PARTICLE SWARM OPTIMIZATION AND ARTIFICIAL BEE COLONY APPROACHES TO OPTIMIZE OF SINGLE INPUT-OUTPUT FUZZY MEMBERSHIP FUNCTIONS

Ebru TURANOĞLU[a,1], Eren ÖZCEYLAN[b], Mustafa Servet KIRAN[c]

[a] *Selcuk University, Industrial Engineering Department, Turkey,ebruturanoglu@gmail.com,+903322232041*
[b] *Selcuk University, Industrial Engineering Department, Turkey, eozceylan@selcuk.edu.tr, +903322232075*
[c] *Selcuk University, Computer Engineering Department, Turkey, mskiran@selcuk.edu.tr, +903322231992*

Abstract: Determination of the fuzzy membership functions in a given fuzzy logic system is the key factor for resulting in the best performance. Thus, in this study, particle swarm optimization (PSO) and artificial bee colony (ABC), relatively new member of swarm intelligence, are used to adjust the shape of fuzzy membership functions, respectively. Proposed methods have been implemented and compared for a single input-output (SI-O) fuzzy system. The obtained results show that using PSO and ABC methods are capable and effective to find optimal values of fuzzy membership functions in a reasonable time.

Keywords: artificial bee colony, fuzzy membership functions, optimization, particle swarm optimization

1. Introduction

Fuzzy logic is conceived as a better method for sorting and handling data, also it has proven to be an excellent choice for many control system applications due to mimicking human logic. It uses an imprecise but very descriptive language to deal with input data more like a human operator. The key component of a fuzzy system is the membership function for defining the fuzzy sets and the rule base for producing the output (Cheong and Lai, 2007). Thus, establishing the fuzzy rule base and defining the limits of the best membership functions of input/output variables become important. Although fuzzy logic control encodes expert knowledge in a direct and easy way using rules with linguistic labels, there are some drawbacks to design approach: (i) relying on a priori knowledge of the human expert, depending on the quality of expertise and consuming much time to design the membership functions; (ii) leading to poor controller performance and possibly to instability on account of wrong fuzzy rules or membership functions (Yeh and Li, 2004). Besides mentioned issues, determination of the most appropriate membership functions and its boundaries consisting of the rule base make significant impacts on the final performance of the controller. In order to get the best performance from fuzzy logic system (FLS), the membership functions must be optimally determined (Bağış, 2003).

Applying optimization techniques are required to determine the parameters that define the membership functions. Being a daunting task of parameter optimization problem, some methods such as genetic algorithm (GA) (Arslan and Kaya, 2001; Kaya et al., 2001; Kaya and Alhajj, 2006; Zhang et al. 2009), tabu search (TS) (Bağış, 2003; Li et al. 2004), simulated annealing (SA) (Yanar and Akyürek, 2011), clonal selection (CS) (Şakiroğlu and Arslan, 2007; Acılar and Arslan, 2011), artificial neural network (ANN) (Abraham, 2002) and particle swarm optimization (PSO) (Al-Jaafreh and Al-Jumaily, 2007; Omizegba and Adebayo, 2011) have been applied to optimize the parameter of fuzzy membership functions. Arslan and Kaya (2001) adjusted the shape of membership functions and determined membership functions of a single input and output fuzzy system using GA approach. Kaya and Alhajj (2006) proposed a GA-based approach to adjust and optimize membership functions, which was essential in finding interesting weighted association rules from quantitative transactions, based on support and confidence specified as linguistic terms. Zhang (2009) presented to automatically design and optimized the fuzzy membership function's parameters using genetic learning and turning based on real-coded GA. Bağış (2003) examined a simple and effective method for the optimum determination of the membership functions based on the use of TS algorithm. Yanar and Akyürek (2011) utilized SA algorithm to fine-tune Mamdani type fuzzy models and reported experiments

