2014 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)
July 6-11, 2014, Beijing, China

Structure and Parameter Optimization of FNNs Using Multi-objective ACO for Control And Prediction

Chia-Feng Juang and Chia-Hung Hsu

Abstract—**Design of a fuzzy neural network (FNN) consists of optimization of network structure and parameters. The objectives are to minimize the network model size with minimum training error at the same time, causing a conflict between the two objectives in the design problem. To address this problem, the multi-objective, rule-coded, advanced, continuous-ant-colony optimization (MO-RACACO) is applied to design FNNs in this paper. The MO-RACACO-designed FNNs are applied to time sequence prediction and nonlinear control problems to verify its performance. Performance of this approach is verified through three simulation examples with comparisons with various multi-objective population-based optimization algorithms and detailed discussions of the results. The results show that the MO-RACACO-based FNN design approach outperforms the multi-objective population-based algorithms used for comparisons in the control and prediction examples.**

I. Introduction

FUZZY neural networks (FNNs) have been successfully applied to different areas such as control, prediction, and pattern recognition [1]-[6]. FNNs are generally built via learning from data and the design consists of structure and parameter learning. Structure learning includes the determination of rules and the number of fuzzy sets in each input variable. Parameter learning determines the optimal antecedent and consequent parameters in fuzzy rules. One popular approach for structure learning is clustering in the input or input-output space. Based on this approach, different learning algorithms have been proposed. In [1][5], the firing strength of a rule is used as the criterion for the generation of a new rule. For a given datum, if the maximum firing strength is smaller than a threshold, then a new rule is generated to cover it. This structure learning algorithm ensures that all input data are properly covered by the rules. This approach has been extended to structure learning in a self-evolving interval type-2 fuzzy neural network in which the average of upper and lower rule firing strengths is used as the criterion for the generation of an interval type-2 fuzzy rule [7]. An online version of subtractive clustering was proposed for structure learning in [2]. In this approach, the potential of a data was compared against the potential of existing rules to determine whether or not a new rule should be added. For these clustering-based structure learning approaches, only one structure is determined at a time. As to the parameter optimization, a gradient descent algorithm has been widely used; however, this algorithm suffers from the local solution problem. Several evolutionary fuzzy systems that optimize fuzzy systems using genetic algorithms [8], particle swarm optimization [9], or continuous ant colony optimization [10] have been proposed aiming to avoid the local-solution problem. However, these studies focus on the optimization the fuzzy systems with a fixed structure. In general, an FNN with a larger number of rules (i.e., a larger network size) tends to show a smaller training error after optimization. Therefore, there is a trade-off between network size and training error. This design problem can be regarded a multi-objective optimization problem, and a list of Pareto optimal solutions would be helpful to the users for selection according to their preferences. To address the above problems, multi-objective population–based optimization (MOPO) algorithms may be employed.

Several MOPO algorithms have been proposed in literature, among which the strength Pareto-evolutionary algorithm 2 (SPEA2) [11] and the non-dominated sorting genetic algorithm II (NSGA-II) [12] have been applied to the optimization of fuzzy systems for regression problems [13][14]. These approaches select significant fuzzy rules from all the possible rules generated from a grid-type partition, which faces the curse of dimensionality in candidate rule base for high-dimensional inputs. A multi-objective, rule-coded, advanced, continuous-ant-colony optimization (MO-RACACO) was proposed to address the problem of fuzzy control of a mobile robot for wall-following control [15], where the input space in a fuzzy system is flexibly partitioned. This paper applies the MO-RACACO to FNN-based control and sequence prediction problems and studies its performance in the application. Both the control/prediction error and the network size are used to evaluate the performance of an individual (FNN). Distributions of the Pareto-optimal solutions in the two-objective function space provide clear observation of the results and analysis of the performance among different optimization algorithms. Performance of the MO-RACACO is compared with various MOPO algorithms to show its advantage in the design of FNNs for control and prediction applications.

This paper is organized as follows. Section II describes the structure and node functions in an FNN and the configuration of the MO-RACACO-based FNN control and prediction problems. Section III describes the MO-RACACO algorithm for FNN optimization. Section IV presents simulation results of the MO-RACACO in three control and prediction examples and

The authors are with the Department of Electrical Engineering, National Chung-Hsing University, Taichung 402, Taiwan, R.O.C. (e-mail: cfjuang@dragon.nchu.edu.tw).

This work was supported by the National Science Council, Taiwan, under Grant NSC 102-2221-E-005-056-MY2.

