UDC 621.391
DOI: 10.15587/1729-4061.2021.245222

IMPLEMENTATION OF NEW HYBRID EVOLUTIONARY ALGORITHM WITH FUZZY LOGIC CONTROL APPROACH FOR OPTIMIZATION PROBLEMS

Maan Afathi
Doctor of Computer Sciences, Teacher
Department of Computer Science
College of Education for Pure Sciences
University of Mosul
Alkendi str., 14, Mosul, Iraq, 41002
E-mail: maan@gamma.ttk.pte.hu

The main purpose of using the hybrid evolutionary algorithm is to reach optimal values and achieve goals that traditional methods cannot reach and because there are different evolutionary computations, each of them has different advantages and capabilities. Therefore, researchers integrate more than one algorithm into a hybrid form to increase the ability of these algorithms to perform evolutionary computation when working alone. In this paper, we propose a new algorithm for hybrid genetic algorithm (GA) and particle swarm optimization (PSO) with fuzzy logic control (FLC) approach for function optimization. Fuzzy logic is applied to switch dynamically between evolutionary algorithms, in an attempt to improve the algorithm performance. The HEF hybrid evolutionary algorithms are compared to GA, PSO, GAPSO, and PSOGA. The comparison uses a variety of measurement functions. In addition to strongly convex functions, these functions can be uniformly distributed or not, and are valuable for evaluating our approach. Iterations of 500, 1000, and 1500 were used for each function. The HEF algorithm's efficiency was tested on four functions. The new algorithm is often the best solution, HEF accounted for 75 % of all the tests. This method is superior to conventional methods in terms of efficiency

Keywords: evolutionary computations, GA, PSO, FLC, optimization, hybrid evolutionary algorithm

Received date 01.10.2021
Accepted date 29.11.2021
Published date 16.12.2021

How to Cite: *Afathi, M. (2021). Implementation of new hybrid evolutionary algorithm with fuzzy logic control approach for optimization problems. Eastern-European Journal of Enterprise Technologies, 6 (4 (114)), 6–14. doi: https://doi.org/10.15587/1729-4061.2021.245222*

1. Introduction

Evolutionary computation helps solve difficult optimization problems. The approach's simplicity, robust response to changing circumstances, flexibility, and other features are all advantages but standard evolutionary algorithms have limited use in practical architectural design tasks. This may be due to the poor search efficiency and the lack of diversity of the result. With scientific progress, many ways emerged to overcome these weaknesses.

To determine this problem, look at the performance and quality of algorithms. It is common to refer to genetic algorithms as evolutionary algorithms or evolutionary computation as, evolutionary strategies [1], learning classifier systems [2], evolutionary programming [3], differential evolution [4], genetic programming [5], and estimation of distribution algorithms as evolutionary algorithms or evolutionary computation [6]. These algorithms share the same conceptual framework for simulating individual structure evolution but differ in issue description, selection method, and estimation of distribution algorithms.

Most evolutionary techniques start with random generation of an initial population, and then reckon fitness value for each subject. After that, they reproduce into a new population based on fitness values and finally stop when the requirements are met. Otherwise, the reproduction step is repeated. The procedure demonstrates that PSO and GA have a great deal in common. Both techniques begin with a collection of randomly created populations and use fitness values to evaluate them. Both methods rely on random algorithms to keep the population fresh and find the best solution. Neither system is certain to succeed. PSO, on the other hand, does not employ genetic operators such as crossover and mutation. Particles self-update in response to their internal velocity. Additionally, they have memory, which is critical for the method.

The mechanism for sharing information in PSO differs significantly from that of GAs. In GAs, chromosomes communicate with one another. Thus, the entire population moves in unison toward an optimal area. Only the global best divulges information in PSO. It is a mechanism for one-way information sharing. Evolution seeks only the optimal solution. In comparison to GA, all particles tend to converge quickly to the optimal solution, even in the case of a local version. Because the two methods are conceptually equivalent, we can use them sequentially. There will be two options: either to begin with GA and conclude with PSO, or vice versa. Both methods enable us to arrive at the solution more quickly than either method alone. The difficulty in determining which method to use to arrive at the optimal value is dependent on the type of problem at hand. It is possible to begin with either of the methods in the proposed algorithm in order to obtain the optimal value.

Therefore, a large number of researchers are focusing on constructing hybrid evolutionary algorithms. It is critical to increase the accuracy of procedures and predictability to reach optimal values. A hybrid algorithm uses two or more algorithms to solve a problem; and hybrid algorithms are very common in the optimized real world.

2. Literature review and problem statement

Several research papers on fuzzy system evolutionary design can be found in the literature [7], the majority of which concern the automatic design or optimization of fuzzy logic controllers by either adapting the fuzzy membership functions or by learning the fuzzy if-then rules [8]; on the other hand, FLC have been successfully applied to nonlinear control problems [9] and automatic construction of membership functions and fuzzy rules from training instances using evolutionary algorithms [10]. A fuzzy rule-based system is based on an agent-based evolutionary framework and multi-objective optimization [11]. In [12], a new system is presented that can be used to search for the best feature subset for dimensional reduction, acting as a genetic feature selector wrapper; while in [13], both genetic algorithm (GA) and particle swarm optimization (PSO) have had success when it comes to the many combinatory issues that require near-optimal solutions. In some cases, hybrid algorithms have been created to mitigate some of the poor behavior exhibited by GA and PSO.

