ADAPTIVE SCHEDULING AND CONTROL USING ARTIFICIAL NEURAL NETWORKS AND EXPERT SYSTEMS FOR A HIERARCHICAL/DISTRIBUTED FMS ARCHITECTURE

Luis Carlos Rabelo and Sema Alptekin

Computer Integrated Manufacturing Laboratory
Engineering Management Department
University of Missouri-Rolla
Rolla, Missouri 65401

ABSTRACT

An adaptive expert scheduler was developed that learns by itself and adapts to the dynamic FMS environment. This hybrid system uses a symbiotic architecture composed of expert systems (ESs) and artificial neural networks (ANNs) and provides a learning scheme guided by past experience. The artificial neural networks recognize patterns in the tasks to be solved in order to select the best scheduling rule according to different criteria. The expert systems, on the other hand, drive the inference strategy and interpret the constraints and restrictions imposed by the upper levels of the control hierarchy of the flexible manufacturing system. The level of self-organization achieved provides a system with a higher probability of success than traditional approaches.

INTRODUCTION

Flexible manufacturing systems (FMSs) are automated manufacturing systems consisting of computer numerical control (CNC) machine tools, material handling devices, automated inspection stations, in-process storage areas, and a computational (hardware-software processing-communications) scheme to provide database handling, supervisory, and monitoring functions. Flexible manufacturing systems are characterized by high flexibility and complexity. Consequently, the scheduling of jobs, machines, and other resources in an FMS to achieve the production goals assigned, taking into consideration their decision making time frame, is a difficult task (4,6).

As an approach to solve the FMS scheduling problem, several researchers have stressed the need for real-time scheduling systems designed with an augmented level of intelligence, using Artificial Intelligence (AI) including O'Grady and Lee (9), Gross (4), Alptekin and Rabelo (1), Park et al. (10), and Kusiak (7). However, the most common use of AI in FMS scheduling is the development of expert systems which emulate intelligent behavior. Nevertheless, the concept of FMS scheduling incorporates several AI disciplines such as feature extraction, data interpretation, distributed decision-making, and optimization. Therefore, it is appropriate to integrate several AI technologies to form systems that can meet the requirements of such an environment.

In this paper, we present the design and implementation of an intelligent scheduling system for FMS (ISS/FMS) that utilizes AI technologies so that expected performance levels can be accomplished. ISS/FMS utilizes ESs, distributed knowledge bases, and artificial neural networks (ANNs) in order to find a good solution for the FMS scheduling/rescheduling problem.

ANNs are used as a method for predicting the behavior of the dispatching rules available in ISS/FMS rather than utilizing rules based on statistical models (15). Their ability to learn from examples provides a self-acting strategy to the knowledge acquisition process and therefore a direct contribution to support self-organization schemes. Expert Systems are utilized to interpret the goals and commands from the different elements of the hierarchical FMS architecture, interact with the user, monitor the performance and develop retraining strategies to enhance the artificial neural network structures, and to implement sophisticated scheduling procedures.

BACKGROUND ON ARTIFICIAL NEURAL NETWORKS

Artificial neural networks (ANNs) are information/processing systems whose development has been motivated by the goal of reproducing the cognitive processes and organizational models of neurobiological systems. By virtue of their computational structure, ANNs feature attractive characteristics such as graceful degradation, robust recall with fragmented and noisy data, speed inherent to parallel distributed architectures, generalization, and the most interesting one: learning. In this section, basic concepts about ANNs, derivation and an example of the use of the learning scheme utilized in this research will be presented.

General Description

Artificial neural networks are information/processing systems composed of a large number of interconnected processing elements (PEs). The characteristics of an ANN is a product of the network paradigm. The network paradigm is given by the network architecture and the neuro-dynamics utilized.

0-8186-1966-X/90/0000/0538/$01.00 © 1990 IEEE

Network Architecture. The network architecture defines the arrangement of processing elements and how they are interconnected. This establishes which PEs are interconnected--inputs from and outputs to PEs, the groups or layers of PEs, and how the information flows in the network. For example, a sequential network will feedback its output to the input units of the network, in a feed-forward network the information will flow strictly from the input to the output (See Figures 1 and 2).

