

Hybrid Multievolutionary System to Solve Function Optimization Problems

Krzysztof Pytel

Faculty of Physics and Applied Informatics
University of Lodz, Poland
Email: kpytel@uni.lodz.pl

Abstract—Evolutionary algorithms are optimization methods inspired by natural evolution. They usually search for the optimal solution in large space areas. In Evolutionary Algorithms it is very important to select an appropriate balance between the ability of the algorithm to explore and exploit the search space. The paper presents a hybrid system consisting of a Genetic Algorithm and an Evolutionary Strategy designed to optimize the function of many variables. In this system, we combined the ability of the Genetic Algorithm to explore the search space and the ability of the Evolutionary Strategy to exploit the search space. Optimization performed by the Genetic Algorithm and the Evolutionary Strategy runs at the same time, so it is possible to perform parallel computations. The results of the experiments suggest that the proposed system can be an effective tool in solving complex optimization problems.

I. INTRODUCTION

EVOLUTIONARY Algorithms (EA) are widely used in solving complex problems of optimization. This is a group of methods inspired by observation of nature. These methods are based on the principles of natural selection of living organisms developed by Charles Darwin. According to this principle, well-adapted individuals have more chances of survival - and transfer of their genetic material to the next generation. A list of terms which are used to describe Evolutionary Algorithms is closely related to genetics and evolution. The Evolutionary Algorithm processes the population of individuals (each individual in the form of a chromosome represents a potential solution to the problem). The Evolutionary Algorithm works in certain environments, which can be defined on the basis of the problem solved by the algorithm. Depending on how much a given individual (ie. chromosome) is adapted to the environment in which it is located, a numeric value that determines the quality represented by its solution is assigned to it. This number is called fitness of the individual and is a major factor that describes the ability of an individual to act as a parent for the next generation of population. Evolutionary Algorithms do not guarantee finding the global optimum, but generally provide a good enough solution in an acceptable period of time. Hence, the main use of these algorithms is in very sophisticated problems in a large search space for which there are no specialized techniques. A characteristic feature of Evolutionary Algorithms is that in the process of evolution they do not use the knowledge specific for a given problem, except for the fitness function assigned to all individuals. The Evolutionary Algorithm must keep the

right balance between exploration and exploitation of the search space. Exploration is the process of searching for a new region of a search space where an optimum can exist. Exploitation is the process of searching for regions within the neighborhood of previously visited points. Examples of Evolutionary Algorithms are Genetic Algorithms (GA) and Evolutionary Strategies (ES).

The Genetic Algorithm is an optimization method that simulates the process of natural evolution. The GA uses the mechanism of natural evolution (selection, mutation, cross-over of individuals and reproduction). The individuals of the GA could be coded by binary strings (binary representation), real numbers (a real number representation) or composite structures of genes. The main parameters of the GA, affecting the ability to explore and exploit of the search space, are probability of selection and probability of mutation. The type of cross-over and mutation operators are very important. Reproduction in GAs is closely related to maintaining the diversity of the population, the selection pressure and avoiding premature convergence to the local optima. In Genetic Algorithms the exploitation is done through the selection process. Cross-over and mutation are both methods of exploring the search space. Very important problem is always finding the right balance between exploration and exploitation. If the exploration ability is too large, the algorithm can get stuck in the local optima. If the exploration ability is too large, the algorithm will waste time on poor solutions and cannot focus on solutions found till now.

More information on Genetic Algorithms can be found in publications [3][5][7].

Evolution Strategies uses primarily mutation and selection as search operators. These operators are applied in a loop until the termination criterion is met. ES are based on the principle that small changes have small effects. The mutation is usually performed by adding a normally distributed random value to each individual's genes. After a certain number of fitness function calls or a number of generations, it is essential to adjust the parameters of mutation. At first, the ratio of successful mutations over all mutations is evaluated. If the ratio is less than the specified threshold, the mutation parameters are increased to obtain greater diversity of individuals. If the ratio is greater than the specified threshold, the mutation parameters are decreased to increase the accuracy of the search and accelerate the convergence of the algorithm. The simplest

Evolution Strategy $(1 + 1) - ES$ operates on a population of two individuals: the current point (a parent) and the result of its mutation (a child). If the child's fitness is equal to or better than the parent's fitness, the child becomes the parent in the next generation. Otherwise, the new child is created in the next loop. In strategy $(1 + \lambda) - ES$, λ children can be generated and compete with the parent. In $(1, \lambda) - ES$ the best child becomes the parent in the next generation while the current parent is always disregarded. Evolution Strategies $(\mu/\rho+, \lambda) - ES$ can use the population of ρ parents and also recombination as an additional operator. This makes them less prone to get stuck in the local optima.