[1] Corresponding author

performed to determine the effects of SA parameters on the final prediction error. A CS algorithm was used to adjust the shape of membership functions for a single input and output fuzzy system by Şakiroğlu and Arslan (2007). Acılar and Arslan (2011) optimized membership functions of the multiple input -output (MI-O) fuzzy system using a method based on CS algorithm and coded using GA and binary particle swarm optimization (BPSO) for comparisons. They obtained that CS algorithm was advantageous than GA and BPSO determining optimum values of membership functions for a fuzzy system. Abraham (2002) used to determine the parameters of fuzzy inference system a meta-heuristic approach combining neural network learning and evolutionary computation. Al-Jaafreh and Al-Jumaily (2007) proposed a new method to optimize parameters of the primary membership functions of type-2 fuzzy system by PSO to improve the performance and increase the accuracy of type-2 fuzzy system model. Omizegba and Adebayo (2009) presented a method for determining the optimal shapes and span of membership function of a fuzzy system based on a modeling performance measure using PSO algorithm.

In this paper, we optimize membership functions of the single input-output fuzzy system (SI-O) using particle swarm optimization (PSO) and artificial bee colony (ABC) and compare each other. As it seen from literature above, there has been no study which uses ABC algorithm to optimize membership functions of any fuzzy system. To the best knowledge of the authors, this is the first study which uses ABC algorithm to optimize membership functions of the single input-output fuzzy system (SI-O). The rest of the paper is organized as follows. Section 2 presents the concept of PSO and ABC algorithms. In Section 3, how the SI-O membership function is optimized using PSO and ABC algorithms shown through an illustrative example and experimental results are generated. Finally, the study is concluded in Section 4 with suggestions on future researches.

2. Swarm Intelligence

Swarm intelligence (SI) focuses on insect intelligent behaviors in order to develop some meta-heuristics which can mimic the abilities of insects in solving their problems. Interactions between insects contribute to the collective intelligence of social insect colonies and these interactions have been successfully adapted to solve scientific and real world optimization problems (Oliveira and Schirru, 2011). PSO and ABC algorithms are some of the well-known algorithms that mimic insect behavior in problem modeling and solution.

2.1 Particle Swarm Optimization

PSO is one of the recent meta-heuristic techniques proposed by Kennedy and Eberhart (1995) based on natural flocking and swarming behavior of birds and insects. The PSO algorithm operates iteration by iteration and solution produced in each iteration is compared self-local best and global best of swarm. The particles try to achieve to global minimum by using global and local best information. V velocity of particle, X particle vector and N number of particle, the new position of particle is computed according to Eqs. (1-2).

$$v_{id}(t+1) = w.v_{id}(t) + c_1.rand_1.\left(pbest_{id}(t) - x_{id}(t)\right) + c_2.rand_2.\left(gbest_{id}(t) - x_{id}(t)\right) \quad (1)$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1) \quad (2)$$

where c_1 and c_2 determine the relative influence of the social and cognitive components (bias factors), $rand_1$ and $rand_2$ denote two random numbers uniformly distributed in the interval [0, 1]. w is a parameter called inertia weight used to control the impact of the previous velocities on the current one (Shi and Eberhart, 1998). Inertia weight is dynamically calculated by depending on iteration number as follow:

$$w = (t - i)/t \quad (3)$$

where t is the maximum iteration number and i is the current iteration index. In this way, while the particles approach to global minimum, velocities of particles are decreased during the iterations.

2.2 Artificial Bee Colony Algorithm

ABC algorithm was proposed by Karaboğa (2005), inspired by the foraging and waggles dance behaviors of honeybees. There are three kinds of honey bee in ABC to forage food source. They are employed bees, scout bees and onlooker bees. The tasks of these bees are to collect nectar around the hive, to random search for the nectar sources and to find the food sources based on the information received from the employed bees waggle dance. In ABC, food searching and nectar foraging around the hive are performed by scout, onlooker and employed bees collectively. In the algorithm ABC, for generating an initial solution for employed bee b, the Eq. (4) is used.

$$x_b^j = x_{min}^j + rand[0,1] * \left(x_{max}^j - x_{min}^j\right), for\ all\ j = 1,2,\dots,D \tag{4}$$

where, x_b^j is a parameter to be optimized for the employed bee b on the dimension j of the D-dimensional solution space, x_{max}^j and x_{min}^j are the upper and lower bounds for x_b^j, respectively. In both onlooker bee and employed bee phases, the food positions in the dimension *j* are obtained by the Eq. (5).

