KASIA Approach vs. Differential Evolution in Fuzzy Rule-Based Meta-Schedulers for Grid Computing

R. P. Prado, S. García-Galán and J. E. Muñoz Expósito
Telecommunication Engineering Department.
University of Jaén. Alfonso X el Sabio, 28 Linares, Jaén. Spain.

***Abstract*—Many efforts have been made in the last few years to solve the high-level scheduling problem in Grid computing, i.e., the efficient resources utilization and allocation of workload within resources domains. Nowadays, some trends are based on the consideration of Fuzzy Rule-Based Systems, whose performance is critically conditioned to theirs knowledge bases quality. In this sense, Genetic Algorithms have been extensively used to obtain such knowledge bases, mainly founded on Pittsburgh approach. However, new strategies are recently emerging showing improvement over genetic-based learning methods. In this work, comparative results of two non-genetic learning strategies derived from bio-inspired algorithms, Differential Evolution and Particle Swarm Optimization, are presented for the evolution of fuzzy rule-based meta-schedulers in Grid computing.**

***Index Terms*—Grid Computing, Scheduling, Fuzzy Rule-Based Systems, Differential Evolution, Particle Swarm Optimization**

I. Introduction

Grid computing is increasingly emerging as a large-scale computational framework to solve problems in science, technology and engineering where traditional distributed systems are unable to cope with high computational demands [1]. It is mainly featured by the cooperation of a wide set of heterogeneous and geographically distributed resources, interconnected through high-speed networks. Moreover, resources belong to different resources domains (RDs) or sites and impose their own access and sharing constraints [2]. Furthermore, a grid consists of different RDs, making up a global virtual entity or Virtual Organization (VO). A long standing problem in Grid computing is the efficient allocation of jobs to resources or scheduling which is known to be NP-hard [3]. In this regard, computational grids are generally described as hierarchical structures concerning two scheduling levels; the RD and the VO levels [4], [5]. On the one hand, local schedulers or local resource managers are responsible for the allocation of jobs or scheduling within their associated domains. On the other hand, a meta-scheduler distributes workload among the different RDs and thus, it conducts the overall scheduling in the VO.

Several heuristics have been proposed to solve the scheduling problem in the highly demanding and distributed environments of computational grids. Heuristics can be broadly categorized into dynamic and static [6]. The main difference resides in the number of jobs taken into account at every scheduling stage or setting up of the objective jobs set. Static heuristics comprise OLB (Opportunistic Load Balancing), MET (Minimum Execution Time) and MCT (Minimum Completion Time) [6], [7]. In addition, representative dynamic heuristics are Min-Min, Max-min, RR (Round Robin), DFPLTF (Dynamic FPLTF) and WQ (Work Queue) [8]. It must be pointed out that these methods essentially base their schedules on the estimation of resources performance and jobs requirements. Nevertheless, in the light of the high demand for scheduling techniques able to satisfy complex QoS criteria, alternative methodologies such as adaptive strategies [9], are emerging. As discussed in [10], the grid system state must be born in mind by a scheduling strategy in order to be able to offer QoS demands. In this sense, adaptive scheduling strategies propose the prevention of the grid system performance degradation taking into account the current and future grid states at every schedule. However, given the high dynamism, changing behaviour and inherent uncertainty of the grid state, a precise grid state characterization cannot be provided. Thus, new approaches are focused on the application of mechanisms able to tolerate the noisy and imprecise grid systems information. In this regard, the role of Fuzzy Rule-Based Systems (FRBSs) must be underlined.

FRBSs are expert systems increasingly attracting the grid research community attention to solve the scheduling problem. This is mainly due to their ability to cope with uncertain information and simulation of human reasoning [11], [12]. However, given the high dependence of FRBSs performance with the quality of theirs knowledge bases (KBs), the learning problem in fuzzy rule-based schedulers emerges as a critical issue. It must be noted that the direct consideration of experts knowledge is not feasible in the great majority of application fields of FRBSs and thus, an automatic acquisition or learning of KBs must generally be born in mind. Many learning strategies for the acquisition of fuzzy rules can be found in literature. Genetics Algorithms (GAs) are evolutionary strategies which have shown their efficiency in their adaptation to the learning of KBs [13], [14], [15]. To be precise, two acknowledged strategies must be indicated, namely, Pittsburgh [16] and Michigan approaches [17]. Nevertheless, new bio-inspired learning strategies are emerging for the evolution of fuzzy rules showing a more efficient performance in its application to meta-schedulers in Grid computing. In this work, comparatives results of the adaptation of two non-genetic

