Dynamic Scheduling of Manufacturing Systems with Machine Learning.

Article *in* International Journal of Foundations of Computer Science · December 2001

DOI: 10.1142/S0129054101000849 · Source: DBLP

David de la Fuente
University of Oviedo
164 PUBLICATIONS **676** CITATIONS
SEE PROFILE

Alberto Gomez
University of Oviedo
78 PUBLICATIONS **534** CITATIONS
SEE PROFILE

Javier Puente
University of Oviedo
64 PUBLICATIONS **393** CITATIONS
SEE PROFILE

DYNAMIC SCHEDULING OF MANUFACTURING SYSTEMS WITH MACHINE LEARNING

PAOLO PRIORE, DAVID DE LA FUENTE, ALBERTO GOMEZ and JAVIER PUENTE
Dpto. de Administración de Empresas y Contabilidad, Universidad de Oviedo
Campus de Viesques, Gijón, 33204, España

ABSTRACT

A common way of scheduling jobs dynamically in a manufacturing system is by means of dispatching rules. The drawback of this method is that the performance of these rules depends on the state the system is in at each moment, and no one rule exists that overrules the rest in all the possible states that the system may be in. It would therefore be interesting to use the most appropriate rule at each moment. To achieve this goal, a scheduling approach which uses machine learning is presented in this paper. The methodology proposed in this paper may be divided into five basic steps. Firstly, definition of the appropriate control attributes for identifying the relevant manufacturing patterns. In second place, creation of a set of training examples using different values of the control attributes. Subsequently, acquiring of heuristic rules by means of a machine learning program. Then, using of the previously calculated heuristic rules to select the most appropriate dispatching rules, and finally testing of the performance of the approach. The approach that we propose is applied to a flow shop system and to a classic job shop configuration. The results demonstrate that this approach produces an improvement in the performance of the system when compared to the traditional method of using dispatching rules.

Keywords: Dynamic scheduling, machine learning, dispatching rules, manufacturing systems, discrete simulation

1. Introduction

Traditional approaches to resolving scheduling problems use simulation, analytical models, heuristics, or a combination of the three. In this respect there is much work that uses simulation to analyse the effects of different dispatching rules (see for example, [2,14]). Analytical models can also be applied. However, even the most simplified models are not capable of being optimally solved for real size problem because of the complexity of the scheduling problem.

One of the most commonly applied ways of solving a scheduling problem is by using dispatching rules. Many researchers have tried in the past to evaluate the performance of dispatching rules in manufacturing systems (see for example, [2,14]). The conclusion to be drawn from such studies is that the performance of these rules depends on the criteria that are chosen, on the configuration of the system and on its conditions (loading of the system, due date tightness, etc.). Changing the dispatching rules at the right time would thus be of interest. There are basically two approaches to doing this: a look-ahead simulation approach, and a knowledge-based approach. Using the former, the dispatching rule is determined for each time period by simulating the predetermined set of dispatching rules and selecting the one with the best performance (see for example, [5,15]). Using the latter, which belongs to the field of artificial intelligence, a set of the system's earlier simulations (training examples) are used to determine the best rule for each of the system's possible states. This knowledge is then used to make intelligent decisions in real time (see for example, [9,12]).

At the same time, over the last fifteen years a great amount of work has gone into developing Artificial Intelligence (AI) methods for solving scheduling problems (see for example, [4,6,13]). Machine learning is a paradigm that lies in the field of AI. Machine learning is a rapidly growing research area for studying methods for developing AI systems that are capable of learning [8]. The ability to learn and improve is essential for intelligent systems; however, little work has so far been carried out applying machine learning to scheduling tasks in manufacturing systems. Aytug et al. [1] present a review of machine learning used to solve scheduling problems.

The paper is organised as follows. Section 2 describes the main characteristics of machine learning; section 3 then deals with how machine learning can be applied to solving the scheduling problem; in section 4, an experimental study over a flow shop and a job shop configuration for evaluating the relative merit of this method over the single dispatching rule approach is presented. Finally, we conclude in section 5 with the major contribution of the paper and future research work.