More information on Evolution Strategies can be found in publications [1][2].

Hybrid intelligent system is a system which employs, in parallel, a combination of methods and techniques from artificial intelligence subfields, for example Genetic Algorithms, the Fuzzy Logic, Artificial Neural Networks etc. Such systems are able to take advantage of the methods and techniques of artificial intelligence while avoiding their disadvantages. They can be used to improve effectiveness of the methods or where simple methods do not produce the expected results.

Hybrid intelligent systems using artificial intelligence methods can be used in different optimization problems, for example in multiobjective optimization [10] [11], Connected Facility Location Problem [12] or Clustering Problem [13].

II. PROBLEM FORMULATION

Optimization is the process of finding the greatest or the smallest value. The Function Optimization Problem (FOP) is a problem in which certain parameters (variables) need to be determined to achieve the best measurable performance (objective function) under given constraints. For function $f(x)$, called the objective function, that has a domain of real numbers of set S , the maximum optimal solution occurs where $f(x_0) > f(x)$ over set S and the minimum optimal solution occurs where $f(x_0) < f(x)$ over set S .

Formally, optimization is the minimization or maximization of a function subject to constraints on its variables. Let's denote:

- x is the vector of variables (parameters);
- $f(x)$ is the objective function that we want to maximize or minimize;
- c is the vector of constraints that the variables must satisfy. It may consist of several restrictions that we place on the variables.

The function optimization problem (e.g a function maximizing problem) can be stated as follows:

$$\begin{cases} \max & f(x) \\ \text{subject to:} & c_i \leq 0 \text{ for } i = 1, 2, \dots, k \\ & x \in S \end{cases} \quad (1)$$

where:

- $x = [x_1, x_2, x_3, \dots, x_n] \in \mathfrak{R}; n \in N$ - is an n-dimensional vector of decision variables,

- $f(x)$ - is the objective function of variables x ,
- $c_i(x)$ - are constrains,
- S - the search area.

The FOP problem can be used as a benchmark for testing optimization methods. Various methods of solving the FOP are discussed in literature, for example [4][6].

III. THE PROPOSED MULTIEVOLUTIONARY SYSTEM

Genetic Algorithms use cross-over and mutation operators to search the space for possible solutions. One of the drawbacks of Genetic Algorithms is low efficiency in the final search stage. The Evolutionary Strategy uses primarily mutation and selection operators. Evolutionary Strategies are at risk of getting stuck in sub-optimal solutions. The proposed system (GA-ES) consists of a Genetic Algorithm and an Evolutionary Strategy. It combines the ability of a GA to find the areas of possible optima and the ability of ESs to quickly converge to the optima. Both of them are types of Evolutionary Algorithms and can use the same individuals' representation, operators of selection and mutation.

In the system, both algorithms start with the same initial population and work in parallel. After a predetermined number of generations, the best individuals from both algorithms will be compared. Depending on the result of this comparison, transposition of individuals between the algorithms may be performed:

- if the best individual in the GA is better than the best individual in the ES, then the new area of the optima is found. The best individual in the GA replaces the parent in the ES.
- if the best individual in the ES is better than the best individual in the GA, then it means that a new optimal solution is found as a result of the ES. The best individual in the ES replaces the worst individual in the GA population.

The system block diagram is shown in Figure 1.