978-1-61284-050-5/11/$26.00 ©2011 IEEE

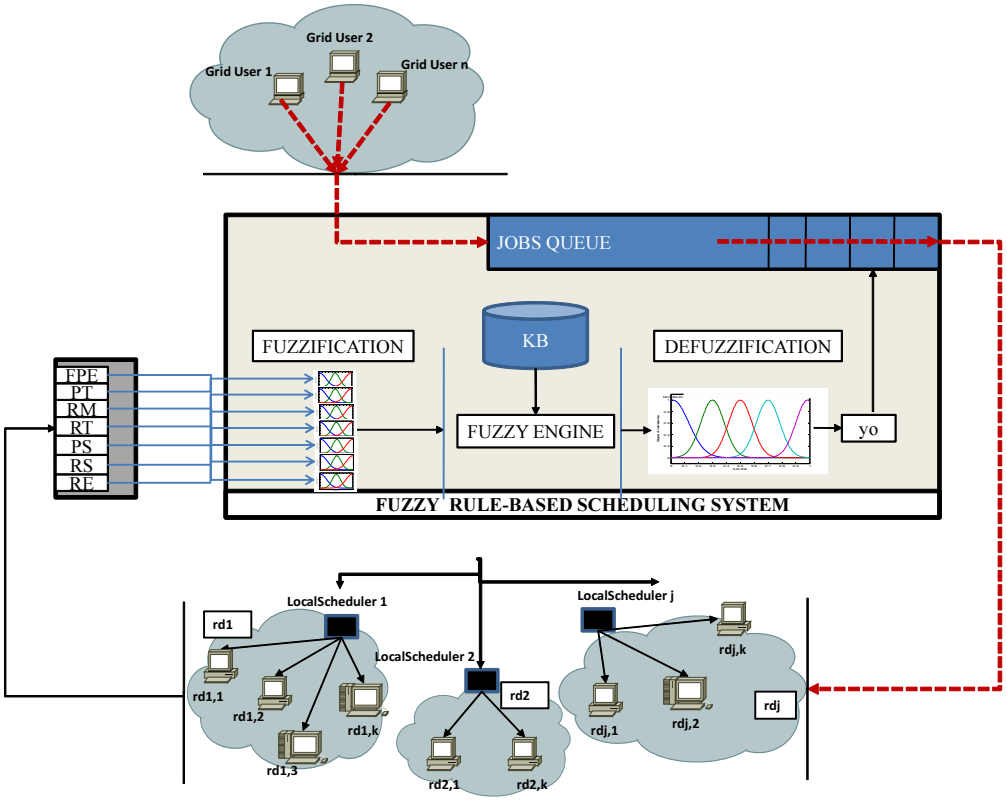

Figure 1. Fuzzy meta-scheduler structure within grid environment.

algorithms, i.e., Differential Evolution (DE) [18] and Particle Swarm Optimization (PSO) [19], to knowledge acquisition in fuzzy rule-based schedulers [20], *DE-learning* [21] and *Knowledge Acquisition with a Swarm Intelligence Approach (KASIA)* [22] are presented. It is shown that these strategies results improve those of Pittsburgh approach regarding final result and convergence behaviour. Comparing these learning strategies with Pittsburgh approach is specially interesting since they use a similar encoding of the fuzzy rule bases and they consider the evaluation of a fixed set of rule bases in every iteration. Hence, not only can these strategies be fairly compared in terms of final accuracy considering the same overall computational effort but also, convergence can be compared step by step in every iteration or generation taking into account an equal number of rule bases evaluations. Moreover, given the fact that a fixed size for the knowledge bases, i.e., the same number of fuzzy rules, are born in mind in both *DE-learning* and *KASIA* strategies, the evaluation of a single rule base requires the same number of operations.

The rest of the paper is organized as follows. First, Section II describes the structure of the FRBS applied to the scheduling problem in Grid computing. The proposed learning strategies for the fuzzy rule-based meta-scheduler are presented in Section III. In Section IV simulation and comparative results are presented. Finally, Section V concludes the paper.

II. Fuzzy Meta-Scheduler Structure

The knowledge-based meta-scheduler structure and organization within the grid environment are illustrated in Figure 1. The general schema of FRBSs can be observed for the meta-scheduler, i.e., *fuzzification, inference and defuzzification systems* and associated *knowledge base*. Hence, the basic operation of the meta-scheduler can be described as the joint operation of these systems and it is summarized as follows. At every scheduling step, the meta-scheduler examines all the available RD state information provided by local schedulers and it features such states through a limited and normalized collection of variables as shown in Table I. Then, the meta-scheduler starts the process of translating the RDs state fuzzy characterization into a RD selector index that represents the suitability of the considered RD to be selected for the next schedule. To be precise, the *fuzzification system* is in charge of providing a fuzzy set from the crisp value obtained for each RD state variable. This way, the retrieved RD state information is associated to a linguistic label considering its inherent uncertainty. Next, in the *inference system,* a fuzzy output is obtained through the application of the system knowledge, in the form of "IF-THEN" rules, to the input fuzzy characterization. Finally, a crisp value that corresponds to a fuzzy RD selector is inferred through the *defuzzification system*. Specifically, "center of gravity" is taken into account as defuzzification strategy. Thus, as described, the workload allocation among RDs is founded of the simultaneous consideration of a fuzzy

grid state characterization and acquired knowledge of the fuzzy meta-scheduler.

Moreover, it must be mentioned that the choice of these variables is based on the achievement of an adaptive scheduling [9]. Also, the considered membership functions for grid state are depicted in Figure 2. As shown, input features are represented by three gaussian-shaped sets corresponding to *low, medium* and *high* linguistic labels, while the output variable is described by five gaussian-shaped sets, i.e. *very low, low, medium high* and *very high* fuzzy sets. Considering the associated negation sets, null set for the seven features and output and connectors, this leads the learning system to contemplate (7^7-1)·11·2 (i.e., 18,117,924) different rules. It must be mentioned that a larger number of sets may be considered to provide a more complete characterization of the grid system state. However, with the aim of avoiding the expansion of the search space and the consequent complexity of the learning process, this selection attempts to achieve a balance between a good characterization of the fuzzy system state and computational effort of the knowledge acquisition processes.