978-1-4799-2072-3/14/$31.00 ©2014 IEEE

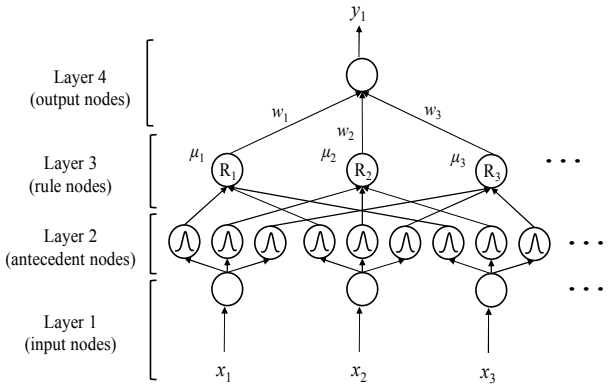


Fig. 1 Structure of the FNN.

performance comparisons with various MOPO algorithms. Finally, Section V presents conclusions.

II. Evolutionary FNN For Control And Prediction

A. FNN Structure and Functions

The FNN to be optimized consists of zero-order Takagi-Sugeno (TS)-type fuzzy rules, each of which is of the following form:

$$\begin{aligned} &\text{Rule } k\text{: If } x_1 \text{ is } A^{k1} \text{ And, ... , And } x_n \text{ is } A^{kn}, \\ &\qquad\qquad \text{Then } u \text{ is } w^k \ , \ k = 1,...,r \end{aligned} \tag{1}$$

where x_i is an input variable, A^{ki} is a fuzzy set, u is a output variable, w^k is a real consequent value, and r is the total number of rules. Fig. 1 shows the structure of the FNN with the followings detail the function in each layer.

Layer one: Each node in this layer represents one input variable x_i. The node transmits the input variable to the next layer after a proper scaling operation so that each scaled input variable falls in the same search range.

Layer two: Each node in this layer represents a fuzzy set A_{kj} and functions as a membership function, so the node output is a membership value. This paper uses the Gaussian membership function described as follows:

$$M^{ki}(x_i) = \exp\left\{-\left(\frac{\left(x_i - m^{ki}\right)^2}{(\sigma^{ki})^2}\right)\right\} \tag{2}$$

where m^{ki} and σ^{ki} denote the center and width of the fuzzy set. In this layer, the number of nodes connected to each input variable is equal to the number of rule nodes in layer three and is optimized through the MO-RACACO.

Layer three: The number of nodes in this layer is equal to the number of rules r and determines the network size. A node represents a fuzzy rule and computes the firing strength of a rule by using the following algebraic product operation:

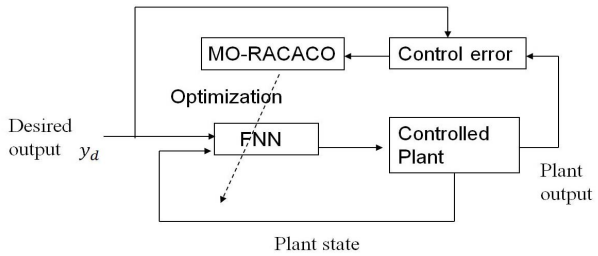


Fig. 2 The evolutionary FNN control configuration using the MO-RACACO.

$$\mu^k(\vec{x}) = \prod_{i=1}^{n} M^{ki}(x_i) \tag{3}$$

where $\vec{x} = [x_1,...,x_n]$.

Layer four: Each node in this layer functions as a defuzzifier by using a weighted average operation. The consequent parameter w^k functions as a link weight. The defuzzified output is given as follows:

$$u = \left(\sum_{k=1}^{r} w^k \mu^k\right) / \sum_{k=1}^{r} \mu^k \ . \tag{4}$$

Construction of the FNN consists of structure and parameter determination. This paper applies the MO-RACACO to optimize the FNN structure and parameter. The optimization objectives include minimization of the network size (i.e., the number of rule nodes) and training error. Because of the trade-off between the two objectives, the MO-RACACO is used to find Pareto optimal solutions of the optimization problem.