2. Machine Learning

From a computational view point, the problem of learning is to create computable procedures to perform tasks of which only partial descriptions about the way the system is desired to work are known. The machine learning scheme is depicted, in very general terms, in Fig. 1. A particular case of machine learning is when the poorly structured information source which the machine learning algorithms make use of is given by a collection of examples (known as training examples) which can be considered as a sample of the performance from which the system wishes to learn.

A series of paradigms, such as neural networks, genetic algorithms, case-based reasoning and so on, can be discerned within the field of machine learning. The particular paradigm used in this paper is known as inductive learning, where the problem-solving model is a set of rules generated from the training examples. In this particular case, the examples are described by a series of attributes, amongst which there is a special attribute known as the class. The aim of the problem-solving model is to try to learn to classify or ascertain the class of new cases similar to those which make up the set of training example, but about which all the attributes are known except the class.

Amongst the different existing ways of representing this knowledge, the most usual are rules and trees, the former having the following form:

$$\text{IF} \quad (b_{i1} \geq a_1 \geq c_{i1}) \quad \text{AND.....} \quad (b_{in} \geq a_n \geq c_{in}) \quad \text{THEN} \quad C_i$$

where a_j represents the *j-th* attribute, b_{ij} and c_{ij} define the range for a_j, and C_i denotes the class. At the same time, the branches of the trees are used to represent the different value ranges of the attributes, and the terminal nodes (leaves) indicate the class to which the new example to be classified presumably belongs.

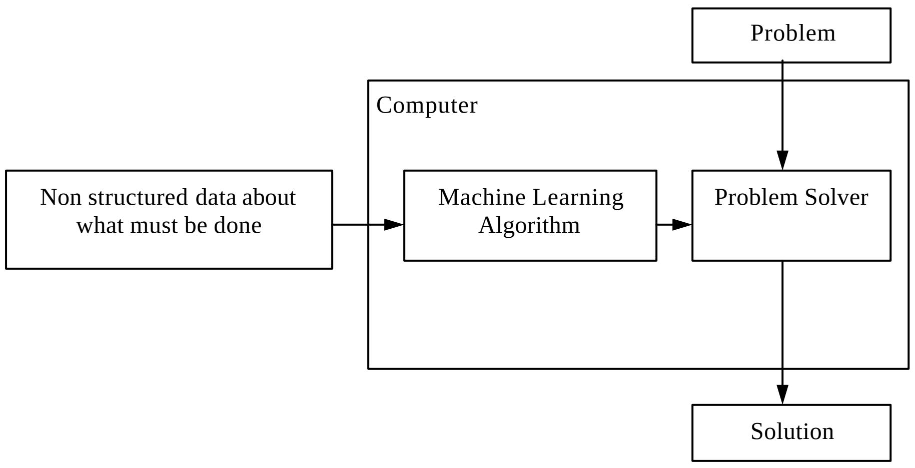
Problem
Computer
Non structured data about what must be done
Machine Learning Algorithm
Problem Solver
Solution

Fig. 1. Machine learning general scheme.

3. Machine learning-based scheduling

One of the most common methods applied to solving dynamic scheduling problems is to use dispatching rules. Many researchers have attempted to evaluate the performance of dispatching rules in manufacturing systems, and the general conclusion to be drawn from such research is that performance of the rules depends on the criterion that is chosen, on the system's configuration and on its conditions. It would thus be of interest to be able to change the dispatching rules at the appropriate time. To do so, this paper puts forward a methodology that can be broken down into five steps, shown in Fig. 2:

- Definition of the appropriate attributes for identifying the relevant manufacturing patterns. Obviously, it is not possible to take all possible attributes into account, so the most significant ones must be chosen. The selected attributes will be called control attributes.