The PSO algorithm comes in a variety of flavors. PSO's capabilities, as well as those of other algorithms like hybrid PSO, have been included in some of these variations. The PSO method makes use of a variety of evolution techniques, including selection, mutation, and crossover, as well as other types of variation. The Genetic Fuzzy System (GFS) keeps track of the structural health of composite helicopter rotor blades through the internet [14]. The authors created both global and local GFS models. The global GFS detects matrix cracking and deboning/delamination all over the blade, but the local GFS only sees it in a few spots. Tunnel ventilation systems use a GA-based fuzzy controller [15]. Due to the system's nonlinear and complex behavior, the FLC method was used, and the FLC was optimized using GA. For a difficult task provided by a machine supplier, a genetic-fuzzy system can be used to provide online scheduling solutions automatically [16]. There is a rule framework in place that categorizes various scheduling circumstances and provides a scheduling approach to each one. The researchers compared and contrasted two different approaches. The first method assigns a similar scheduling strategy to all situation classes through an iterative process. Using symbiotic evolution, we can build different circumstance classes and then assign appropriate scheduling methods based on the Gaussian membership function parameter values.

The paper [17] proposed first running the PSO algorithm and then utilizing the results as a starting population for the GA approach. While [18] looked at two different approaches to combining the GA and PSO methods. In the first, the GA population was used to start with PSO, and in the second, the PSO swarm was used to initiate the GA population.

This approach proposed a new hybrid technique combining PSO with the genetic algorithm. Using the genetic algorithm operator increases population variety and facilitates escape from the local minimum. As stated previously, the influence factor *pe* modulates genetic operators. So it may affect the algorithm's convergence. Choosing the right value depends on the problem and can be tricky. They recommended first running the PSO algorithm and then utilizing the results as an initial population for the GA approach. It's also fine to make changes while the algorithm is running. When the PSO algorithm produces new, better solutions, the influence of genetic algorithms (number of particles modified by genetic operators) should be reduced. To combat this, genetic operators' effect should be increased. Leave the algorithm in the local optimal if it is [19].

To wrap up this section, hybrid approaches are one of the most well-established areas of optimization; numerous studies have been conducted, and optimization methods have been improved. However, hybrid optimization techniques such as GA-Fuzzy, GAPSO, and others are the most prevalent. PSO hybridization with GA requires a significant amount of computation and memory. This is particularly true in series hybridization, where changes from one algorithm to another cannot be returned to the first, thereby negating the first's advantage and making it impossible to reach the optimal value.

The use of a single classical algorithm, or a hybrid of two classical algorithms, to solve a problem is preferable because it is more appropriate to re-apply the algorithm in every case of change encountered. It is from this point that the importance of applying multiple exchanges between the classical algorithms will become apparent. The properties of both optimization algorithms (GA and PSO) will be taken advantage of, and the defects of both optimization algorithms (GA and PSO) will be avoided, by switching between them in an adaptive manner.

3. The aim and objectives of the study

The aim of the study is to investigate differential design for generating intelligent paradigms with evolutionary algorithms. To accompany this evolutionary design architecture, the most modern evolutionary design architecture is given. It will explain the fuzzy control that has an important course within the proposed algorithm.

To achieve the aim, the following objectives were set:

– to implement a new algorithm called the Hybrid Evolutionary Algorithm with FLC (HEF);

– to apply the Hybrid Evolutionary Algorithm with FLC (HEF) using MATLAB in addition to the traditional GA and PSO and two hybrid GAPSO and PSOGA algorithms;

– to compare and evaluate the performance of HEF and GA, PSO, GAPSO and PSOGA. There is evidence that HEF utilized the FLC optimization switching agent to connect several optimization strategies;

– to compare the efficiency of new and old methods, some researchers use optimization functions such as Rosenbrock, Sphere, Rastrigin, and Griewank to provide a wide range of difficulties and challenges.

4. Materials and methods

4. 1. Research hypothesis

A hybrid evolutionary algorithm solves discrete optimization issues in a broad way. As a result, several aspects of the algorithm must be thoroughly examined before they can be used to solve automation problems and determine a set of fixed elements and an optimization approach.

As in Fig. 1, the purpose of this study is to develop a feasible, reliable, and cost-effective method for transitioning from evolutionary optimization techniques to stochastic operations. Specifically, the GA and PSO techniques are combined with fuzzy to optimize HEF simulation, with the goal of increasing overall operational efficiency through result minimization. FLC's primary objective is divided into two categories. The first step is to develop a generic and adaptable mechanism for resolving switching problems between evolutionary algorithms. The second task is to monitor and control the entire system, as well as to determine the termination condition.

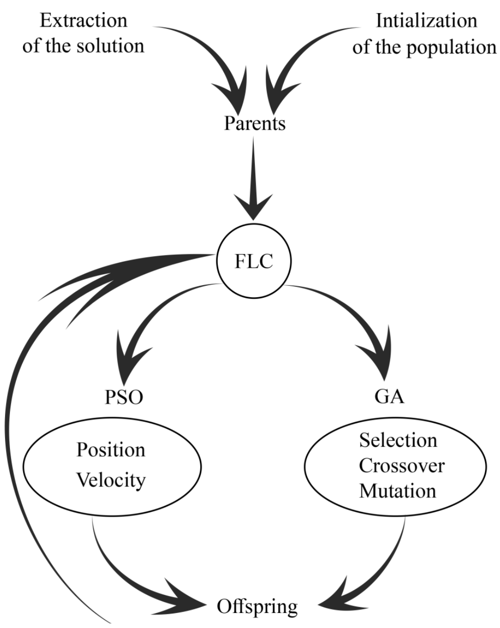

Fig. 1. GA and PSO techniques are applied with fuzzy to optimize

4. 2. HEF Algorithm

These algorithms (GA, PSO, GAPSO, and PSOGA) each have their own set of advantages for dealing with situations of various degrees of complexity. Rather than combining the two algorithms, we wish to mix them in order to take advantage of their individual benefits. On the other hand, to avoid GA and PSO becoming locked in local minima and to overcome the stagnation problem, an FLC-based evolutionary algorithm is being investigated for a number of applications. Fig. 2 graphically depicts the hierarchical structure of the HEF system. When a random particle from the swarm is chosen, the operation proceeds to other locations within the search area until the entire swarm is investigated.