B. MO-RACACO-based FNN Control and Prediction

MO-RACACO-based FNN control the output of a nonlinear plant to track a desired trajectory is investigated for a comparison with the Pareto-optimal solutions obtained with various MOPO algorithms and to demonstrate the superiority of the MO-RACACO. Fig. 2 shows the configuration of the evolutionary FNN control, where the approach can be applied to the plant with unknown mathematical model. Because the precise controller input-output training data is either costly to obtain or unavailable, the MO-RACACO algorithm is adopted for controller design. In this configuration, no data are collected in advance; all data are generated online when control begins. Each individual in the MO-RACACO represents an FNN. The inputs of an FNN are the desired control output $y_d(t+1)$ and current states(s) of the controlled plant. At each time step t, an FNN is applied to control the plant to generate a new controlled output $y(t+1)$. The error between $y(t+1)$ and the desired output $y_d(t+1)$ is computed for control performance evaluation.

For the sequence prediction problem, the input-output sequence is collected off-line in advance for FNN optimization. The inputs of an FNN are the past values and the output is the predicted value. The errors between the predicted and the actual outputs are used to evaluate the prediction performance.

In the FNN-based control and prediction problems, two

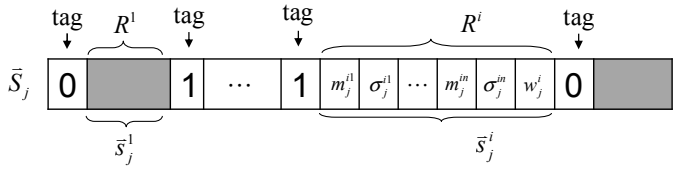


Fig. 3. Rule-coded individual (solution) in the MO-RACACO, where the shaded region represents a null solution.

objective functions, minimization of the number of rule nodes (function f_1) and minimization of the tracking/prediction error (function f_2), are defined to evaluate the performance of an FNN. Advantage of using the MO-RACACO in the FNN optimization problem is verified through performance comparisons with various MOPO algorithms in three examples with detailed performance analyses.

III. MO-RACACO Algorithm

The MO-RACACO was proposed for fuzzy control of a mobile robot in [15]. This paper applies the MO-RACACO to different FNN-based control and prediction problems. Fig. 3 shows the coding of an individual representing an FNN. Each rule R^i is assigned with a tag taking the value of "1" or "0" representing an active or a null rule. The maximum number of possible rules is $\tilde{M}$. Each solution vector $\vec{S}_j$ is described as

$$\vec{S}_j = [\vec{s}_j^{\,1}, \vec{s}_j^{\,2}, \ldots, \vec{s}_j^{\,\tilde{M}}] \in \mathbb{R}^{(2n+1)\tilde{M}}. \quad (5)$$

If the rule R^i is active, then $\vec{s}_j^{\,i}$ is described as

$$\begin{aligned} \vec{s}_j^{\,i} &= [s_j^{i1}, s_j^{i2}, \ldots, s_j^{iD}] \\ &= [m_j^{i1}, \sigma_j^{i1}, \ldots, m_j^{in}, \sigma_j^{in}, w_j^i] \in \mathbb{R}^D,\ D = 2n+1 \end{aligned}; \quad (6)$$

otherwise, $\vec{s}_j^{\,i} \in \mathbb{R}^D$ is a null vector. The MO-RACACO works with a fixed colony size of N solutions (FNNs). The solutions are sorted form the best to the worst according the non-dominated sorting approach and the crowding distance in the NSGA-II. At each iteration I_c, N new solutions are generated using the three-phase approach described below. Among the N original and the N new solutions, only the top-half best performing solutions are reserved. The algorithm ends when a pre-defined maximum number of iterations $I_{\max}$ is reached.

Fig. 4 shows the three-phase operation in generating a new solution. A pheromone level, τ_i, is deposited on a path segment with $\tau_1 > \tau_2 > \cdots > \tau_N$. A path segment with a stronger pheromone level is selected with a higher probability. In phase one, a temporary solution $\tilde{\vec{S}}_j = [\tilde{\vec{s}}_j^{\,1}, \ldots, \tilde{\vec{s}}_j^{\,\tilde{M}}]$ is generated from an ant path selected from an elite or a tournament selection. At each iteration I_c, the elite tournament selection generates $L = N \cdot I_c / I_{\max}$ solutions and the tournament selection generates the others. In phase two, a rule-based mutation