- Creation of a set of training examples using different values of the control attributes. These values must be the most common ones in the manufacturing system studied. The class of each training example will be obtained from the dispatching rule that leads to best performance. In order to do this, a simulation model must be constructed of the system under study and for each set of values of the control attributes (training example), the performance of the system must be tested against the different dispatching rules that are intended to be used.
- Acquiring of heuristic rules by means of an inductive learning program. Each heuristic rule should look like the following:

  IF $(b_{i1} \geq a_1 \geq c_{i1})$ AND..... $(b_{in} \geq a_n \geq c_{in})$ THEN Rule Dispatching

  where a_j is the *j-th* control attribute.
- Use of the previously calculated heuristic rules to select the most appropriate dispatching rule depending on the set of values that the control attributes present at each moment.
- Comparison of the performance of the manufacturing system using the heuristic rules obtained from the inductive learning program with the single dispatching rule which best performs overall. Should the latter method produce an improved system performance, then go back to the second step.

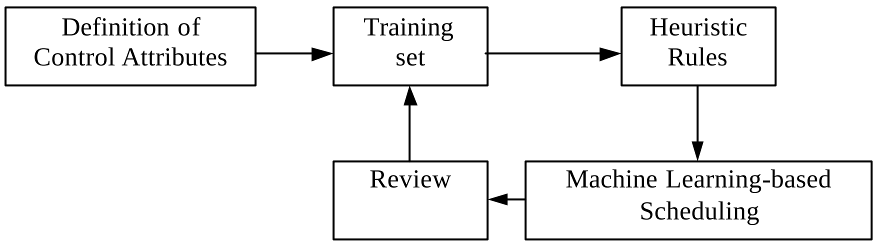
Definition of Control Attributes
Training set
Heuristic Rules
Review
Machine Learning-based Scheduling

Fig. 2. Machine learning-based Scheduling.

4. Experimental study

The proposed approach is applied to two configurations. The first of them is a flow shop system with two work-centres. The second is a classic job shop configuration with four work-centres.

4.1. The flow shop system

Most papers up to date make use of two types of control attributes to identify the relevant manufacturing patterns. On the one hand, some authors have used global attributes such as overall system utilisation, variability in machine workload, etc. (see for example, [12]). On the other hand, local information such as minimum slack time or mean slack time of candidate jobs for a machine is made use of in systems like those mentioned by [3,9]. The problem that global attributes pose is that sometimes it is difficult to obtain the same values with different dispatching rules in a training case. In contrast, with local attributes the problem is that it is difficult to measure the system's characteristics at the individual machine level, and hence to generate training examples [7].

Therefore, this paper will use a set of global control attributes that is different from the ones that are usually applied. The advantage of these attributes is that, being system input parameters, they have the same value for each of the rules, which distinguishes them from the ones most authors use. The control attributes selected for generating the necessary training examples for the inductive learning algorithm to be able to obtain heuristic rules are the following: mean arrival rate of jobs to the system (l); mean process time at work-centre *i* (*PTWCi*), and *F* (Flow Allowance Factor), which allows due dates to be calculated. The set of dispatching rules applied are: SPT (Shortest Processing Time); LPT (Longest Processing Time); EDD (Earliest Due Date), and FCFS (First Come First Served).

The due date of job *i* (d_i) and of operation *j* of job *i* (d_{ij}) is calculated, following [2], by the following expressions:

$$\begin{aligned} d_i &= t + F * p_i \\ d_{ij} &= d_{i,j-1} + p_{ij} * F \end{aligned} \tag{1}$$

where *F* is the flow allowance factor which measures due date tightness; p_{ij} is the processing time of operation *j* of job *i*; p_i is the total processing time of the job *i*, and *t* the moment the scheduling decision is taken.

The system performance criterion, used in this paper, is mean tardiness, as it is the criterion most commonly used in all manufacturing systems. The mean tardiness is defined as follows:

$$\frac{\sum T_i}{N} \tag{2}$$

where, $T_i = max\ \{0, L_i\}$; *N* is the number of complete jobs, and L_i is the difference between the end of the job and the agreed due date d_i.

The training examples needed for the learning step are obtained through simulation, using the Witness program. It is assumed that the jobs arrive at the system following Poisson's distribution and that the processing times conform to an exponential distribution. The arrival rate varies in such a way that the overall use of the system fluctuates between 60 and 95%. At the same time, the value of factor *F* ranges between 2 and 6. In all, 1000 different combinations of the four control attributes are randomly generated. In each attribute combination, the mean tardiness values resulting from employing each one of the four dispatching rules in isolation are determined, and the best rule was chosen as the class for that combination, thereby creating a training example.

