Acquisition and Refinement of State Dependent FMS Scheduling Knowledge Using Neural Network and Inductive Learning

Chang-Ouk Kim*, Hyeung-Sik Min**, Young Hae Lee***

인공신경망과 귀납학습을 이용한 상태 의존적 유연생산시스템 스케쥴링 지식의 획득과 정제

김창욱* · 민형식** · 이영해***

ABSTRACT

The objective of this research is to develop a knowledge acquisition and refinement method for a multi-objective and multi-decision FMS scheduling problem. A competitive neural network and an inductive learning algorithm are integrated to extract and refine necessary scheduling knowledge from simulation outputs. The obtained scheduling knowledge can assist the FMS operator in real-time to decide multiple decisions simultaneously, while maximally meeting multiple objectives desired by the FMS operator. The acquired scheduling knowledge for an FMS scheduling problem is tested by comparing the desired and the simulated values of the multiple objectives. The result shows that the knowledge acquisition and refinement method is effective for the multi-objective and multi-decision FMS scheduling problems.

Key Words : Inductive Learning, Neural Network, Real-Time Scheduling, Flexible Manufacturing System

* 고려대학교 정보통신기술공동연구소
** LG-EDS CIM사업부
*** 한양대학교 산업공학과

1. Introduction

Flexible manufacturing systems (FMSs) have been applied to increase the flexibility and the productivity of various discrete part manufacturing. These highly integrated systems are becoming more complex, hence difficult to control situations that cause several simultaneous decision problems. In particular, the scheduling task of FMS, the control problem during the operation, is much more complex than transfer lines because of simultaneous machining and alternative routing of different part types. In general, the scheduling task of FMS involves several real-time operation decisions such as the selection of machine by part, selection of part by machine, selection of part by material handling system, and so on. Good scheduling method must be able to successfully control the early and late completion of parts, hold minimum work-in-process, and highly utilize resources. In addition, if a computer-aided method is used for the scheduling, it must be run often and fast. These requirements often dictate a simplistic scheduling model that trades off solution time for realism and solution quality.

The objective of the paper is to develop an acquisition and refinement method of FMS scheduling knowledge. The scheduling knowledge is constructed such that the FMS operator can decide the multi-decision variables timely, and yet maximally meet the multiple relative objectives. For the knowledge acquisition and refinement method for the multi-objective and multi-decision FMS scheduling problem, the applicability of a competitive neural network and an inductive learning algorithm is investigated. By incorporating the competitive neural network, the inductive learning algorithm, and a simulation, a knowledge-based FMS scheduler is built up. It can control the behavior of part flows to accomplish the multiple relative objectives by generating the best decision rules. Moreover, the knowledge-based FMS scheduler has the ability to respond in real-time in the sense that the scheduler suggests good decision rules in a few seconds whenever the FMS operator inputs the multiple relative objectives.

The remainder of this paper is organized as follows. Previous scheduling methods using artificial intelligence techniques are reviewed in Section 2. In Section 3, the multi-objective FMS scheduling problem is formally defined, and an FMS simulation scenario is explained. The scenario is used later in this paper to investigate the effectiveness of the proposed knowledge acquisition and refinement method. In Section 4 and 5, the neural network model and the inductive learning are presented, respectively. Finally, the results of the simulation and the conclusions of this research are discussed in Section 6.

II. Previous Artificial Intelligence Approaches to Scheduling

The applications of expert systems to FMS scheduling problems have drawn a great deal of interest over the last fifteen years. Numerous studies have been published in recent years because it gives fast and acceptable solutions, although not guaranteeing optimal solutions (Shaw, 1989 ; Yih, 1990). The main idea behind the applications of expert systems to FMS scheduling problems is that each scheduling system is unique to a given environment and, therefore, a wide variety of technical knowledge and expertise should be taken into account in solving these scheduling problems.

However, despite the advantage of the expert system technique, few applications have been reported in real FMS scheduling due to the difficulty of knowledge acquisition by human experts (Fox, 1990). The behavior of most FMSs is so complex that it is unrealistic to find good scheduling knowledge unless the system is quite simple, or running over a long period enough to represent the best scheduling knowledge. To overcome this disadvantage, some researchers applied the search algorithms of artificial intelligence to automatically find the scheduling knowledge of a given job shop problem (Fox and Smith, 1984 ; Sadeh *et al.* 1995). However, their approaches cannot be applied to real-time FMS scheduling problems because their algorithms require immoderate computational times.

