World Academy of Science, Engineering and Technology
International Journal of Industrial and Manufacturing Engineering
Vol:8, No:3, 2014

Decision Tree Based Scheduling for Flexible Job Shops with Multiple Process Plans

H.-H. Doh, J.-M. Yu, Y.-J. Kwon, J.-H. Shin, H.-W. Kim, S.-H. Nam, D.-H. Lee

***Abstract*—**This paper suggests a decision tree based approach for flexible job shop scheduling with multiple process plans, i.e. each job can be processed through alternative operations, each of which can be processed on alternative machines. The main decision variables are: (a) selecting operation/machine pair; and (b) sequencing the jobs assigned to each machine. As an extension of the priority scheduling approach that selects the best priority rule combination after many simulation runs, this study suggests a decision tree based approach in which a decision tree is used to select a priority rule combination adequate for a specific system state and hence the burdens required for developing simulation models and carrying out simulation runs can be eliminated. The decision tree based scheduling approach consists of construction and scheduling modules. In the construction module, a decision tree is constructed using a four-stage algorithm, and in the scheduling module, a priority rule combination is selected using the decision tree. To show the performance of the decision tree based approach suggested in this study, a case study was done on a flexible job shop with reconfigurable manufacturing cells and a conventional job shop, and the results are reported by comparing it with individual priority rule combinations for the objectives of minimizing total flow time and total tardiness.

***Keywords*—**Flexible job shop scheduling, Decision tree, Priority rules, Case study.

I. Introduction

THIS study considers the scheduling problem in flexible job shops with multiple process plans, i.e. each job can be processed through alternative operations, each of which can be processed on alternative machines. The problem, which is an extension of the conventional job shop scheduling problem, can be found in various types of manufacturing systems, especially in flexible or reconfigurable manufacturing systems since each operation can be processed on one or more machines. Here, the reconfigurable manufacturing system is the one designed at the outset for rapid changes in its hardware/software components to quickly adjust its production capacity and functionality in response to sudden market changes or intrinsic system changes. In case that one or more reconfigurable manufacturing cells are introduced to a conventional job shop system for the purpose of increasing system capacity and flexibility, the resulting hybrid system becomes a flexible job shop.

Since the flexible job shop scheduling problem with multiple process plans is very complicated, the previous studies suggest meta-heuristics or report the performances of priority rule combinations. Here, the priority rule combinations are required since the scheduling problem has two main decisions: (a) selecting operation/machine pairs; and (b) sequencing the jobs assigned to each machine. See Ozguven et al. [1] and Doh et al. [2] for recent studies on flexible job shop scheduling problem with multiple process plans. Although there are some merits, the meta-heuristic and the priority scheduling approaches have disadvantages. First, the meta-heuristic approach requires too much computation time to be used in real-time environment. Also, the priority scheduling approach has the burdens required for developing simulation models and carrying out simulation runs to select the best rule for a specific system state.

In this study, we suggest a decision tree based approach for the flexible job shop scheduling problem in which the decision tree is used to select a priority rule combination appropriate for a specific system state and hence the burdens required for developing simulation models and carrying out simulation runs can be eliminated. In general, the decision tree, one of the data mining techniques, is a kind of classifier expressed as a recursive partition of the instance space. In this study, we adopt the decision tree for flexible job shop scheduling since it has several advantages, i.e. simple to understand and interpret, containing value even with little hard data, adding possible scenarios, etc. See Deng et al. [3] for more advantages of the decision tree.

There are several previous studies on production scheduling based on the decision tree. One of earlier studies is Shinichi and Taketoshi [4] that suggest a learning algorithm that generates a decision tree using the empirical data obtained by simulation runs, where the decision tree is used to decide the priority rule at decision points during the actual production operations. Also, Shaw et al. [5] suggest a scheduling framework to classify manufacturing patterns and to generate a decision tree that dynamically selects the priority rule for a given set of system attributes, and later, Piramuthu et al. [6] and Park et al. [7] applied the framework to flexible manufacturing systems after certain modifications. See Lee et al. [8], Arzi and Iaroslavitz [9], Su and Shiue [10], Kwak and Yih [11], Priore et al. [12], Shiue et al. [13] and Choi et al. [14] for other decision tree based scheduling approaches and applications.

