Real-Time Scheduling of Flexible Manufacturing Systems Using Support Vector Machines and Case-Based Reasoning

Paolo Priore, Raúl Pino, José Parreño, Javier Puente, and Borja Ponte

***Abstract*—Dispatching rules are usually applied to schedule jobs in Flexible Manufacturing Systems (FMSs) dynamically. Despite their frequent use, one of the drawbacks that they display is that the state the manufacturing system is in dictates the level of performance of the rule. As no rule is better than all the other rules for all system states, it would highly desirable to know which rule is the most appropriate for each given condition, and to this end this paper proposes a scheduling approach that employs Support Vector Machines (SVMs) and case-based reasoning (CBR). Using these latter techniques, and by analysing the earlier performance of the system, "scheduling knowledge" is obtained whereby the right dispatching rule at each particular moment can be determined. A module that generates new control attributes is also designed in order to improve the "scheduling knowledge" that is obtained. Simulation results show that the proposed approach leads to significant performance improvements over existing dispatching rules.**

***Index Terms*—CBR, FMSs, scheduling, simulation, SVMs,**

I. Introduction

One of the most commonly applied solutions to the scheduling problem in FMSs involves using dispatching rules, which have been evaluated for performance by many researchers (see for example, [1]-[3]). Almost all the above studies point to the fact that rule performance depends on the criteria that are chosen, and the system's configuration and conditions (utilization level of the system, relative loading, due date tightness, and so on). It would thus be interesting to be able to change dispatching rules at the right moment dynamically.

The literature describes two basic approaches to modify dispatching rules. The first approach is to select a rule at the appropriate moment by simulating a set of pre-established dispatching rules and opting for the one that provides the best performance (see for example, [4]-[7]). The second approach, involving artificial intelligence, requires a set of earlier system simulations (training examples) to determine what the best rule is for each possible system state. A machine learning algorithm [8] is trained to acquire knowledge through these training examples, and this knowledge is then used to make intelligent decisions in real time (see for example, [9]-[11]). Aytug [12] and Priore [13] provide a review in which machine learning is applied to solving scheduling problems.

Nevertheless, there are hardly any studies in the literature that compare the different types of machine learning algorithms used in scheduling problems. This paper therefore presents a scheduling approach that uses and compares SVMs and CBR. To improve the manufacturing system's performance, a new approach is also proposed whereby new control attributes that are arithmetical combinations of the original attributes can be determined.

The rest of this paper is organized as follows. Machine learning algorithms used in this paper are first described. An approach to scheduling jobs that employs machine learning is then presented. This is followed by the experimental study, which describes a new approach to determine new control attributes from the original ones. The two machine learning algorithms used are also compared. Finally, the proposed scheduling approach is compared with the alternative of using a combination of dispatching rules constantly. A summary of the results obtained concludes the paper.

II. Case-based Reasoning and Support Vector Machines

The nearest neighbour algorithm is one of the most popular of CBR algorithms [14]. The formulation of this algorithm, called NN, or k-NN in the more sophisticated version is extremely simple. To calculate a new case's class using this method, the distances between the case and the training examples have to be calculated and the shortest distance found. The new case's class will be the same as the 'nearest' training example's class. The following formulation is employed to calculate the distance ($d(x,e)$) between a new case (x) and a training example (e):

$$d(x,e) = \sqrt{\sum_{i=1}^{n} w_i \times (a_{ix} - a_{ie})^2} \quad (1)$$

where n is the number of attributes considered; a_{ix} is the value of attribute i in case x; a_{ie} is the value of attribute i in example e, and w_i is the weight assigned to attribute i as a function of its importance.

Another two more sophisticated versions of the nearest neighbor algorithm have also been applied in this study. In the first of the two, an integer value of k of three is employed, so that the three nearest neighbors are calculated for each x.

Manuscript received August 1, 2013; revised November 5, 2013. This work has been supported by the Government of the Principality of Asturias, through Council of Economy and Employment. (Project Reference: SV-PA-13-ECOEMP-74).

