Computing and Informatics, Vol. 33, 2014, 810–830

FLEXIBLE FUZZY RULE BASES EVOLUTION WITH SWARM INTELLIGENCE FOR META-SCHEDULING IN GRID COMPUTING

Rocío Pérez Prado, José Enrique Muñoz Expósito
Sebastián García-Galán

Telecommunication Engineering Department
University of Jaen
Alfonso X el Sabio, 28 Linares
Jaen, Spain
e-mail: {rperez, jemunoz, sgalan}@ujaen.es

Abstract. Fuzzy rule-based systems are expert systems whose performance is strongly related to the quality of their knowledge and the associated knowledge acquisition processes and thus, the design of effective learning techniques is considered a critical and major problem of these systems. *Knowledge acquisition with a swarm intelligence approach* is a recent learning strategy for the evolution of fuzzy rule bases founded on swarm intelligence showing improvement over classical knowledge acquisition strategies in fuzzy rule based systems such as Pittsburgh and Michigan approaches in terms of convergence behaviour and accuracy. In this work, a generalization of this method is proposed to allow the simultaneous consideration of diversely configured knowledge bases and this way to accelerate the learning process of the original algorithm. In order to test the suggested strategy, a problem of practical importance nowadays, the design of expert meta-schedulers systems for grid computing is considered. Simulations results show the fact that the suggested adaptation improves the functionality of knowledge acquisition with a swarm intelligence approach and it reduces computational effort; at the same time it keeps the quality of the canonical strategy.

Keywords: Knowledge-based systems, fuzzy logic, optimization, evolutionary computation, grid computing

1 INTRODUCTION

Fuzzy Rule-Based Systems (FRBSs) [1, 2] are knowledge based systems founded on the simulation of human reasoning extensively used in a wide range of applications such as computational network management [3, 4], speech and music discrimination [5], image retrieval [6] and other sciences such as risk modeling and decision making [7, 8]. The success of these systems is based on the quality of the knowledge they represent, which cannot be always obtained from an expert in the field of the considered problem. Moreover, it is generally not feasible to obtain this knowledge in advance and thus it is necessary to resort to automatic learning strategies that allow the extraction of the necessary knowledge to be incorporated to the expert system.

Diverse strategies for the evolution of expert fuzzy bases can be found in literature. However, the importance of Genetic Algorithms (GAs) is to be underlined [9, 10]. The application areas of GAs range from optimization in network coding [11], fuzzy systems design for mobile robot navigation [12] and surface prediction and power management computational in grinding [13] to production scheduling [14]. In the learning of fuzzy rule bases (RBs), these strategies take each individual of a given population as a chromosome that can consist of a single rule or a collection of rules. The population is subject to a genetic competitive process through generations where chromosomes are selected for the next-generation population based on a performance index or objective function [2]. Specifically, the role of two genetic fuzzy learning strategies must be pointed out: Pittsburgh [15] and Michigan approaches [16]. The former, Pittsburgh approach, encodes a complete RB as an individual of the population to be evolved whereas the latter, Michigan approach, considers each rule as a chromosome and the best suited collection of rules is chosen as the final expert base.

On the other hand, Particle Swarm Optimization (PSO) [17] is an evolutionary strategy based on swarm intelligence widely used for optimization in multidimensional problems in diverse application areas such as renewable energies [18] or electromagnetics [19]. The main strengths of this strategy over GAs are given by its implementation simplicity and the small number of control parameters. PSO does not require the use of genetic operators, such as crossover and mutation, but individuals of the population or swarm, called particles, evolve by means of internal velocities. Besides, particles have memory and unidirectional information exchanges are considered which reduces the number of communications needed in the evolutionary process. In addition, the use of PSO allows a greater control of the convergence of the swarm particles than GAs what decreases the computational effort. PSO is used in many engineering applications as shown by works on artificial neural network training [20], ad hoc wireless networks [21], or simultaneous localization and mapping problems [22]. Also, the combination of PSO and fuzzy logic has been considered in areas such as nonlinear identification [23, 24], collective robotic search [25], and power system stabilizers design [26] and new versions for PSO have recently emerged from this combination [27]. Furthermore, in [28],

PSO has recently been proposed for knowledge acquisition in FRBSs in such a way that a RB as a whole represents a swarm individual. However, in spite of the good results obtained by this learning strategy, called KASIA, in comparison to those of GAs, it presents the counterpart that it can only work with fuzzy rule bases of a fixed size, i.e., knowledge bases presenting the same number of fuzzy rules, what derives in a high computational cost since the optimum number of fuzzy rules is initially unknown and previous setting stages are necessary to properly configure the strategy.

In this work, a learning strategy, Generalized Knowledge Acquisition with a Swarm Intelligence Approach (KASIA-G) is presented. KASIA-G is an adaptation of the original KASIA algorithm [28] that allows its generalization to be used with fuzzy rule bases of diverse dimensions and thus to reduce the number of setting stages required by the canonical strategy. In order to show the effectiveness of the proposed algorithm, KASIA-G, as a knowledge acquisition strategy, its application to a problem of practical importance such as the design of meta-schedulers for Grid Computing is considered. Further, comparative results with KASIA and GAs (Pittsburgh approach) are analyzed.

The rest of the paper is organized as follows. In Section 2, the fundamentals of canonical learning strategy KASIA are presented. The suggested knowledge acquisition strategy is introduced in Section 3. In Section 4, KASIA-G is evaluated as a learning strategy for fuzzy rule based meta-schedulers for computational grids and results are compared to that of other classical scheduling approaches. Finally, Section 5 concludes the paper.

2 BACKGROUND

As introduced before, the contribution of this work consists of the adaptation of the knowledge acquisition strategy KASIA. Hence, it is important to present the main features of the canonical strategy first in order to understand the suggested modifications. A detailed description of KASIA and its characteristics such as computational cost and convergence control can be found in [28].

In KASIA learning strategy, a set of NP particles made up a swarm where each individual P_i represents a complete RB. The objective of the algorithm is to move particles within the search space to achieve the optimum location for these particles, i.e., the collection of rules where the optimal performance fitness f is obtained. Each particle P_i in the swarm has the form

$$P_i = \begin{bmatrix} a^i_{1,1} & a^i_{1,2} & \dots & a^i_{1,n} & b^i_1 & c^i_1 \\ a^i_{2,1} & a^i_{2,2} & \dots & a^i_{2,n} & b^i_2 & c^i_2 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ a^i_{m,1} & a^i_{m,2} & \dots & a^i_{m,n} & b^i_m & c^i_m \end{bmatrix} \quad (1)$$

where each row describes a fuzzy rule, n indicates the number of input variables and m is the number of rules. Antecedents of rules $a^i_{j,k}$ are encoded as an integer in the

interval

$$a^i_{j,k} \in [-NF_{in},\, NF_{in}]\,,\; j \in \{1, 2, \ldots, m\},\; k \in \{1, 2, \ldots, n\} \tag{2}$$

with NF_{in} the number of fuzzy sets of input j. Also, consequents b^i_j are encoded analogously, where NF_{out} denotes the number of output sets

$$b^i_j \in [-NF_{out},\, NF_{out}]\,,\; j \in \{1, 2, \ldots, m\} \tag{3}$$

$$NF_{in},\, NF_{out} \in \mathbb{N}. \tag{4}$$

Also, rules connectives c^i_j are represented by "1" for connective "AND" and "2" for connective "OR".