As stated earlier, this study suggests a decision tree based scheduling approach for flexible job shops, especially with multiple process plans. In this approach, the decision tree is

Hyoung-Ho Doh, Jae-Min Yu, Yong-Ju Kwon, Jeong-Hoon Shin, and Hyung-Won Kim are with the Department of Industrial Engineering, Hanyang University, Seoul, Korea (e-mail: hojin81@gmail.com, tenaper@gmail.com, kylekwon@hanyang.ac.kr, boost7hooney@gmail.com, vnawlf1003@hanyang.ac.kr, respectively).

Sung-Ho Nam is with the Micro Machining Process Team, Korea Institute of Industrial Technology, Ansan, Korea (e-mail: goddad@kitech.re.kr).

Dong-Ho Lee is with the Department of Industrial Engineering, Hanyang University, Seoul, Korea (Corresponding author, phone: 82-2-2220-0475; fax: 82-2-2297-0475; e-mail: leman@ hanyang.ac.kr).

used to select a priority rule combination adequate for a specific system state and hence the burdens required for developing simulation models and carrying out simulation runs of the priority scheduling approach can be eliminated. The decision tree based scheduling approach suggested in this study consists of two main steps: construction and scheduling steps. In the construction step, a decision tree is constructed by a systematic procedure using the empirical data obtained by simulation runs, and in the scheduling module, a priority rule combination is selected under a specific system state using the decision tree. To show the performance of the decision tree based approach, a case study was done on a flexible job shop with reconfigurable manufacturing cells and a conventional job shop for each of the objectives of minimizing total flow time and total tardiness, and the results are reported by comparing it with individual priority rule combinations.

The paper is organized as follows. In the next section, the problem is described in more details. The decision tree based scheduling approach is explained in the third section, and the case study is reported in the fourth section. The final section concludes the paper with a summary and discussion of future research.

II. Problem Description

The flexible job shop scheduling problem considered in this study can be briefly explained as follows. For a given set of jobs, the problem is to determine the operation and machine pairs of each job and the sequence of the jobs assigned to each machine according to the process routes.

As explained earlier, each job is processed according to a multiple process plan that specifies alternative operations, their sequence, and alternative machines on which an operation is to be processed. In other words, each job can be processed through alternative operations, each of which can be processed on alternative machines. The decision variables are: (a) process route of each job, i.e. operation/machine selections; and (b) sequence of the jobs assigned to each machine, i.e. job shop scheduling. The two objectives are considered in this study. They are minimizing total flow time and total tardiness. Note that each objective is a function of job completion times.

This study considers a static and deterministic version of the problem. In other words, all jobs are ready for processing at time zero, i.e. zero ready times, and the job descriptors, such as multiple process plans, processing times, due dates, etc., are deterministic and given in advance. Other assumptions made for the problem are: (a) each machine can process only one operation at a time; (b) setup times are sequence-independent and hence can be included in processing times; (c) preemption is not allowed, i.e. once a job is processed on a machine, it will stay on that machine until its completion; (d) transportation times among machines are ignorable or can be included in processing times; and (e) due dates are not enforced as hard constraints. See Doh et al. [2] for more details.

III. Decision Tree Based Scheduling Approach

A. Decision Tree

As a data-mining technique, a decision tree is a kind of classifier represented as a recursive partition of the instance space. More specifically, a decision tree is a rooted one that consists of non-leaf and leaf nodes, where non-leaf nodes represent a choice among alternatives, i.e. splitting the instance space into two or more sub-spaces according to a certain discrete function of the input attributes values, while leaf nodes represent classification or decision.

Before explaining the decision tree in details, an example data set is summarized in Table I. In the table, there are twelve objects, four conditional attributes and one decision attribute. For example, object 1 implies that the decision is 1 (X = 1) if the values of conditional attributes A, B, C and D are 1, 2, 2, and 1, respectively.