The authors are with the Department of Business Administration, Polytechnic School of Engineering (University of Oviedo), Campus de Viesques s/n, CP 33204, Gijón (Asturias), Spain (e-mail: priore@uniovi.es, pino@uniovi.es, parreno@uniovi.es, jpuente@uniovi.es, uo183377@uniovi.es).

The class of x is then determined as a function of the class of the majority of the three neighbors. In the second version, a value of k of five is used.

Support vector machines [15] were originally designed for binary classification. Let (x_1, y_1), (x_2, y_2), … , (x_n, y_n) be a group of data belonging to Class 1 or Class 2, where $x_i \in R^N$ and the associated labels be y_i=1 for Class 1and -1 for Class 2 (i=1, … , n). The formulation of SVMs is as follows:

$$Min \quad \frac{1}{2} w^T w + C \sum_{i=1}^{n} \xi_i \tag{2}$$

subject to the constraints:

$$\begin{aligned} & y_i (w^T \phi(x_i) + b) \geq 1 - \xi_i \quad i = 1,...,n \\ & \xi_i \geq 0 \quad i = 1,...,n \end{aligned} \tag{3}$$

where w is the weight vector; C is the penalty weight; ξ_i are non-negative slack variables; b is a scalar, and x_i are mapped into a higher dimensional space by a non-linear mapping function ϕ. Mapping function ϕ needs to satisfy the following equation:

$$k(x_i, x_j) = \phi(x_i)^T \phi(x_j) \tag{4}$$

where $k(x_i, x_j)$ is called kernel function.

Minimizing $(1/2) \cdot w^T w$ implies that SVMs tries to maximize $2/\|w\|$, which represents the margin of separation between both classes. The data that satisfy the equality in (3) are called support vectors. Moreover, by adding a set of non-negative Lagrange multipliers α_i and β_i to generate the Lagrangian, the upper- mentioned constrained optimization problem can be worked out with the dual form shown below:

$$Max \quad \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j k(x_i, x_j) \tag{5}$$

subject to the constraints:

$$\begin{aligned} & \sum_{i=1}^{n} \alpha_i y_i = 0 \\ & 0 \leq \alpha_i \leq C \quad i = 1,...,n \end{aligned} \tag{6}$$

Having obtained the support vectors (SVs), the decision function for an unseen data (x) is as follows:

$$y = sign \left\{ \sum_{SVs} \alpha_i y_i k(x, x_i) + b \right\} \tag{7}$$

III. Scheduling Using Case-based Reasoning and Support Vector Machines

Two contrasting features need to be fulfilled for a real-time scheduling system that dynamically modifies dispatching rules to work properly [16]:

1) Rule selection must take into account a variety of information about the manufacturing system in real time.
2) Rule selection must be completed fast enough for real operations not to be delayed.

One way of doing this is to employ some class of knowledge about the relationship between the manufacturing system's state and the rule to be applied at that moment. However, one of the most difficult problems is precisely how this knowledge is to be acquired. Machine learning algorithms, such as SVMs or CBR, are used to do this. However, the training examples and the learning algorithm must be adequate for this knowledge to be useful. Moreover, in generating the training examples, the attributes selected are crucial to the performance of the scheduling system [17].

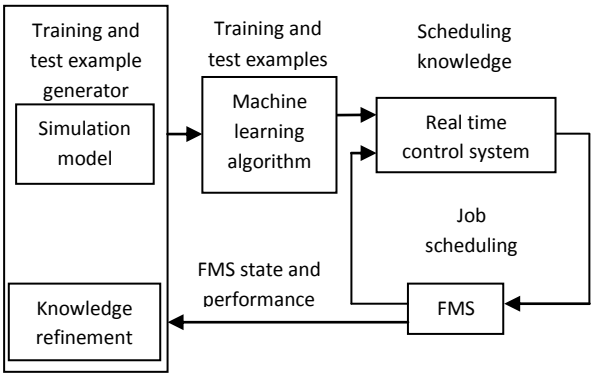

Fig. 1. General overview of a knowledge-based scheduling system.