$$c^i_j \in \{1, 2\} \tag{5}$$

On the other hand, the velocity associated to each particle is formulated as

$$V_i = \begin{bmatrix} v^i_{1,1} & v^i_{1,2} & \cdots & v^i_{1,n} & v^i_{1,n+1} & v^i_{1,n+2} \\ v^i_{2,1} & v^i_{2,2} & \cdots & v^i_{2,n} & v^i_{2,n+1} & v^i_{2,n+2} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ v^i_{m,1} & v^i_{m,2} & \cdots & v^i_{m,n} & v^i_{m,n+1} & v^i_{m,n+2} \end{bmatrix} \tag{6}$$

$$v^i_{j,k} \in [V_{min},\, V_{max}]\,,\; j \in \{1, 2, \ldots, m\},\; k \in \{1, 2, \ldots, n+2\} \tag{7}$$

where V_{min} and V_{max} denote the velocity upper and lower restrictions, respectively. As can be observed in particles formulation (Equation (1)), the search space for KASIA strategy consists of three subsections associated to antecedent, consequent and connector parts. Hence, Equations (2), (3) and (5) describe the search space for every element category in the RB. In addition, in order to effectively guide particles, limits for velocity $[V_{min},\, V_{max}]$ or considering symmetry $[-V_{max},\, V_{max}]$ must be imposed. As discussed in [29], a limitation of velocity parameter V_{max} allows the contention of explosion of the velocity of particles. This way, the following expression must be considered [30] for velocity:

$$v^i_{j,k} = sign(v^i_{j,k},)min(|v^i_{j,k}|, V_{max}) \tag{8}$$

with V_{max} calculated as $p \times s$ and $0.1 \leq p \leq 1.0$.

In KASIA, the updating process of particles location is an iterative process conducted by a fitness value or performance index f that indicates the quality of every particle or RB at every iteration. Hence, a particle velocity is updated taking into account its own inertia, the best quality location reached by the particle, $P^{\#}(t)$ or inner tendency to return to its best location, and the best location found by the whole swarm $P^*(t)$ up to the current iteration or social component

$$\begin{aligned} V_i(t+1) \;=\; & w \otimes V_i(t) \oplus (c_1 * r_1) \otimes (P_i(t) \ominus P^{\#}(t)) \\ & \oplus (c_2 * r_2) \otimes (P_i(t) \ominus P^*(t)) \end{aligned} \tag{9}$$

with $\otimes$, $\oplus$ and $\ominus$ being the multiplication, addition and regular difference of matrices, respectively. Finally, each particle P_i updates its position by the following

expression

$$P_i(t+1) = P(t)_i \oplus V_i(t+1). \tag{10}$$

As shown, KASIA does not allow the utilization of knowledge bases P_i with different sizes, given the difference operator used in Equation (9), $\ominus$, which demands regular matrices difference. Thus, this expression can only be used considering particles with the same size, i.e., the same number of rules. Note that considering a fixed size for rule bases makes it necessary to work independently with different sized rule bases in the training stages of the expert system to obtain a suited base for the problem under consideration. This requirement significantly increases the computational cost of the learning process. Furthermore, it can be observed that the difference expression does not take into account the simultaneous existence of two particles with identical rules located in different rows of the matrices. I.e., let us consider two particles P_1 and P_2 in Equation (11):

$$\begin{aligned} P_1 &= \begin{bmatrix} a^1_{1,1} & a^1_{1,2} & \dots & a^1_{1,n} & b^1_1 & c^1_1 \\ a^1_{2,1} & a^1_{2,2} & \dots & a^1_{2,n} & b^1_2 & c^1_2 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ a^1_{m,1} & a^1_{m,2} & \dots & a^1_{m,n} & b^1_m & c^1_m \end{bmatrix} \\ P_2 &= \begin{bmatrix} a^1_{2,1} & a^1_{2,2} & \dots & a^1_{2,n} & b^1_2 & c^1_2 \\ a^1_{1,1} & a^1_{1,2} & \dots & a^1_{1,n} & b^1_1 & c^1_1 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ a^1_{m,1} & a^1_{m,2} & \dots & a^1_{m,n} & b^1_m & c^1_m \end{bmatrix} \end{aligned} \tag{11}$$

Particle P_2 equals P_1, since only some rule positions are changed which does not alter the associated knowledge. However, the canonical expression does not identify identical rules located in different positions in two particles and thus, the convergence of the algorithm can be affected. In the next section, a modification in the particles differentiation process that allows the use of rule bases of different sizes and considers the position of identical rules in the expert bases is presented.

3 PROPOSED METHOD FOR FLEXIBLE RULE BASE EVOLUTION WITH KASIA

As introduced before, a modification of canonical KASIA is presented in this work that incorporates a new differentiation expression in the original algorithm and a reformulation of Equations (9) and (10) for the updating of particle positions. The goal of this new version of KASIA is to allow the consideration of RBs of different sizes which makes preliminary stages to select the best configuration of RBs unnecessary and thus it reduces the computational effort associated to RBs evaluation and generalizes the algorithm. Furthermore, the reformulation of Equations (9) and (10) makes it possible for KASIA to associate a null distance when comparing identical rules even though these rules are not located in the same position of the bases. The new differentiation method is implemented as an iterative algorithm which looks

for the minimum Euclidean distance between every rule or row in the minuend and all the rows in the subtrahend at every iteration. Once the minimum distance is found for a rule in the minuend, the difference between the involved fuzzy rules is calculated and the result is added as a new row to the so-called differentiation matrix. The only condition that must be imposed to the particles is that they must consider the same number of columns, i.e., the same number of antecedents and consequents must be constant through iterations as in the canonical algorithm.

Given particle P_1 as minuend and particle P_2 as subtrahend, with dimensions $m \times (n+2)$ and $p \times (n+2)$ respectively, and m and p the number of rules in both particles such as illustrated in Equation (12):

$$P_1 = \begin{bmatrix} a^1_{1,1} & a^1_{1,2} & \dots & a^1_{1,n} & b^1_1 & c^1_1 \\ a^1_{2,1} & a^1_{2,2} & \dots & a^1_{2,n} & b^1_2 & c^1_2 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ a^1_{m,1} & a^1_{m,2} & \dots & a^1_{m,n} & b^1_m & c^1_m \end{bmatrix}$$
$$P_2 = \begin{bmatrix} a^2_{1,1} & a^2_{1,2} & \dots & a^2_{1,n} & b^2_1 & c^2_1 \\ a^2_{2,1} & a^2_{2,2} & \dots & a^2_{2,n} & b^2_2 & c^2_2 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ a^2_{p,1} & a^2_{p,2} & \dots & a^2_{p,n} & b^2_p & c^2_p \end{bmatrix} \tag{12}$$

In the first stage of the algorithm, the Euclidean distance between all the rules in P_1 and all the rules in P_2 is pursued and a matrix Δ with $m \times p$ elements is obtained.

$$\Delta = \begin{bmatrix} d_{1,1} & \cdots & d_{1,j-1} & d_{1,j} & d_{1,j+1} & \cdots & d_{1,p} \\ d_{2,1} & \cdots & d_{2,j-1} & d_{2,j} & d_{2,j+1} & \cdots & d_{2,p} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ d_{i-1,1} & \cdots & d_{i-1,j-1} & d_{i-1,j} & d_{i-1,j+1} & \cdots & d_{i-1,p} \\ d_{i,1} & \cdots & d_{i,j-1} & d_{i,j} & d_{i,j+1} & \cdots & d_{i,p} \\ d_{i+1,1} & \cdots & d_{i+1,j-1} & d_{i+1,j} & d_{i+1,j+1} & \cdots & d_{i+1,p} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ d_{m,1} & \cdots & d_{m,j-1} & d_{m,j} & d_{m,j+1} & \cdots & d_{m,p} \end{bmatrix} \tag{13}$$

where every Euclidean distance $d_{i,j}$ is calculated as

$$d_{i,j} = \sqrt{\sum_{k=1}^{n} (a^1_{i,k} - a^2_{j,k})^2 + (b^1_i - b^2_j)^2 + (c^1_i - c^2_j)^2}. \tag{14}$$

Once matrix Δ has been obtained, an iterative process considering an identical number of iterations to the number of rules of particle P_1 starts. For m iterations the minimum value of every row in matrix Δ is identified. E.g., let $d_{i,j}$ satisfy the minimum value condition. This makes reference to the fact that rule i of base P_1 is located in the minimum distance of rule j of P_2. Thereby, given two rules, its difference is calculated and incorporated to differentiation matrix D (dimension of

Algorithm 1 Calculation of the proposed difference between particles.