IV. COMPUTATIONAL EXPERIMENT

The goal of our experiments is verification of the idea of the hybrid multievolutionary algorithm in solving the function optimization problem. We used functions of a wide range of complexity in a diverse environment. For tests we used a set of 3 functions:

- $f1(x_1, x_2)$ - an easy function of two variables (similar to cosinemixture [14] function). The function has many local optima and was used for testing the algorithm's ability to find the global optimum. The function is given by formula:

$$f1(x_1, x_2) = (\sin(x_1) + 0.6 * \sin(20 * x_1)) * \sin(x_2) \quad (2)$$

where:

$$x_1, x_2 \in (0, \pi) \quad (3)$$

The value of maximum 1.6 at point $(\pi/2, \pi/2)$.

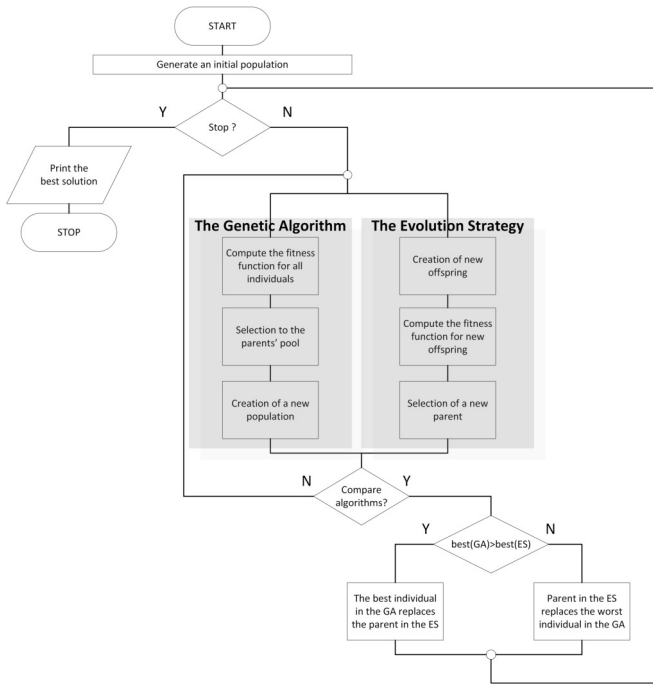


Fig. 1. The system block diagram

- $f_2(x_1, x_2, \dots, x_{10})$ - the function of multiple variables (similar to alpine2 [14] function). A low inclined function, used for testing the ability of the algorithm to determine the exact solution. The function is given by formula:

$$f_2(x_1, x_2, \dots, x_{10}) = \prod_{i=1}^{10} \sin(x_i) \quad (4)$$

where:

$$x_1, x_2, \dots, x_{10} \in (0, \pi) \quad (5)$$

The value of maximum 1 at point $(\pi/2, \pi/2, \pi/2, \pi/2, \pi/2, \pi/2, \pi/2, \pi/2, \pi/2, \pi/2)$.

- f_3 - the function proposed in [9]. It is a sophisticated function with many local optima with different values, which permits to estimate the ability of the algorithm to solve difficult optimization problems. The first generation of individuals was placed in the local optimum (point [5, 5]). The algorithm should find the total optimum (point [50, 50]), avoiding the local optima. The function is given by formula:

$$f_3(x_1, x_2) = \sum_{i=1}^7 h_i * e^{-\mu_i * ((x_1 - x_{i1})^2 + (x_2 - x_{i2})^2)} \quad (6)$$

where: $h_1 = 1.5, h_2 = 1, h_3 = 1, h_4 = 1, h_5 = 2, h_6 = 2, h_7 = 2.5$

$\mu_1 = \mu_2 = \mu_3 = \mu_4 = \mu_5 = \mu_6 = \mu_7 = 0.01$

$(x_{11}, x_{12}) = (5, 5), (x_{21}, x_{22}) = (5, 30), (x_{31}, x_{32}) = (25, 25), (x_{41}, x_{42}) = (30, 5), (x_{51}, x_{52}) = (50, 20), (x_{61}, x_{62}) = (20, 50), (x_{71}, x_{72}) = (50, 50)$