Fig. 1 shows a scheduling system that employs machine learning. The example generator creates different manufacturing system states via the simulation model and choose the best dispatching rule for each particular state. The machine learning algorithm employs the examples to generate the knowledge required to make future scheduling decisions. The real time control system using the 'scheduling knowledge', the manufacturing system's state and performance, choose the best dispatching rule for job scheduling. Further examples may possibly be needed in order to refine the knowledge about the manufacturing system depending on the performance of the latter.

IV. Experimental Study

A. The Proposed FMS

The selected FMS consists of a loading station, an unloading station, four machining centres and a material handling system. Two types of decision are studied in the FMS proposed. The first is the selection by the machine of parts assigned to it using the following dispatching rules: SPT (Shortest Processing Time), EDD (Earliest Due Date), MDD (Modified Job Due Date), and SRPT (Shortest Remaining Processing Time). These rules were selected because of their fine performance in earlier studies (see for example, [18]). The second type of decision involves the selection of the machines by the parts, as an operation can be processed on different machines. The dispatching rules employed in this FMS are: SPT (Shortest Processing Time), NINQ (Shortest Queue), WINQ (Work in Queue), and LUS (Lowest Utilized Station).

B. Generating Training and Test Examples

The control attributes used to describe the manufacturing system state must first be defined in order to generate training and test examples. In this particular FMS these are: F, flow allowance factor which measures due date tightness [19]; NAMO: number of alternative machines for an operation; MU: mean utilization of the manufacturing system; U_i: utilization of machine *i*; WIP: mean number of parts in the system; RBM: ratio of the utilization of the bottleneck machine to the mean utilization of the manufacturing system; RSDU: ratio of the standard deviation of individual machine utilizations to mean utilization.

The training and test examples needed for the learning stage are obtained by simulation using the WITNESS programme. The following suppositions were made to do this: (1) Jobs arrive at the system following a Poisson distribution; (2) Processing times for each operation are sampled from an exponential distribution with a mean of one; (3) The actual number of operations of a job is a random variable, equally distributed among the integers from one to four; (4) The probability of assigning an operation to a machine depends on the parameters PO_i (percentage of operations assigned to machine *i*). These percentages fluctuate between 10% and 40%. It is also assumed that the first two machines have a greater workload; (5) The number of alternative machines for an operation varies between one and four; (6) The job arrival rate varies so that the overall use of the system ranges between 55% and 95%; (7) The value of factor *F* fluctuates between one and ten.

As mean tardiness and mean flow time in the system are the most widely used criteria to measure system performance in all manufacturing systems, they are also employed in this study. In all, 1100 different control attribute combinations were randomly generated. For each combination of attributes, mean tardiness and mean flow time values resulting from the use of each of the dispatching rules in isolation were calculated. Sixteen simulations were actually needed to generate a training or test example, as there are four rules for each of the decisions to be taken.

C. The Application of Case-Based Reasoning

One of the major drawbacks associated with the nearest neighbor algorithm is that its ability to classify new cases depends on the weights w_i that are chosen. Each one of these weights is assigned as a function of the importance of the corresponding attribute. However the value of each w_i is not known a priori. To get around this problem, a genetic algorithm [20] is designed which determines the optimum values of w_i, so that classification error is kept to a minimum. A scheme of this system, where one can see that the genetic algorithm employs the nearest neighbor method to calculate the classification error for given w_i values, is shown in Fig. 2. This error is the fitness for a given set of w_i weights. After a certain number of generations the genetic algorithm will identify the optimum values of w_i. The codification used in the genetic algorithm proposed uses integer numbers. The most appropriate values of population size (*N*), crossover probability (*CP*), mutation probability (MP), and maximum number of generations (NG_{max}) were also studied. Values obtained are: $N = 50$, $CP = 0.7$, $MP = 0.025$ and $NG_{\max} = 100$.

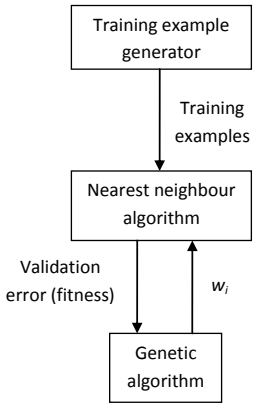

Fig. 2. Method for calculating optimum w_i weights.