Initialization

1. Calculate matrix Δ with distance between given particles P_1 and P_2.
   for every rule i of P_1,
   for every rule j of P_2,
   (a) Calculate distance between from rule i to rule j:
   $(d_{i,,j})^2 = \sum_{k=1}^{n}(a_{i,k}^1 - a_{j,k}^2)^2 + (b_i^1 - b_j^2)^2 + (c_i^1 - c_j^2)^2$
   (b) Add distance $d_{i,j}$ to Δ, Equation (13)
   end
   end
2. Initialize D, differentiation matrix
3. $M = 2max\,(\Delta)$,

Do

1. Update $min(d_{i,,j})$ distance value from Δ .
2. Select rule i from P_1 and rule j from P_2.
   (a) Calculate $r_i = (a_{i,1}^1 - a_{j,1}^2) \cdots (a_{i,n}^1 - a_{j,n}^2)(b_i^1 - b_j^2)(c_i^1 - c_j^2)$
   difference between rule i from P_1 and rule j from P_2
   (b) Add r_i to D matrix, Equation (15)
   (c) Discard the minimum distance in following iterations
   $d_{i,k} = M$, $\forall k \in [1 \cdots p]$ and $d_{l,j} = M$, $\forall l \in [1 \cdots m]$ from Δ,
   Equation (16)

While(Num_rules of P_1)

Return solution: D

$m \times n + 2$) which aggregates the minimum distances of rules and it is constructed at every step of the iterative process. In this first stage, matrix D has the form presented in Equation (15).

$$D = \left[\begin{array}{cccc} (a_{i,1}^1 - a_{j,1}^2) \ldots (a_{i,n}^1 - a_{j,n}^2) & (b_i^1 - b_j^2) & (c_i^1 - c_j^2) \end{array} \right] \tag{15}$$

In order to avoid this minimum distance to be taken into account in following iterations, the position located in row i and column j of matrix Δ is updated with a high value e.g., the double of the maximum of initial Δ matrix, i.e., $M = 2 \cdot max\,(\Delta)$. This condition can be observed in Equation (16).

$$\Delta = \left[\begin{array}{ccccccc} d_{1,1} & \cdots & d_{1,j-1} & M & d_{1,j+1} & \cdots & d_{1,p} \\ d_{2,1} & \cdots & d_{2,j-1} & M & d_{2,j+1} & \cdots & d_{2,p} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ d_{i-1,1} & \cdots & d_{i-1,j-1} & M & d_{i-1,j+1} & \cdots & d_{i-1,p} \\ M & \cdots & M & M & M & \cdots & M \\ d_{i+1,1} & \cdots & d_{i+1,j-1} & M & d_{i+1,j+1} & \cdots & d_{i+1,p} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ d_{m,1} & \cdots & d_{m,j-1} & M & d_{m,j+1} & \cdots & d_{m,p} \end{array} \right] \tag{16}$$

At the end of the iterative process, i.e., m iterations, differentiation matrix D has the form:

$$D = \begin{bmatrix} r_{1,1} & r_{1,1} & \cdots & r_{1,n+2} \\ r_{2,1} & r_{2,2} & \cdots & r_{2,n+2} \\ \cdots & \cdots & \cdots & \cdots \\ r_{m,1} & r_{m,2} & \cdots & r_{m,n+2} \end{bmatrix} \tag{17}$$

The calculation of the differentiation matrix can be summarized in Algorithm 1. The incorporation of the differentiation operator suggested makes necessary the reformulation of the velocity updating expression used in KASIA (Equation (9)) to adapt it to its generalization to different sizes of particles. In order to keep the coherence in the operations, the new calculation of the velocity is done following the expression:

$$\begin{aligned} V_i(t+1) \;=\; & w \otimes V_i(t) \ominus (c_1 * r_1) \otimes (P^{\#}(t) \ominus P(t)) \\ & \ominus (c_2 * r_2) \otimes (P(t)^* \ominus P(t)) \end{aligned} \tag{18}$$

where $\ominus$ denotes the proposed difference function. On the other hand, the position of particles is updated regarding the present value of particles and the global and local best suited RB stored through the swarm evolution as in the canonical algorithm

$$P_i(t+1) \;= P(t)_i \oplus V_i(t+1). \tag{19}$$

As can be observed, the proposed subtraction operator does not alter the size of particles during the learning process but it allows the consideration of particles of diverse dimensions. This is translated in a reduction in the computational cost since previous setting processes can be suppressed (i.e., reduction of the number of RBs or particles evaluations to use KASIA strategy). Note that modifying the way the difference is calculated in the proposed strategy involves a higher computational effort than in the canonical strategy. However, the associated computational cost of the evaluation of RBs or particles to obtain their fitness or quality in many applications of FRBSs is so high (as the evaluation of RBs for expert meta-schedulers in real grid computing scenarios proposed in this work) that this increase is not significant.

4 SIMULATION RESULTS

In this section, conducted simulations to test the proposed schema are analyzed. Specifically, the performance of a fuzzy meta-scheduler within a grid computing environment with KASIA-G learning is studied and results are compared to those of classic scheduling systems. Grid scheduling is a problem of practical importance nowadays in the use of FRBSs that let the evaluation of the proposed learning strategy in an environment where the knowledge acquisition process is critical.

Learning results of the meta-scheduler are studied in comparison to canonical KASIA and genetic algorithms, Pittsburgh approach, in diverse settings. The different learning strategies characteristics are analyzed both in terms of final result,