Compared to the expert system technique, the neural network models do not require the knowledge acquisition phase. Instead, they implicitly acquire necessary scheduling knowledge through a training phase. In the training phase, the weighting vectors of neural network is adjusted such that the inputs produces the right outputs (Bharath and Drosen, 1994). Due to the advantage of the implicit knowledge acquisition, many researchers have applied the neural network models to FMS scheduling (Chryssolouris et al. 1991 ; Nygard *et al.* 1991 ; Yih *et al.* 1991 ; Sim *et al.* 1994).

The main problem with using neural network models is that the acquired knowledge through the training phase cannot be explicitly represented. Thus, it is difficult to modify the knowledge if necessary. Knowledge modification is often demanded because the state of FMS continuously changes.

Recently, an inductive learning algorithm called ID3 (Quinlan, 1986), one of machine learning techniques, has been applied for inducing scheduling knowledge from a limited set of scheduling knowledge (Shaw *et al.* 1990 ; Shaw *et al.* 1992). The limited set of scheduling knowledge is called examples, which are usually obtained from simulation outputs. Each of the examples include attribute values and a pre-defined class. Attributes are used to describe the characteristics of an example. A class is defined as the group to which examples with similar characteristics are assigned. The class in FMS scheduling, for example, can be the best dispatching rule for a given set of FMS attributes. The inductive learning method infers the concept of a class from the examples of the class. The concept can be represented either by a rule or a decision tree.

The main disadvantage of inductive learning is that the classes to which examples are assigned must be pre-defined. For example, for a given set of FMS attributes, the best dispatching rule (class) can be selected after a simulation is run for each dispatching rule. This process becomes an intolerable time-consuming task as the number of attributes is large.

To resolve the above problem, we first apply a competitive neural network model to identify classes, then feed the result of the competitive neural network to an inductive learning software called C4.5 (Quinlan, 1993). The C4.5 implements an advanced version of Quinlan's ID3 algorithm for refining and generating rule-based scheduling knowledge. From the next section, our proposed solution approach is explained in detail.

III. Multi-objective FMS Scheduling Problem

3.1. The scheduling problem

To define the scheduling problem dealt with in this paper formally, we formulate the problem as follows :

$$\textit{Minimization} \quad \sum_{i \in S} | s^d_{i,t} - s^a_{i,t} | + \sum_{j \in P} | p^d_{j,t} - p^a_{j,t} |$$

$$\textit{S.T.} \quad \textit{Routing sequences of all parts}$$
$$v_k \in V_k \quad k=1, 2, \dots, D$$

In the above formulation, v_k is the k th decision variable, which can take one of its candidate decision rules. A decision variable is a decision point which is determined by the FMS operator or computer. For instance, whenever a machine completes the processing of a part and is ready to process the waiting parts, the machine should select the next processing part among the waiting parts. This decision variable is called the selection of part by machine (or dispatching decision). V_k is the set of candidate decision rules of v_k, and D is the number of decision variables. S and P denote the set of system status variables and the set of performance measures, respectively. $s^d_{i,t}$ is the desired average value of the ith system status variable at the end of production interval t, and $s^a_{i,t}$ is the actual average value of the ith system status variable at the end of production interval t. A production interval may be a day or a month depending on the operational strategy of the company. In the same way, $p^d_{j,t}$ and $p^a_{j,t}$ are the desired and actual average values of the jth performance measure at the end of production interval t, respectively. Hereafter, we often use the term evaluation criterion to designate a system status variable or a performance measure. The operational scenario of the FMS is as follows: the desired value of each evaluation criterion is provided by the FMS operator at the beginning of the next production interval. More specifically, for each evaluation criterion, the FMS operator gives the scheduler the difference (change) between the actual average value of the evaluation variable at the end of the current production interval and the desired average value of the evaluation criterion at the end of the next production interval. Then the scheduler provides the operator with a good decision rule for each decision variable at the beginning of the next production interval. In this research, the difference is called relative objective value.

3.2. The FMS scenario

The FMS studied in this research emulates a Mazak FMS. The model chosen for simulation consists of four machining centers, a washing machine, thirty nine work-in-process storage racks, and a crane for material handling. Figure 1 shows the system. Each machining center has one input and one output buffer. In addition, each machining center has several interchangeable tool magazines in order to process various operations by mounting different tool magazines. Therefore, the Mazak system has great routing flexibility. This also implies that scheduling problems in such a system are much more complex.

In this study, we assume that the Mazak system has existing policies for tool management. Alternative machine options exist for various operation types. A crane is assigned for transporting parts among machine centers, loading/unloading station and work-in-process storage racks. The parts are transferred by the crane with a speed of 70 feet/min. on the rail and of 20 feet/min. vertically at the storage rack area. When the crane completes the transfer, it either stays at the same station or goes to the designated station depending on the control policy.