III. Non-Genetic Knowledge Acquisition for Fuzzy Rule-Based Meta-Schedulers

Next, the adaptation of two optimization strategies to knowledge acquisition in FRBSs based on DE and PSO, *DE-learning* [21] and *KASIA* [22], are presented. As it is shown, these strategies consider each individual of the population as a whole rule base and they present many similarities in terms of rule bases encoding that make their features alike. Furthermore, the size of the rule bases considered by these two learning strategies is fixed through the whole learning process what allows a fair comparison regarding computational effort.

A. Differential Evolution Learning

The adaptation of DE to knowledge acquisition in fuzzy rule-based meta-schedulers [21] follows the general operation of evolutionary algorithms, i.e., initialization, mutation, crossover and selection. First, in the initialization stage, the encoding of rules and rule bases to be evolved must be specified. In this approach, every rule is represented by antecedents (corresponding to n input features), consequent and connectives of "IF-THEN" Mamdani rules [23]. Hence, founded on the considered rules encoding, population individuals have the form:

$$RB_g^i = \begin{bmatrix} x_{1,1,g}^i & x_{1,2,g}^i & \cdots & x_{1,n+2,g}^i \\ x_{2,1,g}^i & x_{2,2,g}^i & \cdots & x_{2,n+2,g}^i \\ \cdots & \cdots & \cdots & \cdots \\ x_{m,1,g}^i & x_{m,2,g}^i & \cdots & x_{m,n+2,g}^i \end{bmatrix}, \; i = 1, 2, ..., N. \tag{1}$$

where m indicates the number of fuzzy rules and g denotes the generation number. As shown, canonical DE algorithm has been modified to consider RBs as candidate individuals where every row represents the codification of a single fuzzy rule. This way, parameters vectors of the original algorithm are extended to $[n+2, m]$ dimension matrices describing whole sets of rules. Specifically, for the proposed scheduler setting, taking into account seven input variables for RD state characterization (see Table I), rule connector and one output variable, the number of rules is $n+2=9$. Moreover, for every rule base, RB^i, each rule component is randomly initialized bearing in mind each individual lower and upper limits for antecedents, consequents and connectives of rules:

$$x_{j,k,g}^i \in [-NF_{in},\, NF_{in}],\; j \in \{1, 2, ..., m\},\; k \in \{1, 2, ..., n\} \tag{2}$$

$$x_{j,k,g}^i \in [-NF_{out},\, NF_{out}],\; j \in \{1, 2, ..., m\},\; k = n+1 \tag{3}$$

$$x_{j,k,g}^i \in \{1, 2\},\; j \in \{1, 2, ..., m\},\; k = n+2 \tag{4}$$

with NF_{in} and NF_{out} the number of input and output fuzzy sets, respectively. Also, two connectives are considered ("AND" and "OR", encoded by 1 and 2, respectively). Note that rules weights are equal to unity in this approach and thus they do not influence the rules relevance.

Next, after population initialization, every individual RB_g^i is subject to mutation, crossover and selection procedures. First, mutation is used in a such a way that local optimums are avoided as it is generally considered in evolutionary algorithms. Specifically, in this approach, for every target base RB_g^i, three individuals, $RB_g^{r_1}$, $RB_g^{r_2}$ and $RB_g^{r_3}$ are randomly selected in a way that r_1, r_2 and r_3 are different to i and the weighted difference of $RB_g^{r_2}$ and $RB_g^{r_3}$ is added to the first rule base $RB_g^{r_1}$:

$$DB_{g+1}^i = RB_g^{r_1} + F(RB_g^{r_2} - RB_g^{r_3}) \tag{5}$$

where F denotes the mutation factor and DB_{g+1}^i indicates the donor base for RB_g^i. It must be mentioned that rules elements of auxiliary bases such as donor bases, are subject to the closer bound of their associated search space shown in Eqs. 2, 3 and 4 in order to avoid invalid chromosomes. Next, a crossover stage is entailed. In this stage, the trial base TB_{g+1}^i is obtained though the crossing of rules elements of both target, RB_g^i and donor bases, DB_{g+1}^i. To be precise, components of DB_{g+1}^i are included to TB_{g+1}^i with probability CR:

$$TB_{j,k,g+1}^i = \begin{cases} DB_{j,k,g+1}^i & if\, rand_{j,k}^i \leq CR \\ RB_{j,k,g}^i & if\, rand_{j,k}^i > CR \end{cases} \tag{6}$$

with $j \in \{1, 2, ..., NR\}$, $k \in \{1, 2, ..., D\}$. It must be mentioned that a uniform distribution is considered for crossover in this approach. Finally, in the selection process,

Table I
INPUTS FEATURES FOR THE FUZZY META-SCHEDULER.

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.
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.

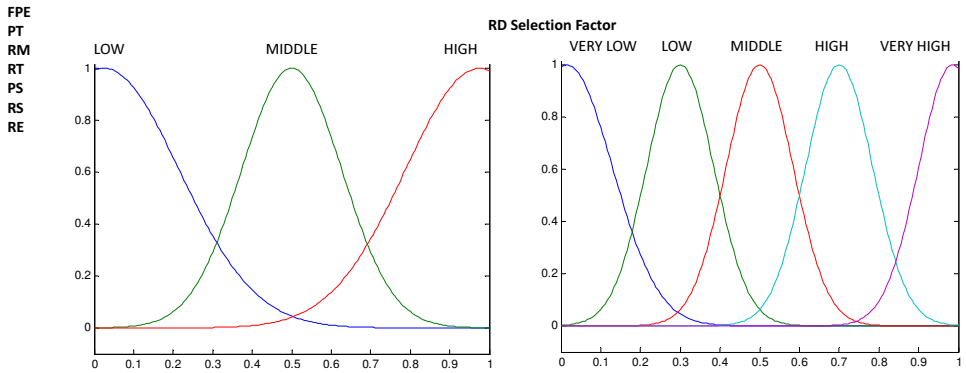

Figure 2. membership functions for the meta-scheduler inputs and output.