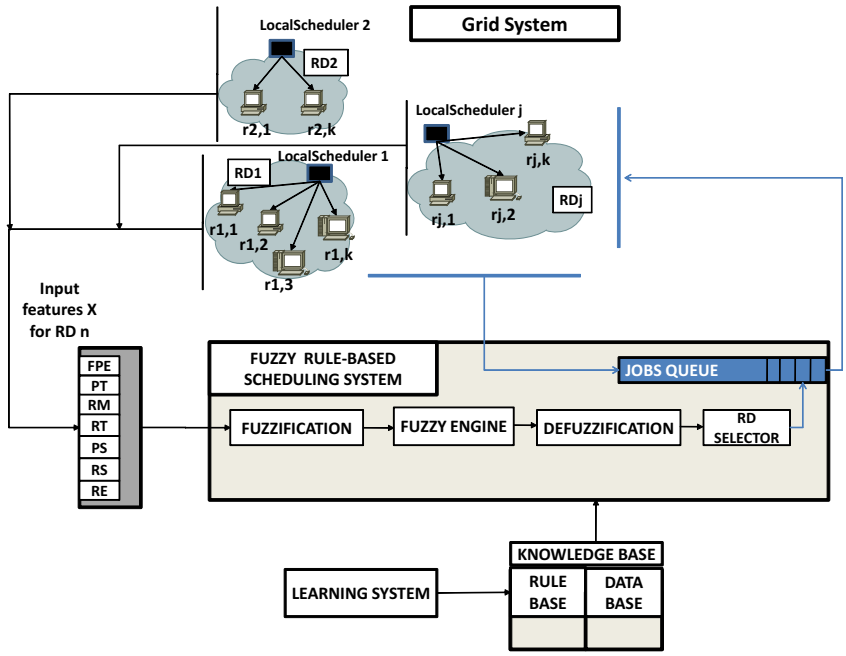

Figure 1. Fuzzy Meta-scheduling system in a grid computing network

convergence behaviour and computational effort. Furthermore, the fuzzy scheduling strategy is compared to other scheduling strategies in grid computing EASY-BF and ESG-LS periodical.

Nowadays, science is based on large-scale numeric simulations and data analysis and collaborative environments able to integrate theoretical and experimental efforts that are made by individual entities are sought [31]. Grid computing has an important role in diverse science areas such as e-Science, bio-informatics, meteorology, medicine and physic [32]. A computational grid is made up of a set of heterogeneous and geographically distributed resources sharing their capabilities with the aim of achieving a common goal. Hence, a computational grid is generally described as a collection of H_j machines which are allocated among diverse resources domains, $RD_j = \{r_{j,1}, r_{j,2}, \ldots, r_{j,H_j}\}$ making up a global resource domain or virtual organization, $VO = \{RD_1, RD_2, \ldots, RD_G\}$. The objective of a meta-scheduler is to distribute a collection of L jobs $J = \{J_1, J_2, \ldots, J_L\}$ to the involved resource domains, RD_j, in the organization. Besides, local schedulers are located in every resource domain and they are in charge of the scheduling of jobs within their associated RD. Jobs are heterogeneous and diverse in terms of demanded number of processors, memory, deadline times and other characteristics, such as the CPU type and operating system. Also, resources domains RD_j are heterogeneous and featured by properties such as the number of machines H_j, the number of CPU, operating

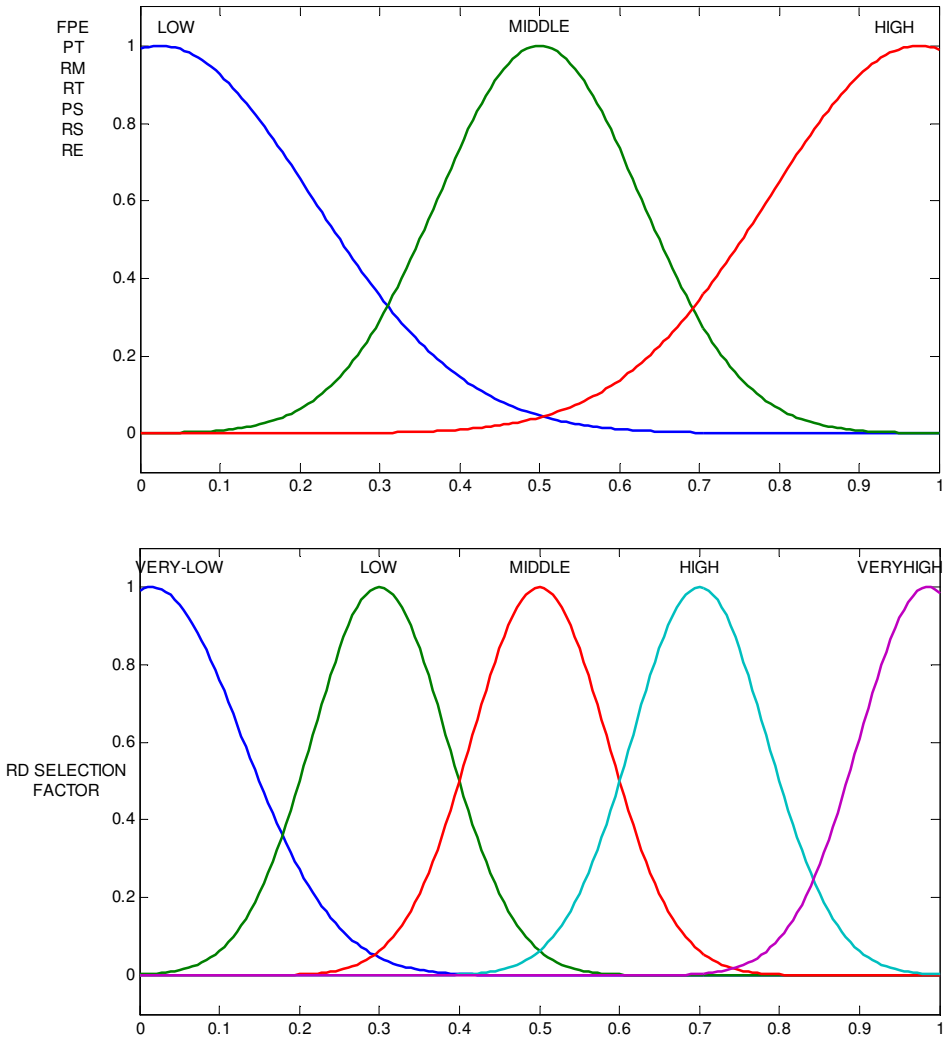

Figure 2. Fuzzy sets for grid features and resource domain selector

system or memory size of each machine. Recent works such as [3] and [33, 34] suggest the utilization of expert systems to work as a meta-scheduler. The architecture of this kind of systems can be observed in Figure 1. In this figure the classical schema of a fuzzy system adapted to work in a Grid environment is shown. The fuzzy system provides a selection factor y_0 at every stage of the scheduling process that indicates the suitability of an specific resource domain to be selected on the basis of the knowledge of the state of the resources domains and its own knowledge of the

grid system. The state of the resources in the grid system is described using several variables that inform about the computational capabilities over time. An example of these variables are the number of busy processing elements, the number of machines in use, resources own makespan, deadlines, delays and failures of the different resources. The state of the resources is featured considering both their current usage and performance during its operation with time. Hence, the scheduler can base its decisions bearing in mind a wider description of resources state than the only consideration of their current availability. In this sense, it is necessary to find a balance between accuracy in the featuring of the resources state and the complexity of the learning strategy to obtain a scheduling based on a good description of the system state without highly increasing the complexity of the system knowledge extraction.

Feature	**Description**
Number of free processing elements (**FPE**)	Number of free processing element within RD_i.
Previous Tardiness (**PT**)	Sum of tardiness of all finished jobs in RD_i.
Resource Makespan (**RM**)	Current makespan for RD_i.
Resource Tardiness (**RT**)	Current tardiness of jobs within RD_i.
Previous Score (**PS**)	Previous deadline score of already finished jobs in RD_i.
Resource Score (**RS**)	Number of non delayed jobs so far in RD_i.
Resources In Execution (**RE**)	Number of Resources currently executing jobs within RD_i.

Table 1. Inputs features for the fuzzy meta-scheduler

Table 1 summarizes the system variables in this work.