TABLE I
Data Set for Constructing a Decision Tree: Example

Objects	Conditional attributes				Decision attribute
	A	*B*	*C*	*D*	*X*
1	1	2	2	1	1
2	1	2	3	2	1
3	1	2	2	3	1
4	2	2	2	1	1
5	2	3	2	2	2
6	1	3	2	1	1
7	1	2	3	1	2
8	2	3	1	2	1
9	1	2	2	2	1
10	1	1	3	2	1
11	2	1	2	2	2
12	1	1	2	3	1

Using the data given in Table I, various decision trees can be constructed. Fig. 1 shows an example in which a path from the root node to each lead node corresponds to a decision. For example, if the values of conditional attributes A, B and C are 2, 3, and 1, the resulting decision is 1, i.e., d = 1.

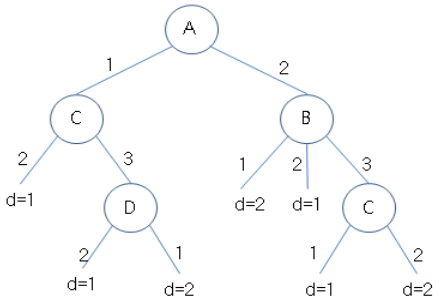

Fig. 1 Decision tree: example

B. Decision Tree Based Scheduling

Fig. 2 shows the decision tree based scheduling framework suggested in this study. As can be seen in the figure, the framework consists of storage, construction and scheduling, each of which is explained below.

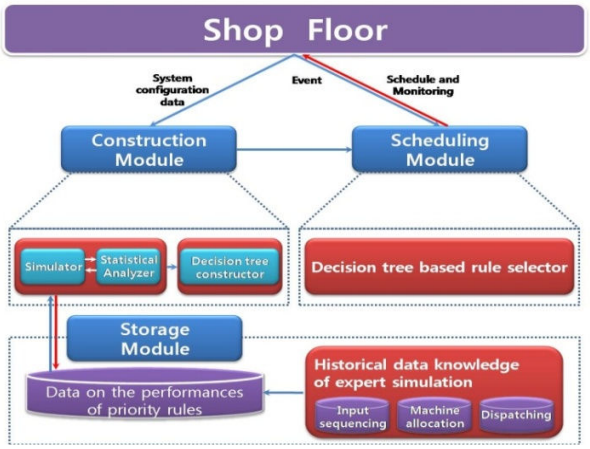

Fig. 2 Decision tree based scheduling: framework

1) Construction Module

In this module, a decision tree is built up using simulator, statistical analyzer and decision tree constructor. The simulator provides the performances of priority dispatching rules under various system states, which are used by the statistical analyzer that analyzes the data associated with constructing a decision tree. In our application, the conditional and decision attributes in the data set correspond to the system states and the selection of priority dispatching rule, respectively. If the simulation technique is used to construct a decision tree, an object in the data set, i.e., each row in Table I, is obtained by performing a simulation run under a given system state and identifying the best dispatching rule.

Based on the data obtained from simulation, the decision tree constructor builds up a decision tree that will be used to select a priority rule expected to perform the best under a specific system state. For this purpose, we suggest an algorithm that consists of four stages, each of which is explained below (The method is a modified version of Kwon et al. [15]).

Stage 1. Specifying the Attributes

In this stage, the system attributes are specified that may affect the system performance measures under consideration. For this purpose, various methods, e.g. knowledge of system experts, simulation, etc., can be used.

Stage 2. Selecting the Eligible Attributes

Among the specified system attributes in stage 1, the eligible ones, which are expected to be highly influential to the system performance under consideration, are selected. For this purpose, we suggest the correlation coefficient based method.

The detailed step-by-step procedure is given below.

Step 1. Calculate correlation coefficient value (r_a) between an attribute and the performance measure under consideration, i.e.

$$r_a = \sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y}) / \sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y_i - \bar{y})^2},$$

where x_i and y_i denote the ith values of attribute and performance measure, respectively.

Step 2. Classify the attributes as follows.

Group 1: Attributes with $0.7 < |r_a| \le 1.0$
Group 2: Attributes with $0.3 < |r_a| \le 0.7$
Group 3: Attributes with $0.1 < |r_a| \le 0.3$
Group 4: Attributes with $0 \le |r_a| \le 0.1$

Step 3. Determine the eligible attributes by selecting some higher ranked groups.