The performance of the algorithm is described with steps of list a proposed HEF algorithm is given below:

1. Define all initial parameters for GA, PSO and FLC.
2. Initial input rate of change (RCH) Optimization progress (OpP).
3. FLC. It is depending on the input variables and the previous method PSO or Ga.
4. If the previous method is PSO, the GA procedure will be called or If the previous method is GA, the PSO procedure will be called.
5. Check the condition of the end of the algorithm depending on No of iterations or minimum error value.
6. If not stop Calculation of new RCh & OpP for the next round then go to 3.

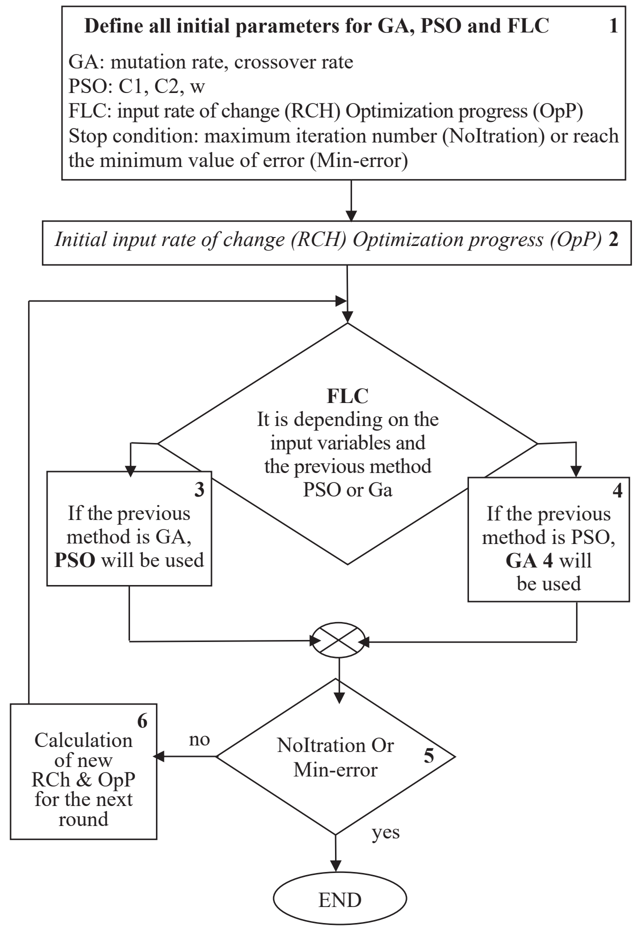

Fig. 2. HEF architecture

4. 3. Mechanism FLC controlling on switching in the HEF algorithm

The rate of change (RCh) in the fitness function and the optimization progress (OpP) are chosen as inputs to the fuzzy system. The two output variables are the change of the used optimization method (COp) and the next stage duration (NSD). The rate indicates the effectiveness of the proposed solution discovered thus far by HEF. A variety of performance metric settings are required for various optimization challenges. In order to develop a fuzzy system that can be utilized to solve a wide range of optimization issues, one of the inputs must be the rate, RCh must be translated into a normalized format:

$$RCh=\frac{V_{int}-V_{end}}{V_{int}}, \quad (1)$$

where V_{int} is the initial value of the fitness function and V_{end} is the final value of the fitness function after the last iteration. The optimization progress (OpP) is therefore:

$$\mathrm{OpP}=\frac{N}{N_{\max}}, \quad (2)$$

where N is the number of optimization iteration at termination and $N_{\max}$ is the maximum number of iterations. All the three fuzzy variables, i.e. the two input variables (RCh, OpP) and one of the output variables (N), are three fuzzy sets defined as LOW, MEDIUM and HIGH with associated membership functions as LowTriangle, MedTriangle and

HiTriangle, respectively. These three membership functions are defined as follows.

Low triangle membership function:

$$F_{LowTriangle}(x)=\begin{Bmatrix}1 & \text{if } x<x_1\\ \dfrac{x_2-x}{x_2-x_1} & \text{if } x_1<x<x_2\\ 0 & \text{if } x>x_2\end{Bmatrix}. \quad (3)$$

Medium triangle membership function:

$$F_{MedTriangle}(x)=\begin{Bmatrix}0 & \text{if } x<x_1\\ 2\dfrac{x-x_1}{x_2-x_1} & \text{if } x_1<x\le\dfrac{x_2-x_1}{2}\\ 2\dfrac{x_2-x}{x_2-x_1} & \text{if } \dfrac{x_2+x_1}{2}<x\le x_2\\ 0 & \text{if } x>x_2\end{Bmatrix}. \quad (4)$$

High triangle membership function:

$$F_{HiwTriangle}(x)=\begin{Bmatrix}0 & \text{if } x<x_1\\ 2\dfrac{x-x_1}{x_2-x_1} & \text{if } x_1<x<x_2\\ 1 & \text{if } x>x_2\end{Bmatrix}. \quad (5)$$

The other output variable COp has two fuzzy sets: «Change» and «NoChange», with Change and NoChange as membership functions, respectively. These two membership functions are defined as follows:

$$F_{\mathrm{COp}}(x)=\begin{Bmatrix}1 & \text{if } x<x_1\\ \dfrac{x_2-x}{x_2-x_1} & \text{if } x_1<x<x_2\\ 0 & \text{if } x>x_2\end{Bmatrix}. \quad (6)$$

$$F_{\mathrm{NCOp}}(x)=\begin{Bmatrix}0 & \text{if } x<x_1\\ \dfrac{x-x_1}{x_2-x_1} & \text{if } x_1<x\le x_2\\ 1 & \text{if } x>x_2\end{Bmatrix}. \quad (7)$$