Before describing the knowledge acquisition process for the fuzzy meta-scheduler it is necessary to show how its knowledge is represented and the fitness used for training. In this work, a rule R_i is made up of a set of antecedents and a consequent, using a Mamdani type encoding [35]: both antecedents and consequents are encoded using linguistic variables associated to fuzzy sets [36]. This type of rules, when using multiples inputs and a single output can be represented as follows:

$$R_i = \text{IF } x_1 \text{ is } A_{i,1} \text{ and/or } \ldots x_n \text{ is } A_{i,n} \text{ THEN } y \text{ is } B_i \tag{20}$$

where $(x_1, \ldots, x_n)$ indicates the features considered for the resources state, and $A_{i,m}$ and B_i correspond to one of the possible NF_{in} fuzzy sets for input and NF_{out} for output, respectively. System features describing the system state are illustrated in Table 1, and for their description Gaussian membership functions are considered,

which are formulated as follows:

$$\mu_i^{(x_m)}(z) = \frac{1}{\sigma_i^{(x_m)}\sqrt{2\pi}} \exp\left\{ \frac{-\left(z - \tau_i^{(x_m)}\right)^2}{2\sigma_i^{(x_m)2}} \right\} \tag{21}$$

$$\left\{ z \in \mathbb{R}^+ \mid z \leq 1 \right\}. \tag{22}$$

These membership functions are driven by two parameters $\tau_i^{(x_m)}$ and $\sigma_i^{(x_m)}$, mean and standard deviation. The goal of using this type of function is to smooth the transition between areas of fuzzy sets. This is a desirable characteristic for the system in order to be able to provide a contribution in a wide range of system conditions [3]. Figure 2 represents the fuzzy sets for the normalized input features and output. As shown, inputs are featured by three fuzzy sets indicating *low*, *medium* and *high* values and the output or selector factor relevance are described by five fuzzy areas, to wit, *very low*, *low*, *medium*, *high* and *very high*. Finally, *makespan* or workload finalization time is considered as the optimization criteria in this work,

$$makespan = \max_{j \in J} T_j \tag{23}$$

where T_j denotes the execution time of job j. *Makespan* is extensively found in literature as a general grid system productivity indicator and its minimization is typically pursued [37, 38].

The performance of the scheduler is tested through simulations with Alea software [39]. Alea is a grid scheduling simulation toolkit based on GridSim where grid scenarios and traces from real world can be used. To be precise, the considered grid environment in our tests is based on Czech National Grid Infrastructure Metacentrum project [40]. This grid is made up by 806 CPUs of heterogeneous types (i.e., Opteron and Xeon) and speed (i.e., 1 500–3 200 MHz) located in 210 machines running Linux. Moreover, queues configuration parameters, maintenance and reservation behaviour of machines, and jobs characterization are obtained from traces of Metacentrum facilities [41]. Firstly, the fuzzy scheduler training results are presented. As introduced before, the scheduler is trained with *makespan* as performance index or fitness [37, 38]. This way, the goal is the minimization of the finalization time of the last job in the simulation or *makespan*,

$$\min_{S_i \in Sched} \left\{ \max_{j \in J} T_j \right\} \tag{24}$$

where *Sched* indicates all the possible schedules. The training is done for 100 iterations where the scheduler must allocate 2 000 jobs in the grid. Results are obtained for the fuzzy scheduler using KASIA-G and KASIA [28] and genetic Pittsburgh approach for the evolution of rules. KASIA-G, KASIA and Pittsburgh approach are configured considering parameters shown in Tables 2 and 3.

Parameters configuration for swarm-based learning of fuzzy rules				
KASIA-G	$\omega = 0.9$	$d_1 = 2$, $d_2 = 2$	Number of particles/RBs (NP) = 18	$8 \leq \text{RB}_{size} \leq 15$
KASIA	$\omega = 0.9$	$d_1 = 2$, $d_2 = 2$	Number of particles/RBs (NP) = 18	$\text{RB}_{size} = 10$

Table 2. Parameters configuration for KASIA-G and KASIA

Parameters configuration for genetic learning of fuzzy rules				
Pittsburgh-Elitism	Elitism Selection	Selection rate $\lambda = 0.9$	Mutation rate = $0.1e^{(-iter/Num_{iter})}$	Population size (PS) = 20
Pittsburgh-Tournament	Tournament Selection	Two-point crossover		init max $\text{RB}_{size} = 20$

Table 3. Parameters configuration for Pittsburgh approach

The configuration for the well-known strategies (i.e., KASIA and Pittsburgh approach) is based on previous works in the usage of these strategies for learning of meta-schedulers for grid computing [28, 34, 42] and the configuration of genetic learning strategies [2, 43]. On the other hand, KASIA-G follows the same configuration as the canonical strategy KASIA, since both methods are to be compared in the same conditions to observe the effect of the alternative fuzzy rule bases subtraction mechanism in its performance. It must be underlined that KASIA is optimally configured in terms of rule bases size for the problem under consideration in contrast to KASIA-G which does not need this previous setting.

Also, note the size of the swarm size and the genetic population are fixed to allow a fair comparison in terms of computational cost at every iteration. The different approaches are computationally rated according to the required number of cost function evaluations (FEs) (i.e., RBs evaluations in the grid) [44]. On the one hand, in KASIA, each particle or RB is evaluated at every iteration, and thus, computational effort for an experiment can be formulated as

$$CE_{KASIA} = NP * num_{iter} \tag{25}$$

where NP denotes the size of the swarm and num_{iter} is the number of iterations or stopping condition. Regarding computational effort for KASIA-G, note that the proposed difference operator allows the consideration of particles of diverse dimensions which is translated in a reduction in computational cost, given previous setting processes can be suppressed (i.e., reduction of the number of RBs or particles evaluations to use KASIA strategy). Hence, once KASIA is optimally configured, computational effort of KASIA-G in FEs is the same as the computational effort of KASIA. On the other hand, with the aim of providing a fair step by step (i.e., in every iteration) comparison in terms of computational effort in FEs through generations, Pittsburgh strategy [2, 45] is proposed for comparison in several configurations, see Table 3. Computational effort for Pittsburgh approach is given by the following expression

$$CE_{Pitts} = PS + PS * \lambda * (num_{iter} - 1) \tag{26}$$

where PS denotes the number of individuals or RB population size, λ represents the selection factor. Pittsburgh population is made up of a set of RBs and genetic operators are applied at this level. Hence, as in canonical KASIA and suggested

KASIA-G, every generation requires a number of selected population FEs. This way, the different strategies can be compared at every generation considering the same computational effort what motivates the selection of Pittsburgh strategy instead of other genetic learning strategies.

