Rule-Based Cooperative Continuous Ant Colony Optimization to Improve the Accuracy of Fuzzy System Design

Chia-Feng Juang, *Senior Member, IEEE*, Chi-Wei Hung, and Chia-Hung Hsu

***Abstract*—This paper proposes a cooperative continuous ant colony optimization (CCACO) algorithm and applies it to address the accuracy-oriented fuzzy systems (FSs) design problems. All of the free parameters in a zero- or first-order Takagi–Sugeno–Kang (TSK) FS are optimized through CCACO. The CCACO algorithm performs optimization through multiple ant colonies, where each ant colony is only responsible for optimizing the free parameters in a single fuzzy rule. The ant colonies cooperate to design a complete FS, with a complete parameter solution vector (encoding a complete FS) that is formed by selecting a subsolution component (encoding a single fuzzy rule) from each colony. Subsolutions in each ant colony are evolved independently using a new continuous ant colony optimization algorithm. In the CCACO, solutions are updated via the techniques of pheromone-based tournament ant path selection, ant wandering operation, and best-ant-attraction refinement. The performance of the CCACO is verified through applications to fuzzy controller and predictor design problems. Comparisons with other population-based optimization algorithms verify the superiority of the CCACO.**

***Index Terms*—Ant colony optimization, cooperative evolution, evolutionary fuzzy systems, swarm intelligence (SI).**

I. Introduction

EVOLUTIONARY fuzzy systems (FSs) that design FSs via population-based evolutionary computation techniques, such as genetic algorithms (GAs) and swarm intelligence (SI) algorithms, have drawn attention in the past two decades. In contrast with neural fuzzy systems (NFSs) that use the gradient descent algorithm, this technique is less likely to become stuck at a local minimum. Among the well-known SI optimization algorithms are particle swarm optimization (PSO) [1] and ant colony optimization (ACO) [2]. These two methods were developed from observations of the social behavior of animals in nature, such as bird flocking, fish schooling, and ants foraging for food. Several advanced GAs and PSO algorithms have been proposed to solve various optimization problems in the past decade [3]–[14], and one promising approach is the modification of single-population topology to multiple populations [3], [5], [9]–[13]. A cooperative PSO (CPSO) was proposed in [5]. A CPSO uses multiple swarms to optimize different components of a solution vector cooperatively for function optimization. Based on CPSO, different advanced CPSO have been proposed to address FS optimization problems [11], [13]. A hierarchical cluster-based multispecies particle swarm optimization (HCMSPSO) for FS optimization was proposed in [13]. The HCMSPSO uses multiple swarms to optimize different fuzzy rules. In this paper, the idea of multiple-population topology is incorporated into ACO, and a cooperative continuous ACO (CCACO) is proposed.

In contrast to PSO, ACO is a relatively new optimization approach. ACO is a multiagent approach, which was originally proposed to solve difficult discrete combinatorial optimization problems [2]. Different discrete ACO models have been applied to design the consequent part of an FS [15]–[17]. These approaches find a parameter set in a discrete space and, therefore, are not suitable for accuracy-oriented FS optimization problems. To find solutions in a continuous space, a few continuous ACO algorithms have been proposed [18]–[24]. Among them, a basic approach is continuous ACO in real space $(\mathrm{ACO}_{\mathbb{R}})$ [20]. In $\mathrm{ACO}_{\mathbb{R}}$, the domain of an optimized variable is continuous, and its sampling is based on a continuous probability density function (PDF). New solutions are generated by sampling the PDF, which uses a Gaussian kernel. Based on the $\mathrm{ACO}_{\mathbb{R}}$, the design of fuzzy rules using continuous ACO (RCACO) was proposed in [21]. The RCACO is based on the node-path framework of the original discrete ACO, and its operation is explained by using nodes and paths in a graph as the discrete ACO. The RCACO was shown to outperform several advanced PSO and continuous ACO algorithms in different accuracy-oriented fuzzy control problems [21]. As opposed to previous continuous ACO algorithms that all work with a single population of solutions, the proposed CCACO uses a new optimization framework that incorporates the technique of multiple population topology and cooperative framework into continuous ACO.

This paper proposes a CCACO algorithm to address accuracy-oriented FS design problems. For an FS with a given number of rules, the CCACO aims to optimize all of the free parameters in it. If the number of fuzzy rules is r, then the CCACO creates r ant colonies with each colony optimizing only a single fuzzy rule. In other words, the number of rules is equal to the number of colonies in the CCACO. A combination of r rules selected from each of the r colonies forms a complete FS. To

Manuscript received May 1, 2013; revised; accepted June 20, 2013. Date of publication July 9, 2013; date of current version July 31, 2014. This work was supported by the National Science Council, Taiwan, under Grant NSC 101-2221-E-005-036-MY2.

The authors are with the Department of Electrical Engineering, National Chung-Hsing University, Taichung 402, Taiwan (e-mail: cfjuang@dragon.nchu.edu.tw; a0934016211@yahoo.com.tw; smallturtle@ms44.url.com.tw).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TFUZZ.2013.2272480

optimize a fuzzy rule in each of the r colonies, the CCACO uses a new continuous ACO optimization approach that consists of the techniques of pheromone-based tournament ant path selection, the ant wandering operation, and global-best ant attraction refinement. The CCACO is applied to five FS optimization problems, and its performance is compared with those of different SI algorithms to demonstrate the superiority of the CCACO.

This paper is organized as follows. Section II introduces the FSs to be optimized. Section III introduces the rule-based multiple colony structure and cooperative framework in the CCACO. Section IV introduces the ant-based continuous solution update process. Section V presents the simulation results of the CCACO for FS optimization problems. Section VI presents discussions on the CCACO performance. Finally, Section VII presents conclusions.