$$v_b^j = x_b^j + \Phi\left(x_b^j - x_k^j\right)\ j \in \{1,2,\dots,D\}, k \neq b\ and\ k \in \{1,2,\dots.,n\} \tag{5}$$

where, x_b^j is employed bee b, v_b^j is the new solution for x_b^j in the dimension j, x_k^j is a neighbor bee of x_b^j in employed bee population. Here Φ is a number randomly selected in the range [-1, 1], *n* is number of the employed bees, and $j \in \{1,2,\dots,D\}\ and\ k \in \{1,2,\dots.,n\}$ are selected randomly. In order to generate a new food position, every onlooker bee memorizes the solution of one of *n* employed bees based on fitness values of the employed bees. The probability p_b of that an onlooker bee will select the selection of the solution of the employed bee *b* is obtained as follows:

$$p_b = \frac{fit_b}{\sum_{j=1}^{D} fit_j} \tag{6}$$

where, fit_b is the fitness value of employed bee *b* obtained is as follows:

$$fit_b = \begin{cases} \frac{1}{1+f_b} & if\ (f_b \geq 0) \\ 1 + abs(f_b) & if\ (f_b \leq 0) \end{cases} \tag{7}$$

where, f_b is the object function. In addition, in the scout bee phase of the ABC, Eq.(4) is used in order to generate new solution for the scout bee and all the onlooker bees use the Eq. (5) of so as to improve the solution.

3. Computation of the membership function using PSO and ABC Algorithms

If rules table and shape of membership functions of a fuzzy system is known previously, a suitable placement of membership functions could be determined. However, a membership function should be used out under condition which is pre-known its mathematical function's model. Because of this, the membership optimization problem can be reduced to parameter optimization problem (Acılar and Arslan, 2011). In this study, mentioned parameter optimization problem is solved by PSO and ABC algorithms. Proposed two approaches which were implemented in MATLAB 2009 are applied for SI-O fuzzy system and placed the membership functions on input and output axis and presented below.

3.1 Single Input-Output Fuzzy System Implementation

It is assumed that there are two different membership functions for input and output of considered system by Arslan and Kaya (2001). The membership functions are called input (*x*) and output (*y*), *x* uses slow and fast; *y* uses easy and difficult. In this case, the linguistic rules are determined as follow:

Rule 1: If x is slow then y is easy

Rule 2: If x is fast then y is difficult

If the ranges of input and variables are assumed as [0-7] and [0-49], respectively, then the triangular membership functions of fuzzy system will be as shown in Figure 1.

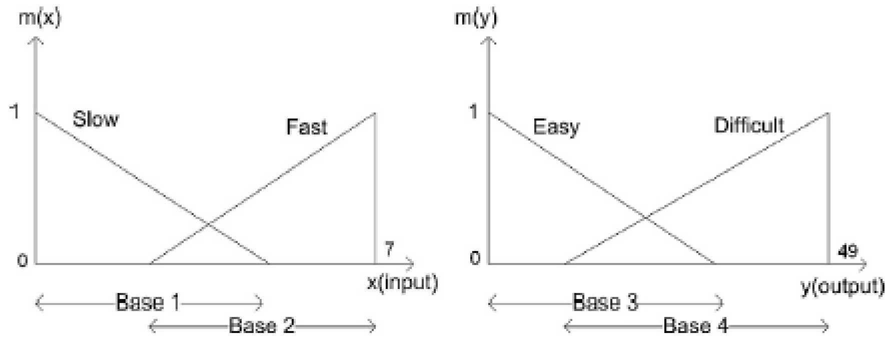

Figure 1. The membership functions of a fuzzy system for input and output (Arslan and Kaya, 2001)

As it seen from figure, right sides of right triangle are restricted by lower and upper limits. The output values have to be known amongst the input values to compute of fitness function of the problem. For that reason, following input and output values are used.