Table 1. Training stage for the flow shop system.

Number of training cases	Number of rules	Error on training data	Error on test data
100	7	4%	23%
200	9	3.5%	19%
300	11	3%	15%
400	22	2.5%	14%
500	23	4.2%	15%
600	24	3.8%	14%
700	25	5%	15%
800	22	5.2%	15%
900	28	4.6%	14%

The C4.5 learning program was used to obtain the heuristic rules [11]. The learning process for C4.5 consists of a sequence of specialisation steps guided by an information entropy function for evaluating class membership. Different sized training sets are used (up to a maximum value of 900 examples), and the last 100 are used to check the system's forecasting capacity. Table 1 shows the results of the training stage. According to these results, it was decided to choose the obtained rules with four hundred training cases.

Twenty-two heuristic rules were obtained. The rule or class EDD was chosen in eleven of them, while that of SPT was selected in the remaining eleven. The default class was the EDD rule. The set of rules obtained using the C 4.5 program was the following:

Rule 1:IF $\lambda \leq 6.8$ AND PTWC1 > 8 AND F ≤ 2 THEN SPT
Rule 2:IF $\lambda \leq 6$ AND PTWC1 > 6.8 AND F ≤ 2 THEN SPT
Rule 3:IF $\lambda \leq 5.4$ AND F ≤ 3 THEN SPT
Rule 4:IF $\lambda \leq 5.9$ AND PTWC1 > 7.5 AND F ≤ 3 THEN SPT
Rule 5:IF $\lambda \leq 6$ AND PTWC2 > 7.9 AND F ≤ 2 THEN SPT
Rule 6:IF $\lambda \leq 5.3$ AND PTWC1 > 7.6 AND PTWC2 > 6.9 AND F ≤ 5 THEN SPT
Rule 7:IF $\lambda \leq 5.8$ AND PTWC2 > 9.6 THEN SPT
Rule 8:IF $\lambda \leq 5.9$ AND PTWC1 > 7.8 AND PTWC1 ≤ 8.2 AND PTWC2 > 7.5 AND F > 3 AND F ≤ 5 THEN SPT
Rule 9:IF $\lambda > 6$ AND $\lambda \leq 6.2$ AND PTWC1 > 8.6 AND PTWC1 ≤ 9.3 THEN SPT
Rule 10:IF $\lambda \leq 5$ AND PTWC2 > 6.6 THEN SPT

Rule 11:IF $\lambda \leq 5.9$ AND PTWC1 > 6.3 AND PTWC1 ≤ 6.4 AND F > 2 THEN SPT
Rule 12:IF PTWC2 ≤ 6.6 AND F > 3 THEN EDD
Rule 13:IF PTWC1 > 6.4 AND PTWC1 ≤ 7.6 AND PTWC2 ≤ 9.6 AND F > 3 THEN EDD
Rule 14:IF $\lambda > 6$ AND PTWC1 ≤ 8 THEN EDD
Rule 15:IF $\lambda \leq 5.9$ AND PTWC1 > 6.4 AND PTWC2 ≤ 9.6 AND F > 5 THEN EDD
Rule 16:IF $\lambda > 5.3$ AND T ≤ 5.9 AND PTWC1 > 6.4 AND PTWC2 ≤ 7.5 AND F > 3 THEN EDD
Rule 17:IF $\lambda > 5.3$ AND PTWC1 > 9.5 AND PTWC2 ≤ 9.6 AND F > 3 THEN EDD
Rule 18: IF $\lambda > 6.8$ THEN EDD
Rule 19:IF $\lambda > 5.9$ AND F > 2 THEN EDD
Rule 20:IF $\lambda > 5.4$ AND PTWC1 ≤ 7.5 AND F > 2 AND F ≤ 3 THEN EDD
Rule 21: IF PTWC1 ≤ 6.3 THEN EDD
Rule 22 or default rule: EDD

where:
l: mean time between job arrivals
PTWCi: mean processing time in work centre *WCi*
F: flow allowance factor

The default rule or class (EDD) is used in the case when none of the remaining rules is chosen at the time at which they are evaluated.