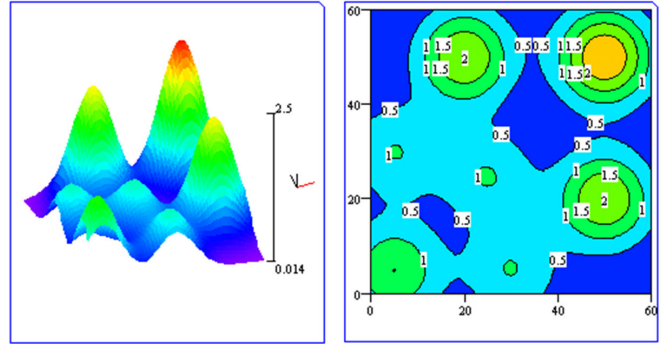

 Fig. 2. Function f_3

TABLE I

THE AVERAGE TIME AND NUMBER OF FITNESS FUNCTION CALLS NEEDED TO REACH THE PREDETERMINED VALUE OF THE OPTIMIZED FUNCTION

Function	The predetermined value of algorithm termination	SGA		GA-ES	
		Time [s]	The number of fitness function calls	Time [s]	The number of fitness function calls
f1	1.596	0.210	37960	0.092	3978
f2	0.999	1.378	80158	0.480	8450
f3	2.499	0.307	56940	0.072	7124

The value of maximum 2.5 at point (50,50).

The function f_3 is shown in Figure 2.

The values of parameters of the Genetic Algorithm and the Evolution Strategy was fixed during the initial experiments. In the experiments we accepted the following values of parameters of the Genetic Algorithm:

- the genes of individuals are represented by real numbers,
- the probability of cross-over = 0.8,
- the probability of mutation = 0.15,
- the number of individuals in the population = 25.

For the Evolutionary Strategy, model $(1+1) - ES$ was chosen and the mutation performed by adding a number generated randomly according to normal distribution.

The best individuals from both algorithms were compared after every 50 generations and, depending on the result of this comparison, transposition of individuals was performed between the algorithms. The system was stopped when the best individual reached the predetermined value of the optimized function.

In the experiment, we compared the results of the proposed GA-ES and the standard genetic algorithm (SGA) described in [8], and adapted it to optimize the test functions. Each algorithm was executed 10 times on a standard PC computer (CPU: Intel i3, RAM: 8GB, Windows 10 operating system). Table 1 shows the average time and number of fitness function calls needed to reach the predetermined value of the optimized function.

The graph in Figure 3 shows the average running time of the Genetic Algorithm (SGA) and the proposed system (GA-ES).

The graph in Figure 4 shows the average number of fitness function calls in the Genetic Algorithm (SGA) and the

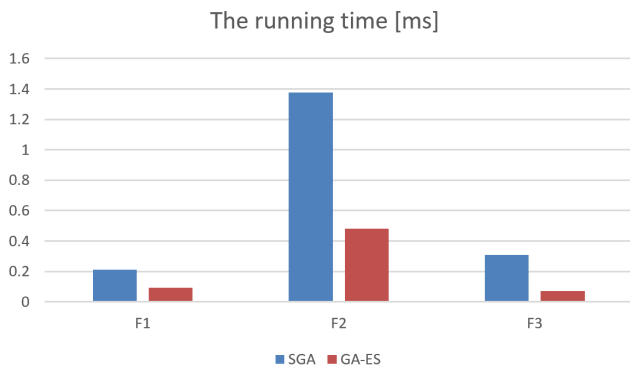


Fig. 3. The average running time of the Genetic Algorithm (SGA) and the proposed system (GA-ES)

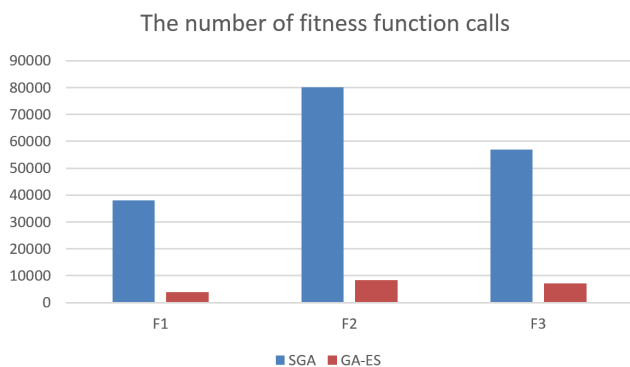


Fig. 4. The average number of fitness function calls in the Genetic Algorithm (SGA) and the proposed system (GA-ES)

proposed system (GA-ES).