Input: x_i*: {1, 3, 5, 7} Output:* y_i*: {1, 9, 16, 49}, i= 1, 2, 3 and 4.*

Expected from PSO and ABC models is to find the base lengths (Bases 1, 2, 3 and 4 in Figure 1) of the right triangles. Here, it should be noted that both [0-7] input range and [0-49] output range are represented by 4 dimensions in proposed models. The affinity function which is the fitness function of the proposed model is calculated as follows (Acılar and Arslan, 2011):

Affinity= MaxError-TotalError (8)

MaxError and *TotalError* are calculated using Eqs (9) and (10) given below, respectively.

MaxError= $\sum_{i=1}^{n}(y_i - y_{max})^2$ (9)

TotalError= $\sum_{i=1}^{n}(y_{PSO\ and\ ABC} - y_i)^2$ (10)

where y_i is the output of i^{th} reference input and output, $y_{PSO\ and\ ABC}$ obtained by PSO and ABC algorithms, is input of output, and *n* is the number of input-output data pairs given. The aim of the proposed models is to approximate the TotalError to zero as close as possible. Because *TotalError* is as likely to approach to zero, $y_{PSO\ and\ ABC}$ is close to y_i too, this case is the suitable solution. In order to prevent affinity function from getting negative values, the maximum number 4480 is used and this number is also *MaxError* at the same time (Arslan and Kaya, 2001). This number is obtained using Eq. (9).

Using mentioned input and output values, PSO and ABC models were coded with MATLAB 2009 and run on an i3, 1.33 GHz, 4 GB RAM notebook computer. While c_1 and c_2 values are fixed as 2, *w* is started with 1 and decreased per iteration for PSO algorithm. Particle and bee population are set to 20 and models are run until 100 iteration ends for both algorithm. All models are run 10 times and best values are considered. The optimal solution obtained PSO and ABC algorithms are given in Table 2 and their membership function shapes are depicted in Figure 2 and 3, respectively.

Table 1. Optimal solutions of PSO and ABC

Approaches	Base 1	Base 2	Base 3	Base 4	y_1(x=1)	y_2(x=3)	y_3(x=5)	y_4(x=7)	Affinity	CPU time
PSO	5.27	3.98	24.17	24.92	8.16	9.59	36.18	40.88	4362.45	73.12
ABC	7.00	2.91	25.53	25.46	8.51	9.47	25.00	41.34	4364.56	33.44

According Table 1, for the best result of PSO, the base values for x are 5.27 and 3.98. In the same way, the base values for y are 24.17 and 24.92. Here, the centroid method is used as defuzzification process in Mamdani system and outputs are calculated as 8.16, 9.59, 36.18 and 40.88 whereas the real outputs are 1, 3, 5 and 7. So *TotalError* at this point equals to 117.55. For the best results of ABC, *TotalError* is obtained as 115.44 with a 0.04% improvement when all calculations are done. As a result, ABC gives better results than PSO in both affinity and CPU time.