Subsequently, a series of experiments were carried out in which the values of the control attributes were varied with time so as to observe how the system performed using the heuristic rules obtained from C4.5 and to compare this with the performance when using a single dispatching rule constantly. In the latter case, only the SPT and EDD rules were used since they were the ones that generated the least mean tardiness in the majority of the training cases studied. It was observed that on certain occasions, the heuristic rules did not perform as expected, and were bettered by the SPT or the EDD rule employed individually.

The explanation of this phenomenon lies in the system that is fed with heuristic rules reacting precipitously to changes in control attributes that were only transitory, thus giving rise to poor performance. Hence, a mechanism to dampen these transitory scenarios in the control attributes was designed. It consisted of weighting each of the dispatching rules that are chosen by the heuristic rules. Only when this weighting exceeds a certain limiting value (*WL*) will the dispatching rule be chosen. The calculation of the weighting for each rule (*WR*) is carried out by means of the following expression [10]:

$$WR_i = \sum_{j=1}^{n} \ddot{a}_{ij} \acute{a}^{n-j} \tag{3}$$

where WR_i is the weighting of the rule i; d_j is a parameter which is equal to 1 when dispatching rule i is selected at time j and 0 otherwise; a is a number less than 1 used to give a lesser weighting to the rules selected in the past, and n is the total number of changes that have taken place in the control attributes since the dispatching rule was last modified.

Table 2 presents an example of the application of this mechanism for assigning weighting (assuming $WL=3$ and $\alpha=0.8$) in a scenario in which 9 changes are produced in the control attribute values. In the first time interval, even though the heuristic rules choose SPT rule, as its weighting is less than 3 (*WL*), this dispatching rule is not to be chosen immediately; being kept the rule chosen in the preceding interval. The variation in the weightings for the SPT and EDD rules may be observed, the first being chosen only when its weighting exceeds the *WL* value (in the seventh interval). From this moment onwards, the SPT rule is maintained until the weighting of the EDD rule exceeds the *WL* value.

Table 2. Mechanism for assigning weighting.

No. of changes in the control attributes	Dispatching rule selected by the heuristic rules	WR_{SPT}	WR_{EDD}	Dispatching rule selected by the proposed mechanism
1	SPT	1	0	Preceding
2	SPT	1.8	0	Preceding
3	EDD	1.44	1	Preceding
4	EDD	1.152	1.8	Preceding
5	SPT	1.922	1.44	Preceding
6	SPT	2.537	1.152	Preceding
7	SPT	3.030	0.921	SPT
8	EDD	0	1	SPT
9	EDD	0	1.8	SPT

Subsequently, 100 experiments were carried out using different *WL* values so as to ascertain the most appropriate *WL* value to use. At the same time, the frequency of changes in the system's control attributes was modified, varying the number of parts manufactured with constant attribute values (*NP*). Values for *NP* used in the different experiments ranged between 50 and 1400. It was observed that the optimum value of *WL* changed as a function of *NP*; however, 5 (with $\alpha = 0.9$) was chosen as a value of *WL* in a large number of experiments. These showed that for 67% of cases, the proposed approach was superior to the best rule used individually, and in 29% of cases its performance was the same. The proposed approach was inferior in only 4% of cases. The results lead to the conclusion that by applying heuristic rules there is an improvement in mean tardiness of 10% and 13% respectively when compared to using either the SPT or EDD rules constantly.

4.2. The job shop system

The manufacturing system under consideration is a classic job shop configuration with four work-centres. This has already been used by other authors studying scheduling problems (see for example, [2]). The set of rules used in this configuration are: SPT (Shortest Processing Time); EDD (Earliest Due Date); MDD (Modified Job Due Date), and MOD (Modified Operation Due Date). The control attributes are the following: Mean arrival rate of jobs to the system (***l***); percentage of operations assigned to machine *i* (PO_i), and *F*. The criterion used to measure system performance is once again mean tardiness. It is assumed that the jobs arrive at the system following Poisson's distribution and that the processing times conform to an exponential distribution with a mean of 1. The actual number of operations of a job is a random variable, equally distributed among the integers from 1 to 4. The probability of assigning an operation to a machine depends on the parameters PO_i . The arrival rate varies in such a way that the overall use of the system fluctuates between 60 and 95%. At the same time, the value of factor *F* ranges between 2 and 6.