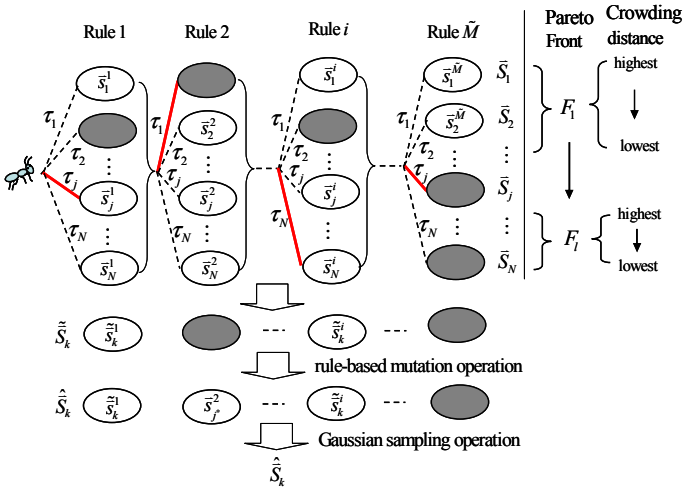


Fig. 4. Graphic representation of the three-phase new solution generation process in the MO-RACACO, where a dark node represents a null rule.

operation is introduced to activate the null rules in the temporary solution vectors with a mutation probability of 0.5. The objective is to avoid premature convergence to FNNs with smaller network sizes and spread the non-dominated solutions over different network sizes. An activated (mutated) rule $\vec{s}_j^{\,i}$ is replaced with the highest ranked active solution vector $\vec{s}_{j*}^{\,i}$ in the same column node. In phase three, a Gaussian sampling operation is applied to the active solution component $\tilde{s}_j^{ih}$ in $\tilde{\vec{s}}_j^{\,i}$ to generate a new value $S(g(\tilde{s}_j^{ih}))$. The component $\tilde{s}_j^{ih}$ serves as the mean of the Gaussian probability density function $g(\tilde{s}_j^{ih})$. The standard deviation (STD) b of $g(\tilde{s}_j^{ih})$ dynamically changes with the iteration number and is described by

$$b = \max\{b_{\min}, b_{\max} \times \left(1 - \frac{I_c}{I_{Max}}\right)\} \in [b_{\min},\ b_{\max}]. \quad (7)$$

where b_{max} =0.1 and $b_{\min}$ =0.01. The application of the Gaussian sampling PDF operation to the active rule solution $\tilde{\vec{s}}_j^{\,i}$ generates a new rule solution $\hat{\vec{s}}_j^{\,i}$ described by

$$\hat{\vec{s}}_j^{\,i} = S(g(\tilde{\vec{s}}_j^{\,i})) = [S(g(\tilde{s}_j^{i1})), \ldots, S(g(\tilde{s}_j^{iD}))],\ j = 1,\ \ldots,\ N. \quad (8)$$

IV. Simulations

This section shows the performance of the MO-RACACO-designed FNN via comparisons with various MOPO algorithms, including a multi-objective elite genetic algorithm (MO-EGA) using the elite GA [16] for new solution generation, the NSGAII [12], and a multi-objective ant colony optimization in real space (MO-$ACO_{\mathbb{R}}$) using the $ACO_{\mathbb{R}}$ [17] for new solution generation with two selection coefficients ($q = 0.01$ and $q = 10$), These algorithms use the same rule coding and solution sorting and replacement strategy as in the MO-RACACO. All algorithms used the same population size (N=50) and the same number of performance evaluations. The performance indicators employed were the coverage metric C

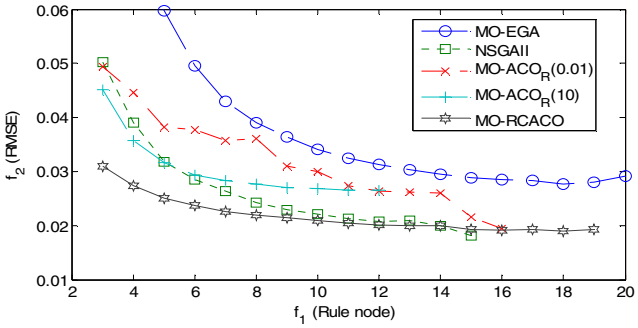
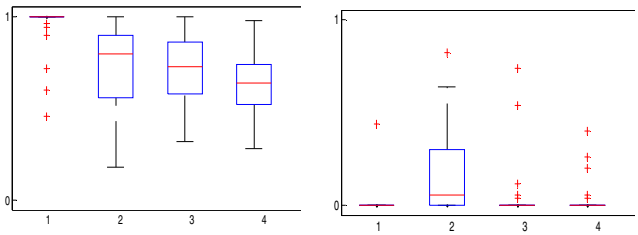


Fig. 5. The average non-dominated solutions from the MO-RACACO and various MOPO algorithms in Example 1.