V. CONCLUSIONS

The proposed Genetic Algorithm-Evolution Strategy system was able to find a solution near the optimum for all tested functions. Optimization of function F2 (a low inclined function) has shown that the system has greater convergence and accuracy in comparison to the SGA. Optimization of function F3 (a sophisticated function with many local optima) has shown that the system is more resistant to premature convergence to the local optimum compared to the ES.

The GA-ES running time on a PC was very short (less than 2 seconds). The time was 56, 65 and 76 percent shorter than the running time of SGA for function f1, f2 and f3 respectively.

The number of fitness function calls in GA-ES system was decreased by nearly 90 percent in relation to the number of fitness function calls in the SGA.

In the proposed system it is possible to perform parallel calculations by the Genetic Algorithm and the Evolutionary Strategy, eg. by using multiple processors or processor cores.

The proposed system is an efficient tool for solving function optimization problems. It could be used for solving very wide range of optimization problems.

REFERENCES

- [1] Bäck T., Hoffmeister F., Schwefel H.-P., *A survey of evolution strategies*, Proceedings of the Fourth International Conference on Genetic Algorithms, Morgan Kaufmann, s. 2-9, 1991.
- [2] Beyer H.-G. Schwefel H.-P., *Evolution Strategies: A Comprehensive Introduction*. Journal Natural Computing, 1(1):3-52, 2002.
- [3] Goldberg, David E. *Genetic Algorithms in Search, Optimization, and Machine Learning* Reading, MA: Addison-Wesley, 1989.
- [4] Jensi R., Wiselin Jiji G., *An improved krill herd algorithm with global exploration capability for solving numerical function optimization problems and its application to data clustering*, Appl. Soft Comput. 46: 230-245, 2016.
- [5] Michalewicz Z., *Genetic Algorithms + Data Structures = Evolution Programs*, Springer Verlag, Berlin (1992).
- [6] Karaboga, D., Basturk B., *A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm*, JOURNAL OF GLOBAL OPTIMIZATION Volume: 39 Issue: 3, Pages: 459-471, 2007.
- [7] Kwasnicka H., *Obliczenia ewolucyjne w sztucznej inteligencji*. Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław, 1999, (in Polish).
- [8] Potter M.A., De Jong K.A., *A cooperative coevolutionary approach to function optimization*. In: Davidor Y., Schwefel HP., Männer R. (eds) Parallel Problem Solving from Nature - PPSN III. PPSN 1994. Lecture Notes in Computer Science, vol 866. Springer, Berlin, Heidelberg 1994.
- [9] Pytel K., Nawarycz T., *Analysis of the Distribution of Individuals in Modified Genetic Algorithms* [in] Rutkowski L., Scherer R., Tadeusiewicz R., Zadeh L., Zurada J., Artificial Intelligence and Soft Computing, Springer-Verlag Berlin Heidelberg, 2010.
- [10] Pytel K., *The Fuzzy Genetic Strategy for Multiobjective Optimization*, Proceedings of the Federated Conference on Computer Science and Information Systems, Szczecin, (2011).
- [11] Pytel K., Nawarycz T., *The Fuzzy-Genetic System for Multiobjective Optimization*, [in] Rutkowski L., Korytkowski M., Scherer R., Tadeusiewicz R., Zadeh L., Zurada J., Swarm and Evolutionary Computation, Springer-Verlag Berlin Heidelberg, 2012.
- [12] Pytel K., Nawarycz T., *A Fuzzy-Genetic System for ConFLP Problem*, Advances in Decision Sciences and Future Studies, Vol. 2, Progress & Business Publishers, Krakow 2013.
- [13] Pytel K., *Hybrid Fuzzy-Genetic Algorithm Applied to Clustering Problem*. Proceedings of the 2016 Federated Conference on Computer Science and Information Systems, Gdańsk, 2016, doi: 10.15439/2016F232.
- [14] http://infinity77.net/global_optimization/test_functions.html.