Table 3. Training stage for the job shop system.

Number of training cases	Number of rules	Error on training data	Error on test data
100	9	15%	29%
200	14	8%	25%
300	19	7%	22%
400	17	11%	22%
500	22	11%	18%
600	28	11%	16%
700	26	12%	21%
800	24	11%	17%
900	34	10%	19%

In all, 1000 different combinations of the six control attributes are randomly generated, and the last 100 serve as test examples. Table 3 shows the results of the training stage. According to these results, it was decided to choose the obtained rules with six hundred training cases. Twenty-eight heuristic rules were obtained, seventy and eleven of them corresponding to the MDD and MOD rule or class, respectively. The default class, or rule, is MOD, which will apply when none of the remaining twenty-seven heuristic rules is chosen when evaluation is made. As SPT and EDD rules were chosen as best in few training cases (and even when they were chosen tardiness differences compared to other rules were minimal), it was decided to exclude them, as test error was thus lower. The set of rules obtained using the C4.5 program was the following:

Rule 1: IF F = 6 AND PO3 > 0.26 THEN MDD

Rule 2: IF F = 6 AND λ > 0.77 AND λ <= 0.95 THEN MDD
Rule 3: IF F = 4 AND PO3 > 0.25 AND PO3 <= 0.27 AND λ > 0.9 AND λ <= 1.04 THEN MDD
Rule 4: IF F = 5 AND PO2 <= 0.25 AND PO3 <= 0.25 AND λ > 0.78 AND λ <= 1.08 THEN MDD
Rule 5: IF PO3 > 0.25 AND PO4 > 0.25 AND λ <= 0.82 THEN MDD
Rule 6: IF F = 5 AND PO4 <= 0.23 THEN MDD
Rule 7: IF F = 6 AND PO1 > 0.26 AND λ > 0.77 THEN MDD
Rule 8: IF PO3 > 0.24 AND λ <= 0.68 THEN MDD
Rule 9: IF F = 5 AND PO2 <= 0.27 AND PO3 <= 0.25 AND λ > 0.9 AND λ <= 1.08 THEN MDD
Rule 10: IF F = 3 AND PO2 > 0.23 AND PO3 > 0.26 AND λ <= 0.84 THEN MDD
Rule 11: IF F = 5 AND PO3 > 0.25 AND λ <= 1.07 THEN MDD
Rule 12: IF F = 4 AND λ > 1.01 AND λ <= 1.05 THEN MDD
Rule 13 IF F = 4 AND PO1 > 0.24 AND PO1 <= 0.25 AND PO2 <= 0.26 AND PO4 <= 0.26 AND λ > 0.75 AND λ <= 0.99 THEN MDD
Rule 14: IF F = 1 AND PO3 > 0.25 AND λ <= 0.71 THEN MDD
Rule 15: IF F = 3 AND λ <= 0.66 THEN MDD
Rule 16: IF F = 6 AND λ > 1.09 THEN MDD
Rule 17: IF F = 3 AND PO3 <= 0.24 AND λ > 1 THEN MDD
Rule 18: IF F = 2 AND PO2 > 0.23 AND PO3 <= 0.25 THEN MOD
Rule 19: IF F = 2 AND λ > 0.7 AND λ <= 1.02 THEN MOD
Rule 20: IF F = 1 AND λ > 0.71 THEN MOD
Rule 21: IF PO3 <= 0.25 AND PO4 > 0.23 AND λ > 0.67 AND λ <= 0.78 THEN MOD
Rule 22 IF F = 3 AND PO3 > 0.25 AND PO3 <= 0.26 AND PO4 > 0.23 AND PO4 <= 0.25 THEN MOD
Rule 23: IF F = 3 AND PO2 > 0.23 AND λ > 0.84 AND λ <= 0.9 THEN MOD
Rule 24: IF F = 4 AND PO3 <= 0.25 AND λ > 0.69 AND λ <= 0.99 THEN MOD
Rule 25: IF F = 3 AND PO3 <= 0.25 AND λ > 0.66 THEN MOD
Rule 26: IF PO3 > 0.25 AND PO3 <= 0.27 AND PO4 <= 0.25 AND λ > 1.04 THEN MOD
Rule 27: IF F = 6 AND PO1 <= 0.26 AND PO3 <= 0.25 AND λ > 0.95 AND λ <= 1.09 THEN MOD
Rule 28 or default rule: MOD