Stage 3. Determining the Number of Levels for Each Attribute

In this stage, the number of levels for each eligible attribute is determined using its correlation coefficient value. More formally, calculate the number of levels as

$$\left\lceil \left| |r_a| - \left(\left| \sum_{k=1}^{n_F} r_k \right| / n_F \right) \right| \times 10 \right\rceil,$$

where n_F denotes the total number of eligible attributes. Also, $\lceil \bullet \rceil$ denotes the smallest integer greater than or equal to $\bullet$.

Stage 4. Constructing a Decision Tree

In this stage, a decision tree is constructed using the data obtained from the simulation results on the performances of candidate priority rules under various system configurations, i.e. combinations of all the levels of eligible system attributes. Among various algorithms to generate a decision tree, we use the ID3 algorithm since it is simple but proved to be effective. See Quinlan [16] for more details of the ID3 algorithm.

The ID3 algorithm is based on the entropy function to select the conditional attributes of a decision tree, where the entropy function of conditional attribute j is defined as

$$entropy_j = \sum_{c=1}^{C_j} - p(w_{cj}|j) \cdot \log_2 p(w_{cj}|j),$$

where C_j denotes the number of different conditional attribute values, e.g., C_A, C_B, C_C, and C_D are, 2, 3, 3, and 3 for the example in Table I. Also, $p(w_{cj} | j)$ denotes the proportion of value w_{cj} in conditional attribute c. For example, $p(1|A) = 8/12$ and $p(2|B) = 4/12$ in Table I.

The detailed procedure of the ID3 algorithm is given below.

Step 1. Create the root node using the conditional attribute with the smallest entropy value and let the root node be the current node.

Step 2. For each conditional attribute value of the current node, create and connect a child node whose conditional attribute is set to the one with the smallest entropy value after updating the data set, i.e. entropy values are calculated after removing the conditional attribute of the current node and the objects with the conditional attribute value of the current node.

Step 3. If all conditional attributes are considered, stop. Otherwise, let one of the unconsidered child nodes be the current node and go to Step 2.

2) Scheduling Module

In this module, a priority rule is selected using the decision tree. The decision tree can be used in two ways. First, for the static flexible job shop scheduling problem considered in this study, the decision tree is used once at the beginning of scheduling period. On the other hand, if the decision tree is used for dynamic scheduling, the decision tree can be used to select a priority rule when the current rule needs to be changed due to system changes. Here, there may be various rescheduling strategies that determine the points of time when the current priority rule is to be changed.

3) Storage Module

This module collects and stores the date required for the construction and scheduling modules, e.g. simulation results or historical data on the performances of priority rules in order to construct the decision tree or perform rescheduling.

IV. Computational Experiments

The performance of the decision tree based scheduling approach was tested using the data on a flexible job shop case, and the results are reported in this section.

As explained earlier, the flexible job shop considered in this study consists of reconfigurable manufacturing cells (RMCs) and a conventional job shop. The RMC, a state-of-the-art manufacturing technology that overcomes the limitations of flexible manufacturing systems, is the one designed for rapid changes in its hardware and software components to quickly adjust its production capacity and functionality. See Koren et al. [17] for more details on reconfigurable manufacturing systems. Also, the job shop is a conventional legacy system that consists of dedicated and flexible machines, such as marking machines, numerical control machines, cleaning machines, etc. Note that when RMCs are introduced to a conventional job shop, the resulting hybrid system becomes a type of flexible job shop. Here, an RMC can be utilized as an alternative processor that can replace the conventional job shop. Therefore, the hybrid system can be considered as a parallel system in which the operations can be done on either the RMC or the job shop. However, the two systems are different in operations and processing times even for the same part type.

The RMC consists of numerical control (NC) machines, a loading/unloading (L/U) station and a central buffer. Each machine has an automatic tool changer and a tool magazine with limited capacities. A part can be fed into the RMC through the loading/unloading station after it is clamped onto a pallet with a required fixture type. Note that common pallets are used in the RMC, i.e., any fixture types can be mounted on a pallet. Also, a fixture type can be used for a specific set of part types. One or more tools are required to perform an operation on a part type, and each tool requires one or more slots in the tool magazine. The central buffer which is an automatic storage and retrieval system (AS/RS) is used to store in-process parts within the RMC. Since the RMC has a limited central buffer, an upper limit is imposed on the number of parts circulating in the system. After released into the RMC, a part with a required fixture type on a pallet goes into the central buffer and waits for processing. Each part stored in the central buffer is sent to the machines for operations. After the required operations are finished, the part leaves the system through the L/U station and removed from the pallet together with the fixture. Table II summarizes the system components.