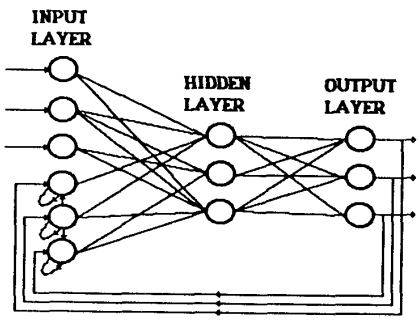

Figure 1. A sequential network

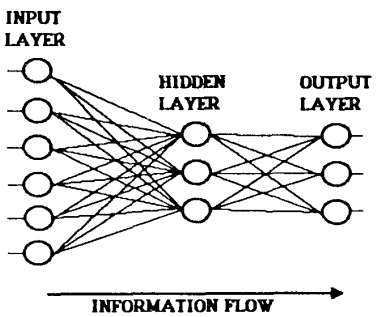

Figure 2. A feedforward network

Neuro-Dynamics. The PEs (also called neurons) have a number of inputs which are modified by adaptive coefficients (weights) and generate an output signal (See Figure 3). Neuro-dynamics specifies how the inputs to the PE are going to be combined together, and what type of function or relationship is going to be used to develop the output, and how the weights are going to be modified.

The inputs to the PE are weighted and often are combined together using the summation function. This is also called "internal activation". This internal activation is utilized to generate the output of the neuron using a continuous or noncontinuous transfer function.

The learning mechanism which handles modifications to the weights and any other organization of the network can be classified under supervised learning, unsupervised

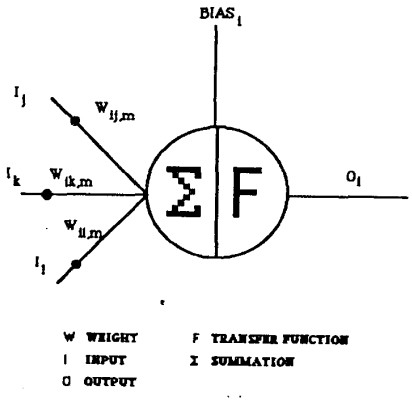

Figure 3. A processing element

learning, or self-supervised learning (14). Supervised learning takes place when the network is trained using pairs of input and desired outputs. In unsupervised learning, inputs are entered and the network is able to self-organize its own categories. Self-supervised learning adds feedback to unsupervised learning to correct errors in the pattern classification process.

Among the different rules and procedures developed, it is possible to mention: The Generalized Delta Rule (13), Counterpropagation (5), Adaptive Resonance Theory (2), Hopfield (8). They have their limitations and strengths and it is possible to identify suitable applications areas for which they are intended. In the next subsections, the generalized delta rule, a supervised training method for feedforward networks, is explained.

The Generalized Delta Rule

The generalized delta rule is a learning procedure developed by Rumelhart (13) which learns adequate internal representations using deterministic units to provide a mapping from input to output. This procedure involves the calculation of a set of output vectors O using the current weights W (set composed of matrixes $W_{m-2}....W_{m=1}$ where $W_{m=2}$ would be the matrix of weights between the input and the first hidden layer and $W_{m\,1}$ the matrix of weights between the last hidden layer and the output layer) and θ (set composed of matrixes $\theta_{m\,2}...\theta_{m=1}$ where $\theta_{m=2}$ would be the matrix of biases of the first hidden layer and $\theta_{m=1}$ the matrix of biases corresponding to the output layer) state of the network and a given set of input vectors I. This set of calculated output vectors will be compared to a target set of output vectors T and an error is estimated by using an error function. This error function is defined for an specific I_p and T_p as follows:

$E_p = 1/2\sum(t_i - o_{il})^2$

where the index p represents each input vector/target output vector that conforms the input vector set I and

target output vector set T, i represents the output nodes of the output layer in the network, and l is the total number of layers (i.e., layer m = l is the output layer, layer m = 1 is the input layer). t is the targeted output for the ith output node and o is the response obtained from the ith output node using the corresponding I_p. Thus the total error will be determinate as:
$E = \sum_p E_p$.

The learning procedure minimizes E_p by performing steepest descent and therefore obtaining appropriate W and θ.