where:
l: mean time between job arrivals
POi: percentage of operations assigned to machine i
F: flow allowance factor

Subsequently, 100 experiments using different *WL* and *NP* values were carried out using the weighting mechanism described in the previous subsection. They demonstrated, as did the earlier system, that the optimum value of *WL* changes as a function of *NP*; however, 5 was once again selected in a large number of experiments. They also demonstrated that for 60% of cases, the proposed approach was superior to the best rule used individually, and in 33% of cases performance was the same. The proposed approach was inferior in only 7% of cases. Globally, improvement using this approach was 8% with respect to the MOD rule, the dispatching rule that best performs when used individually.

5. Conclusions

In this paper, a new approach to scheduling jobs using machine learning has been proposed. In general terms, it has been proven that the performance of this method is an improvement on that of using a single dispatching rule constantly. One of the drawbacks, however, of using this technique is the need to execute a large number of simulations in order to generate the training examples. Nevertheless, these simulations need only be carried out once.

Our research effort is currently aimed at studying the performance of this approach in job shop configurations with a greater number of machines, as well as in Flexible Manufacturing Systems, for which new attributes must be considered which characterise said systems. At the same time, it would be of interest to study how this approach performs in the case of machine breakdowns occurring. Analysing new mechanisms to assign weighting to a system's transitory state would also be an interesting field of study.

6. References

1. H. Aytug, S. Bhattacharyya, G. J. Koehler and J. L. Snowdon, "A review of machine learning in scheduling," *IEEE Transactions on Engineering Management* **41** (1994) 165-171.
2. K. R. Baker, "Sequencing rules and due-date assignments in a job shop," *Management Science* **30** (1984) 1093-1104.
3. C. Chiu and Y. Yih, "A learning-based methodology for dynamic scheduling in distributed manufacturing systems," *International Journal of Production Research* **33** (1995) 3217-3232.
4. M. S. Fox and S. F. Smith, "ISIS: A knowledge-based system for factory scheduling," *Expert Systems* **1** (1984) 25-49.
5. N. Ishii and J. Talavage, "A transient-based real-time scheduling algorithm in FMS," *International Journal of Production Research* **29** (1991) 2501-2520.
6. A. Kusiak and M. Chen, "Expert systems for planning and scheduling manufacturing systems," *European Journal of Operational Research* **34** (1988) 113-130.

7. C.-Y. Lee, S. Piramuthu and Y.-K. Tsai, "Job shop scheduling with a genetic algorithm and machine learning," *International Journal of Production Research* **35** (1997) 1171-1191.
8. R. S. Michalski, "A theory and methodology of inductive learning," in Machine Learning, eds. R. S. Michalski, J. Carbonell and T. Mitchell (Tioga, Palo Alto, CA, 1983).
9. S. Nakasuka and T. Yoshida, "Dynamic scheduling system utilizing machine learning as a knowledge acquisition tool," *International Journal of Production Research* **30** (1992) 411-431.
10. S. Piramuthu, N. Raman and M. J. Shaw, "Learning-based scheduling in a flexible manufacturing flow line," *IEEE Transactions on Engineering Management* **41** (1994) 172-182.
11. J. R. Quinlan, *C4.5: Programs for Machine Learning* (Morgan Kaufmann Publishers, San Mateo, 1993).
12. M. J. Shaw, S. Park and N. Raman, "Intelligent scheduling with machine learning capabilities: The induction of scheduling knowledge," *IIE Transactions* **24** (1992) 156-168.
13. M. J. Shaw and A. Whinston, "An artificial intelligence approach to scheduling in flexible manufacturing systems," *IIE Transactions* **21** (1989) 170-183.
14. K. E. Stecke and J. J. Solberg, "Loading and control policies for a flexible manufacturing system," *International Journal of Production Research* **19** (1981) 481-490.
15. S. 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* **27** (1989) 1603-1623.