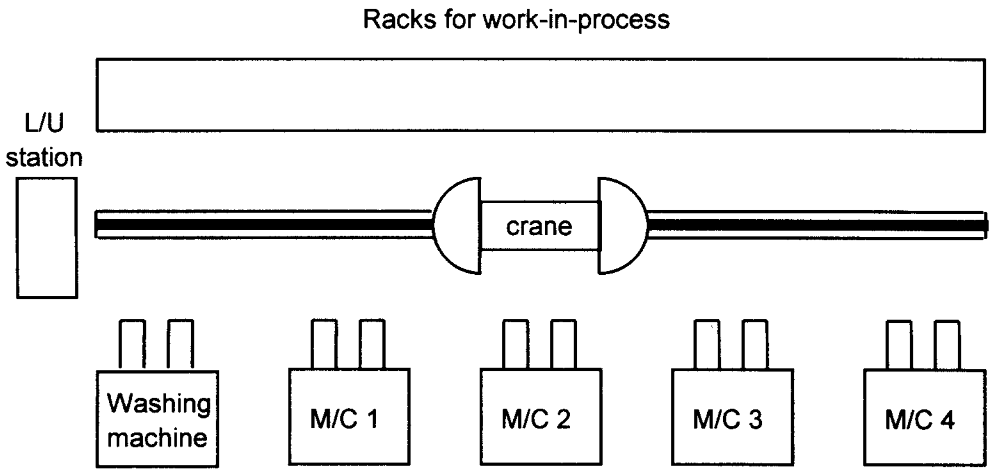

Fig. 1. Configuration of the experimental FMS

Five types of part are processed in the FMS, and each part type can be processed by several flexible routing sequences. The routing sequences and machining times at each alternative machine for each part type are listed in Table 1. Also, it is assumed that the inter-arrival times of all parts are exponentially distributed, but if there are too many parts waiting in the racks or loading/unloading station, the arrived parts will not be allowed to enter into the FMS until a rack is free. In this case study, four decision variables are used.

Table 1. Input data of the simulation

Part type	Operation Number	Machining time (min./part) on alternative machine						
		1	2	3	4	WM	Avg.	Total
1	1	21	23	22	18	–	21	
	2	14	–	15	20	–	16.3	60.3
	3	15	–	11	13	–	13	
	4	–	–	–	–	10	10	
2	1	13	9	–	–	–	11	
	2	–	16	–	24	–	20	69.3
	3	8	–	9	12	–	9.7	
	4	20	18	16	19	–	18.3	
	5	–	–	–	–	10	10	
3	1	11	–	10	14	–	11.7	
	2	21	20	–	–	–	20.5	61.2
	3	–	17	–	21	–	19	
	4	–	–	–	–	10	10	
4	1	12	–	13	9	–	11.3	
	2	–	18	–	21	–	19.5	
	3	20	16	–	–	–	18	94.1
	4	24	20	19	22	–	21.3	
	5	12	–	16	–	–	14	
	6	–	–	–	–	10	10	
5	1	–	18	–	19	–	18.5	
	2	18	19	17	16	–	17.5	59
	3	10	–	15	14	–	13	
	4	–	–	–	–	10	10	

The simulation model of the Mazak FMS is developed by the FORTRAN based SLAM II simulation language. A long simulation period is divided into many short production intervals. In each production interval, we apply a random combination of decision rules, thereby obtaining system status and performance measure data at the end of each production interval. The same procedure is applied in the next production interval. After two production intervals end, the differences of the evaluation criteria between the two intervals are collected together with the two sets of decision rules. Then the differences and the two sets of decision rules are collected as an input vector for training the neural network. The length of each production interval is set to 480 minutes and 3500 input data are collected.

3.3. Decision variables and evaluation criteria

The definition of each decision variable and associated decision rules is given in Table 2. In an FMS, changing the scheduling rules in real time will have an effect on the system status and performance measures. There are numerous criteria that can be used in evaluating the system performance and system status of the FMS. Some of these are based on completion-times, some on due-dates, and some on the inventory and processing times. In this study, we select three performance measures : (1) mean Tardiness, (2) maximum Tardiness, (3) mean flow time, and two system status variable : (1) slack and (2) crane utilization.