next generation population of rule bases are selected. With this aim, both target bases and trial bases are tested and those obtaining a better performance become members of the next population:

$$RB^i_{g+1} = \begin{cases} TB^i_{g+1} & if\ f(TB^i_{g+1}) \leq f(RB^i_g) \\ RB^i_g & othewise \end{cases} \quad (7)$$

The general procedure of the algorithm, i.e., mutation, crossover and selection, is iterated until the stopping condition is satisfied. Specifically, the number of generations, G, is proposed as stopping condition. Note other versions for DE, considering exponential crossover or x/rand differential variation [18], are possible. However, this approach proposes the use of canonical DE strategy for rules evolution.

B. KASIA approach

KASIA (*Knowledge Acquisition with a Swarm Intelligence Approach*) [22] is the adaptation of PSO to the learning of rule bases in FRBSs. In this approach, each individual or particle, P_i, represents a whole RB of the population so-called swarm and the aim is to achieve the optimum location for particles or equivalently, RBs with the highest quality. In other words, the rule set providing the best performance in terms of a selected function fitness f (desirable for the whole operation of the system) is sought. Hence, every rule base, RB_i, or particle P_i in KASIA approach is represented as

$$P_i = \begin{bmatrix} a^i_{1,1} & a^i_{1,2} & \dots & a^i_{1,n} & b^i_1 & c^i_1 \\ a^i_{2,1} & a^i_{2,2} & \dots & a^i_{2,n} & b^i_2 & c^i_2 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ a^i_{m,1} & a^i_{m,2} & \dots & a^i_{m,n} & b^i_m & c^i_m \end{bmatrix} \quad (8)$$

with every row denoting a fuzzy rule and n, m indicating the number of input features and rules, respectively. As far as rules representation is concerned, antecedents, $a^i_{j,k}$, are encoded as entire values in the interval

$$a^i_{j,k} \in [-NF_{in},\ NF_{in}]\,,\ j \in \{1, 2, ..., m\},\ k \in \{1, 2, ..., n\} \quad (9)$$

where NF_{in} indicates the number of fuzzy sets for input variable j, corresponding to three in the proposed schema, see Figure 2. Further, an analog reasoning can be followed for consequents b^i_j with NF_{out} the number of output fuzzy sets

$$b^i_j \in [-NF_{out},\ NF_{out}]\,,\ j \in \{1, 2, ..., m\} \quad (10)$$

$$NF_{in},\ NF_{out} \in \mathbb{N} \quad (11)$$

Also, connectives are encoded considering two values; AND connective is denoted as "1" and "2" indicates the OR operator,

$$c_j^i \in \{1, 2\} \tag{12}$$

In the initial stage of KASIA algorithm, each particle or RB, P_i, and an associated velocity, V_i, are randomly generated. In this algorithm, velocity is considered for the updating of the particles position during the search process and thus, velocity matrix drives each RB modification,

$$V_i = \begin{bmatrix} v_{1,1}^i & v_{1,2}^i & \dots & v_{1,n}^i & v_{1,n+1}^i & v_{1,n+2}^i \\ v_{2,1}^i & v_{2,2}^i & \dots & v_{2,n}^i & v_{2,n+1}^i & v_{2,n+2}^i \\ \dots & \dots & \dots & \dots & \dots & \dots \\ v_{m,1}^i & v_{m,2}^i & \dots & v_{m,n}^i & v_{m,n+1}^i & v_{m,n+2}^i \end{bmatrix} \tag{13}$$

$$v_{j,k}^i \in [V_{min},\, V_{max}]\,,\; j \in \{1, 2, ..., m\},\; k \in \{1, 2, ..., n+2\} \tag{14}$$

where V_{min} and V_{max} indicate the limits for velocity. Some comments must be made about V_{min} and V_{max}. Every particle in this learning approach updates its position founded of its own inertia, own best experience and global social experience in the optimization process. Furthermore, particles move in the search space in a constrained range $[-s, s]$. Specifically, the search space includes three subsections, as shown in particle or RB formulation, Eq. 8, regarding antecedent, consequent and connector search spaces. This way, the search space is defined by Eq. 9, Eq. 10 and Eq. 12 for each element of RBs, respectively. Moreover, with the final goal of effectively driving particles in the search space, the maximum range for velocity is bounded to a range [V_{min}, V_{max}], or taken symmetry into account, [$-V_{max}$, V_{max}]. As stated in [24], explosion in PSO strategies has generally been controlled through the consideration of a V_{max} parameter, which constraints velocity. Thus, the following is born in mind for particles velocity [11]:

$$v_{j,k}^i = sign(v_{j,k}^i,)min(|v_{j,k}^i|, V_{max}) \tag{15}$$

V_{max} is generally defined as $p \times s$, with $0.1 \leq p \leq 1.0$. To be precise, in this work, p is selected in such a way that V_{max} is the unity, i.e., $p = 1/3$ for antecedents, $p = 1/5$ for consequents and $p = 1/2$ for antecedents.