The shape and placement of the function are determined by the essential parameters x_2 and x_1. Other membership function definitions exist, but the authors found that they were beneficial in a variety of situations and were simple to implement in microcontrollers and microprocessors. The controller in this study has two input variables and different linguistic variables. The first variable (input) is Rch, which consists of three fuzzy sets, each with a different dynamic range (0, *l*). Each fuzzy set is given a membership function. The first is a LowTriangle function with two critical values of 0 and 0.4; the second is a MedTriangle function with two critical parameters of 0.1 and 0.9; and the third is a Hitriangle function with two critical parameters of 0.6 and 1. The present inertial time, OpP, is the second input variable. There are two output variables on the controller. The change of inertial change and Nochange are two linguistic variables in the first output variable COp. Like the input variables, the second output variable, NCOp, has three linguistic variables.

This results in at most 3·3=9 rules to characterize the nature of those inputs in relation to their discourse universe. Although each scenario has a related entry in this example, it is possible to leave a particular space blank, implying that the controller does nothing (i.e. the output remains unchanged). The rule base for systems with two inputs can be built as shown in Tables 1, 2.

Table 1

FLC rule base for rate condition

RCh \ OpP	Low	Medium	High
Low	Change	Change	Change
Medium	Change	Change	Change
High	No change	No change	No change

Table 2

FLC rate base for Noltration condition

RCh \ OpP	Low	Medium	High
Low	High	Medium	Medium
Medium	Medium	Medium	Medium
High	High	High	High

The degree of non-linearity of the FLC used in the HEF model can be gauged by viewing the FLC control surface, two views of which are shown in Fig. 3. A linear control surface would be consistent with a plane, and while the control surface above exhibits some near-linear behavior in the central section of the surface, non-linear behavior is evident at extreme values of rate and time and by «bumpy» and sharp-changing regions throughout. There are three primary sources for the non-linear surface in an FLC [17, 20].

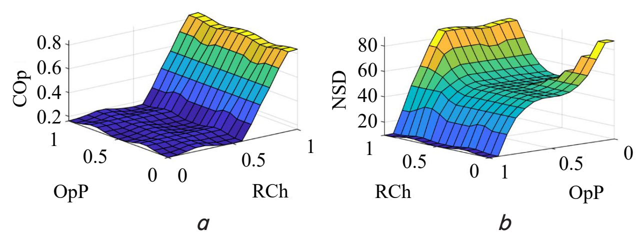

Fig. 3. That's rate of change (RCh) and optimization progress (OpP) are selected as inputs to FLC: *a* – Change of Optimization (COp); *b* – Next Stage Duration (NSD)

The experiment has been divided into two parts. The first experiment respectively compares the traditional algorithms (GA, PSO), the hybrids (GAPSO, PSOGA) and the proposed algorithm (HEF). The second experiment applied our algorithm (HEF) in difficult and different circumstances.

5. Results of implementing the HEF algorithm

5. 1. Experimental settings of the HEF algorithm

We tested the proposed HEA algorithm on the maximum and minimum functions in order to ensure its validity. To confirm the efficiency of HEF, some regularly used optimization functions were chosen. Many papers [12, 13, 21, 22] mention Rosenbrock, Sphere, Rastrigin, and Griewank as benchmark functions for comparison in this study. These are common test functions that have been utilized in prior evolutionary optimization studies and provide a wide variety of challenges. Rosenbrock and Sphere (f8 and f9) with singlepeak were employed in this study to verify the algorithm's convergence accuracy and rate. Due to the fact that many physical problems have several peaks, the optimization search may end up in a local optimum along the way to the global optimum. Multimodal functions should be used to test the optimization algorithm. To test the algorithm's global optimization capacity, the Rastrigin and Griewank functions (f10 and f11) with multi-peak were used.

In order to account for the valley's non-linearity, many algorithms cover a large area slowly since they have to modify the direction of their searches on a regular basis. The Rosenbrock function is two-dimensional and unimodal in nature, with a parabola-shaped deep valley that can be found on it, which will eventually reach its global minimum and there are no separate dimensions to this function. This function is defined by the equation below:

$$f_{ros}(x)=\sum_{i=1}^{p-1}\left[100\left(x_{i+1}-x_i^2\right)^2+\left(1-x_i\right)^2\right]. \qquad (8)$$

The Sphere function, which is unimodal and separable, is the third function. The following equation describes strongly convex and simple function:

$$F_{Sphere}(x)=\sum_{i=1}^{p-1}\left(x_i^2\right). \qquad (9)$$

The second function is more generic. Rastrigin is a multimodal function that was created by combining Sphere with a modulator term $\alpha\cos(2\pi x_i)$. Its contour is formed by a large number of local minima whose values grow in proportion to the distance between them and the global minimum defined by (9):

$$F_{ras}(x)=10p\sum_{i=1}^{n-1}\left[x_i^2-10\cos\left(2\pi x_i\right)\right]. \qquad (10)$$

The generalized Griewank function, which is multimodal and regularly distributed but not separable, is the final function A product term is used to establish the interdependence of the variables in the model. It is possible to optimize each variable separately using the equation (11):

$$F_{Griewank}(x)=\sum_{i=1}^{p}\frac{x_i^2}{4000}-\prod_{i=1}^{d}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1. \qquad (11)$$

The above-mentioned four functions have a variety of different situations and they are used to increase the accuracy and demonstrate the algorithm proposed in this research.

5. 2. Comparison of traditional algorithms and HEF

Three different dimensions of sizes 10, 30 and 50 are tested for all four functions. The maximum number of generations is set to 500, 1,000, and 1,500. One way to investigate the performance of the HEF algorithm is to vary the size of the populations employed in each iteration. This size variation happens with varying dimensions.