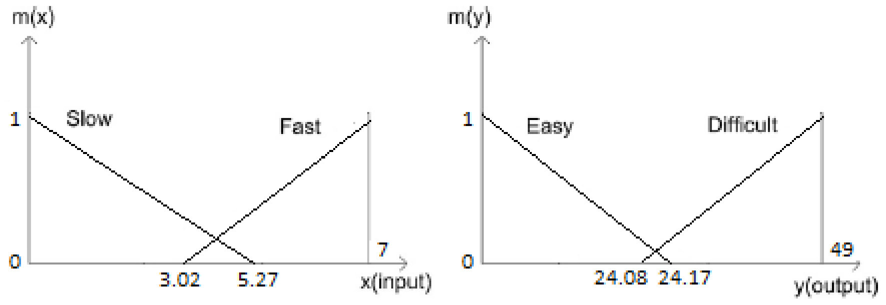

Figure 2. The membership function shape of the optimal solution obtained by PSO

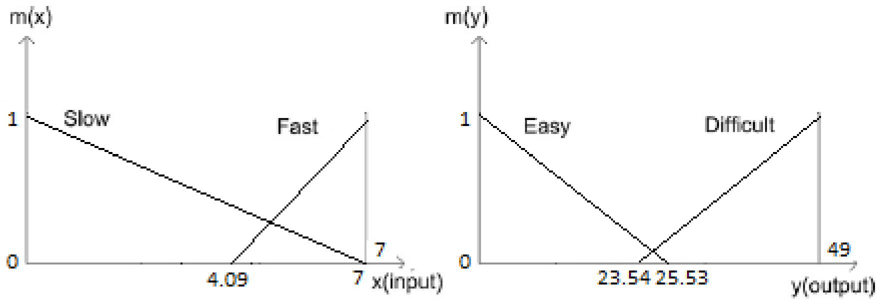

Figure 3. The membership function shape of the optimal solution obtained by ABC

3.2 Experimental Results of Parameter Changing

Although there are not many parameters that need to be tuned in PSO and ABC, only population size and iteration number are investigated to see the changes on optimal solutions in this section. In order to compare both algorithms on optimization fuzzy membership function, two types of experiments were performed:
Exp#1: While the population size is 20, iteration number is increased 5 times from 50 to 250 steps 50.
Exp#2: While the iteration number is 100, population size is increased 5 times from 10 to 50 steps 10.

Above experiments are applied according to example in Section 3.1, 10 runs were executed for each algorithm and best results are considered.

Table 2. Obtained results according to *Exp#1*

Iteration	Approaches	Base 1	Base 2	Base 3	Base 4	y_1	y_2	y_3	y_4	Affinity	CPU time
50	PSO	7.00	4.08	22.78	26.22	7.58	9.89	28.47	40.42	4350.80	37.96
	ABC	7.00	2.85	23.72	25.32	7.90	8.79	24.92	40.72	4363.74	14.94
100	PSO	5.27	3.98	24.17	24.92	8.16	9.59	36.18	40.88	4362.45	73.12
	ABC	7.00	2.91	25.53	25.46	8.51	9.47	25.00	41.34	4364.56	33.44
150	PSO	5.26	3.99	23.52	25.47	7.93	9.33	36.23	40.67	4362.47	110.85
	ABC	7.00	2.92	25.40	23.65	8.47	9.42	25.03	41.27	4364.34	43.15
200	PSO	5.44	4.02	23.26	25.73	7.83	9.57	34.65	40.58	4361.68	141.61
	ABC	7.00	2.92	25.46	23.55	8.49	9.44	25.02	41.31	4364.51	57.14
250	PSO	5.27	3.99	24.01	24.98	8.10	9.52	36.21	40.83	4362.50	186.36
	ABC	7.00	2.87	24.02	24.97	8.00	8.91	24.97	40.83	4364.28	73.57

While the population size is 20 and iteration number is increased, as it seen affinity of the models are also generally increased (Table 2). In all iteration rows, ABC gives better results than PSO in better CPU time. Best result (4364.56) is obtained with ABC in 33.44 seconds while the iteration number is 100.

Table 3. Obtained results according to *Exp#2*

Pop Size	Approaches	Base 1	Base 2	Base 3	Base 4	y_1	y_2	y_3	y_4	Affinity	CPU time
10	PSO	7.00	4.08	23.03	25.96	7.66	9.95	28.44	40.50	4350.77	41.40
	ABC	6.91	2.75	19.51	29.52	6.47	7.22	25.08	39.32	4353.15	14.54
20	PSO	5.27	3.98	24.17	24.92	8.16	9.59	36.18	40.88	4362.45	73.12
	ABC	7.00	2.91	25.53	25.46	8.51	9.47	25.00	41.34	4364.56	33.44
30	PSO	7.00	4.09	21.48	27.51	7.14	9.82	28.67	39.99	4350.73	108.45
	ABC	7.00	2.89	24.62	24.48	8.20	9.13	25.03	41.00	4364.03	43.60
40	PSO	5.29	4.00	23.19	25.80	7.82	9.18	35.86	40.56	4362.14	139.13
	ABC	7.00	2.90	25.34	23.75	8.45	9.40	24.98	41.24	4364.11	58.96
50	PSO	5.28	4.00	23.54	25.45	7.94	9.33	36.05	40.67	4362.39	174.85
	ABC	7.00	2.85	23.42	25.68	7.80	8.68	24.97	40.60	4363.08	74.92

Table 3 gives the results of changing population size of two approaches. As it seen from table, ABC approach gives also the best results in all population size classifications. According to Table 3, while population size is 20, best affinity is obtained with ABC. Finally, these experiment shows that population size and iteration number should be fixed as 20 and 100, respectively to obtain desired results in fuzzy membership functions.