Hence, at every iteration $t + 1$, a performance indicator or fitness is obtained for every particle of the swarm and the velocity matrix is updated as follows

$$\begin{aligned} V_i(t+1) = w \otimes V_i(t) \oplus (c_1 * r_1) \otimes (P^{\#}(t) \ominus P_i(t)) \\ \oplus (c_2 * r_2) \otimes (P^*(t) \ominus P_i(t)) \end{aligned} \tag{16}$$

where $\otimes$ and $\oplus$ indicate the multiplication and sum of matrices, respectively, $P^{\#}(t)$ denotes the best knowledge (in terms of considered fitness f) for particle P_i and $P^*(t)$ represents the best knowledge achieved for the whole swarm up to the considered iteration. Thereby, each particle P_i is updated in regard of the present value or inertia, and the global and local best value reached though the swarm optimization and this process is iterated until the stopping condition is met

$$P_i(t+1) = P(t)_i \oplus V_i(t+1) \tag{17}$$

As it is found in literature with respect to FRBSs knowledge acquisition in large-scale computing environments [25], the specification of the stopping condition for bio-inspired algorithms is generally founded on statistical analysis of the system performance in the specific environment. In this work, the number of iterations is suggested as stopping condition. In addition, given the fact that antecedents, consequents and connectives may violate the search space limits as a results of location updating, a set of constrains are imposed for these components at the end of every iteration with the aim that RBs coherence is kept as in DE strategy i.e., prevention of invalid chromosomes,

$$a_{j,k}^i = \begin{cases} NF_{in} & if\, a_{j,k}^i > NF_{in} \\ -NF_{in} & if\, a_{j,k}^i < -NF_{in} \end{cases} \tag{18}$$

$$b_{j,k}^i = \begin{cases} NF_{out} & if\, b_{j,k}^i > NF_{out} \\ -NF_{out} & if\, b_{j,k}^i < -NF_{out} \end{cases} \tag{19}$$

$$c_j^i = \begin{cases} 1 & if\, c_j^i < 1 \\ 2 & if\, c_j^i > 2 \end{cases} \tag{20}$$

Moreover, two remarks must be born in mind. On the one hand, rules presenting null antecedents are considered to be non consistent (i.e., a controller output is provided regardless of the grid system state) and thus, they must be initialized in case they are obtained in the updating process

$$if\ \sum_{k=1}^{n} a_{j,k}^i = 0 \Rightarrow init\, a_j^i \tag{21}$$

On the other hand, it is to be noted that KASIA algorithm, as it also happens with DE-learning, does not lead the fuzzy sets modification. Thereby, rules interpretability is not altered in the knowledge acquisition process.

Finally, it is important to specify the associated computational effort for both strategies in terms of number

Table II
METACENTRUM-BASED GRID STRUCTURE.

Cluster	CPU speed (MHz)	Main memory size (KB)	CPU type	Operating system	number of machines	Total number of CPUs
cluster_0	1500	48,000,000	Itanium2	Linux	1	8
cluster_1	2200	32,000,000	Opteron	Linux	1	16
cluster_2	3200	1,009,000	Xeon	Linux	10	10
cluster_3	2600	131,182,840	Opteron	Linux	5	80
cluster_4	1600	1,005,000	AthlonMP	Linux	16	32
cluster_5	2400	1,048,576	Xeon	Linux	32	64
cluster_6	2659	15,565,060	Xeon	Linux	36	148
cluster_7	3056	2,021,000	Xeon	Linux	35	70
cluster_8	1600	1,024,000	Opteron	Linux	10	20
cluster_9	2400	4,000,000	Opteron	Linux	3	6
cluster_10	2000	4,000,000	Opteron	Linux	23	92
cluster_11	3000	4,546,800	Xeon	Linux	19	152
cluster_12	2660	27,343,000	Xeon	Linux	8	64
cluster_13	2360	15,200,000	Xeon	Linux	11	44

of RB evaluations. Computational effort for DE-learning can be formulated as

$$CE_{DE} = I * num_{gen} \tag{22}$$

where I denotes the number of candidates or individuals in the DE population and num_{gen} represents the number of generations. On the other hand, computational effort for KASIA strategy can be denoted as

$$CE_{KASIA} = NP * num_{iter} \tag{23}$$

with NP the number of particles, as introduced above, and num_{iter} the number of iterations for swarm updating. Hence, considering the associated encoding for every RB and taking the same number of individuals for the population and swarm for DE-learning and KASIA, respectively, strategies can be fairly compared in terms of overall and step by step computational effort.

IV. SIMULATION RESULTS

The proposed non-genetic knowledge acquisition strategies for the learning of meta-schedulers in Grid computing are tested with Alea [26]. Alea is a GridSim-based simulation software for the study of scheduling strategies in computational grids. To be precise, the grid scenario and workload traces are obtained from Czech National Grid Infrastructure Metacentrum [27]. The final aim of CESNET (operator of academic network of the Czech Republic -National Research and Education Network, NREN) project, Metacentrum, is to cooperate in the achievement a high-performance computing framework by the sharing and cooperation of a large set of institutions resources all around the world. Specifically, the grid scenario consists of 210 machines, including 806 heterogeneous CPUs (i.e., Opteron and Xeon, 1500-3200 MHz), running Linux and distributed in 14 RDs, see II. On the other hand, jobs are retrieved from Metacentrum traces from January 2009 (available at [27]). Workload traces provide relevant information about jobs such as job identifier or job ID, corresponding job owner priority, collection of properties to be satisfied in the end machine (i.e., number of CPU, CPU type, operating system, etc.) or job arrival time. Precisely, 2000 and 2400 jobs are involved in learning and validation scenarios, respectively. Furthermore, the dynamism of the grid resources behaviour considering aspects such as reservation, failure or availability times are retrieved from Maintenance & Reservation traces. Also, *makespan* [9] is suggested as training index (i.e., objective function) for the knowledge acquisition process,