(a) (b)

Fig. 6. Box plots based on the coverage metric C values in Example 1, (a) C(MO-RACACO, B) and (b) C(B, MO-RACACO), where the algorithm used to obtain B is 1: MO-EGA, 2: NSGAII, 3: MO-ACO$_R$ ($q = 0.01$), 4: MO-ACO$_R$ ($q = 10$).

and the diversity metric D [18]. The solution space of the number of rule nodes was the set of integers between 3 and 20, i.e., $\tilde{M}$ =20. The total number of iterations was set to 10^4, and so the total number of performance evaluations was $10^4 \times N = 5 \times 10^5$ in a run. For the statistical evaluation of the learning performance, 30 runs were conducted for each MOPO algorithm.

Example 1 (nonlinear plant control). The nonlinear plant is described by

$$y(t+1) = \frac{y(t)}{1+y^2(t)} + u^3(t), \tag{9}$$

where $y(t) \in [-2, 2]$ with $y(0) = 0$, and $u(t) \in [-1, 1]$ is the control input. The objective is to control the output $y(t)$ to track the following desired trajectory using an MO-RACACO–designed FNN:

$$y_d(t) = \sin(\pi t/50)\cos(\pi t/30), \ 1 \le t \le 250. \tag{10}$$

The inputs of the FNN were $y_d(t+1)$ and $y(t)$, and the output was $u(t)$. The control root-mean-squared error (RMSE) over the 250 time steps was used as the second objective function f_2. Fig. 5 shows the distribution of the average non-dominated solutions obtained from the various MOPO algorithms over 30 runs. Fig. 6 shows box plots of the coverage metric C when the

Table I. The Average Coverage Metric C of The Solution Set A Obtained by the MO-RACACO And The Solution Set B Obtained by Various Algorithms in Example 1.

Algorithm B	MO-EGA	NSGAII	MO-ACO$_R$ ($q = 0.01$)	MO-ACO$_R$ ($q = 10$)
C(MO-ACACO, B)	0.9716	0.7372	0.7328	0.6272
C(B, MO-ACACO)	0.0088	0.1624	0.0300	0.0204

Table II. The Average Diversity Metric D of Various Algorithms In Example 1.

method	MO-EGA	NSGAII	MO-ACO$_R$ ($q = 0.01$)	MO-ACO$_R$ ($q = 10$)	MO-RACACO
Average D	14.8447	9.8568	5.8962	4.5348	11.5922

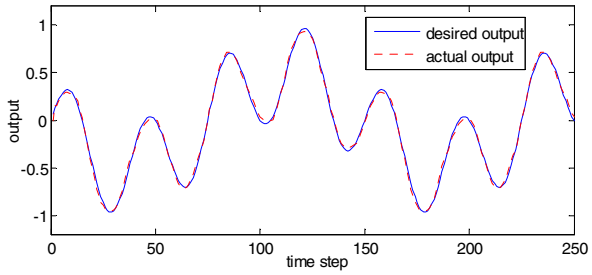


Fig. 7. The control result of the MO-RACACO in Example 1.

solutions obtained from the MO-RACACO were compared against the solutions (denoted as B) obtained from each of the MOPO algorithms used for comparison. Table I shows the average coverage metric C, which indicates that C (MO-RACACO, B) was greater than C (B, MO-RACACO) for each pair in the comparison. Table I and Figs. 5 and 6 show that the non-dominated solutions of the proposed MO-RACACO cover most of the solutions found by the algorithms used for comparison. Table II shows the diversity metric D for the various optimization algorithms. The results show that the MO-RACACO produces more diverse solutions than the algorithms used for comparison, with the sole exception of the MO-EGA. The MO-EGA produces a better spread mainly because of the much greater tracking error (f_2), especially when the rule node number (f_1) is small. The solutions of the MO-EGA algorithm are almost completely dominated by those of the MO-RACACO, as shown in Fig. 5.

