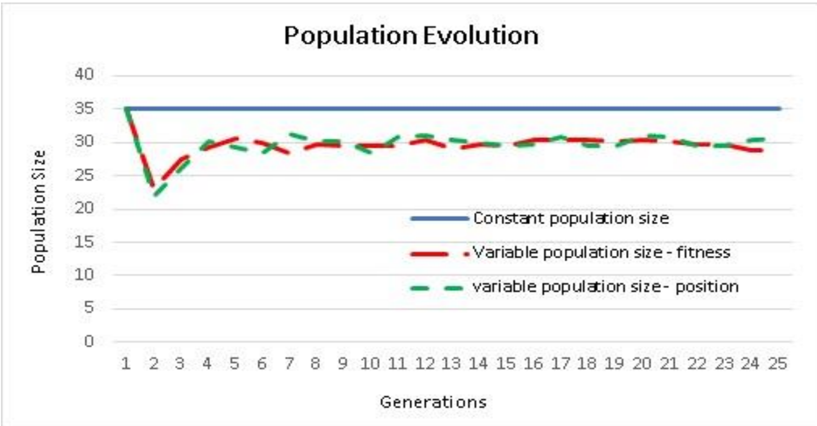

Figure.6. Population evolution

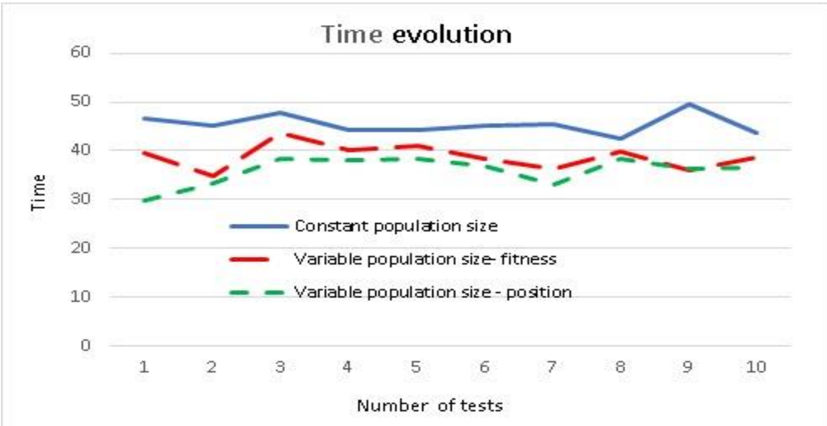

Figure.7. Time evolution

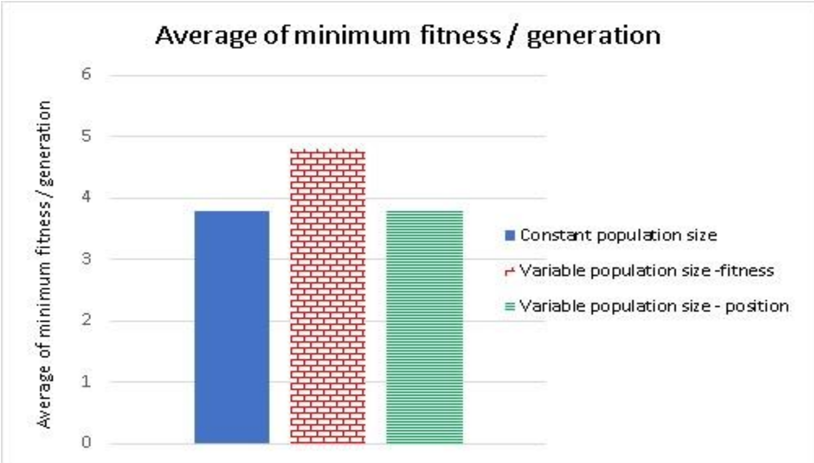

Figure.8. Minimum fitness

5. Conclusions

Taking in consideration the tests results, both the adaptive version of the fitness-based populations and based on the position of the chromosomes in search space lead to a decrease in the running time of the genetic algorithm. If is used an initial population with a reduced number of chromosomes, decreasing the size reduces the quality of the solutions produced (Figure 4), although the evolution of the whole generation's fitness has the best values when using variable population populations depending on fitness. The decrease in running time is significant (Figure 3) and the best solutions are offered in this case by the classic genetic algorithm (Figure 4).

For larger populations (35 chromosomes), the best solutions are generated by the classical genetic algorithm and genetic algorithm with variable population sizes determined by chromosomes position in search space (Figure.8.). The total fitness evolution has the best values for the classic genetic algorithm (Figure.5.). Also, in this case, the use of variable-size generations produces a significant decrease in running time (Figure.7.).

As a result, if chromosome populations are reduced, the use of genetic algorithms with fixed-size populations is the best choice because it offers the best results, even if the running time is higher. If the populations have larger size (so the running time of the genetic algorithm is high), the genetic algorithm that uses variable populations according to the location of the chromosomes in the search space offers a quality of similar solutions offered by the genetic algorithm classic, but running time is significantly reduced.

References

[1] J. Arabas, Z. Michalewicz and J. Mulawka, “GAVaPS-a genetic algorithm with varying population size” in *Proceedings of the First IEEE Conference on Evolutionary Computation. IEEE World Congress on Computational Intelligence*, Orlando, FL, 1994, pp. 73-78 vol.1. doi: 10.1109/ICEC.1994.350039

[2] R. Hinterding, Z. Michalewicz, and T C. Peachey. “Selfadaptive genetic algorithm for numeric functions”, in *Parallel Problem Solving from Nature*, PPSN IV, pages 420–429, 1996

[3] A. E. Eiben, E. Marchiori, and V. A. Valko. “Evolutionary algorithms with onthe-fly population size adjustment”. In *X. Yao et al., editors, Parallel Problem Solving from Nature* PPSN VIII, LNCS 3242, pages 41–50. Springer, 2004

[4] M. Affenzeller, S. Wagner and S Winkler. “Self-adaptive population size adjustment for genetic algorithms.” In *Proceedings of the 11th international conference on Computer aided systems theory (EUROCAST'07)*, Roberto Moreno Díaz, Franz Pichler, and Alexis Quesada Arencibi (Eds.). Springer-Verlag, Berlin, Heidelberg, 820-828.a

[5] M. Affenzeller and S. Wagner. “Offspring Selection: A New Self-Adaptive Selection

Scheme for Genetic Algorithms”. *Adaptive and Natural Computing Algorithms* (2005) 218–221
[6] B.R. Rajakumar and Aloysius George, “APOGA: An Adaptive Population Pool Size based Genetic Algorithm”,*AASRI Procedia*,Volume 4,2013,Pages 288-296,ISSN 2212-6716
[7] R. Maniu, "Adaptive mutation in genetic algorithms for shortest pathrouting problem", *7th International Conference on Electronics,Computers and Artificial Intelligence (ECAI)*, Bucharest, 2015, pp. S-69-S-74