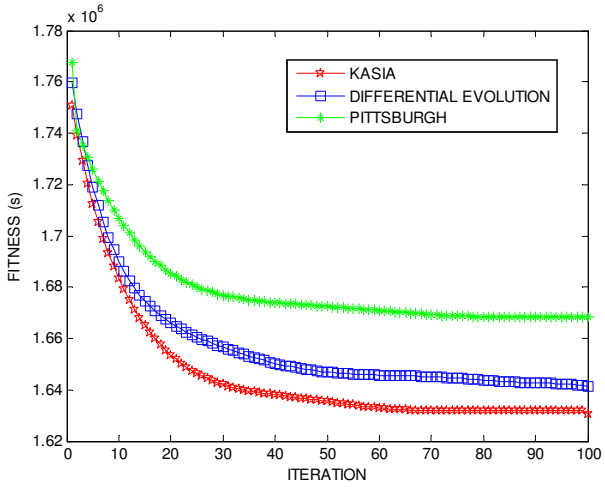

Figure 3. Convergence comparative results of KASIA and DE.

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

with T_j the finalization time for job J_j, $Sched$ all the possible schedules and J the set of considered workload jobs.

Regarding the implementation considered for each learning method to compare results, DE is configured with binomial crossover with probability CR=0.5, mutation factor F=0.8 with 1/rand differential variation, population size $I = 18$ and RB size, $RB_{size} = 10$ rules. On the other hand, KASIA is

Table III
SIMULATION RESULTS FOR 30 SIMULATIONS IN GRID METACENTRUM. TRAINING FITNESS MAKESPAN (S).

Results	Average	Standard Deviation	Confidence Interval (95%)
KASIA	1630479.5	36778.7	1617318.41, 1643640.50
Differential Evolution	1641823.2	22234.7	1633520.70, 1650125.85
GA-Pittsburgh	1667586.2	19757.8	1660515.96, 1674656.43

Table IV
RESULTS SUMMARIZATION RELATED TO PITTSBURGH APPROACH IN VALIDATION SCENARIO.

Approach	Configuration	% Improvement
Differential evolution	NP=18, Rules=10	1,10
KASIA	NP=18, Rules=10	1,58

configured with $\omega = 0.9$, $d_1 = 2$, $d_2 = 2$, $NP = 18$ and $RB_{size} = 10$. Further, Pittsburgh approach is set up with selection rate $\lambda = 0.9$ and population size $I_P = 20$, with the same RB size as the other bio-inspired strategies. In these conditions, the comparison in terms of computational cost is fair since the same number of particles or individuals is considered for every method and each knowledge base bears in mind the same number of rules.

In Figure 3, convergence behaviour of the different strategies during the learning processes is presented. Curves represent the best RB associated fitness at every iteration (30 simulations on average). It is illustrated that KASIA initial convergence behavior does not show any relevant difference in the stating iterations (approximately up to iteration 10) in comparison to DE. Nevertheless, from this point on, a faster convergence for KASIA learning strategy is observed. It must be noted that a faster convergence is translated into a more reduced computational effort as it can be derived from Eqs. 22 and 23. Also, statistics of the learning data are shown in Table III. Furthermore, in Table IV, validation results considering the relative improvement of the presented learning strategies with respect to the genetic knowledge acquisition with better results in fuzzy rule-based meta-schedulers, i.e., Pittsburgh approach, are shown. As it can be observed, DE-based strategy improves Pittsburgh approach by 1.10% whereas KASIA outperforms the genetic strategy by 1.58%. Hence, the PSO-based, KASIA, strategy presents both better performance in terms of convergence behaviour and accuracy with respect to DE-learning strategy and requiring the same step by step computational effort. Finally, Tables V and VI show best rule bases obtained by DE and KASIA strategies, respectively.

V. CONCLUSIONS

The design of efficient scheduling strategies is a major issue to harness the high potential of grids. Given the inherent changing behaviour of grid environments, dynamics models are progressively calling grids research community attention. Fuzzy rule-based meta-schedulers are expert systems founded on the fuzzy characterization of the grid system state and the application of expert knowledge to provide an efficient scheduling. However, due to the critical importance of the quality of the expert knowledge and the way this knowledge is acquired, learning strategies arise as one of the major challenges of these systems. In this work, scheduling results obtained with fuzzy rule-based meta-schedulers considering two non-genetic automatic learning strategies have been analyzed. On the one hand, an adaptation of DE to knowledge acquisition and on the other hand, a new learning strategy based on PSO (KASIA). Both strategies have proved their capability to acquire knowledge in FRBSs that are adapted to meta-schedulers in a grid system using makespan as criteria to measure the offered QoS to users. To be precise, the obtained results using both methodologies improve those achieved with classical strategies based on genetic algorithms regarding the same number of rule bases evaluations in every iteration. Furthermore, results show that KASIA outperforms DE by 0.5% and can faster reach convergence with the consequent reduction in computational effort.

ACKNOWLEDGMENT

This work has been financially supported by the Andalusian Government (Research Project P06-SEJ-01694).

REFERENCES

[1] I. Foster and C. Kesselman, "The grid: Blueprint for a new computing infrastructure," USA, 2004.

[2] J. M. Marín and S. B. Cámara, *Las Tecnologías Grid de la Información como Nueva Herramienta Empresarial*. Jaén (España): Septem Ediciones, 2008.