These populations are at 50, 100, and 200 individuals each. A total of 100 runs for each experimental setting were conducted. The search space for the experiments is [–10, +10] for all the functions. The algorithm's settings for specific parameters. Firstly, for GA, the population size is equal to (100), crossover rate is (0.6) and mutation rate is (0.3). Secondly, for PSO parameters, coefficient C1 and social coefficient C2 are equal (2) and inertial weight w is between 29.9 and 0.3.

Hybrid algorithms and HEF used the same parameters. Samples of test functions results are shown in Tables 3, 4. For each of the 100 runs of the HEF algorithm, the search space and average best fitness for the best particle are computed, which leads to the following results: with the functions (F8–F11), it produced good results while completing the convergent phase more quickly than the other competing algorithms.

For the purpose of inspection and testing of the algorithm, HEF conducted all the tests possible, such that the number of these tests is 594. Table 3 represents only some of the tests that show that the HEF algorithm is better than the other algorithms.

Table 3

Some testing results (average value of 100 runs) of the following functions: a) Rosenbrock; b) Rastrigin; c) Sphere; d) Griewank; the results show that HEF has the best result

a) Rosenbrock function							
dim	popu	Itra	GA	PSO	GAPSO	PSOGA	HEF
10	100	1,500	54.9050390	3.0986129	3.5012260	4.1840576	2.1422217
	200	1500	25.8339254	3.3489086	3.2979943	4.1255705	1.7295450
30	100	1500	1595.4309558	146.4905519	81.0923249	200.126111	53.3110017
	200	500	555.7968170	154.2657570	120.931273	279.787945	99.9235301
		1000	564.4494005	120.1217701	61.7329330	189.296672	61.0480409
		1500	553.0114423	89.9231590	69.7676323	126.032776	63.5426128
50	50	500	19103.102875	7279.3190056	3745.39274	13495.8964	2324.10906
	200	1500	3968.6416594	266.3573605	217.927704	438.101363	163.780868
b) Rastrigin function							
10	50	500	20.9995809	6.2763435	7.0240323	10.4223827	5.6125994
	100	1500	211.3057540	51.8426738	71.8419924	65.7992447	41.3134580
		500	114.4197907	80.0115066	62.2497541	87.6102558	47.0579405
50	50	500	526.4267855	335.0863013	346.398007	327.879789	273.549064
	100	500	312.9898250	281.4605339	210.729705	236.201480	150.107662
		1500	318.0132520	121.4187847	136.908739	166.265919	95.3724370
	200	500	201.4782142	238.5881391	131.455705	170.881719	92.1868103
		1000	178.3804510	136.3013245	90.9524070	170.308027	59.9357941
c) Sphere function							
10	200	1500	0.0576697	0.0000000	0.0000000	0.0000000	0.0000000
30	50	500	9.6281266	0.3105080	0.29655964	2.6503327	0.2896150
	100	500	4.8499440	0.0856588	0.2061771	0.9534102	0.0815745
50	50	500	36.4090684	12.4308898	7.6227767	27.882661	4.3087127
		1000	34.9863164	3.3272009	1.8375423	6.6395920	1.2019434
	100	1000	21.4725031	1.1254789	0.7804462	2.9096162	0.2490245
		1500	21.5177332	7.0032946	0.1271885	1.3943089	0.0435033
		1500	9.8283309	0.0037393	0.0331918	0.1701024	0.0027109
d) Griewank function							
10	200	1500	0.0092926	0.0123052	0.0044834	0.0114812	0.0035968
30	50	500	0.1295082	0.1097727	0.0360240	0.0932734	0.0245314
	200	500	0.0330478	0.0300050	0.0063523	0.0314659	0.0046551
		1500	0.0414372	0.0254638	0.0096768	0.0264402	0.0089887
50	50	500	0.1493082	0.0956985	0.0416602	0.1085254	0.0289331
	100	1000	0.1371774	0.1160969	0.0188092	0.0954646	0.0159163
	200	1500	0.0774445	0.1180633	0.0136650	0.0934671	0.0130475

Table 4

All possible results (average value of 100 runs) of the following functions: a) Rosenbrock; b) Rastrigin; c) Sphere; d) Griewank; the results show that HEF has a few drawbacks

dim	popu	Itra	GA	PSO	GAPSO	PSOGA	HEF
a) Rosenbrock function							
10	100	500	66.1868549	4.8317257	6.0130655	11.5317982	6.0593365
	200	500	24.3443255	8.0424355	4.5250319	4.3355859	4.5776259
50	50	1000	15758.153354	607.3546398	981.909540	451.96917	663.325609
		1500	15263.166238	439.6065380	618.243662	1153.54796	454.433765
b) Rastrigin function							
10	50	1000	27.6899555	6.2846837	5.6370697	6.3492710	6.6956957
		1500	24.7271925	2.5570749	5.1329931	3.8063875	4.4375153
	100	500	12.6556343	4.8232295	4.2988138	3.3118377	4.4258424
		1000	12.2761426	4.2555092	3.3201413	2.2968709	2.8655842
		1500	11.7771524	1.9023348	2.5777617	3.1072136	2.5968483
	200	1000	5.4747189	1.5536038	1.2043218	3.0348732	1.9949591
		1500	6.7492654	1.8409374	1.4030058	0.8287218	1.1541525
30	50	1000	218.8286347	52.6664627	90.7339091	79.6194346	70.4025930
	200	1500	60.9333014	29.9782374	12.1612736	42.5684754	22.2893929
50	50	1500	505.4940637	134.6591321	201.257473	187.244442	168.734600
c) Sphere function							
10	50	500	0.4833277	0.0000000	0.0000070	0.0000419	0.0000092
	100	500	0.1665356	0.0000042	0.0000011	0.0000069	0.0000023
	200	500	0.0544333	0.0000005	0.0000006	0.0000004	0.0000013
30	50	1000	9.5841473	0.0404308	0.0376095	0.0230273	0.0292281
		1500	9.6673474	1.0000010	0.0019731	0.0047108	0.0048875
	100	1000	4.5722550	0.0021638	0.0047276	0.0217913	0.0011693
		1500	4.8639225	0.0000982	0.0001934	0.0000100	0.0000805
	200	1000	1.8455636	0.0000922	0.0000298	0.0038836	0.0000856
50	50	1500	34.1238462	12.0182145	0.4548737	3.2010603	0.4798060
d) Griewank function							
10	50	500	0.0475908	0.0328842	0.0250986	0.0363188	0.0253389
	100	1500	0.0250486	0.0214224	0.0128176	0.0214746	0.0144882
	200	1000	0.0135937	0.0192313	0.0087693	0.0211199	0.0100248
30	200	1000	0.0373370	0.0317528	0.0086188	0.0251994	0.0096874
50	50	1500	0.1713549	0.1379658	0.0131923	0.1133198	0.0135191