Among the non-dominated solutions in the 30 runs of the MO-RACACO, the minimum RMSE achieved was $f_2 = 0.0178$ and the corresponding rule number was $f_1 = 17$. For the solution that achieved the minimum rule number ($f_1 = 3$), the corresponding minimum RMSE was $f_2 = 0.0282$. Fig. 5 shows that the average RMSE of the MO-RACACO was much smaller than those of the algorithms used for comparison when the rule number was three. This result also shows the superior optimization capability of the MO-RACACO. Fig. 7

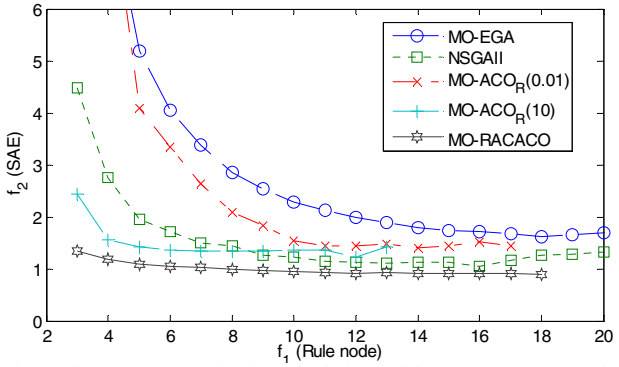


Fig. 8. The average non-dominated solutions of the MO-RACACO and various multi-objective population-based optimization algorithms in Example 2.

Table III. The Coverage Metric C Of The Solution Set A Obtained By The MO-RACACO And The Solution Set B Obtained By Various Algorithms.

Algorithm B	MO-EGA	NSGAII	MO-ACO$_R$ ($q = 0.01$)	MO-ACO$_R$ ($q = 10$)
C(MO-ACACO, B)	0.9992	0.8628	0.8552	0.7704
C(B, MO-ACACO)	0	0.0616	0.0356	0.0768

Table IV. The Average Diversity Metric D Of Various MOPO Algorithms In Example 2.

Algorithms	MO-EGA	NSGAII	MO-ACO$_R$ ($q = 0.01$)	MO-ACO$_R$ ($q = 10$)	MO-RACACO
Average D	30.3309	16.5272	18.6518	12.2848	12.3506

shows the desired and the actual control outputs using the MO-RACACO-designed FC with only three control rules. The result shows that the control outputs were close to the desired outputs.

Example 2 (nonlinear dynamic plant control). In this example, the dynamic plant is described by

$$y(t+1) = \frac{y(t)\times y(t-1)\times (y(t)+2.5)}{1+y^2(t)+y^2(t-1)} + u(t), -1.2 \le y(t) \le 1.2 \tag{11}$$

where the $u(t)$ is the control input and $-1.2 \le u(t) \le 1.2$. The output $y(t+1)$ was controlled to track the following desired trajectory given by

$$y_d(t+1) = 0.2\times\left[0.6y_d(t) + 0.2y_d(t-1) + r(t)\right],\ 0 \le t \le 250, \tag{12}$$

where

$$r(t) = 0.2\sin(2\pi t/25) + 0.4\sin(\pi t/32). \tag{13}$$

The inputs of the FNN were $y_d(t)$, $y(t-1)$, and $y(t)$, and the output was $u(t)$. The tracking error in f_2 was described by the following sum of absolute error (SAE),

$$f_2 = \sum_{t=1}^{250} \left| y_d(t) - y(t) \right|. \tag{14}$$

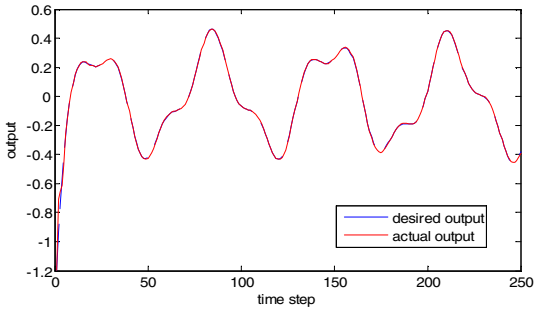


Fig. 9. The control results of the MO-RACACO using different non-dominated solutions in Example 2.

Fig. 8 shows the average of the non-dominated solutions obtained from various MOPO algorithms. Tables III and IV show that the average coverage metric C and diversity metric D, respectively, of various algorithms. Table III and Fig. 8 indicate that the non-dominated solutions of proposed MO-RACACO cover most solutions of the algorithms used for comparison. Table IV shows that the MO-EGA, NSGAII, and MO-ACO$_R$ ($q = 0.01$) achieve more diverse solutions than the MO-RACACO. As shown in Fig. 8 and Table III, the reason is that most solutions of the MO-RACACO are closer to the actual Pareto front than these algorithms. Fig. 9 shows the desired and actual outputs of the solution with the minimum rule number ($f_1 = 3$), where the corresponding SAE is $f_2 = 1.001$.