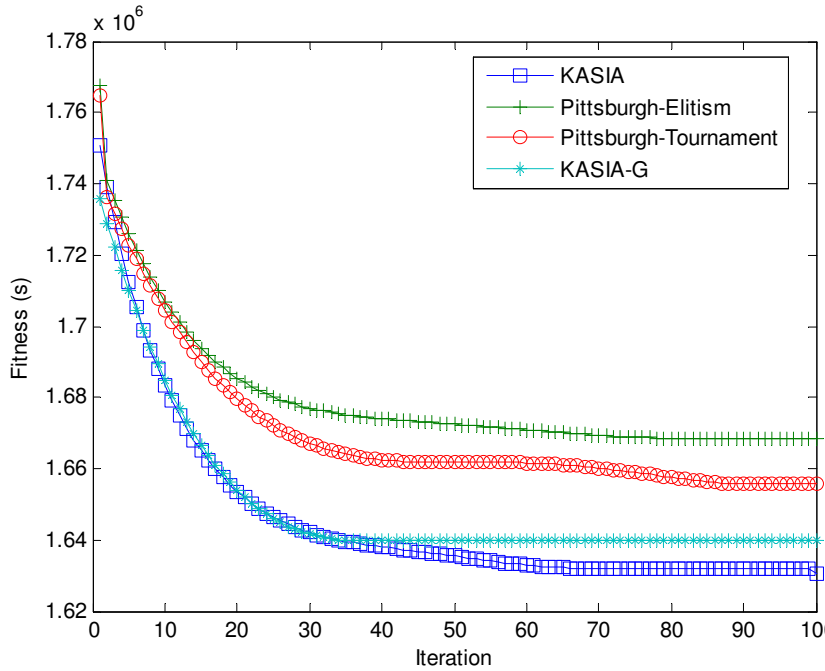

Figure 3. Learning strategies convergence behaviour with *makespan* as performance index

Figure 3 represents the evolution of the learning strategies, KASIA-G, KASIA and Pittsburgh in its diverse settings (i.e., Pittsburgh-Elitism and Pittsburgh-Tournament) during the training of the scheduler. Curves show the obtained fitness by the best knowledge base in every iteration (average result of 30 simulations). It can be observed that KASIA-G presents a convergence speed slightly higher than KASIA and significantly higher than Pittsburgh approaches. To be precise, KASIA-G converges in 49 iterations in contrast to 64, 78 and 82 in KASIA, Pittsburgh-Elitism and Pittsburgh-Tournament, respectively, which is translated into a reduction in computational cost. Associated computational effort for these strategies is presented in Table 4. Also, the final value of the fitness for KASIA-G is similar to that of KASIA and improves Pittsburgh-Elitism and Pittsburgh-Tournament approaches by 1.66 % and 1.05 % on average as presented in this table.

Furthermore, with the aim of analyzing the scheduler performance and validating results, different tests are conducted using several QoS criteria such as *flow-*

Strategy/Parameter (30 experiments)	Avg. Convergence fitness-makespan (s)	Iteration Iteration	Computational Effort (FEs)
KASIA-G	1 639 877.838	49	1 152
KASIA	1 630 479.572	64	1 152
Pittsburgh-Elitism	1 667 586.227	78	1 406
Pittsburgh-Tournament	1 657 267.364	82	1 478

Table 4. Training results with *makespan* as performance index

time, *weighted usage*, *classic usage*, *tardiness* and *slowdown*, beside the training fitness. Additionally, the Metacentrum scenario is modified with different reservation and machine failures behaviour and workload is increased by 17 %. Table 5 presents the results for the fuzzy scheduler with the obtained knowledge bases with KASIA-G, KASIA and Pittsburgh approaches (i.e., Pittsburgh-Elitism and Pittsburgh-Tournament) in validation scenario. It can be observed that KASIA-based strategies improve the genetic schedulers results in terms of *flowtime, weighted usage* and *classic usage*. Moreover, results are compared with those of two extended scheduling strategies EASY-Backfilling (EASY-BF) and the ESG Local Search periodical strategy (ESG + LS periodical) [46]. Results show their better performance compared to the rest of strategies. As commented earlier, the scheduler trained with KASIA-based strategies is able to obtain the highest-quality knowledge bases in terms of *makespan* and as can be observed, this quality remains in different conditions as shown in validation results. In addition, an improvement in *weighted* and *classical usage* in comparison to EASY-BF can be appreciated, although, as expected, the optimization of *makespan* deteriorates performance in *flowtime, slowdown* and *tardiness*.

Metric/Strategy	KASIA-G	KASIA	Pittsburgh-	Pittsburgh-	EASY-BF	ESG + LS
Metric/Strategy	KASIA-G	KASIA	Elitism	Tournament		period.
Makespan (s)	1 647 228.268	1 633 719.8	1 659 926.3	1 659 201.625	1 749 586.008	1 973 151.408
Flow-time (s)	88 397.133	87 765.539	88 865.534	89 184.388	87 491.471	83 379.182
Weighted usage (%)	46.659	48.184	46.420	46.341	44.47	34.74
Classic usage (%)	57.178	59.360	56.528	56.843	47.01	40.91
Tardiness (s)	4 766.554	4 683.250	4 757.286	4 834.579	3 235.311	1 274.822
Slowdown (s)	194.999	197.036	192.049	199.384	184.352	17.522

Table 5. Scheduling results of the fuzzy scheduler with KASIA-G, KASIA, Pittsburgh-Elitism and Pittsburgh-Tournament and EASY-BF and ESG + LS periodical in validation scenario with obtained expert knowledge

In many situations it is desirable to check if the distribution of a variable is the same in two populations, or if this variable tends to be higher or lower in one of the two groups based on sampling data. In our case, it can be interesting to study the objective function for the jobs subject to evolutionary schedulers analyzed (KASIA-G, KASIA and Pittsburgh approaches). To this end, two comparisons are made: KASIA-G versus Pittsburgh in its two settings and KASIA-G versus KASIA. Firstly, it is necessary to determine if it is possible to conduct parametric tests to

the obtained values of the function cost f_{obj}. As suggested in [10], Kolmogorov-Smirnov and Shapiro-Wilk are used to check the samples normality. The obtained results for the three strategies show that there exists no normality in the samples and thus it is necessary to use a nonparametric test to check if there exist differences among them. Two nonparametric tests for comparison are done based on the sum of ranges of Wilcoxon or Mann-Whitney test. This widely used method is available in different statistical applications such as KEEL [47] and SPSS [48]. In order to make the comparison, two observations regarding KASIA-G with respect to the canonical strategy, KASIA and the classical strategy Pittsburgh in its best setting, Pittsburgh-Tournament, are considered following previous works in statistical comparisons with Wilcoxon tests [10, 49]:

Observation 1: Average of the fitness function in two different populations (KASIA-G and Pittsburgh-Tournament). The evidence against the null hypothesis is that the sum of the ranges in Pittsburgh-Tournament approach is higher than that of KASIA-G. Wilcoxon W parameter results in 310 and the obtained p-value is 0.03839; thus, the evidence against the null hypothesis is significant with the obtained p-value.

Observation 2: Average of the fitness function in two different populations (KASIA-G and KASIA). The evidence against the null hypothesis is that the sum of the ranges in KASIA approach is lower than that of KASIA-G. In this case, the obtained p-value is 0.36 and thus, there is no significant evidence against the null hypothesis.

In the light of these statistical tests results, it can be derived that KASIA-G has a better performance with a p-value of 0.03839 than Pittsburgh approach and it cannot be concluded that KASIA-G results are worse than those of canonical KASIA.

5 CONCLUSIONS

In this work, an adaptation of the learning strategy KASIA based on PSO, KASIA-G has been presented. The proposed knowledge acquisition strategy allows the utilization of knowledge bases of different sizes with the canonical algorithm. This property reduces the number of required experiments in the training stage since tests with different configurations can be suppressed and thus it significantly decreases computational cost. Moreover, the suggested methodology improves the performance of the canonical learning strategy since it contemplates the existence of equal rules in the calculation of particles of the involved RBs distance although they are located in different positions within these bases with the following increase in the algorithm efficacy. In order to evaluate the efficiency of the methodology, it is applied to the learning of fuzzy rule-based meta-schedulers in grid computing where the knowledge acquisition stage is critical and KASIA-G results are compared to those of canonical KASIA and Pittsburgh approach in diverse settings. It is observed that KASIA-G

results are close to those of KASIA and outperform Pittsburgh results in diverse settings. Furthermore, it is shown that the difference between the obtained results with KASIA-G and KASIA is not statistically significant. In addition, it has been checked that the obtained results with the suggested methodology outperform those of traditional strategies such as EASY-BF (5.85 %) and EGS + LS periodical (16.52 %) in terms of training fitness. In sum, the proposed learning methodology, KASIA-G is a generalization of the original strategy, KASIA, that improves its functionality and setting simplicity without reducing its quality.