HEF normally assigns the facilities quickly, whereas GA took the longest and had the worst fitness function value. This experiment revealed that GAPSO results are the nearest to HEF algorithm results, thus, the next experiment will explore the advantage of the proposed algorithm over GAPSO. Other models do not produce satisfactory outcomes. However, with a larger population and more generations, better solutions are possible.

While Table 4 includes all the results of the tests in which the HEF algorithm failed. The size of the tests where the failure occurs is 25 %. This 25 % is distributed among the other 4 methods (GA, PSO, GPSO and PSOG), thus this percentage will not affect the effectiveness of the HEF algorithm.

Fig. 4 gives all comparisons between all the algorithms showing that HEF has the lowest fitness function with different parameters. Population size, dimension, and number of iterations.

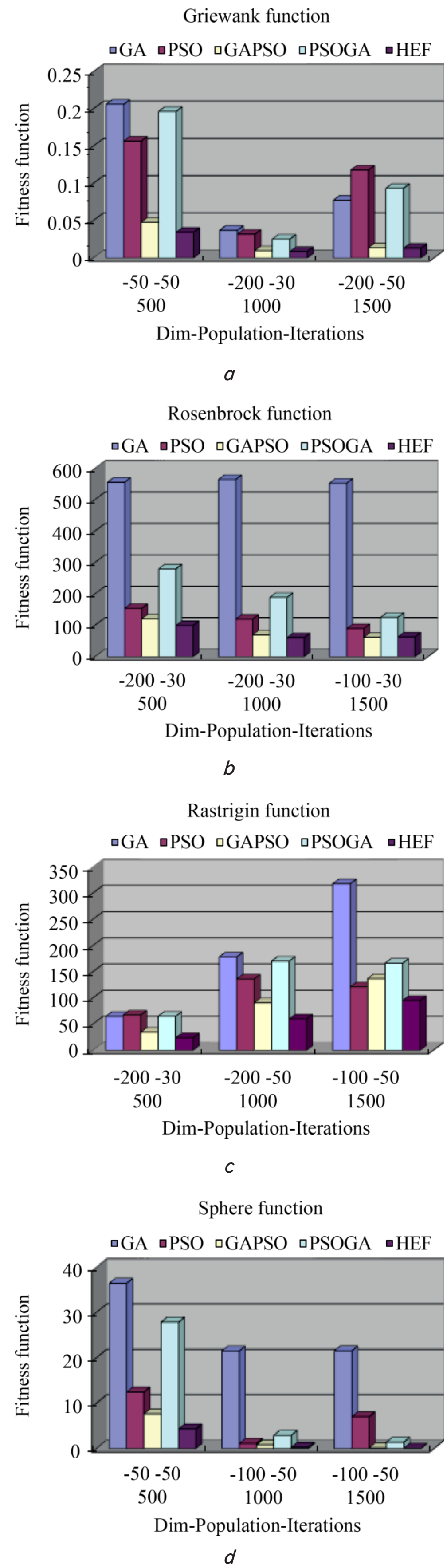

Fig. 4. All comparisons for: *a* — Griewank; *b* — Rosenbrock; *c* — Rastrigin; *d* — Sphere

5. 3. Using HEF in difficult and different situations

Fig. 5 presents the second experiment graphs showing the best average, both HEF and GAPSO had the highest average fitness. Experiments using a unimodal (Rosenbrock) and a multimodal (Griewank) test function of 10, 30, 50 dimensions and 100 populations are depicted in the graphs.

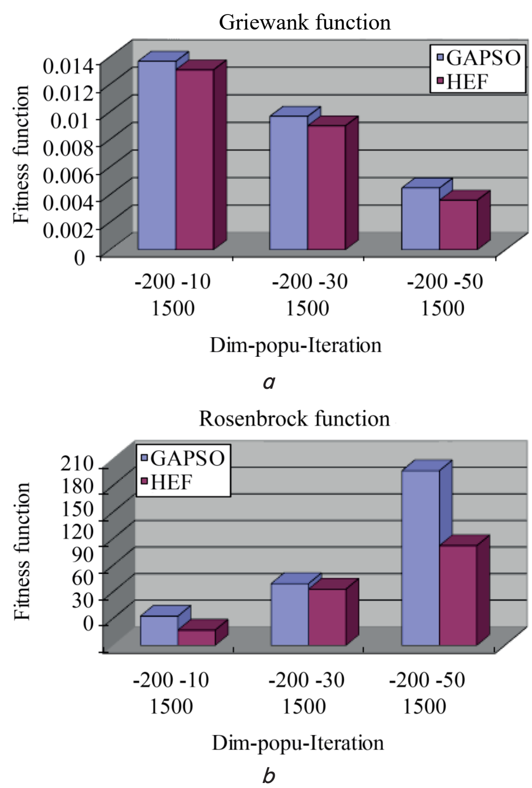

Fig. 5. GAPSO versus HEF model for: *a* – Griewank; *b* – Rosenbrock

Also, Fig. 5 shows that GAPSO has reached a fail point, while HEF is still looking for a better position.