Table 2. Decision variables and associated decision rules

Decision variable	Associated rules
Selection of machine by part	1. FWJM - fewest waiting jobs for a machine 2. CYC - cyclic priority 3. LAUF - lowest average utilization first 4. SFTO - shortest flow time at an operation
Selection of rack by part	1. SDR - shortest distance to rack 2. SDMR - shortest distance to the nearest rack from a machine of next operation 3. RANDOM - random selection
Selection of part by machine	1. SIO - shortest imminent processing time 2. FCFS - first come and first serve 3. SRPT - shortest remaining processing time 4. EDD - earliest due date
Selection of part by crane	1. SDP - shortest distance to part 2. SRPT - shortest remaining processing time 3. EDD - earliest due date 4. MSLACK - minimum slack

Ⅳ. Knowledge Acquisition : A Competitive Neural Network Approach

For the knowledge acquisition of multi-objective FMS scheduling problem, a competitive neural network is applied. The competitive network can learn to detect regularities and correlations in its input vectors (patterns) and adapt future responses to the input vectors accordingly. Figure 2 depicts the architecture of the competitive neural network applied to classify the outputs of the simulation. A class contains a set of input vectors, some outputs of the simulation which shows the similar differences of the evaluation criteria. In Figure 2, N is the size of the input vector, and M is the size of the output vector. Also, x_i and y_j indicate the ith input node and jth output node, respectively. W_j is the weighting vector connected to jth output node from all input nodes. In this study, the competitive neural network is trained with Kohonen's learning rule (Bharath and Drosen, 1994) using 3500 data obtained from the simulation. Neural network toolbox in the Matlab software (Demuth and Beale, 1995) is applied for training.

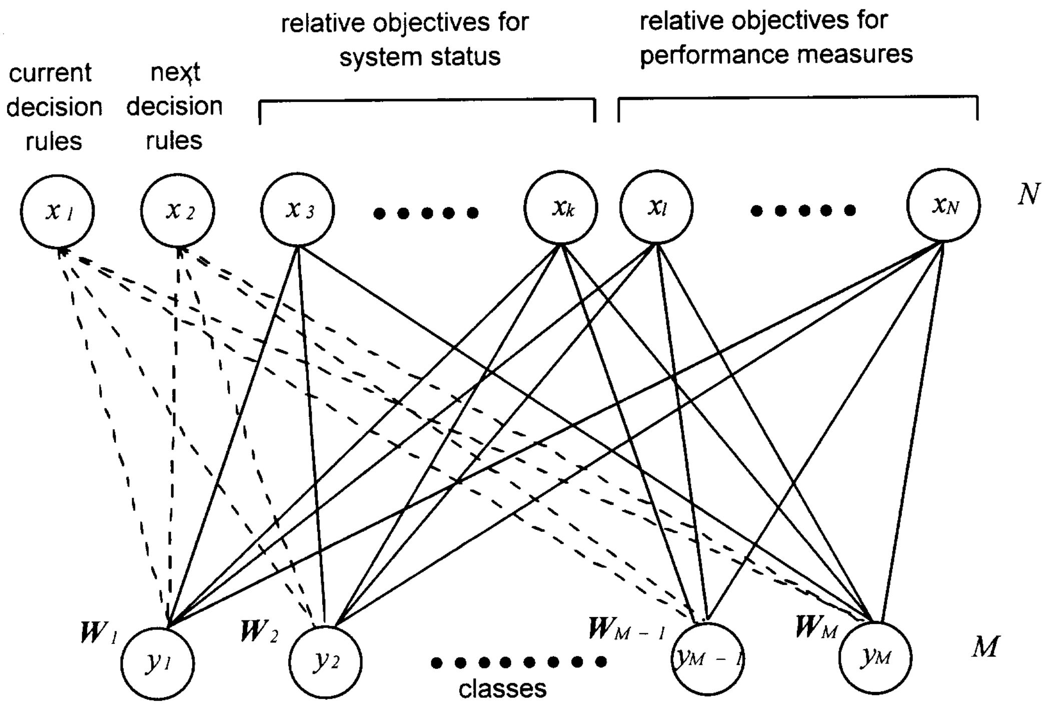

Fig. 2. The competitive neural network

In this research, the neural network model is designed to be trained such that it can (1) generate next decision rules based on the current decision rules, system status and performance measures, and (2) control the levels of system status as well as those of performance measures of the next production interval. To reflect the above features in the competitive neural network, we present the relative objectives of the all evaluation criteria together with the current decision rules and the next decision rules as an input vector to the neural network. In Figure 2, x_1 and x_2 represent the current decision rules and the next decision rules, respectively. Also, x_i, $i=3,\ldots,k$ are the relative objectives of the system status, and x_j, $j=k+1,\ldots,N$ are the relative objectives of performance measures. Unlike other input nodes, x_1 and x_2 are not connected to output nodes through weighting vectors. We do not make the connections deliberately in order that x_1 and x_2 cannot affect the classification. In other words, only the relative objectives contribute on the classification ; x_1 and x_2 are included in the input vector to be used in inductive learning.