4. Conclusion

How the membership functions compute as a parameter optimization problem using ABC and PSO is described for single input–output (SI–O) fuzzy system in this study. ABC which is applied first on this area and PSO models coded by MATLAB 2009 are compared through a small SI-O fuzzy system. Firstly, the algorithms are run 10 times and best results are examined. Secondly, two experiments based on population sizes and iteration numbers are generated to see the effects of changing parameters of algorithms. As a conclusion, it could be said that ABC and PSO can be used while determining optimum values of membership functions for a fuzzy system and using of ABC is advantageous than using PSO in a reasonable time. For future, different membership functions, multiple input output fuzzy systems and different heuristic algorithms should be considered.

References

Abraham, A., 2002. EvoNF: a framework for optimization of fuzzy inference systems using neural network learning and evolutionary computation. IEEE International Symposium on intelligent Control, Vancouver, Canada, 27-30.

Acılar, M.A., Arslan A., 2011. Optimization of multiple input-output fuzzy membership functions using clonal selection algorithm. Expert Systems with Applications 38, 1374-1381.

Al-Jaafreh, M., Al-Jumaily, A., 2007. Training type-2 fuzzy system by particle swarm optimization. IEEE Congress on Evolutionary Computation, 3442-3446.

Arslan, A., Kaya, M., 2001. Determination of fuzzy logic membership functions using genetic algorithms. Fuzzy Sets and Systems 118, 297-306.

Bağış, A., 2003. Determining fuzzy membership functions with tabu search-an application to control. Fuzzy Sets and Systems 139, 209-225.

Cheong, F., Lai, R., 2007. Designing a hierarchical fuzzy logic controller using the differential evolution approach. Applied Soft Computing 7, 481-491.

Kaya, M., Karcı, A., Arslan, A., 2001. Determination of membership functions in multiple input-output fuzzy systems by genetic algorithms. In Proceeding of 3rd International Symposium on Intelligent Manufacturing System. Sakarya. Turkey.

Karaboğa, D., 2005. An idea based on honeybee swarm for numerical optimization. Technical Report TR06. Erciyes University. Engineering Faculty. Computer Engineering Department.

Kaya, M., Alhajj, R., 2006. Utilizing genetic algorithms to optimize membership functions for fuzzy weighted association rules mining. Applied Intelligence 24(1), 7-15.

Kennedy, J., Eberhart, R.C., 1995. A new optimizer using particles swarm theory. In: Proceedings of Sixth International Symposium on Micro Machine and Human Science, 39-43.

Li, C, Liao, X., Yu, J., 2004. Tabu search for fuzzy optimization and applications. Information Sciences 158, 3-13.

Şakiroğlu, A.M., Arslan, A., 2007. Clonal selection principle for fuzzy membership function optimization. Lecture Notes in Computer Science 1443). Berlin, Heidelberg: Springer-Verlag.

Oliveira, I.M.S., Schirru, R., 2011. Swarm intelligence of artificial bees applied to in-core fuel management optimization. Annals of Nuclear Energy 38(5), 1039-1045.

Omizegba, E.E., Adebayo , G.E., 2009. Optimizing fuzzy membership functions using particle swarm optimization. IEEE International Conference on Systems, Man and Cybernetics, 3866-3870.

Shi, Y., Eberhart, R., 1998. A Modified particle swarm optimizer. Evolutionary Computation Proceedings, IEEE World Congress on Computational Intelligence, 69-73.

Zhang, H.-X., Wang, F., Zhang, B., 2009. Genetic optimization of fuzzy membership functions. Proceedings of the International Conference on Wavelet Analysis and Pattern Recognition, 465-470.

Yanar, T.A., Akyürek, Z., 2011. Fuzzy model tuning using simulated annealing. Expert Systems with Applications 38(7), 8159-8169.

Yeh, Z-M., Li, K-H., 2004. A systematic approach for designing multistage fuzzy control systems. Fuzzy Sets and Systems 143, 251-273.