5. 4. Summary and comparison

Table 5 summarizes and compares the final results of HEF with various optimization approaches showing the results of comparison between them.

It is very clear from the foregoing tabulated results that the hybrid evolutionary algorithm with fuzzy logic control has improved FLC's performance significantly for all the test functions.

Table 5

Percentage of average best tests

Time rate	F1	F2	F3	F4	Total		% best
					best	fail	
GA	0	0	0	0	0	108	0
PSO	1	4	1	0	6	102	5.5
GAPSO	1	3	4	5	13	95	12
PSOGA	2	3	3	0	8	100	7.4
HEF	23	17	19	22	81	27	75

5. 5. Simulation technique to apply the HEF algorithm

Mathworks developed MATLAB, a multi-paradigm numerical computing environment and proprietary programming language. It integrates computing, visualization, and programming in a user-friendly environment using mathematical equations. We compared the suggested approach to other algorithms using the simulation results. Our algorithm has the best optimization, according to the comparison results.

From the above results, we conclude that the optimizations and problem-solving efficiency are greatly improved by the proposed HEF algorithm. It can be seen that HEF is superior to traditional GA and PSO and two hybrid GAPSO and PSOGA, which means that the result proves the positive performance of the new algorithm.

6. Discussion of optimization comparison among GA, PSO, GAPSO, POSGA and HEF

To carry out a fair comparison among the GA, PSO, GAPSO, POSGA and HEF algorithms, the initial population was the same in all the experiments. Many researchers provide that small populations may not sufficiently cover the solution space and therefore, given limited function evaluations, may be prone to premature convergence. Larger populations, while providing greater diversity for the search, allow exploration of fewer generations per unit of computational overhead and for limited function evaluations may not converge at all. Nonetheless, in this evaluation, such a comparison provides a good and easy compromise. HEF was examined in the same experimental setup that had previously been used. Tables 3, 4 provide only a few representative functions from each group that were tested and examined for brevity and clarity. Table 5 sums up the cases in which the proposed HEF algorithm outperforms the other algorithms, whereas, Table 4 presents the situations in which HEF's performance is inferior to its counterparts.

In Table 5, even after evaluating the value of the 100 runs, the proposed HEF performs better than all its counterparts. To put it another way, HEF performs slightly better than GAPSO, but still rather close to its counterparts, and clearly better than other techniques. Overall, the results demonstrate that HEF continues to perform admirably on multimodal functions with many minima, as well as exceptionally well on multimodal and unimodal functions with only a small number of local minima. HEF has shown a considerable improvement in performance in all four functions.

For all of the functions in the above experiments, the HEF population size was the same for evolutionary methods and it is realistic to predict even better outcomes. It has been mentioned several times in this paper that HEF performs better than other techniques because of its fuzzy way of continuous switching between GA and PSO during the implementation of the algorithm. Thus, HEF has the benefit of selecting between GA and PSO both at the beginning and at any other point during the run time of the problem. The increase of the population size and number of iterations leads to a more accurate solution in HEF. It can be noticed that the solution with parameter (popu=200, itration=1500) is almost the best solution for all test functions.

To demonstrate the HEF algorithm's efficiency, in this research, several tests were applied to four functions. Each function was applied using five different techniques (algorithms), with each approach testing three distinct populations, and each population applied with three different iterations, for a total of 4·5·3·3·3=540 tests. Although HEF is not always the greatest option, as illustrated in Table 5, it had the highest value with a ratio of 75 % of total tests.

Although successful in some application fields, most of these approaches are ad hoc in design. Without a shared framework, comparing hybrid systems conceptually and comparing their performance is challenging. Due to the limitations of the evolutionary algorithm, the proposed algorithm also has limitations.

To reduce the effect of limitation on evolutionary algorithms, it is open to use other techniques such as Neural network, Ant colony optimization, Bacterial foraging, etc. Just as Fuzzy logic was introduced with GA, PSO to assist the evolutionary algorithm in this research algorithm.

7. Conclusions

1. In this paper, we proposed the Hybrid Evolutionary Algorithm with FLC. It is based on introducing the Evolutionary Algorithm population whose individuals are modified by switching PSO and GA, respectively.

2. The purpose of this switching operation is to prevent premature convergence. The key point of the proposed method is the use of fuzzy system to control the switching operation and when must stop. It allows increasing the influence of the genetic algorithm on the search process when the PSO algorithm is stagnating. And vice versa allows the PSO algorithm to increase its impact on the search process when the GA algorithm is stagnating.

3. We have developed an HEF algorithm as an optimization tool on the MATLAB platform with an accompanying graphical user interface in addition to traditional GA and PSO and two hybrid GAPSO and PSOGA. The performance of the proposed HEF algorithm was confirmed on the four benchmark functions such as Rosenbrock, Sphere, Rastrigin, and Griewank.

4. The results have been compared with some existing well-known methods (GA, PSO, GAPSO and PSOGA). According to the comparison results, our algorithm is capable of clearly giving satisfactory solutions for four benchmark functions.

Acknowledgments

We thank our colleagues from the University of Mosul College of Education for Pure Sciences who provided insight and expertise that greatly assisted the research, especially our colleagues from the computer science department, who may agree with all of the interpretations and conclusions of this paper.