Table I provides a summary of the results obtained using different-sized sets of examples for the criteria of mean tardiness and mean flow time. Generally, it can be seen that as the number of examples increases, test example error (examples that have not previously been dealt with) decreases considerably. Table I also shows that test error fluctuates between 11% and 9% upwards of 500 examples for the criterion of mean tardiness (using the 10-fold cross-validation and k=5). Furthermore, for the criterion of mean flow time, the nearest neighbor algorithm (k-NN with k=1) obtains zero test error upwards of 300 examples. Errors for this latter criterion are lower due to there being five dispatching rule combinations (SPT+SPT, SPT+NINQ, SPT+WINQ, MDD+WINQ, SRPT+WINQ) that are really used. In contrast, twelve combinations are used for the criterion of mean tardiness.

TABLE I: TEST ERROR USING THE NEAREST NEIGHBOUR ALGORITHM FOR THE CRITERIA OF MEAN TARDINESS (MT) AND MEAN FLOW TIME (MFT)

Number of examples	Test error (MT)	Test error (MFT)
200	16%	2%
300	15%	0%
400	14%	0%
500	11%	0%
600	11%	0%
700	10%	0%
800	10%	0%
900	10%	0%
1000	9%	0%
1100	9%	0%

D. The Application of Support Vector Machines

The scheduling problem is essentially a multi-class classification problem as several dispatching rule combinations are employed in the FMS. This study uses the one-against-one method to extend the binary SVMs to generate the multi-class scheduler since this method is more suitable for practical use than other methods [21]. In the same way, in this study, the radial basis function (RBF) and the polynomial function have been used as kernel functions. After several preliminary tests, it has been decided to make use of the RBF Kernel since it is the one that shows a better performance. Furthermore, by employing the grid search technique on the examples, the best performance for the SVMs is obtained when C=1,000 and σ=10. Table II provides a summary of the results obtained for the criteria of mean tardiness and mean flow time. Generally, it can be seen

that as the number of examples increases, test example error decreases considerably. Table II also shows that test error fluctuates between 11% and 10% upwards of 700 examples for the criterion of mean tardiness. Furthermore, for the criterion of mean flow time, test error drops to 1% upwards of 700 examples.

TABLE II: TEST ERROR USING SVMS FOR THE CRITERIA OF MEAN TARDINESS (MT) AND MEAN FLOW TIME (MFT)

Number of examples	Test error (MT)	Test error (MFT)
200	16%	6%
300	15%	5%
400	15%	2%
500	16%	2%
600	12%	2%
700	11%	1%
800	11%	1%
900	11%	1%
1000	10%	1%
1100	10%	1%

E. Generating New Control Attributes

On occasions, it is necessary to obtain arithmetical combinations of the original attributes to improve the scheduling knowledge. But in many cases these combinations are not known beforehand, and are only found in very simple manufacturing systems after close examinations of their simulation results. For these reasons, a module was designed which automatically selects the 'useful' combinations of the original attributes by using simulation data which originally provided test and training examples. To do this, the basic arithmetic operators considered are adding, subtracting, multiplying and dividing. The pseudo-code for the generator of the new control attributes is as follows:

1) Determination of the combinations of the present attributes.
2) Generation of new training and test examples in the light of earlier combinations.
3) Selection of the 'useful' combinations, which are in the decision tree and in the set of decision rules generated by C4.5 [22].
4) If the new decision tree and/or the set of decision rules has fewer classification errors, go back to step one. If not, stop the algorithm.

However, the decision to continue may also be taken at step four because, even though error may not be improved by the present iteration, it may well be improved during later iteration(s). Fig. 3 provides an overview of this module.

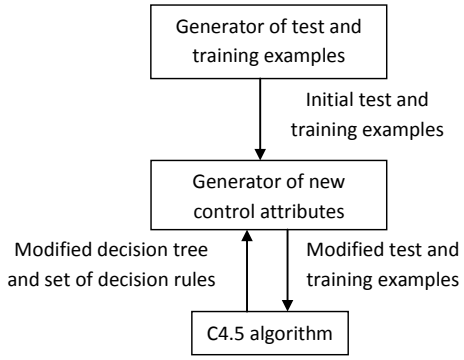

Fig. 3. Generator module of new control attributes.