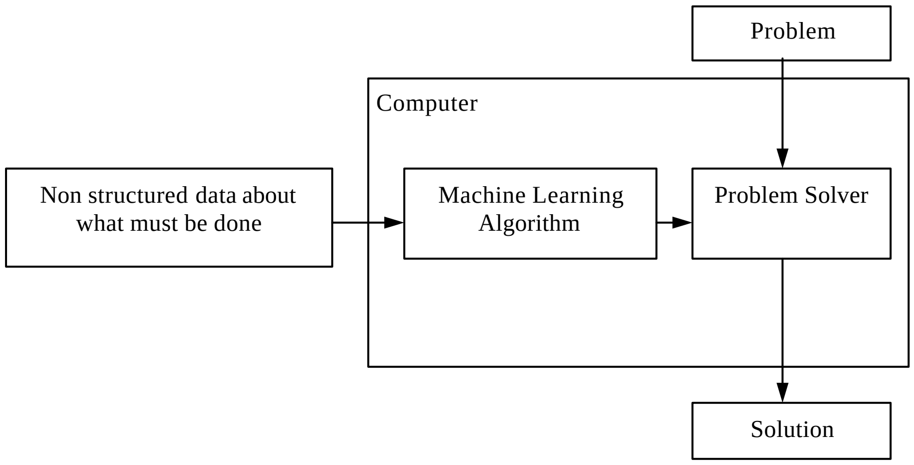
Problem
Computer
Non structured data about what must be done
Machine Learning Algorithm
Problem Solver
Solution

Fig. 1. Machine learning general scheme.

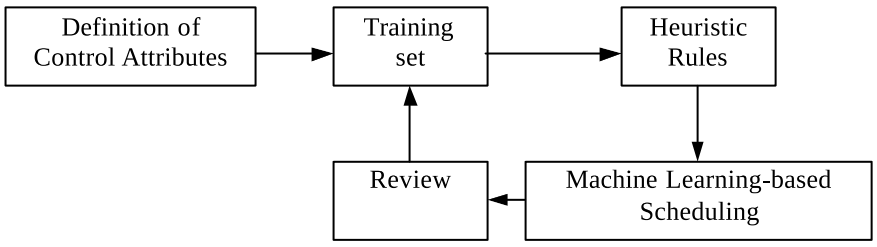
Definition of Control Attributes
Training set
Heuristic Rules
Review
Machine Learning-based Scheduling

Fig. 2. Machine learning-based Scheduling.

Table 1. Training stage for the flow shop system.

Number of training cases	Number of rules	Error on training data	Error on test data
100	7	4%	23%
200	9	3.5%	19%
300	11	3%	15%
400	22	2.5%	14%
500	23	4.2%	15%
600	24	3.8%	14%
700	25	5%	15%
800	22	5.2%	15%
900	28	4.6%	14%

Table 2. Mechanism for assigning weighting.

No. of changes in the control attributes	Dispatching rule selected by the heuristic rules	WR_{SPT}	WR_{EDD}	Dispatching rule selected by the proposed mechanism
1	SPT	1	0	Preceding
2	SPT	1.8	0	Preceding
3	EDD	1.44	1	Preceding
4	EDD	1.152	1.8	Preceding
5	SPT	1.922	1.44	Preceding
6	SPT	2.537	1.152	Preceding
7	SPT	3.030	0.921	SPT
8	EDD	0	1	SPT
9	EDD	0	1.8	SPT

Table 3. Training stage for the job shop system.

Number of training cases	Number of rules	Error on training data	Error on test data
100	9	15%	29%
200	14	8%	25%
300	19	7%	22%
400	17	11%	22%
500	22	11%	18%
600	28	11%	16%
700	26	12%	21%
800	24	11%	17%
900	34	10%	19%