As explained above, the net input to a neuron is expressed as:
$net_{im} = \sum w_{ijm} o_{jm-1} + \theta_{im}$
where w represents the weight between the jth unit of layer m-1 and the ith unit of layer m. In addition, the activation function utilized is the logistic function given by:
$o_{im} = 1/(1 + (e^{-net_{im}}))$.

It is possible to conclude that to minimize E_p and achieve convenient W and θ it is necessary to make adjustments to previous W and θ obtained until the error tolerance imposed by the final desired mapping accuracy is accomplished. Therefore, it is possible to establish
$\Delta W_{ijm} \alpha - \partial E_p / \partial W_{ijm}$

$\Delta \theta_{im} \alpha - \partial E_p / \partial \theta_{im}$.

Then the partial derivative of E with respect to the weights and biases could be expressed as:
$\partial E_p / \partial w_{ijm} = (\partial E_p / \partial net_{im})(\partial net_{im} / \partial w_{ijm})$
$\partial E_p / \partial \theta_{im} = (\partial E_p / \partial net_{im})(\partial net_{im} / \partial \theta_{im})$
and the partial derivative of the error to the net input could be stated as:
$\partial E_p / \partial net_{im} = -\delta_{im}$.

It is possible to replace and get the following terms:
$\partial E_p / \partial w_{ijm} = -\delta_{im} o_{jm-1}$
$\partial E_p / \partial \theta_{im} = -\delta_{im}$.

The variable δ defined above could be calculated by backpropagating the error through the network starting with the output layer where the partial derivative of the error to the output is defined as:

$$\partial E_p / \partial o_{il} = \partial (1/2 \sum (t_i - o_{il})^2) / \partial o_{il}$$
$$= -(t_i - o_{il})$$

and δ_{il} (output layer) is

$$\delta_{il} = -(\partial E_p / \partial o_{il})(\partial o_{il} / \partial net_{il})$$
$$= (t_i - o_{il}) o_{il} (1 - o_{il})$$

and the adjustments are equal to

$$\Delta w_{ijl} = \eta \delta_{il} o_{jl-1}$$

$$\Delta \theta_{il} = \eta \delta_{il}$$

where η is the learning rate.

Learning of an ANN is achieved through a sequence of iterations or epochs (13). An epoch is a pass through the entire training set. The operations to update w and θ can be done in two modes:
a. For each pattern.
b. For the input vector set.

In spite of the capabilities of this proven algorithm, the rate of convergence might be very slow. This rate of convergence is dependent on many factors such as initial weights, learning rates, complexity of the mapping to be performed, number of hidden layers, connections, hidden units, data samples, identification of relevant data samples, etc. Hence a great deal of research is on progress to speed up the rate of convergence and improve the mapping accuracy provided by the generalized delta rule. One of the most used heuristics to speed up the rate of convergence is the utilization of a momentum factor that weights the contribution of the past ΔW and $\Delta \theta$. The updating equations will be modified as follows:
$w_{ijm}(t) = w_{ijm}(t-1) + \Delta w_{ijm}(t) + \alpha \Delta w_{ijm}(t-1)$
$\theta_{im}(t) = \theta_{im}(t-1) + \Delta \theta_{im}(t) + \alpha \Delta \theta_{im}(t-1)$.

Once a training session of an ANN has accomplished the constraints of the mapping accuracy, it should be able to relate input vectors with the appropriate output vectors. If this is not achieved modifications have to be implemented to the network architecture, with the possibility of changes to the input feature space and training set, and the applicability of the learning paradigm to the problem should be evaluated. If the previous step was fulfilled, the next step to satisfy is the utilization of the ANN as a predicting mechanism. This is exercised using patterns that were not previously taught and recording the performance exhibited by the trained ANN. In this step, the generalization capabilities of the network will be an important indication that it is dependent on the architecture achieved, features represented in the input feature space, and the suitability of our training set.