The neural network categorizes the training input vectors according to the similarity base which is defined as the similar amount of the differences of the evaluation criteria between the current production interval and the next production interval. A classes is an output node where similar input vectors are aggregated.

V. Knowledge Reasoning : An Inductive Learning Approach

Using the output of the competitive neural network, the C4.5 systematically constructs a decision tree and production rules, this allows us to obtain the characteristics of each class. C4.5 employs outputs of the neural network as a set of training examples to induce scheduling knowledge. A decision tree and production rules produced by C4.5 will be used as the scheduling knowledge and be attached to a scheduler which selects the desired decision rules. For example, in our FMS scenario, an example includes four attributes and a class number. Two attributes describe decision rules on four decision variables for the current and the next production intervals, respectively. The other two attributes contain the differences of evaluation criteria. The class number is determined by the competitive neural network.

Inductive learning does not require precise concepts and definitions for domain problems in advance. A set of training examples is provided as the input for learning the concept which represents each class. A training example consists of attribute values and the associated class. An obtained concept can be presented by forms of a decision tree or rules which are constructed by the inductive learning process. If a new example satisfies the condition of a rule produced by inductive learning, then it belongs to the corresponding class. The quality of learned concepts depends on training examples, attributes and pre-defined classes. Attributes are used to describe the characteristics of examples. Quinlan's ID3 algorithm is one of the popular inductive algorithms. The ID3 algorithm induces concepts from examples. Its advantages include its clear representation of learned knowledge, the management of complexity, its heuristic for selecting candidate concepts and its potential for handling noisy data (Quinlan, 1993). It uses a set of training examples to induce IF..THEN.. rules, which constructs a decision tree in top-down fashion. A part of the production rules which classify outputs of the neural network is shown in Figure 3.

```
Rule 10 :
          change_max_tardiness>-447.75
          change_max_tardiness <=-248
          change_mean_flow_time>-222.416
          change_mean_flow_time<=-5.76242
          change_crane_util>-0.12704
          change_slack<=104.767
          -> class 4 [91.0%]

Rule 29 :
          change_mean_tardiness>-136.508
          change_max_tardiness>-248
          change_max_tardiness<=-71.4375
          change_mean_flow_time<=-88.1775
          change_slack>46.4568
          change_slack<=169.24
          -> class 14 [95.1%]

Rule 43 :
          change_mean_tardiness>-45.454
          change_max_tardiness>-239.676
          change_max_tardiness<=-71.4375
          change_mean_flow_time>-88.1775
          change_mean_flow_time<=53.1445
          change_slack>-43.3898
          -> class 10 [96.8%]

Rule 45 :
          change_max_tardiness>=-95.5625
          change_mean_flow_time>53.1445
          change_mean_flow_time>=166.495
          -> class 6 [95.8%]
```

Fig. 3. A part of production rules

After inductive learning determines the class to which desired differences of evaluation criteria belong, the search algorithm starts to search the instances of the class one by one until it finds an instance whose current decision rules are matched with the current decision rules given by the FMS operator. An instance corresponds to an output of the simulation. If such an instance is found, the search algorithm extracts the next decision rules from the instance. Otherwise, a voting method is undertaken as follows : for each next decision variable, find the mostly used next decision rule in the class, and take the rule (winning rule) as the decision rule for the next decision variable. The idea behind employing the voting method is that the winning rules can satisfy the desired relative objectives of evaluation criteria maximally with a great chance no matter what rules are applied in the current production interval.

Ⅵ. Results and Future Research Work

Table 3 shows the actual results and the desired results (values) given by the operator. Since the actual results cannot be obtained as long as the FMS is actually operated, we regard simulated results as the actual values. In this table, the comparison result of crane utilization is omitted because of page limit. Initial applied decision rules are [1131], which are identical to FWJM rule for selection of machine by part, SDR for selection of rack by part, SRPT for selection of part by machine, and SDP for selection of part by crane. After production interval [*t0, t1*], the next decision rules are changed by the guidance of the scheduler. As the last column of Table 3 indicates, the decision rules are changed at every production interval. This implies that the evaluation criteria are varied dynamically, and thus there is no potent combination of rules that can achieve all the objectives.