References

1. Ishibuchi, H., Nojima, Y. (2007). Optimization of Scalarizing Functions Through Evolutionary Multiobjective Optimization. Evolutionary Multi-Criterion Optimization, 51–65. doi: https://doi.org/10.1007/978-3-540-70928-2_8
2. Butz, M. V. (2006). Rule-based evolutionary online learning systems. Springer-Verlag, 259. doi: https://doi.org/10.1007/b104669
3. Coello, C. A. C., Lamont, G. B., Van Veldhuizen, D. A. (2007). Evolutionary algorithms for solving multi-objective problems. Springer, 800. doi: https://doi.org/10.1007/978-0-387-36797-2
4. Deng, W., Shang, S., Cai, X., Zhao, H., Song, Y., Xu, J. (2021). An improved differential evolution algorithm and its application in optimization problem. Soft Computing, 25 (7), 5277–5298. doi: https://doi.org/10.1007/s00500-020-05527-x
5. Kuranga, C., Pillay, N. (2021). Genetic programming-based regression for temporal data. Genetic Programming and Evolvable Machines, 22 (3), 297–324. doi: https://doi.org/10.1007/s10710-021-09404-w
6. Lehre, P. K., Nguyen, P. T. H. (2021). Runtime Analyses of the Population-Based Univariate Estimation of Distribution Algorithms on LeadingOnes. Algorithmica, 83 (10), 3238–3280. doi: https://doi.org/10.1007/s00453-021-00862-3
7. Chen, P.-C., Chen, C.-W., Chiang, W.-L., Yeh, K. (2009). A novel stability condition and its application to ga-based fuzzy control for nonlinear systems with uncertainty. Journal of Marine Science and Technology, 17 (4). doi: https://doi.org/10.51400/2709-6998.1985
8. Chang, X., Lilly, J. H. (2004). Evolutionary Design of a Fuzzy Classifier From Data. IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics), 34 (4), 1894–1906. doi: https://doi.org/10.1109/tsmcb.2004.831160
9. Mohammadian, M., Stonier, R. J. (1994). Generating fuzzy rules by genetic algorithms. Proceedings of 1994 3rd IEEE International Workshop on Robot and Human Communication. doi: https://doi.org/10.1109/roman.1994.365902
10. Chen, S.-M., Chen, Y.-C. (2002). Automatically constructing membership functions and generating fuzzy rules using genetic algorithms. Cybernetics and Systems, 33 (8), 841–862. doi: https://doi.org/10.1080/01969720290040867
11. Tsang, C.-H., Kwong, S., Wang, H. (2007). Genetic-fuzzy rule mining approach and evaluation of feature selection techniques for anomaly intrusion detection. Pattern Recognition, 40 (9), 2373–2391. doi: https://doi.org/10.1016/j.patcog.2006.12.009
12. Angeline, P. J. (1998). Evolutionary optimization versus particle swarm optimization: Philosophy and performance differences. Evolutionary Programming VII, 601–610. doi: https://doi.org/10.1007/bfb0040811
13. Eberhart, R. C., Shi, Y. (1998). Comparison between genetic algorithms and particle swarm optimization. Evolutionary Programming VII, 611–616. doi: https://doi.org/10.1007/bfb0040812
14. Pawar, P. M., Ganguli, R. (2007). Genetic fuzzy system for online structural health monitoring of composite helicopter rotor blades. Mechanical Systems and Signal Processing, 21 (5), 2212–2236. doi: https://doi.org/10.1016/j.ymssp.2006.09.006
15. Chu, B., Kim, D., Hong, D., Park, J., Chung, J. T., Chung, J.-H., Kim, T.-H. (2008). GA-based fuzzy controller design for tunnel ventilation systems. Automation in Construction, 17 (2), 130–136. doi: https://doi.org/10.1016/j.autcon.2007.05.011

16. Franke, C., Hoffmann, F., Lepping, J., Schwiegelshohn, U. (2008). Development of scheduling strategies with Genetic Fuzzy systems. Applied Soft Computing, 8 (1), 706–721. doi: https://doi.org/10.1016/j.asoc.2007.05.009
17. Tang, J., Zhang, G., Lin, B., Zhang, B. (2010). A Hybrid PSO/GA Algorithm for Job Shop Scheduling Problem. Advances in Swarm Intelligence, 566–573. doi: https://doi.org/10.1007/978-3-642-13495-1_69
18. Robinson, J., Sinton, S., Rahmat-Samii, Y. (2002). Particle swarm, genetic algorithm, and their hybrids: optimization of a profiled corrugated horn antenna. IEEE Antennas and Propagation Society International Symposium (IEEE Cat. No.02CH37313). doi: https://doi.org/10.1109/aps.2002.1016311
19. Dziwinski, P., Bartczuk, L. (2020). A New Hybrid Particle Swarm Optimization and Genetic Algorithm Method Controlled by Fuzzy Logic. IEEE Transactions on Fuzzy Systems, 28 (6), 1140–1154. doi: https://doi.org/10.1109/tfuzz.2019.2957263
20. Ruan, X., Wang, J., Zhang, X., Liu, W., Fu, X. (2020). A Novel Optimization Algorithm Combing Gbest-Guided Artificial Bee Colony Algorithm with Variable Gradients. Applied Sciences, 10 (10), 3352. doi: https://doi.org/10.3390/app10103352
21. Gao, W., Liu, S. (2012). A modified artificial bee colony algorithm. Computers & Operations Research, 39 (3), 687–697. doi: https://doi.org/10.1016/j.cor.2011.06.007
22. Xue, Y., Jiang, J., Zhao, B., Ma, T. (2017). A self-adaptive artificial bee colony algorithm based on global best for global optimization. Soft Computing, 22 (9), 2935–2952. doi: https://doi.org/10.1007/s00500-017-2547-1