An ANN for Diagnostic in Robotics

To illustrate the backpropagation paradigm an example has been developed in the area of robotics and diagnosis. This example has been designed utilizing real data from a GE P60 robot and it is a small scale version of a larger implementation of an "expert connectionist network" (16). In the case of "expert connectionist networks" (3) the knowledge is not stored by facts or IF-THEN rules or in a specific knowledge base location. Patterns of activation levels in the neurons inherited from the connection strengths and the distributed structure provides the knowledge representation scheme. This representation scheme has several advantages, such as the creation of models with limited knowledge engineering participation and naturally built of fuzzy predicate functions.

In this robotics application, an artificial feedforward neural network was trained on the functional relationships between specific events (symptoms), diagnoses, and recovery. The input parameters for the ANN could be collected using different sensory techniques or human

input--in this case the analog characteristics of ANNs provides significant benefits to interface and develop a real-time system. In order to get a response, all the input-questions should be answered at once (See Table I). The ANN performs the classification task based on the features of the problem. The result of this classification task is the type of malfunction and the error recovery strategy.

	INPUT
I0	DIFFERENTIAL VOLTAGE DOES NOT COME TO 0V.
I1	ARM IS MOVING EVEN THOUGH NO COMMAND TO MOVE IS GIVEN.
I2	MOTORS ARE TURNED ON.
I3	EXCESSIVE CURRENT FLOWED IN MOTOR.
I4	STROKE END LIMIT SWITCH ACTUATED.
I5	THE AXIS CAN NOT BE MOVED FARTHER.
I6	PLAYBACK MODE.
I7	RECORD MODE.
I8	MANUAL MODE.
I9	TEACH MODE.
I10	TEMPERATURE RISE ERROR.
I11	STOP SWITCH ON OPERATION PANEL IS DEPRESSED.
I12	STOP SWITCH ON THE TEACHING CONTROLLER IS DEPRESSED.
I13	TRANSISTOR HEAT SINK IS OVER HEATED.

	OUTPUT
O0	SYSTEM ERROR.
O1	OPERATOR ERROR.
O2	THE SERVO SYSTEM EXCEEDS THE ALLOWABLE. OPERATOR SHOULD NOT BE FORCING THE BRAKES WHILE TURNING ON.
O3	EXCESSIVE WEIGHT ON THE ARM. EXCESS PAYLOAD SHOULD BE REMOVED.
O4	THE ROBOT HAS CRASHED. ERROR RESET PROCEDURE SHOULD BE PERFORMED.
O5	THE BASE ROTATION AXIS HAS EXCEEDED THE ALLOWABLE STROKE. JOG AXIS AWAY OPERATION.
O6	THE BEND AXIS HAS EXCEEDED THE ALLOWABLE STROKE. JOG AXIS AWAY OPERATION.
O7	ARM ANGLE IS TOO LARGE OR TOO SMALL. MOVE THE AXIS IN THE OPPOSITE DIRECTION OR ADD EXTRA POINTS.
O8	TEMPERATURE IN THE CONTROLLER EXCEEDS 55 C. SUCTION PORTS OF THE HEAT EXCHANGE AND THE AIR DUCK SHOULD BE CHECKED.
O9	VENTILATING DUCK FAN IS NOT WORKING PROPERLY. VENTILATING DUCK FAN SHOULD BE CHECKED.
O10	STOP BUTTON AND START COMMAND ARE GIVEN AT THE SAME TIME. REMOVE STOP SIGNAL AND RESTART.

Table I. Input Output for the diagnostic system

Training a One-Hidden Layer Architecture This architecture has 14 inputs that correspond to each specific event, 11 outputs that identify diagnostic and recovery procedures, and two hidden units in the hidden layer (See Figure 4). The training was performed using the training data shown in Table II.

The following two steps were utilized to train the ANN:

(Step 1: Initialization)
1. A standard backpropagation architecture was selected. The output of the input layer is equal to its input, the hidden and the output layers uses a sigmoidal logistic as activation function:
$\lambda/(1 + e^{-\beta \times net})$
where, for our case, $\lambda = 1$ and $\beta = 1$.
2. The weights and biases were initialized using random values between -0.5 and +0.5.
3. The updating mode for W and θ to be utilized is by training set.
4. For this specific network a learning rate of 0.25 and a momentum of 0.9 yielded excellent results (See Figure 5).