As shown in Figure 4 to 7, the scheduler satisfies most of the objectives, but it has the difficulty of exactly achieving all the objectives simultaneously in every production interval. Also, it is observed that behavior of the system in terms of the evaluation criteria is not stable even if the scheduler endeavors to keep the good performance measures and system status by altering the decision rules. This phenomenon can be observed in production interval [*t3, t5*]. In production interval [*t3, t4*], the scheduler succeeds in maintaining good performance measures. For example, the mean tardiness and the maximum tardiness are almost zero, which is also desired value given by the operator. In production interval [*t4, t5*], however, the actual values are increased although the operator wants to keep the previous performance measures. The limitation of the FMS capacity is the factor causing the phenomenon. Because the parts are introduced continuously until available rack exists, the system reaches its processing limit at the end of production interval [*t4, t5*]. Therefore, the performance measures become increasing from production interval [*t4, t5*].

Table 3. Comparison results between actual and desired values

Production interval	Actual values (simulation output)				Desired values				Applied decision rules
	mean tard.	max. tard.	mean flow time	slack	mean tard.	max. tard.	mean flow time	slack	
t0 to *t1*	238	366	516	-85	238	366	516	-85	1131
t1 to *t2*	10.9	203	255	100	97.5	216	366	15	2334
t2 to *t3*	0.31	8.1	206	128	3.5	103	205	100	1331
t3 to *t4*	0.28	12.4	210	104	0.31	8.1	206	128	4142
t4 to *t5*	14	270	273	97.2	0.28	12.4	260	104	2111
t5 to *t6*	119	480	405	-38	114	370	373	47.2	4144
t6 to *t7*	120	380	388	-24	98.5	460	375	-7.5	1312
t7 to *t8*	0	0	167	128	19.8	180	238	75.7	4342
t8 to *t9*	0	0	204	103	0	0	167	128	1313
t9 to *t10*	0	0	196	119	0	0	167	103	2331
t10 to *t11*	0	0	204	124	0	0	167	119	2342
t11 to *t12*	18.9	128	271	81.5	0	0	167	124	2321

This situation is also observed in case of the mean flowtime and the slack. One advantage of using the scheduler is that the operator can understand the capacity limit of the system during the FMS operation, which helps the operator design production plans such as lot sizing and selection of part types to be processed concurrently.

Moreover, the scheduler has the ability to loosen the speed of getting worse of evaluation criteria, and reduce the values of evaluation criteria again. For example, in production interval [*t5*, *t8*], the scheduler reduces the increasing speed of the mean tardiness, maximum tardiness, and the mean flowtime. Also, it reduces the decreasing speed of the slack. Furthermore, at the end of production interval [*t8*, *t9*], the mean tardiness, maximum tardiness, and the mean flowtime are decreased down to zero, which is the ideal value observed in production interval [*t3*, *t4*]. *This happening can be seen in case of slack.*

In Figure 4, all actual mean tardiness values are similar to the corresponding desired value except production interval 〈*t1*, *t2*] in which the actual value is 10.9, and the desired value is 97.5. But this can be considered as a desirable result because the scheduler lowers the mean tardiness more than the operator's desired value. Such a result can be seen several times in the other graphs (see Figure 5 to 7). Some large deviations in the graphs are not thought as discouraged result because we choose a real FMS and take into consideration of multi-objectives and multi-decision variables in the experimentation. Also, the inaccuracy can be compensated by gathering more simulation data, or having a prior knowledge about the system capacity.

Throughout this simulation, the integration model of the competitive neural network and the inductive learning has been proved to be an effective tool for acquiring and refining FMS scheduling knowledge.