Example 3 (chaotic series prediction). The prediction problem uses the Mackey-Glass chaotic time series, which is generated from the following differential equation with time delay

$$\frac{dx(t)}{dt} = \frac{0.2x(t-\tau)}{1+x^{10}(t-\tau)} - 0.1x(t) \tag{15}$$

where $\tau > 17$. The parameter τ was set to be 30 and $x(0) = 1.2$. Four past values were used to predict $x(t)$ using an MO-RACACO-designed FNN. The inputs of the FNN were $x(t-24)$, $x(t-18)$, $x(t-12)$, and $x(t-6)$. The output was the predicted value of $x(t)$. One thousand patterns were generated from t=124 to t = 1123, with the first 500 patterns being used for training and the last 500 for testing. Fig. 10 shows the average of the non-dominated solutions obtained from various MOPO algorithms. Tables V and VI show the average coverage metric C and diversity metric D, respectively, of different optimization algorithms. Table V and Fig. 10 indicate that the non-dominated solutions of proposed MO-RACACO cover most of the solutions from the algorithms used for comparison. Table VI shows the MO-RACACO achieves more diverse solutions than the algorithms used for comparison with the sole exception of the MO-EGA. As in Examples 1 and 2, the reason is that most solutions of the MO-RACACO are closer to the Pareto front than these algorithms, as shown in Fig. 10. To see the prediction result, Fig. 11 shows the actual and predicted outputs of the non-dominated solution with the minimum rule node number ($f_1 = 3$), where

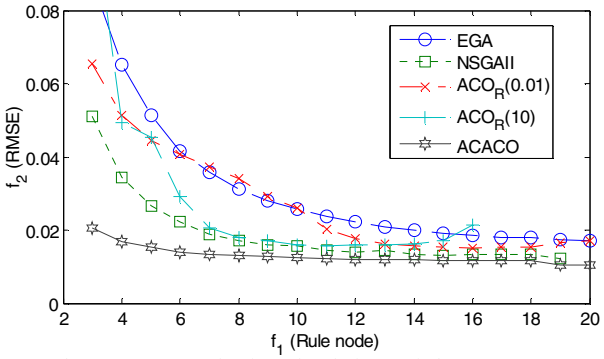


Fig. 10. The average non-dominated solutions of the MO-RACACO and various multi-objective population-based optimization algorithms in Example 3.

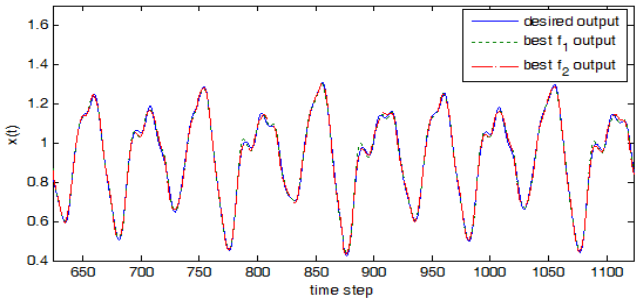


Fig. 11. The prediction results of the MO-RACACO using different non-dominated solutions for the test patterns in Example 3.

Table V. The Coverage Metric C Of The Solution Set A Obtained By The MO-RACACO And The Solution Set B Obtained By Various Algorithms.

Algorithm B	MO-EGA	NSGAII	MO-ACO$_R$ ($q = 0.01$)	MO-ACO$_R$ ($q = 10$)
C(MO-ACACO, B)	0.9944	0.8560	0.9060	0.7780
C(B, MO-ACACO)	0.0024	0.0784	0.0528	0.1276

Table VI. The Average Diversity Metric D Of Various Algorithms In Example 3.

Algorithms	MO-EGA	NSGAII	MO-ACO$_R$ ($q = 0.01$)	MO-ACO$_R$ ($q = 10$)	MO-RACACO
Average D	16.3488	13.2925	11.4220	8.0207	13.5897

the corresponding RMSE is $f_2 = 0.0130$, and the solution with the minimum RMSE $f_2 = 0.0094$, where the correspond rule number is $f_1 = 14$ using the MO-RACACO.

V. Conclusions

This paper applies the MO-RACACO to design FNNs for control and sequence prediction problems. The tradeoff between the network size and training error motivates the use of the MO-RACACO to find the Pareto optimal solutions. Three examples were conducted to verify the optimization ability of the MO-RACACO in the two application areas. Comparisons with various MOPO algorithms using different new solution generation approaches were conducted. Analyses on the distributions of the found Pareto-optimal solution on the objective functions, the coverage metric, and the diversity metric of different MOPO algorithms were made to verify the optimization ability of the MO-RACACO. The results show the superiority of the MO-RACACO in the FNN optimization problems.