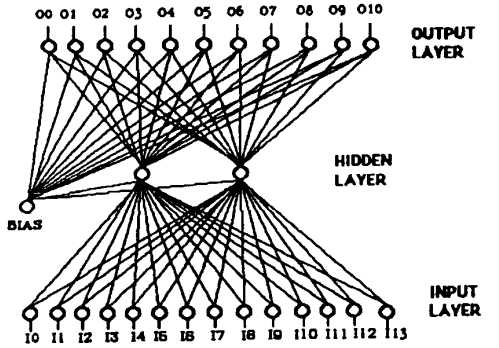

Figure 4. ANN architecture with a hidden layer

5. The constraints to satisfy were the total root mean square (RMS) error and the maximum output error. These errors are defined by:
Total RMS error = $\sqrt{\sum_p \sum_i (t_{ip} - o_{ip})^2 / (\#\text{patterns} \times \#\text{output}}$
Maximum output error $= \max_{p,i} |t_{ip} - o_{ip}|$.
(Step 2: Training)
1. An input vector is presented to the network from the input vector set. The output of each unit of the network is calculated starting from the lowest layer to the output layer. This will require computations of the net input to each neuron and the logistic function for the hidden and output layer units. For the input layer, the outputs of the units will be equal to the input values themselves.
2. Calculation of δ is performed for the output and hidden layer units in that specific order:
$\delta_{i3}(output) = (t_i - o_{i3})o_{i3}(1 - o_{i3})$
$\delta_{j2}(hidden) = o_{j2}(1 - o_{j2})\sum(\delta_{i3}(output)w_{ij3})$
3. Calculation of w' and θ' for the output and hidden layer units as follows:
$w'_{ijm} = \delta_{im}o_{jm-1}$
$\theta'_{im} = \delta_{im}$.

input	11100000000000
output	10110000000
input	00110000000000
output	10011000000
input	00111001000000
output	01000110000
input	00100000001000
output	10000000100
input	00100100100000
output	01000001000
input	00100100010000
output	01000001000
input	00100101000000
output	01000001000
input	00000000000001
output	10000000010
input	00100000000100
output	01000000001
input	00100000000010
output	01000000001

Table II. Training set

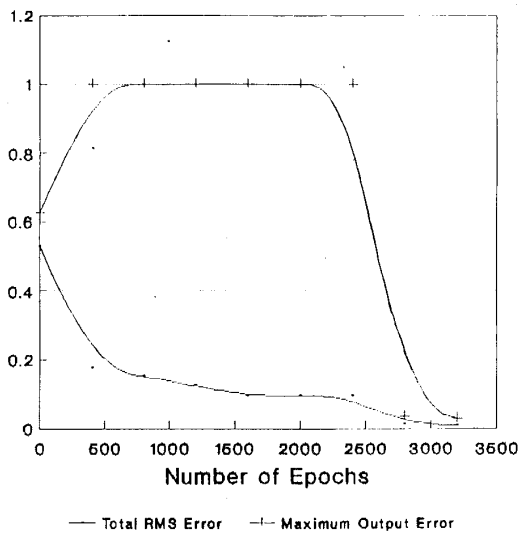

Figure 5. Learning curve

4. Repeat 1, 2, and 3 for the next input pattern and accumulate w and θ for the output and hidden layer units until the entire training set has been processed.
5. Update w and θ using the following equations:
$w_{ijm}(t) = w_{ijm}(t-1) + \eta\Delta w'_{ijm}(t) + \alpha\Delta w_{ijm}(t-1)$
$\theta_{im}(t) = \theta_{im}(t-1) + \eta\Delta\theta'_{im}(t) + \alpha\Delta\theta_{im}(t-1)$.
6. Calculations of the total RMS error and the maximum output error. Comparison with the accuracy requested and decision to stop or continue training.

Results of the training session are shown in Figure 5. It is clear that both criteria, the total RMS error and the maximum output error are important to determine when a network has learned.

INTELLIGENT SCHEDULING SYSTEM FOR FMS

The Intelligent Scheduling System for FMS (ISS_FMS) is designed to support the integration of scheduling and control functions in FMSs (1,12). Therefore several functions are required and are explained as follows (See Figure 6):

Communication. The higher hierarchical level sends data such as the number of jobs to be scheduled, process plans, processing times, due dates, and production goals. This system also communicates with the lower levels of the hierarchy.