[3] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. New York, NY, USA: W. H. Freeman & Co., 1979.

[4] K. Christodoulopoulos, V. Sourlas, I. Mpakolas, and E. Varvarigos, "A comparison of centralized and distributed meta-scheduling architectures for computation and communication tasks in grid networks," *Computer Communications*, vol. 32, no. 7-10, pp. 1172 – 1184, 2009.

[5] S. Phatanapherom, P. Uthayopas, and V. Kachitvichyanukul, "Dynamic scheduling ii: fast simulation model for grid scheduling using hypersim," in *WSC '03: Proceedings of the 35th conference on Winter simulation*. Winter Simulation Conference, 2003, pp. 1494–1500.

[6] L. Tseng, Y. Chin, and S. Wang, "The anatomy study of high performance task scheduling algorithm for grid computing system," *Computer Standards and Interfaces*, vol. 31, no. 4, pp. 713 – 722, 2009.

[7] H. S. T.D. Braun and N. Beck, "A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems," *Journal of Parallel and Distributed Computing*, vol. 61, pp. 810–837, 2001.

[8] N. Fujimoto and K. Hagihara, "A comparison among grid scheduling algorithms for independent coarse-grained tasks," in *SAINT-W '04: Proceedings of the 2004 Symposium on Applications and the Internet-Workshops (SAINT 2004 Workshops)*. Washington, DC, USA: IEEE Computer Society, 2004, p. 674.

[9] F. Xhafa and A. Abraham, "Meta-heuristics for grid scheduling problems," *Metaheuristics for Scheduling: Distributed Computing Environments, Studies in Computational Intelligence, Springer Verlag, Germany, ISBN*, pp. 978–3, 2008.

Table V
OBTAINED RB WITH DE STRATEGY.

Rule Base for DE
1. If (FPE is HIGH) and (PT is HIGH) and (RM is MIDDLE) and (RT is MIDDLE) and (RS is not HIGH) and (RE is HIGH) then (OUTPUT is VERYHIGH) (1)
2. If (FPE is HIGH) or (PT is HIGH) or (RM is LOW) or (RT is HIGH) or (PS is HIGH) or (RS is HIGH) or (RE is HIGH) then (OUTPUT is VERYHIGH) (1)
3. If (PT is HIGH) or (RM is HIGH) or (RT is not HIGH) or (PS is HIGH) or (RS is not LOW) or (RE is HIGH) then (OUTPUT is VERYHIGH) (1)
4. If (FPE is not MIDDLE) and (PT is not LOW) and (RM is LOW) and (PS is HIGH) and (RS is HIGH) and (RE is HIGH) then (OUTPUT is LOW) (1)
5. If (FPE is HIGH) or (RT is not HIGH) or (PS is not MIDDLE) then (OUTPUT is not VERYLOW) (1)
6. If (FPE is MIDDLE) and (PT is HIGH) and (RT is LOW) and (RE is HIGH) then (OUTPUT is not VERYHIGH) (1)
7. If (FPE is HIGH) and (PT is LOW) and (RT is HIGH) and (PS is not LOW) and (RE is HIGH) then (OUTPUT is VERYHIGH) (1)
8. If (FPE is not LOW) or (PT is LOW) or (RM is HIGH) or (PS is HIGH) or (RS is LOW) or (RE is MIDDLE) then (OUTPUT is VERYHIGH) (1)
9. If (FPE is HIGH) or (PT is HIGH) or (RT is not MIDDLE) or (PS is HIGH) or (RS is HIGH) or (RE is not HIGH) then (OUTPUT is VERYLOW) (1)
10. If (FPE is not HIGH) or (RM is HIGH) or (RT is not HIGH) or (PS is LOW) or (RE is not MIDDLE) then (OUTPUT is VERYHIGH) (1)

Table VI
OBTAINED RB WITH KASIA STRATEGY.

Rule Base for KASIA
1. If (FPE is not LOW) and (PT is not HIGH) and (RT is not MIDDLE) and (PS is not HIGH) and (RS is not HIGH) and (RE is not HIGH) then (OUTPUT is not HIGH) (1)
2. If (PT is not HIGH) and (RM is not LOW) and (RT is not HIGH) and (PS is not HIGH) and (RS is not HIGH) and (RE is not MIDDLE) then (OUTPUT is not LOW) (1)
3. If (FPE is not HIGH) and (PT is not HIGH) and (RM is not HIGH) and (RT is not HIGH) and (PS is not LOW) and (RS is not HIGH) and (RE is not LOW) then (OUTPUT is VERYLOW) (1)
4. If (FPE is not HIGH) and (PT is not LOW) and (RM is not HIGH) and (RT is not MIDDLE) and (PS is MIDDLE) and (RS is not HIGH) and (RE is not MIDDLE) then (OUTPUT is not LOW) (1)
5. If (FPE is not HIGH) and (PT is not HIGH) and (RM is not HIGH) and (RT is not HIGH) and (PS is not HIGH) and (RS is not MIDDLE) and (RE is MIDDLE) then (OUTPUT is not VERYHIGH) (1)
6. If (FPE is not MIDDLE) and (RM is not MIDDLE) and (RT is not HIGH) and (PS is not HIGH) and (RS is LOW) and (RE is not HIGH) then (OUTPUT is not MIDDLE) (1)
7. If (FPE is not LOW) and (PT is not MIDDLE) and (RM is not LOW) and (PS is not HIGH) and (RS is not HIGH) and (RE is not HIGH) then (OUTPUT is not VERYLOW) (1)
8. If (FPE is not LOW) and (PT is LOW) and (RM is not LOW) and (RT is not MIDDLE) and (PS is not HIGH) and (RE is not HIGH) then (OUTPUT is not VERYLOW) (1)
9. If (FPE is not HIGH) and (PT is not MIDDLE) and (RM is not HIGH) and (RT is not HIGH) and (PS is not HIGH) and (RS is not HIGH) and (RE is not LOW) then (OUTPUT is VERYLOW) (1)
10. If (FPE is not HIGH) and (PT is not HIGH) and (RM is not MIDDLE) and (RT is not MIDDLE) and (PS is not HIGH) and (RS is not LOW) and (RE is not HIGH) then (OUTPUT is not LOW) (1)