The proposed module rendered the following 'useful' control attribute combinations for the criterion of mean tardiness: U1+U2, U1+U4, U2+U3, U1-U2, U2-U4, U3-U4 and U3/U4. Table III shows the results obtained for the criterion of mean tardiness when the SVMs and the generator module of new control attributes were applied. It can be seen from the results that test error oscillates between 10% and 8% from 600 examples upwards, and that the lowest test error was achieved with 1000 and 1100 examples. The proposed module is then applied for the criterion of mean flow time, and the following combinations of attributes were determined to be 'useful': U1-U2, U3-U4, U1/U2 and U2/U3. The Table shows how test error drops to 0% from 700 examples upwards. If these results are compared with those in Table II, an improvement can be seen to exist. Only sets of 600 examples or more were used, as lower errors are obtained upwards of this set size.

TABLE III: TEST ERROR USING SVMS AND THE GENERATOR MODULE OF NEW CONTROL ATTRIBUTES FOR THE CRITERIA OF MEAN TARDINESS (MT) AND MEAN FLOW TIME (MFT)

Number of examples	Test error (MT)	Test error (MFT)
600	10%	1%
700	9%	0%
800	9%	0%
900	9%	0%
1000	8%	0%
1100	8%	0%

TABLE IV: TEST ERROR USING THE NEAREST NEIGHBOR ALGORITHM AND THE GENERATOR MODULE OF NEW CONTROL ATTRIBUTES FOR THE CRITERIA OF MEAN TARDINESS (MT) AND MEAN FLOW TIME (MFT)

Number of examples	Test error (MT)	Test error (MFT)
600	10%	0%
700	9%	0%
800	9%	0%
900	10%	0%
1000	8%	0%
1100	8%	0%

Test error using the nearest neighbor algorithm and the new attributes generated was likewise calculated. Results are shown in Table IV, where it is again clear that classification error drops compared to the alternative of using the original control attributes. For the criterion of mean tardiness, the nearest neighbor algorithm gives a test error of 8% upwards 1000 examples. Furthermore, for the criterion of mean flow time, the nearest neighbor algorithm gives a test error of zero. It is observed that test errors obtained from both algorithms (SVMs and k-NN) are very similar

F. Learning-Based Scheduling

To select the best combination of dispatching rules according to the FMS's state in real time we must implement the scheduling knowledge in the FMS simulation model. Selecting the monitoring period is another key question because the frequency used to test the control attributes determines the performance of the manufacturing system. To do this, multiples of the average total processing time for a job, which in our particular case are 2.5, 5, 10 and 20 time units, are taken (see for example, [5]-[7]). In view of the results in the previous section, 1000 examples were used for both performance criteria. Five independent replicas of

100000 time units were carried out.

Table V summarizes the results obtained. Mean tardiness and mean flow time values in the Table are the average of the five replicas. Readability has been improved by showing values in the Table that are relative to the lowest mean tardiness and mean flow time obtained (these are assigned a value of one). The monitoring period chosen was 2.5 time units. Table V shows that the best alternative is to employ a knowledge-based strategy and that the SVMs generate the lowest mean tardiness values. The combinations MDD+NINQ and MDD+WINQ are the best of the strategies that use a fixed combination of dispatching rules, but their mean tardiness values are higher than the SVMs alternative by between 14.50% and 15.66%.