Scheduling. A schedule is generated for the jobs to be manufactured at different work stations. This schedule takes into consideration the dynamic status and satisfies the performance criteria imposed by the shop controller.

Human Machine Interface. A user friendly environment should be provided.

Learning. The learning/supervisory functions should check the performance of the problem solving architecture.

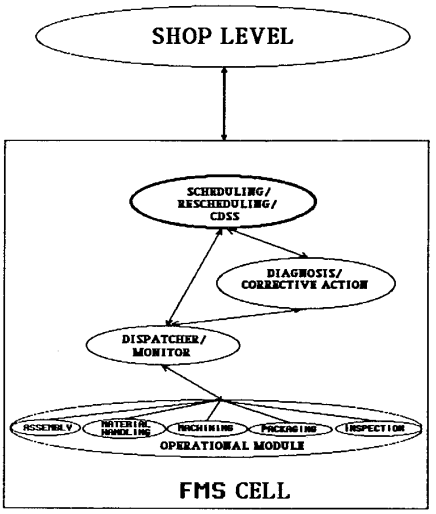

Figure 6. Scheduling and control model for FMS

General Description

The architecture of the ISS/FMS consists of the following subsystems (See Figure 7):
Expert Scheduler;
Heuristics Programs;
Artificial Neural System for Heuristics;
Expert Look-Ahead Scheduler;
Artificial Neural System for Coefficients;
Learning Unit;
Decision Support Unit;
Databases.

This intelligent system has been designed using modularity principles. The ISS/FMS subsystems will be explained in the following subsections.

The Expert Scheduler

The Expert Scheduler is the knowledge controller of ISS/FMS. In order to perform its high order functions the Expert Scheduler consists of several rule-based modules utilizing backward/forward chaining. It is composed of the following rule-based modules: Interpretation and Feasibility, Control of Scheduling Resources, and Discrimination and Evaluation.

Interpretation and Feasibility. This rule-based module interprets the request from the shop level for scheduling. This request is stated by commands and databases. The job database and objectives to be achieved in this scheduling are analyzed using updated information of the current status, and availability of resources and materials of the given FMS cell. If there exists incomplete information, this is requested to that specific level.

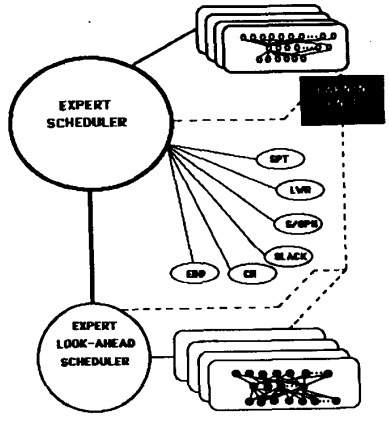

Figure 7. ISS/FMS architecture

Constraints imposed by the shop level can be stated in three categories:
1. Perfomance measure(s) optimization desired;
2. Constraints imposed by the performance measure(s);
3. Time in which the result is expected.

The Interpretation and Feasibility module based on the information obtained concludes whether the generation of a suitable outcome is possible. If the degree of feasibility is perceived to be lower than the required by the request, a message will be sent with possible changes to be done before developing a schedule.

The current implementation of ISS/FMS is strictly based on tardiness, but future enhancements will include minimization of in-process inventory and maximization of machine utilization.

Controller. The Controller is a rule-based module in the Expert Scheduler that takes into consideration the output of the ANN(s). The Controller, based on the request of the higher level and the relative heuristic ranking provided by the ANN(s) develops a criterion. This criterion is used to call specific heuristic(s). Also, based on the time frame provided, the Expert Look Ahead Scheduler (ELAS) will be called.

Discriminator and Evaluator. The Discriminator is a rule-based module in the Expert Scheduler. The discriminator receives the answers to the scheduling problem from the selected heuristics and the Expert Look Ahead Scheduler (ELAS). It selects the best among them, and proceeds to send the answers to the appropriate levels. If the final schedule does not meet some of the high priority constraints, the Discriminator checks the decision time frame available to determine the time constraints. It then makes changes to the job database and recursively proceeds with the process.