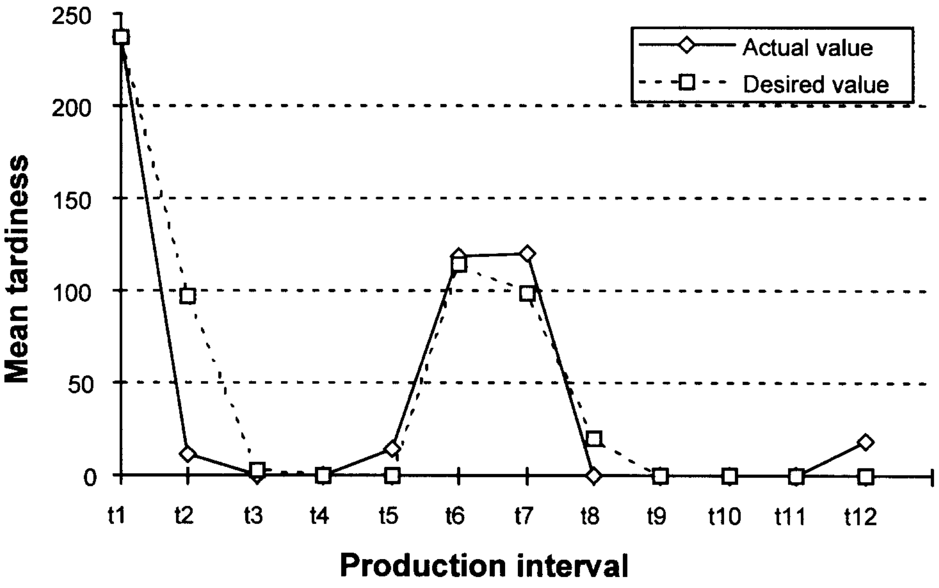

Fig. 4. Comparison result of the mean tardiness

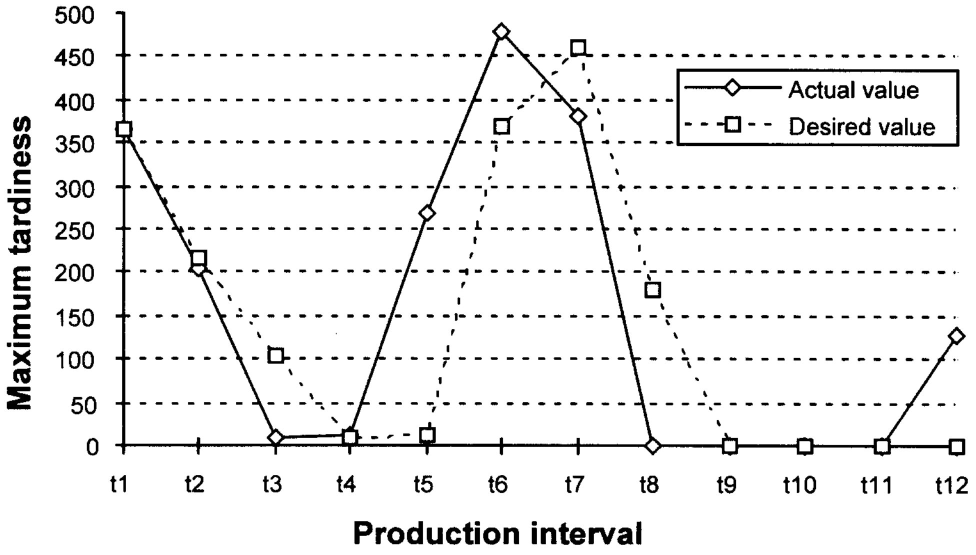

Fig. 5. Comparison result of the maximum tardiness

Since the knowledge can be retrieved once it is stored in computer, and can be modified without difficulty if necessary, the integration model is practical to apply to real FMSs. For a future research work, it is necessary to develop a systematic knowledge modification method so that the knowledge always reflects the dynamic operation conditions of the FMS.