TABLE V: MEAN TARDINESS (MT) AND MEAN FLOW TIME (MFT) FOR THE PROPOSED STRATEGIES

Strategy used	MT	MFT	Strategy used	MT	MFT
SPT+SPT	5.4916	2,4183	**MDD+NINQ**	**1.1450**	1,2569
SPT+NINQ	1.2220	**1,0500**	**MDD+WINQ**	**1.1566**	1,2620
SPT+WINQ	1.2011	**1,0476**	MDD+LUS	2.5326	1,8646
SPT+LUS	2.5920	1,5277	SRPT+SPT	6.1452	2,6706
EDD+SPT	4.7207	2,6260	SRPT+NINQ	1.4174	1,1477
EDD+NINQ	1.6802	1,3991	SRPT+WINQ	1.4089	1,1486
EDD+WINQ	1.6885	1,4030	SRPT+LUS	3.0247	1,7195
EDD+LUS	3.2958	2,0614	**SVMs**	**1.0000**	1,0059
MDD+SPT	4.7620	2,6908	**k-NN**	1,0039	**1.0000**

Moreover, the k-NN algorithm gives the best results for the criterion of mean flow time. Table V also shows that the combinations SPT+NINQ and SPT+WINQ generate the least mean flow time from amongst the strategies that apply a fixed combination of rules. However, mean flow time values are greater than the k-NN alternative by between 4.76% and 5%. Finally, the SVMs-based scheduling system is compared with the other strategies by using ANOVA. The conclusion is that this scheduling system stands out above the other strategies with a significance level of less than 0.05. The only exceptions occur between the k-NN algorithm and the SVMs for both criteria.

V. CONCLUSIONS

An approach for scheduling using SVMs and case-based reasoning is proposed in this study. A generator module of new control attributes is also incorporated, and this reduces test error obtained with the machine learning algorithms leading to better performance of the manufacturing system. The knowledge-based scheduling system is shown to provide the lowest mean tardiness and mean flow time values. Future research might focus on using more decision types for the proposed FMS. However, the more decision types that are used, the more simulations are needed to generate the training and test examples. A simulator could usefully be incorporated to decide which rule to apply when the "scheduling knowledge" provides two or more theoretically correct dispatching rules. Finally, a knowledge base refinement module could also be added, which would automatically modify the knowledge base when major changes in the manufacturing system come about.

REFERENCES

[1] M. Montazeri and L. N. W. Wassenhove, "Analysis of scheduling rules for an FMS," *International Journal of Production Research*, vol. 28, pp. 785-802, 1990.

[2] K. E. Stecke and J. Solberg, "Loading and control policies for a flexible manufacturing system," *International Journal of Production Research*, vol. 19, pp. 481-490, 1981.

[3] L. L. Tang, Y. Yih, and C. Y. Liu, "A study on decision rules of a scheduling model in an FMS," *Computers in Industry*, vol. 22, pp. 1-13, 1993.

[4] N. Ishii and J. Talavage, "A transient-based real-time scheduling algorithm in FMS," *International Journal of Production Research*, vol. 29, pp. 2501-2520, 1991.

[5] K. C. Jeong and Y. D. Kim, "A real-time scheduling mechanism for a flexible manufacturing system: Using simulation and dispatching rules," *International Journal of Production Research*, vol. 36, pp. 2609-2626, 1998.

[6] M. H. Kim and Y. D. Kim, "Simulation-based real-time scheduling in a flexible manufacturing system," *Journal of Manufacturing Systems*, vol. 13, pp. 85-93, 1994.

[7] S. Y. D. Wu and R. A. Wysk, "An application of discrete-event simulation to on-line control and scheduling in flexible manufacturing," *International Journal of Production Research*, vol. 27, pp. 1603-1623, 1989.

[8] R. S. Michalski, J. G. Carbonell, and T. M. Mitchell, *Machine Learning, An Artificial Intelligence Approach*, Tioga Press, Palo Alto, CA, 1983.

[9] P. Priore, D. de la Fuente, R. Pino, and J. Puente, "Dynamic scheduling of flexible manufacturing systems using neural networks and inductive learning," *Integrated Manufacturing Systems*, vol. 14, pp. 160-168, 2003.

[10] Y. R. Shiue, R. S. Guh, and T. Y. Tseng, "GA-based learning bias selection mechanism for real-time scheduling systems," *Expert Systems with Applications*, vol. 36, pp. 11451-11460, 2009.

[11] Y. R. Shiue, R. S. Guh, and K. C. Lee, "Study of SOM-based intelligent multi-controller for real-time scheduling," *Applied Soft Computing*, vol. 11, pp. 4569-4580, 2011.