Dispatching Rule Programs

The dispatching rules utilized in this research prototype are:
SPT: Shortest Process Time;
EDD: Earliest Due Date;
CR: Critical Ratio;
SLACK: Slack Time Remaining;
S/OPN: Slack/Operation;
LWR: Least Work Remaining.

New dispatching rules could be simply added because of the modularity design of ISS/FMS.

Artificial Neural System for Heuristics (ANSH)

The Artificial Neural System for Heuristics, according to the parameters passed by the Interpretation and Feasibility module, decides what ANN(s) to use. The Artificial Neural System for Heuristics has several ANNs based on the number of machines, number of jobs utilized, and performance measure desired. These are three-layer feedforward networks trained using the generalized delta rule. The average size of each ANN is about 15 input units, 65 hidden units, and 6 output units.

Input Feature Space for Tardiness. One has to develop appropriate ways to represent the problem to be learned in order to make it understandable for the network. Without this key feature, the neural network will fail to learn the relationship desired with the efficiency and accuracy desired. For FMS scheduling and tardiness, the input the following dimensions were selected (See Figure 8):
1. Group Technology;
2. Time Remaining Until Due Date;
3. Number of Jobs.

In the current ISS/FMS research prototype data have been selected from an FMS cell with capacity to manufacture starting with three process plans.

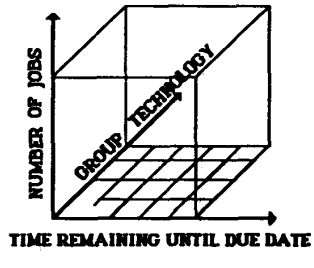

Figure 8. Input feature space

Training. The ANN's were trained with data generated from simulations performed for that purpose. A normal training session was able to spend on average more than 85 hours and using training data files of 32 Kb in a IMB 6152 workstations equipped with INTEL 80286/80287 microprocessors.

Expert Look Ahead Scheduler (ELAS)

This expert system implements a new feedback-based heuristic scheduling procedure. This heuristic procedure, which was utilized mainly for the tardiness criterion (6), has been proven to be effective for the FMS scheduling problem (11). ELAS calls the Artificial Neural System for Coefficients (ANSC) which are utilized to provide coefficients to accelerate the performance of the algorithm and improve its real-time capabilities. ELAS also calls the feedback heuristic algorithm and has a set of rules based on heuristics to make an intelligent search from the results provided by the algorithm using the coefficients predicted by ANSC. This rule-based module is fired when the decision making time frame permits its utilization.

Units Under Development

The Decision Support Aid Unit and the Learning Unit in the current ISS/FMS research prototype are under development. They will provide innovative features to ISS/FMS.

Decision Support Aid Unit. The Decision Support Aid Unit currently provides the generation of Gantt Charts, line charts, and bar charts. Also, it is possible to query the system about possible scenarios and input data from commercial spreadsheets.

Learning Unit. The learning unit is another example of the integration of ESs and ANNs. This unit implements a feedback mechanism needed to identify how well the knowledge encoded in the different ANNs is performing.

The envisioned unit will have a backup of the different artificial neural networks that the FMS scheduler uses. An ES could be used to support the monitoring, diagnosis, and recovery strategy of this unit.

The functions of the ES are the monitoring and diagnosis of the FMS scheduler. If the solution of the FMS scheduler does not meet the required performance level, the supervisory/learning unit will impose a strategy to update the problem solving system. This strategy implies retraining a backup copy of the artificial neural neural structure that performed below accepted limits.

AN EXAMPLE

The following is an example that embodies some of the ideas behind ISS/FMS.

Problem Definition

Suppose that there is a type of four-machine and ten-job scheduling problem. A FMS which could produce three basic products is given.

Training and Results

Simulations were carried out to generate a data set of 200 examples. An initial architecture was selected based on experiences. Trial and error procedures were used to select the architectural parameters. Finally, a successful topology was found. It had 15 input units, 68 hidden units, and 6 hidden units. This network was tested for a "new" set of 100 examples and yielded consistently good results (lowest tardiness in 83 out of 100 cases). Table III compares the tardiness values obtained.