TABLE II
System Components

	M/C Code	Description
RMC 1	RVMC1A	Vertical Machining Center (RMC1)
	RVMC1B	Vertical Machining Center (RMC1)
	RVMC1C	Vertical Machining Center (RMC1)
	RLU	Loading/Unloading station
RMC 2	RVMC2A	Vertical Machining Center (RMC2)
	RVMC2B	Vertical Machining Center (RMC2)
	RVMC2C	Vertical Machining Center (RMC2)
	RLU	Loading/Unloading station
RMC 3	RHMC1A	Horizontal Machining Center (RMC3)
	RHMC1B	Horizontal Machining Center (RMC3)
	RHMC1C	Horizontal Machining Center (RMC3)
	RLU	Loading/Unloading station
RMC 4	RHMC2A	Horizontal Machining Center (RMC4)
	RHMC2B	Horizontal Machining Center (RMC4)
	RHMC2C	Horizontal Machining Center (RMC4)
	RLU	Loading / Unloading station
Job shop (legacy)	MK1	Marking Machine 1
	MK2	Marking Machine 2
	VMC1	Vertical Machining Center 1
	VMC2	Vertical Machining Center 2
	HMC1	Horizontal Machining Center 1
	HMC2	Horizontal Machining Center 2
	HMC3	Horizontal Machining Center 3
	CFM	Cubic Face Milling Machine
	HFMCMM	Horizontal Face Milling/Cutting & Measuring Machine
	GDM	Grinding Machine
	DRGD	Drilling & Grinding Machine
	INS	Inspection
	CLM	Cleaning

It is assumed that part types and their quantities to be produced during the upcoming period are given in advance, and each part type is produced requires a predetermined set of operations. Table III lists all the operations, together with the machines that each operation can be processed.

TABLE III
OPERATIONS AND AVAILABLE MACHINES

Operations	Description	Available machine
OMK	Marking	MK1, MK2
OVR	Vertical Removing	RVMC1A, RVMC1B, RVMC1C RVMC2A, RVMC2B, RVMC2C VMC1, VMC2, CFM, GDM
OHR	Horizontal Removing	RHMC1A, RHMC1B, RHMC1C RHMC2A, RHMC2B, HMC2C HMC1, HMC2, CFM, FMCMM
OCR	Cubic Removing	CFM
OFC	Face Cutting	RVMC1A, RVMC1B, RVMC1C RVMC2A, RVMC2B, RVMC2C
OHFM	Horizontal Face Milling	RHMC1A, RHMC1B, RHMC1C RHMC2A, RHMC2B, MK1
OCFM	Cubic Face Milling	CFM
OINS	Inspection	HFMCMM, INS
OHFMCM	Horizontal Face Milling/Cutting & Measuring	HFMCMM
OGM	Grinding Mark	MK2
ODR	Drilling	VMC1, VMC2
OGD	Grinding	GDM , DRGD
ODRGD	Drilling & Grinding	DRGD

It is assumed that a loading plan is given to specify the assignments of operations and their cutting tools on the machines. See Kim et al. [18] for more details on the loading problem. Also, it is assumed that fixture allocation is done in advance, i.e., the given set of common pallets is divided into mutually exclusive subsets, within which the pallets are equipped with a predetermined fixture type. Finally, we assume that the number of fixtures is enough to clamp parts on pallets.

As explained earlier, each job is processed according to a multiple process plan, where the multiple process plan can be represented as a network that consists of three types of nodes: source, intermediate and sink. The source and sink are dummy nodes that represent the start and the end of a part processing, respectively. Each intermediate node represents alternative machines and operations, together with the corresponding processing times. Also, an arc connecting two nodes represents the precedence relation between the corresponding two operations. In particular, if a part meets an OR relation, it must select one of the corresponding alternative operation/machine pairs. In summary, a part is completed through a path (set of intermediate nodes) from the source to the sink node. Fig. 3 shows an example of the network model for a multiple process plan with 4 OR relations and 15 intermediate nodes. In this figure, we can see that there are five paths from the source to the sink node, e.g. S1–OMK–OVR–OCFM–OINS–F1, where the operation OMK can be processed by either MK1 or MK2 whose processing times are 30 and 60, respectively.

