Real-time Scheduling of Flexible Manufacturing Systems using Support Vector Machines and Neural Networks

P. Priore¹, R. Pino¹, J. Parreño¹, J. Lozano¹ and M. Monterrey¹
¹EPI de Gijón, Campus de Viesques, 33203 Gijón, Asturias, Spain

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 backpropagation neural networks. 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.

Keywords: Scheduling, Neural Networks, SVMs, FMS, Simulation

1 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, [12], [21], [22]). 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 (utilisation 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, [7], [8], [9], [23]). 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 [11] 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, [16], [19], [20]). Aytug [1] and Priore [15] 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 neural networks. 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 organised 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.

2 Neural Networks and Support Vector Machines

“Backpropagation neural networks”, or multilayer perceptron [18], which will be applied in this work, figure amongst those networks that are most well-know and most widely used as pattern classifiers or function approximators ([5], [10]). The backpropagation training algorithm is used in this type of neural networks. This algorithm calculates the most adequate connection weights and thresholds so that the difference between the network output and the desired one is minimised.

Support vector machines [4] were originally designed for binary classification. Let \((x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\) be a group of data belonging to Class 1 or Class 2, where \(x_i \in \mathbb{R}^N\) and the associated labels be \(y_i=1\) for Class 1 and -1 for Class 2 \((i=1, \ldots, n)\). The formulation of SVMs is as follows:
subject to the constraints:

\[ y_i (w^T \phi(x_i) + b) \geq 1 - \xi_i \quad i = 1, \ldots, n \]

\[ \xi_i \geq 0 \quad i = 1, \ldots, n \]  

(1)

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

\[ k(x_i, x_j) = \phi(x_i)^T \phi(x_j) \]

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

Minimizing \( \frac{1}{2} w^T w \) implies that SVMs tries to maximise

\[ \frac{1}{2} \|w\|^2 \]

which represents the margin of separation between both classes. The data that satisfy the equality in Eq. (1) are called support vectors. Moreover, by adding a set of non-negative Lagrange multipliers \( \alpha_i \) and \( \beta_i \) to generate the Lagrangian, the upper-mentioned constrained optimization problem can be worked out with the dual form shown below:

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

subject to the constraints:

\[ \sum_{i=1}^{n} \alpha_i y_i = 0 \]

\[ 0 \leq \alpha_i \leq C \quad i = 1, \ldots, n \]

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

\[ y = \text{sign} \left( \sum_{SVs} \alpha_i y_i k(x_i, x) + b \right) \]

3 Scheduling using Neural Networks 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 [13]:

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 neural networks, 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 [3].

Figure 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.

![Figure 1. General overview of a knowledge-based scheduling system.](image-url)
4 Experimental Study

4.1 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, [14]). 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 Utilised Station).

4.2 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 [2]; NAMO: number of alternative machines for an operation; MU: mean utilisation of the manufacturing system; U$_i$: utilisation of machine $i$; WIP: mean number of parts in the system; RBM: ratio of the utilisation of the bottleneck machine to the mean utilisation of the manufacturing system; RSDU: ratio of the standard deviation of individual machine utilisations 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 $P_O$, (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, and 100 of these were used as test examples. 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.

4.3 The application of neural networks

Backpropagation neural networks are particularly used to solve classification problems such as the one being considered in this work. Table I provides a summary of the results obtained using different-sized sets of training examples for the criteria of mean tardiness and mean flow time. Generally, it can be seen that as the number of training examples increases, test example error (examples that have not previously been dealt with) decreases considerably. Table I also shows that test error fluctuates between 16% and 12% upwards of 400 examples for the criterion of mean tardiness. Furthermore, for the criterion of mean flow time, test error is observed to oscillate between 7% and 4% upwards of 500 examples. Errors for this latter criterion are lower due to there being five dispatching rule combinations (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 neural networks for the criteria of mean tardiness (MT) and mean flow time (MFT).

<table>
<thead>
<tr>
<th>Number of examples</th>
<th>Test error (MT)</th>
<th>Test error (MFT)</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>21%</td>
<td>9%</td>
</tr>
<tr>
<td>300</td>
<td>17%</td>
<td>10%</td>
</tr>
<tr>
<td>400</td>
<td>16%</td>
<td>10%</td>
</tr>
<tr>
<td>500</td>
<td>15%</td>
<td>7%</td>
</tr>
<tr>
<td>600</td>
<td>13%</td>
<td>5%</td>
</tr>
<tr>
<td>700</td>
<td>14%</td>
<td>6%</td>
</tr>
<tr>
<td>800</td>
<td>16%</td>
<td>6%</td>
</tr>
<tr>
<td>900</td>
<td>15%</td>
<td>5%</td>
</tr>
<tr>
<td>1000</td>
<td>15%</td>
<td>4%</td>
</tr>
<tr>
<td>1100</td>
<td>12%</td>
<td>4%</td>
</tr>
</tbody>
</table>

The ideal configuration of the neural network used for the criterion of mean tardiness was found to have 11 input nodes (one for each control attribute), 16 nodes in the hidden layer, and 12 nodes in the output layer (one for each dispatching rule combination). Similarly, the ideal configuration of the neural network employed for the criterion of mean flow time was found to have 11 input nodes, 16 nodes in the hidden layer, and 5 nodes in the output layer.

4.4 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 [6]. 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 $\sigma=10$. 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.

4.5 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 [17].
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).

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 training 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 training set size.

Table II. Test error using SVMs for the criteria of mean tardiness (MT) and mean flow time (MFT).