LOWEST TARDINESS FREQUENCY - 100 problems

SPT	LWR	SLACK	S/OPN	CR	EDD	ANN
12	11	60	62	53	57	83

AVERAGE TOTAL TARDINESS - 100 problems

SPT	LWR	SLACK	S/OPN	CR	EDD	DRC	ANN
82	83	72.9	72.6	73	76.3	69.6	71

DRC Dispatching rules combined

Table III. Results of a ANN trained with 200 examples

SUMMARY

ISS/FMS has been designed based on the integration of several technologies. Artificial neural networks have been utilized as effective prediction tools and scheduling pattern recognition mechanisms. Expert systems, on the other hand, have been utilized as the higher order members that interact with other elements of the FMS hierarchy providing guidance for problem-solving strategy, monitoring the performance of the system, and automating the ANN learning process. As a result, the level of responsiveness achieved may provide the necessary strength for scheduling in FMS.

REFERENCES

1. ALPTEKIN, S. and RABELO, L., 1988, Expert System Applications in CIM. presented at the ORSA/TIMS National Conference, Denver, Colorado.

2. CARPENTER, G. and GROSSBERG, G., 1987, A Massively Parallel Architecture for a Self-Organizing Neural Pattern Recognition Machine. Computer Vision, Graphics, and Image Processing, 37, 54-115.

3. GALLANT, S., 1988, Connectionist Expert Systems. Communications of the ACM, 331(February), 152 - 169.

4. GROSS, J., 1987, Intelligent Feedback Control for Flexible Manufacturing Systems. Ph. D. Thesis, University of Illinois at Urbana-Champaign.

5. HECHT-NIELSEN, R., 1988, Applications of Counterpropagation Networks. Neural Networks, 1(2), 131 - 139.

6. KIRAN, A. and ALPTEKIN, S., 1989, A Tardiness Heuristic For Scheduling Flexible Manufacturing Systems. Proceedings of the 15th Conference on Production Research and Technology: Advances in Manufacturing Systems Integration and Processes, 559-564.

7. KUSIAK, A., 1989, Scheduling Automated Manufacturing Systems: A Knowledge-Based Approach. Proceedings of the Third ORSA/TIMS Conference on Flexible Manufacturing Systems: Operations Research Models and Applications, (Elsevier Science Publishers B. V.), 377-382.

8. LIPPMANN, R., GOLD, B. and MALPASS, M., 1987, A Comparison of Hamming and Hopfield Neural Nets for Pattern Classification. Massachusetts Institute of Technology, Lincoln Laboratory, Technical Report 769.

9. O'GRADY, P. and LEE, K., 1988, An Intelligent Cell Control System for Automated Manufacturing. International Journal of Production Research, 26(5), 845-861.

10. PARK, S., Raman, N., and SHAW, M., 1989, Heuristic Learning for Pattern Directed Scheduling in a Flexible Manufacturing System. Proceedings of the Third ORSA/TIMS Conference on Flexible Manufacturing Systems: Operations Research Models and Applications,(Elsevier Science Publishers B. V.), 369-376.

11. RABELO, L., 1988, Comparison of Heuristics for Scheduling. Technical Report, Computer Integrated Manufacturing Laboratory, UMR.

12. RABELO, L. and ALPTEKIN, S., 1989, Synergy of Neural Networks and Expert Systems for FMS Scheduling. Proceedings of the Third ORSA/TIMS Conference on Flexible Manufacturing Systems: Operations Research Models and Applications, (Elsevier Science Publishers B. V.), 361-366.

13. RUMELHART, D., McCLELLAND, and the PDP Research Group, 1988, Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundations (Cambridge, MA: MIT Press/Bradford Books).

14. STORK, D., 1989, Self-Organization, Pattern Recognition, and Adaptive Resonance Networks. Journal of Neural Network Computing, 26 - 42.

15. THESEN, A. and LEI, L. 1986, An Expert System for Scheduling Robots in a Flexible Electroplating System with Dynamically Changing Workloads. Proceedings of the Second ORSA/TIMS Conference on FMS, 555-566.

16. WU, H., 1989, Neural Networks and Robotics, M. S. thesis, University of Missouri-Rolla.