Acknowledgments

This work has been financially supported by the Spanish Government (Research Project P07-TIC-02713). The Metacentrum workload log was generously provided by the Czech National Grid Infrastructure Metacentrum.

REFERENCES

[1] Alcalá, R.—Ducange, P.—Herrera, F.—Lazzerini, B.—Marcelloni, F.: A Multiobjective Evolutionary Approach to Concurrently Learn Rule and Data Bases of Linguistic Fuzzy-Rule-Based Systems. IEEE Transactions of Fuzzy Systems, Vol. 17, 2009, No. 5, pp. 1106–1122.

[2] Cordón, O.—Herrera, F.—Hoffmann, F.—Magdalena, L.: Genetic Fuzzy Systems: Evolutionary Tuning and Learning of Fuzzy Knowledge Bases. World Scientific Pub. Co. Inc. 2001.

[3] Franke, C.—Hoffmann, F.—Lepping, J.—Schwiegelshohn, U.: Development of Scheduling Strategies with Genetic Fuzzy Systems. Appl. Soft Comput., Vol. 8, 2008, No. 1, pp. 706–721.

[4] Sanchez Santiago, A. J. et al.: A Multi-Criteria Meta-Fuzzy Scheduler for Independent Tasks in Grid Computing. Computing and Informatics, Vol. 30, 2011, No. 6, pp. 1201–1223.

[5] Muñoz Expósito, J. E.—García-Galán, S.—Ruiz-Reyes, N.—Vera-Candeas, P.: Audio Coding Improvement Using Evolutionary Speech/Music Discrimination. IEEE International Conference on Fuzzy Systems (FUZZY-IEEE), London, UK, 2007, pp. 1–6.

[6] Ajorloo, H.—Lakdashti, A.: IRFuM: Image Retrieval via Fuzzy Modeling. Computing and Informatics, Vol. 30, 2011, No. 5, pp. 913–945.

[7] Alonso, S.—Herrera-Viedma, E.—Chiclana, F.—Herrera, F.: Individual and Social Strategies to Deal with Ignorance Situations in Multi-Person Decision Making. International Journal of Information Technology & Decision Making, IJITDM, Vol. 8, 2009, No. 2, pp. 313–333.

[8] Yager, R.: Perception-Based Granular Probabilities in Risk Modeling and Decision Making. IEEE Transactions on Fuzzy Systems, Vol. 14, 2006, No. 2, pp. 329–339.

[9] Casillas, J.—Carse, B.: Special Issue on Genetic Fuzzy Systems, Recent Developments and Future Directions. Soft Comput., Vol. 13, 2009, No. 5, pp. 417–418.

[10] García, S.—Fernandez, A.—Luengo, J.—Herrera, F.: A Study of Statistical Techniques and Performance Measures for Genetics-Based Machine Learning: Accuracy and Interpretability. Soft Comput., Vol. 13, 2009, No. 10, pp. 959–977.

[11] Xing, H.—Qu, R.: A Compact Genetic Algorithm for the Network Coding Based Resource Minimization Problem. Applied Intelligence 2011, pp. 1–15.

[12] Pratihar, D. K.—Deb, K.—Ghosh, A.: A Genetic-Fuzzy Approach for Mobile Robot Navigation Among Moving Obstacles. International Journal of Approximate Reasoning, Vol. 20, 1999, pp. 145–172.

[13] Nandi, A. K.—Pratihar, D. K.: Design of a Genetic-Fuzzy System to Predict Surface Finish and Power Requirement in Grinding. Fuzzy Sets and Systems, Vol. 148, 2004, pp. 487–504.

[14] Chung, S. H.—Chan, F. T. S.—Chan, H. K.: A Modified Genetic Algorithm Approach for Scheduling of Perfect Maintenance in Distributed Production Scheduling. Engineering Applications of Artificial Intelligence, Vol. 22, 2009, No. 7, pp. 1005–1014.

[15] Smith, S. F.: A Learning System Based on Genetic Adaptive Algorithms. Pittsburgh, USA 1980.

[16] Booker, L. B.—Goldberg, D. E.—Holland, J. H.: Classifier Systems and Genetic Algorithms. Artif. Intell., Vol. 40, 1989, No. 1-3, pp. 235–282.

[17] Kennedy, J.—Eberhart, R.: Particle Swarm Optimization. IEEE International Conference on Neural Networks 1995, Vol. 4.

[18] Lopez, P. R.—Jurado, F.—Ruiz Reyes, N.—García Galán, S.—Gomez, M.: Particle Swarm Optimization for Biomass-Fuelled Systems With Technical Constraints. Engineering Applications of Artificial Intelligence, Vol. 21, 2008, No. 8, pp. 1389–1396.

[19] Robinson, J.—Rahmat-Samii, Y.: Particle Swarm Optimization in Electromagnetics. IEEE Transactions on Antennas and Propagation, Vol. 52, 2004, No. 2, pp. 397–407.

[20] Meissner, M.—Schmuker, M.—Schneider, G.: Optimized Particle Swarm Optimization (OPSO) and Its Application to Artificial Neural Network Training. BMC Bioinformatics, Vol. 7, 2006, 7:125. doi: 10.1186/1471-2105-7-125.

[21] Huang, C. J.—Chuang Y. T.—Hu, K. W.: Using Particle Swarm Optimization for QoS in Ad-Hoc Multicast. Eng. Appl. Artif. Intell., Vol. 22, 2009, pp. 1188–1193.

[22] Chatterjee, A.—Matsuno, F.: A Neuro-Fuzzy Assisted Extended Kalman Filter-Based Approach for Simultaneous Localization and Mapping (SLAM) Problems. IEEE Transactions on Fuzzy Systems, Vol. 15, 2007, No. 5, pp. 984–997.

[23] Araujo, E.—dos Santos Coelho, L.: Particle Swarm Approaches Using Lozi map Chaotic Sequences to Fuzzy Modelling of an Experimental Thermal-Vacuum System. Appl. Soft. Comput., Vol. 8, 2008, No. 4, pp. 1354–1364.

[24] dos Santos Coelho, L.—Herrera, B. M.: Fuzzy Identification Based on a Chaotic Particle Swarm Optimization Approach Applied to a Nonlinear Yo-Yo Motion System. IEEE Transactions on Industrial Electronics, Vol. 54, 2007, No. 6, pp. 3234–3245.

[25] Venayagamoorthy, G. K.—Grant, L. L.—Doctor, S.: Collective Robotic Search Using Hybrid Techniques: Fuzzy Logic and Swarm Intelligence Inspired by Nature. Engineering Applications of Artificial Intelligence, Vol. 22, 2009, No. 3, pp. 431–441.

[26] Hussein, T.—Elshafei, A. L.—Bahgat, A.: Comparison Between Multi-Band and Self-Tuned Fuzzy Power System Stabilizers. 16th Mediterranean Conference on Control and Automation 2008, pp. 374–379.