References

[1] C. F. Juang and C. T. Lin, "An on-line self-constructing neural fuzzy inference network and its applications," *IEEE Trans. Fuzzy Systems*, vol. 6, no. 1, pp. 12-32, Feb. 1998.

[2] P. Angelov and D. Filev, "An approach to online identification of Takagi-Sugeno fuzzy models," IEEE Trans. Syst., Man, Cybern. B, vol. 34, no. 1, pp. 484–498, Feb. 2004.

[3] C. F. Juang and C. M. Chang, "Human body posture classification by a neural fuzzy network and home care system application," *IEEE Trans. Syst., Man, and Cyber., Part A: Systems and Humans*, vol. 37, no. 6, pp. 984-994, Nov. 2007.

[4] J. D. Rubio, "SOFMLS: Online self-organizing fuzzy modified least-squares network," *IEEE Trans. Fuzzy Systems*, vol. 17, no. 6, pp. 1296-1309, Dec. 2009.

[5] C. F. Juang, T. C. Chen, and W. Y. Cheng, "Speedup of implementing fuzzy neural networks with high-dimensional inputs through parallel processing on graphic processing units," *IEEE Trans. Fuzzy Systems*, vol. 19, no. 4, pp. 717-728, Aug. 2011.

[6] W. Zhao, K. Li, and G. W. Irwin, "A new gradient descent approach for local learning of fuzzy neural models," *IEEE Trans. Fuzzy Systems,* vol. 21, no. 1, pp. 30-44, Feb. 2013.

[7] C. F. Juang and Y. W. Tsao, "A self-evolving interval type-2 fuzzy neural network with on-line structure and parameter learning," *IEEE Trans. Fuzzy Systems*, vol. 16, no. 6, pp. 1411-1424, Dec. 2008.

[8] C. F. Juang, "Temporal problems solved by dynamic fuzzy network based on genetic algorithm with variable-length chromosomes," *Fuzzy Sets and Systems*, Vol. 142, No. 2, pp. 199-219, March 2004.

[9] C. F. Juang, C. M. Hsiao, and C. H. Hsu, "Hierarchical cluster-based multi-species particle swarm optimization for fuzzy system optimization," *IEEE Trans. Fuzzy Systems*, vol. 18, no. 1, pp. 14-26, Feb. 2010.

[10] C. F. Juang and P. H. Chang, "Designing fuzzy rule-based systems using continuous ant colony optimization," *IEEE Trans. Fuzzy Systems*, vol. 18, no. 1, pp. 138-149, Feb. 2010.

[11] E. Zitzler, M. Laumnns, and L. Thiele, "SPEA2: improving the strength pareto evolutionary algorithm," Computer. Eng. and Network Lab, Swiss Federal Inst. Technol., Zurich, Tech. Rep. 103, 2001.

[12] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGAII," *IEEE Trans. Evolutionary Computation*, vol. 6, no. 2, pp. 182-197, April 2002.

[13] P. Pulkkinen and H. Koivisto, "A dynamically constrained multiobjective genetic fuzzy system for regression problems," *IEEE Trans. Fuzzy Systems*, vol. 18, no. 1, pp. 161-177, Feb. 2010.

[14] R. Alcala, M. J. Gacto, and F. Herrera, "A fast and scalable multiobjective genetic fuzzy system for linguistic fuzzy modeling in high-dimensional regression problems," *IEEE Trans. Fuzzy Syst.*, vol. 19, no. 4, pp. 666-681, Aug. 2011.

[15] C. H. Hsu and C. F. Juang, "Multi-objective continuous-ant-colony-optimized FC for robot wall-following control," *IEEE Computational Intelligence Magazine*, vol. 8, no. 3, pp. 28-40, Aug. 2013.

[16] C. F. Juang, " A TSK-type recurrent fuzzy network for dynamic systems processing by neural network and genetic algorithms," *IEEE Trans. Fuzzy Syst.*, vol. 10, no. 2, pp. 155-170, April 2002.

[17] K. Socha and M. Dorigo, "Ant colony optimization for continuous domain," *European Journal of Operational Research*, vol. 185, pp. 1155-1173, 2008.

[18] H. Li, Q. Zhang, E. Tsang, and J. A. Ford, "Hybrid estimation of distrbution algorithm for multi-objective knapsack problem," in *Proc. EvoCOP* (LNCS 3004), 2004, pp. 145-154.