[12] H. Aytug, S. Bhattacharyya, G. J. Koehler, and J. L. Snowdon, "A review of machine learning in scheduling", *IEEE Transactions on Engineering Management*, vol. 41, pp. 165-171, 1994.

[13] P. Priore, D. de la Fuente, A. Gómez, and J. Puente, "A review of machine learning in dynamic scheduling of flexible manufacturing systems," *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, vol. 15, pp. 251-263, 2001.

[14] D. W. Aha, "Tolerating noisy, irrelevant and novel attributes in instance-based learning algorithms," *International Journal of Man-Machine Studies*, vol. 36, pp. 267-287, 1992.

[15] C. Cortes and V. Vapnik, "Support-vector network," *Machine Learning*, vol. 20, pp. 273-297, 1995.

[16] S. Nakasuka and T. Yoshida, "Dynamic scheduling system utilizing machine learning as a knowledge acquisition tool," *International Journal of Production Research*, vol. 30, pp. 411-431, 1992.

[17] C. C. Chen and Y. Yih, "Identifying attributes for knowledge-based development in dynamic scheduling environments," *International Journal of Production Research*, vol. 34, pp. 1739-1755, 1996.

[18] R. M. O. keefe and T. Kasirajan, "Interaction between dispatching and next station selection rules in a dedicated flexible manufacturing system," *International Journal of Production Research*, vol. 30, pp. 1753-1772, 1992.

[19] K. R. Baker, "Sequencing rules and due-date assignments in a job shop," *Management Science*, vol. 30, pp. 1093-1103, 1984.

[20] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison Wesley, Reading, MA, 1989.

[21] C. W. Hsu and C. J. Lin, "A comparison of methods for multi-class support vector machines," *IEEE Transactions on Neural Networks*, vol. 13, pp. 415-425, 2002.

[22] J. R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, San Mateo, CA, 1993.

Paolo Priore graduated in industrial engineering in 1991 at University of Oviedo and gained his Ph. D. from the University of Oviedo in 2001. He is currently professor at the Gijón Polytechnic School of Engineering. His research interests include machine learning applications in production problems, simulation, and manufacturing. He has published a great number of research papers in a number of leading journals such as: Applied Soft Computing, Engineering Applications of Artificial Intelligence, Applied Artificial Intelligence, Computers & Industrial Engineering, Artificial Intelligence for Engineering Design, Analysis and Manufacturing, etc.

Raúl Pino graduated in Industrial Engineering in 1992 at University of Oviedo (Spain), and gained his Ph. D. from the University of Oviedo in 2000. He is currently professor at the Gijón Polytechnic School of Engineering. His current research interests are in the areas of artificial intelligence, forecasting, simulation and manufacturing. He has published a great number of research papers in a number of leading journals.

José Parreño graduated in industrial engineering in 1991 at University of Oviedo and gained his Ph. D. from the University of Oviedo in 2002. He is currently professor at the Gijón Polytechnic School of Engineering. His research interests include forecasting and artificial intelligence applications in production problems, simulation, and manufacturing. He has published research papers in a number of leading journals such as: Journal of Mechanical Engineering, Engineering Applications of Artificial Intelligence and Applied Artificial Intelligence.

Javier Puente is an associate professor of Operations Management at the University of Oviedo (Spain). His key research topics include: Applied Artificial Intelligence, Production Planning and Control, Quality of service and Supply Chain Management. He has published a great number of research papers in a number of leading journals such as: International Journal of Production Economics, Applied Artificial Intelligence, Computers & Industrial Engineering, Applied Soft Computing or Artificial Intelligence in Medicine among others. He also has published in several international conferences: IC-AI, GECCO, POMS, EUSFLAT. He has been usual referee of several international journals and has taken part of the scientific committee in some international conferences.

Borja Ponte is a Ph.D. student at the Department of Business Administration of the University of Oviedo. His Master's Thesis "The Bullwhip Effect in Supply Chains: An Approach based on Artificial Intelligence Techniques", qualified with honors, represents its introduction into the world of research, trying to combine the fields of Logistics and Multiagent Systems. He has presented papers at two international conferences, and he has participated in the work presented in two others.