[27] Norouzzadeh, M.—Ahmadzadeh, M.—Palhang, M.: LADPSO: Using Fuzzy Logic to Conduct PSO Algorithm. Applied Intelligence 2011, pp. 1–15.

[28] Prado, R. P.—García-Galán, S.—Muñoz Exposito, J. E.—Yuste, A. J.: Knowledge Acquisition in Fuzzy Rule Based Systems with Particle Swarm Optimization. IEEE Transactions on Fuzzy Systems, Vol. 18, 2010, No. 6, pp. 1083–1097.

[29] Clerc, M.—Kennedy, J.: The Particle Swarm – Explosion, Stability, and Convergence in a Multidimensional Complex Space. IEEE Transactions on Evolutionary Computation, Vol. 6, 2002, No. 1, pp. 58–73.

[30] Liu, H.—Abraham, A.—Hassanien, A. E.: Scheduling Jobs on Computational Grids Using a Fuzzy Particle Swarm Optimization Algorithm. Future Gener. Comput. Syst., Vol. 26, 2010, pp. 1336–1343.

[31] Berman, F.: High-Performance Schedulers. The Grid: Blueprint for a Future Computing Infrastructure, Morgan Kaufmann Publishers, San Francisco (USA) 1998.

[32] Marin, J. M. M.—Camara, S. B.: Las Tecnologías Grid de la Información Como Nueva Herramienta Empresarial. Septem Ediciones 2008.

[33] Prado, R. P.—García Galán, S.—Yuste, A. J.—Muñoz Exposito, J. E.—Sanchez Santiago, A. J.—Bruque, S.: Evolutionary Fuzzy Scheduler for Grid Computing. Lecture Notes in Computer Science, Vol. 5517, Springer 2009, pp. 286–293.

[34] Prado, R. P.—García-Galán, S.—Yuste, A. J.—Muñoz Exposito, J. E.: Genetic Fuzzy Rule-Based Scheduling System for Grid Computing in Virtual Organizations. Soft Computing – A Fusion of Foundations, Methodologies and Applications, Vol. 15, 2011, pp. 1255–1271.

[35] Mamdani, E. H.—Assilian, S.: An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller. International Journal of Man-Machine Studies, Vol. 7, 1975, No. 1, pp. 1–13.

[36] Zadeh, L. A.: The Concept of a Linguistic Variable and Its Application to Approximate Reasoning. Information Sciences, Vol. 8, 1975, No. 3, pp. 199–249.

[37] Xhafa, F.—Abraham, A.: Meta-Heuristics for Grid Scheduling Problems. Studies in Computational Intelligence, Vol. 146, 2008, pp. 1–37.

[38] Xhafa, F.—Abraham, A.: Computational Models and Heuristic Methods for Grid Scheduling Problems. Future Generation Computer Systems, Vol. 26, 2010, No. 4, pp. 608–621.

[39] Klusacek, D.—Matyska, L.—Rudova, H.: Alea – Grid Scheduling Simulation Environment. Parallel Processing and Applied Mathemathics, Lecture Notes in Computer Science, Vol. 4967, 2008, pp. 1029–1038.

[40] Šustr, Z.—Sitera, J.—Mulač, M.—Ruda, M.—Antoš, D.—Hejtmánek, L. et al.: MetaCentrum, the Czech Virtualized NGI. 2009.

[41] Infrastructure CNG: MetaCentrum data sets. Available on: `http://www.fi.muni.cz/~xklusac/index.php?page=meta2009`.

[42] Prado, R. P.—García Galán, S.—Yuste, A. J.—Muñoz Exposito, J. E.: A Fuzzy Rule-Based Meta-Scheduler with Evolutionary Learning for Grid Computing. Engineering Applications of Artificial Intelligence, Vol. 23, 2010, No. 7, pp. 1072–1082.

[43] Herrera, F.—Lozano, M.—Sanchez, A. M.: Hybrid Crossover Operators for Real-Coded Genetic Algorithms: An Experimental Study. Soft Computing, Vol. 9, 2005, pp. 280–298.

[44] Zhan, Z.—Zhang, J.—Li, Y.—Chung, H. S.: Adaptive Particle Swarm Optimization. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics. Vol. 39, 2009, No. 6, pp. 1362–1381.

[45] Ishibuchi, H.—Yamamoto, T.—Nakashima, T.: Hybridization of Fuzzy GBML Approaches for Pattern Classification Problems. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics. Vol. 35, 2005, No. 2, pp. 359–365.

[46] Klusacek, D.—Rudova, H.—Baraglia, R.—Pasquali, M.—Capannini, G.: Comparison of Multi-Criteria Scheduling Techniques. In: Gorlatch, S., Fragopoulou, P., Priol, T. (Eds.): Grid Computing: Achievements and Prospects. European Commiss, Network Excellence CoreGRID Fund, Springer 2008, pp. 173–184.

[47] Alcala-Fdez, J.—Sanchez, L.—García, S.—del Jesus, M. J.—Ventura, S.—Garrell, J. M. et al.: KEEL: A Software Tool to Assess Evolutionary Algorithms for Data Mining Problems. Soft Computing – A Fusion of Foundations, Methodologies and Applications, Vol. 13, 2009, pp. 307–318.

[48] Marin Fernández, J.: SPSS User's Guide. Universidad de Murcia 2004, pp. 1–85.

[49] García, S.—Herrera, F.: An Extension on Statistical Comparisons of Classifiers over Multiple Data Sets for all Pairwise Comparisons. Journal of Machine Learning Research, Vol. 9, 2008, pp. 2677–2694.

Rocío Pérez Prado received the M. Sc. degree in telecommunication engineering from Seville University, Seville, Spain, in 2008 and the Ph. D. degree in telecommunication engineering with European Mention from Jaen University, Jaen, Spain, in 2011. At present, she is an Assistant Professor with the Telecommunication Engineering Department, Jaen University, Jaen, Spain, in the chair of Signal Theory and Communications. Her current research interests include artificial intelligence, machine learning, grid computing, cloud computing and scheduling. She is an active member of the research group "Signal Processing for Telecommunication Systems" (Group TIC-188 of the PAI) in the Jaen University; she forms part of the editorial reviewer board of international journals such as IEEE Transactions on Fuzzy Systems, Applied Soft Computing and Soft Computing.

José Enrique Muñoz Expósito received the M. Sc. degree in telecommunication engineering from Malaga University, Spain and Ph. D. in telecommunication engineering from Jaen University, Spain. Since 2003, he has been an Associate Professor at the Telecommunication Engineering Department at Jaen University in the chair of Telematics. His research interest include speech and audio analysis, artificial intelligence, grid and cloud computing. He is currently a senior researcher with the chair Signal Processing and Telecommunication Systems (TIC188 of the PAI), Jaen University, Spain. He is involved in research projects.

Sebastián García-Galán received the M. Sc. and Ph. D. degrees in telecommunication engineering from Malaga University and Technical University of Madrid in 1995 and 2004, respectively. Since 1999, he is an Associate Professor in telematics engineering at the Telecommunication Engineering Department of Jaen University. He is a member of the research group "Signal Processing for Telecommunication Systems" (Group TIC-188 of the PAI) of Jaen University and of the European Society for Fuzzy Logic And Technology (EUSFLAT). His areas of research interest include engineering applications of artificial intelligence, grid/cloud computing, speech and audio analysis. He is also reviewer of several journals indexed in JCR. He has been involved in research projects of the Spanish Ministry of Science and Education (MEC), and private companies.