II. Fuzzy Systems

The CCACO is applied to FSs consisting of zero- or first-order Takagi–Sugeno–Kang (TSK)-type fuzzy rules. The number of fuzzy sets in each input variable is assumed to be equal to the number of fuzzy rules. Each rule in the TSK-type FS is described as follows [25]:

$$\begin{aligned} \mathrm{R}_i:\ & \text{If } x_1(k) \text{ is } A_{i1} \text{ And } \ldots \text{ And } x_n(k) \text{ is } A_{in} \\ & \text{Then } u(k) \text{ is } f_i(x_1, \ldots, x_n) \end{aligned} \tag{1}$$

where k is the time step, $x_1(k), \ldots, x_n(k)$ are input variables, $u(k)$ is the system output variable, and A_{ij} is a fuzzy set. The function f_i in the consequent part is

$$f_i(x_1, \ldots, x_n) = a_{i0} \tag{2}$$

for a zero-order TSK-type FS or

$$f_i(x_1, \ldots, x_n) = a_{i0} + \sum_{j=1}^{n} a_{ij} x_j \tag{3}$$

for a first-order TSK-type FS. As in most NFSs [26]–[32], fuzzy set A_{ij} uses a Gaussian membership function

$$M_{ij}(x_j) = \exp\left\{ -\left(\frac{x_j - m_{ij}}{b_{ij}} \right)^2 \right\} \tag{4}$$

where m_{ij} and b_{ij} represent the center and width of the fuzzy set A_{ij}, respectively. In the inference engine, the fuzzy AND operation is implemented by the algebraic product in fuzzy theory. Thus, given an input dataset $\vec{x} = (x_1, \ldots, x_n)$, the firing strength $\phi_i(\vec{x})$ of rule i is calculated by

$$\phi_i(\vec{x}) = \prod_{j=1}^{n} M_{ij}(x_j) = \exp\left\{ -\sum_{j=1}^{n} \left(\frac{x_j - m_{ij}}{b_{ij}} \right)^2 \right\}. \tag{5}$$

If there are r rules in an FS, the output of the system that is calculated by the weighted average defuzzification method is [25], [31]

$$u = \frac{\sum_{i=1}^{r} \phi_i(\vec{x}) f_i}{\sum_{i=1}^{r} \phi_i(\vec{x})}. \tag{6}$$

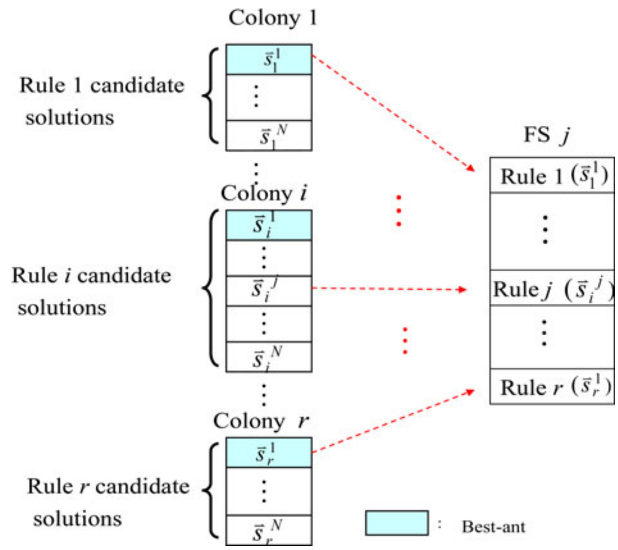

Fig. 1. Multicolony and cooperative structure in the CCACO.

For an FS with a given number of r rules, the CCACO is applied to optimize all of the free parameters m_{ij}, b_{ij}, and a_{ij} in it.

III. Rule-Based Multiple Colony Structure and Cooperative Framework

This section describes the multiple colony structure motivated from the multipopulation cooperative structure in the CPSO [5] and HCMSPSO [13]. In CPSO, instead of having one swarm trying to find an optimal M-dimensional vector, the vector is split into k constituent components such that k swarms optimize the k different vector components independently. In CCACO, as shown in Fig. 1, r ant colonies are formed while optimizing an FS consisting of r rules. In other words, a colony optimizes only the free parameters in a fuzzy rule. The solution vector in the ith ant colony is represented as follows:

$$\vec{s}_i = [m_{i1}, b_{i1}, \ldots, m_{in}, b_{in}, a_{i0}] \in \Re^{2n+1} \tag{7}$$

for a zero-order TSK-type fuzzy rule, and

$$\vec{s}_i = [m_{i1}, b_{i1}, \ldots, m_{in}, b_{in}, a_{i0}, a_{i1}, \ldots, a_{in}] \in \Re^{3n+1} \tag{8}$$

for a first-order TSK-type fuzzy rule. A complete solution vector for optimizing all of the free parameters in a whole FS is constructed by selecting one ant solution from each of the n colonies. That is, $[\vec{s}_1, \ldots, \vec{s}_r]$ represents a complete solution vector of a whole FS and components in this vector are optimized through the proposed CCACO algorithm.

For a whole FS, its performance is evaluated using the error between desired and actual outputs. The size of each colony is assumed to be N. The performance of the jth ant solution $\vec{s}_i^{\,j}$ in colony i is implicitly evaluated according to the performance of the FS in which it participates. To form a whole FS, the accompanying rules for $\vec{s}_i^{\,j}$ should be selected from the other $r-1$ colonies. At the start of the CCACO algorithm, the performances of the solutions in a colony are unknown. In this special case, the accompanying rules are simply set to be