<table>
<thead>
<tr>
<th>Number of examples</th>
<th>Test error (MT)</th>
<th>Test error (MFT)</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>16%</td>
<td>6%</td>
</tr>
<tr>
<td>300</td>
<td>15%</td>
<td>5%</td>
</tr>
<tr>
<td>400</td>
<td>15%</td>
<td>2%</td>
</tr>
<tr>
<td>500</td>
<td>16%</td>
<td>2%</td>
</tr>
<tr>
<td>600</td>
<td>12%</td>
<td>2%</td>
</tr>
<tr>
<td>700</td>
<td>11%</td>
<td>1%</td>
</tr>
<tr>
<td>800</td>
<td>11%</td>
<td>1%</td>
</tr>
<tr>
<td>900</td>
<td>11%</td>
<td>1%</td>
</tr>
<tr>
<td>1000</td>
<td>10%</td>
<td>1%</td>
</tr>
<tr>
<td>1100</td>
<td>10%</td>
<td>1%</td>
</tr>
</tbody>
</table>

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).

<table>
<thead>
<tr>
<th>Number of examples</th>
<th>Test error (MT)</th>
<th>Test error (MFT)</th>
</tr>
</thead>
<tbody>
<tr>
<td>600</td>
<td>10%</td>
<td>1%</td>
</tr>
<tr>
<td>700</td>
<td>9%</td>
<td>0%</td>
</tr>
<tr>
<td>800</td>
<td>9%</td>
<td>0%</td>
</tr>
<tr>
<td>900</td>
<td>9%</td>
<td>0%</td>
</tr>
<tr>
<td>1000</td>
<td>8%</td>
<td>0%</td>
</tr>
<tr>
<td>1100</td>
<td>8%</td>
<td>0%</td>
</tr>
</tbody>
</table>

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).

Test error using backpropagation neural networks 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 backpropagation neural network gives a 12% test error, compared to the 8% error of the SVMs. Furthermore, for the criterion of mean flow time, the SVMs are seen to give zero test error, whereas the backpropagation neural network generates a bigger test error (2%). Finally, mention should be made of the fact that the ideal neural network for the criterion of mean tardiness has 18, 15 and 12 neurons in the input,
hidden and output layers respectively, whilst for the criterion of mean flow time the ideal neural network has 15, 10 and 5 neurons.

**Table IV.** Test error using neural networks and the generator module of new control attributes for the criteria of mean tardiness (MT) and mean flow time (MFT).

<table>
<thead>
<tr>
<th>Number of examples</th>
<th>Test error (MT)</th>
<th>Test error (MFT)</th>
</tr>
</thead>
<tbody>
<tr>
<td>600</td>
<td>15%</td>
<td>4%</td>
</tr>
<tr>
<td>700</td>
<td>14%</td>
<td>4%</td>
</tr>
<tr>
<td>800</td>
<td>12%</td>
<td>4%</td>
</tr>
<tr>
<td>900</td>
<td>14%</td>
<td>2%</td>
</tr>
<tr>
<td>1000</td>
<td>12%</td>
<td>2%</td>
</tr>
<tr>
<td>1100</td>
<td>12%</td>
<td>2%</td>
</tr>
</tbody>
</table>

4.6 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, [8], [9], [23]). In view of the results in the previous section, 1000 examples were used for both performance criteria. Five independent replicas of 100,000 time units were carried out.

**Table V.** Mean tardiness (MT) and mean flow time (MFT) for the proposed strategies.

<table>
<thead>
<tr>
<th>Strategy used</th>
<th>MT</th>
<th>MFT</th>
<th>Strategy used</th>
<th>MT</th>
<th>MFT</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPT+SPT</td>
<td>5.4916</td>
<td>2.4041</td>
<td>MDD+NINVQ</td>
<td>1.1450</td>
<td>1.2495</td>
</tr>
<tr>
<td>SPT+NINVQ</td>
<td>1.2220</td>
<td>1.0438</td>
<td>MDD+WINQ</td>
<td>1.1566</td>
<td>1.2546</td>
</tr>
<tr>
<td>SPT+WINQ</td>
<td>1.2011</td>
<td>1.0415</td>
<td>MDD+LUS</td>
<td>2.5326</td>
<td>1.8537</td>
</tr>
<tr>
<td>SPT+LUS</td>
<td>2.5920</td>
<td>1.5187</td>
<td>SRT+WINQ</td>
<td>1.4089</td>
<td>1.1419</td>
</tr>
<tr>
<td>EDD+SRT+WINQ</td>
<td>4.7207</td>
<td>2.6106</td>
<td>SRT+LUS</td>
<td>1.0427</td>
<td>1.7094</td>
</tr>
<tr>
<td>EDD+NINVQ</td>
<td>1.6885</td>
<td>1.3948</td>
<td>SVMs</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>EDD+WINQ</td>
<td>1.6885</td>
<td>1.3948</td>
<td>SVMs</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>EDD+LUS</td>
<td>3.2958</td>
<td>2.0493</td>
<td>Neural Network</td>
<td>1.0298</td>
<td>1.0055</td>
</tr>
</tbody>
</table>

Moreover, the SVMs give better results than the neural network for the criterion of mean flow time. Table V also shows that the combinations SPT+NINVQ 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 SVMs alternative by between 4.15% and 4.38%. Finally, the SVMs-based 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.

5 Conclusions

An approach for scheduling using SVMs and neural networks 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 SVMs-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.

6 References