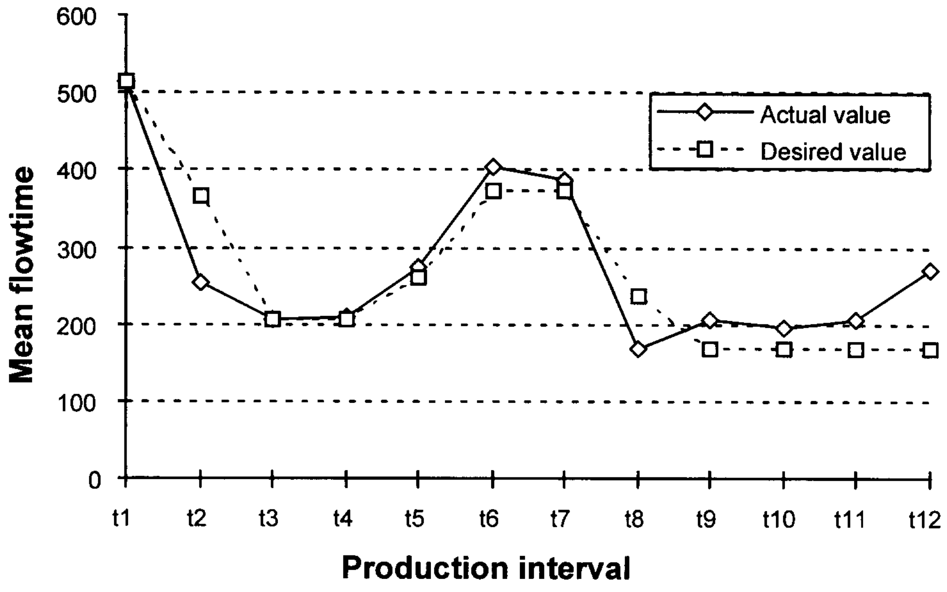

Fig. 6. Comparison result of the mean flow time

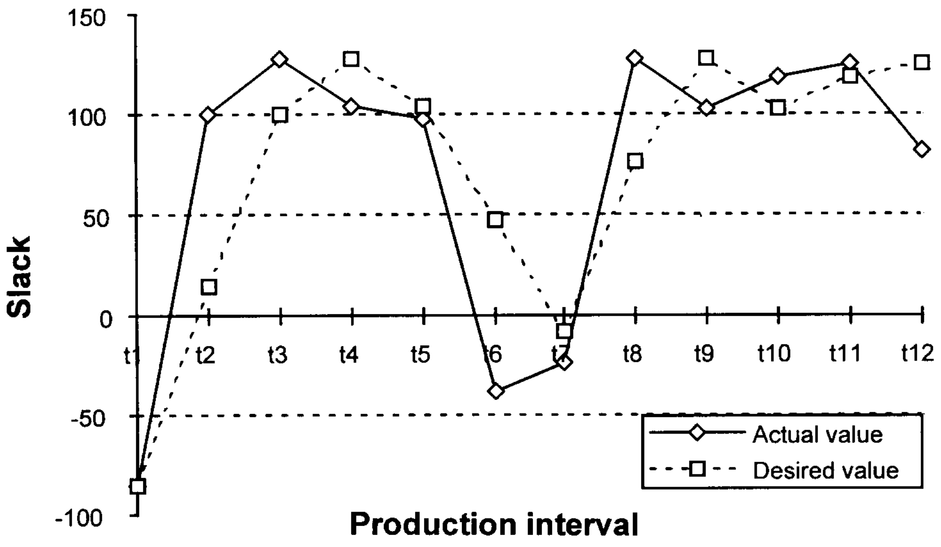

Fig. 7. Comparison result of the slack

References

Bharath, R., and J. Drosen, *Neural Network Computing*, Windcrest, 1994.

Chryssolouris, G., M. Lee, and M. Domroese, "The Use of Neural Network in Determining Operational Policies for Manufacturing Systems," *Journal of Manufacturing Systems*, Vol.10, (1994), 166-175.

Demuth, H. and M. Beale, Neural Network Toolbox, The Math Works, MA, 1995.

Fox, M. S. and S. F. Smith, "ISIS - A knowledge-Based System for Factory Scheduling," *Expert Systems*, Vol.1, (1984), 25-44.

Fox, M. S., "Constraint-Guided Scheduling - A Short History of Research at CMU," *Computers in Industry*, Vol.14, (1990), 79-88.

Nygard, K. E., P. Juell, and N. Kadaba, "Neural Network for Selecting Vehicle Routing Heuristics," *ORSA journal on Computing*, Vol.2, (1991), 353-364.

Philipoom, P. R., L. P. Rees, and L. Wiegmann, "Using Neural Networks to Determine Internally-Set Due-Date Assignments for Shop Scheduling," *Decision Sciences*, Vol.25, (1994), 825-851.

Quinlan, J. R., *C4.5 : Programs for Machine Learning*, Morgan Kaufmann Publishers, CA, 1993.

Quinlan, J. R., "Induction of Decision Trees," *Machine Learning*, Vol.1, (1986), 81-106.

Sadeh, N., K. Sycara, and Y. Xiong, "Backtracking Techniques for the Job Shop Scheduling Constraint Satisfaction Problem," *Artificial Intelligence*, Vol.76, (1995), 455-480.

Shaw, M. J., "A Pattern-Directed Approach to Flexible Manufacturing: A Framework for Intelligent Scheduling, Learning, and Control," *Int. Journal of Flexible Manufacturing Systems*, Vol.2, (1989), 121-144.

Shaw, M. J., J. A. Gentry, and S. Piramuthu, "Inductive Learning Methods for Knowledge-Based Decision Support: A Comparative Analysis," *Computer Science in Economics and Management*, Vol.3, (1990), 147-165.

Shaw, M. J., S. C. Park, and N. Raman, "Intelligent Scheduling with Machine Learning Capabilities : The Induction of Scheduling Knowledge," *IIE Transaction*, Vol.24, (1992), 156-168.

Sim, S. K., K. T. Yeo, and W. H. Lee, "An Expert Neural System for Dynamic Job Shop Scheduling," *Int. Journal of Production Research*, Vol.32, (1994), 1759-1773.

Yih, Y., "Trace-Driven Knowledge Acquisition (TDKA) for Rule-Based Real Time Scheduling Systems," *Journal of Intelligent Manufacturing*, Vol.1, (1990), 217-230.

Yih, Y., T. P. Liang, and H. Moskowitz, "A Hybrid Approach for Crane Scheduling Problems," *Intelligent Engineering Systems Through Artificial Neural Networks*, ASME, New York, (1991), 867-872.