[10] D. Klusacek, "Dealing with Uncertainties in Grids through the Event-based Scheduling Approach," in *Fourth Doctoral Workshop on Mathematical and Engineering Methods in Computer Science (MEMICS 2008)*, vol. 1. Ing. Zdeněk Novotnỳ CSc., Ondráčkova 105, 628 00 Brno Further information, 2008, pp. 978–80.

[11] H. Liu, A. Abraham, and A. E. Hassanien, "Scheduling jobs on computational grids using a fuzzy particle swarm optimization algorithm," *Future Gener. Comput. Syst.*, vol. In Press, Corrected Proof, pp. –, 2009.

[12] J. Zhou, K.-M. Yu, C.-H. Chou, L.-A. Yang, and Z.-J. Luo, "A dynamic resource broker and fuzzy logic based scheduling algorithm in grid environment," in *ICANNGA '07: Proceedings of the 8th international conference on Adaptive and Natural Computing Algorithms, Part I*. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 604–613.

[13] O. Cordon, F. Herrera, and P. Villar, "Generating the knowledge base of a fuzzy rule-based system by the genetic learning of the data base," *Fuzzy Systems, IEEE Transactions on*, vol. 9, no. 4, pp. 667–674, Aug 2001.

[14] S. Garcia, A. Fernandez, J. Luengo, and F. Herrera, "A study of statistical techniques and performance measures for genetics-based machine learning: accuracy and interpretability," *Soft Comput.*, vol. 13, no. 10, pp. 959–977, 2009.

[15] C. Franke, F. Hoffmann, J. Lepping, and U. Schwiegelshohn, "Development of scheduling strategies with genetic fuzzy systems," *Appl. Soft Comput.*, vol. 8, no. 1, pp. 706–721, 2008.

[16] S. F. Smith, "A learning system based on genetic adaptive algorithms," Ph.D. dissertation, Pittsburgh, PA, USA, 1980.

[17] L. B. Booker, D. E. Goldberg, and J. H. Holland, "Classifier systems and genetic algorithms," *Artif. Intell.*, vol. 40, no. 1-3, pp. 235–282, 1989.

[18] R. Storn and K. Price, "Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces," *J. of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1996.

[19] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *IEEE International Conference on Neural Networks, 1995. Proceedings.*, vol. 4, 1995.

[20] R. P. Prado, S. G. Galán, A. J. Yuste, J. E. M. Expósito, A. J. S. Santiago, and S. Bruque, "Evolutionary fuzzy scheduler for grid computing," ser. Lecture Notes in Computer Science, vol. 5517. Springer, 2009, pp. 286–293.

[21] R. P. Prado, S. García-Galán, J. E. M. Expósito, A. J. Yuste, and S. Bruque, "Learning of fuzzy rule-based meta.schedulers for grid computing with differential evolution," in *IPMU(1)*, 2010, pp. 751–760.

[22] R. P. Prado, S. García-Galán, J. E. Muñoz Expósito, and A. J. Yuste, "Knowledge acquisition in fuzzy rule based systems with particle swarm optimization," *Fuzzy Systems, IEEE Transactions on*, vol. 18, no. 6, pp. 1083 –1097, 2010.

[23] E. H. Mamdani and Others, "Application of fuzzy algorithms for control of simple dynamic plant," *Procedings of IEEE*, vol. 121, no. 12, pp. 1585–1588, 1974.

[24] M. Clerc and J. Kennedy, "The particle swarm - explosion, stability, and convergence in a multidimensional complex space," *IEEE transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 58–73, 2002.

[25] C. Franke, J. Lepping, and U. Schwiegelshohn, "On advantages of scheduling using genetic fuzzy systems," in *JSSPP'06: Proceedings of the 12th international conference on Job scheduling strategies for parallel processing*. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 68–93.

[26] D. Klusacek, L. Matyska, and H. Rudova, "Alea - grid scheduling simulation environment," in *Parallel Processing and Applied Mathematics*, ser. Lecture Notes in Computer Science, R. Wyrzykowski, J. Dongarra, K. Karczewski, and J. Wasniewski, Eds., vol. 4967, intel; Microsoft; IBM; Action SA; SIAM. Heidelberger Platz 3, D-14197 Berlin, Germany: Springer-Verlag Berlin, 2008, Proceedings Paper, pp. 1029–1038, 7th International Conference on Parallel Processing and Applied Mathematics, Gdansk, Poland, Sep 09-12, 2007.

[27] C. N. G. Infrastructure, "Metacentrum data sets, http://www.fi.muni.cz/~xklusac/index.php?page= meta2009," 2009. [Online]. Available: http://www.fi.muni.cz/~xklusac/index.php?page= meta2009