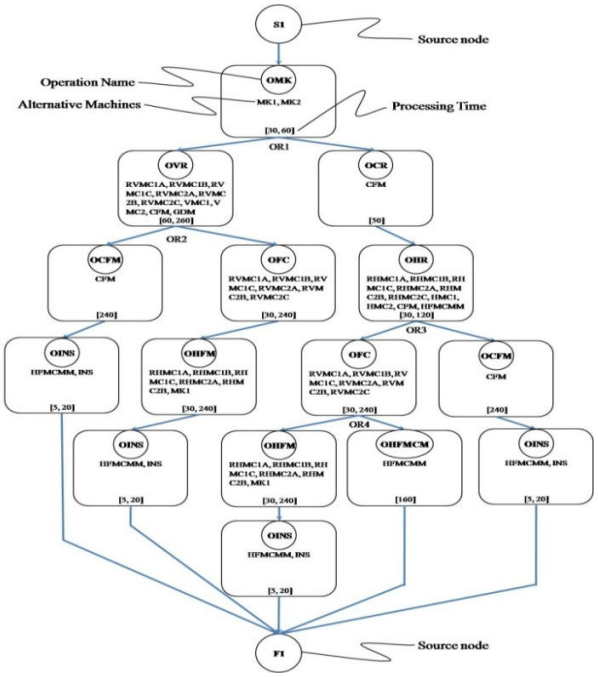

Fig. 3 Multiple process plan: example

Since the RMC is being developed, we could not obtain the real data on part types. Instead, we generated various data based on the experiences of the project partners. More specifically, we generated 5000 instances. In the instances, the number of part types and the production quantities were generated from *DU*(5, 15) and *DU*(1, 10), where *DU*(*a*, *b*) denotes the discrete uniform distribution with a range [*a*, *b*]. Also, the number of operations for each part type and the number of alternative operation/machine pairs were generated from *DU*(10, 32) and *DU*(1, 10). Also, multiple process plans for each part type was generated randomly in order to consider various process routing configurations. The processing times were generated from *DU*(30, 100). Finally, the capacity of the central buffer is 36 and the available number of pallets (with fixtures) was generated from *DU*(5, 20).

In the decision tree based scheduling approach, the decision tree was constructed using the four-stage algorithm explained earlier, where the eligible attributes selected by the correlation coefficient based method for the problem of minimizing the total flow time are summarized below. (Initially, we identified 6 static attributes.)

- Number of part types (for total flow time and tardiness)
- Production quantity (for total flow time and tardiness)
- Processing time (for total flow time and tardiness)

Also, to show the performance of the four-stage algorithm, we constructed another decision tree using the C5.0 algorithm. For this purpose, we used a commercial software package. The detailed decision trees are not represented here due to its size.)

In the test, the decision tree based scheduling approach is compared with 216 priority rule combinations, where the rule combinations are those of 4 input sequencing rules (SPPT, LPPT, SRF/TF and LRF/TF), 3 operation/machine selection rules (SQ, SW and SP) and 18 part sequencing rules (FIFO,

SOPT, WINQ, LWKR, LOPR, SJPT, EDD, CR, ATC, COVERT, MDD, MST, P-FIFO, P-SOPT, P-WINQ, P-LWKR, P-LOPR and P-SJPT). See Doh et al. [19] and Yu et al. [20] for the detailed descriptions of the priority rules tested. (The detailed descriptions are skipped due to the space limitation.) For evaluation of the results, we use the relative performance ratio because we could not obtain the optimal solutions. Here, the relative performance ratio for a test instance is defined as

$$100 \cdot (C_a - C_{best}) / C_{best},$$

where C_a is the objective value obtained using rule combination a for the instance and C_{best} is the best objective value among those obtained from the 216 rule combinations, the four-stage algorithm and the C5.0 algorithm.

Test results are summarized in Fig. 4 that show the average relative performance ratios of the priority rule combinations, the four-stage algorithm and the C5.0 algorithm out of 5000 test instances. In the figure, the x- and y-axis represent the methods and the relative performance ratios, respectively. As can be seen in the graph, the decision tree based scheduling approach (four-stage and the C5.0 algorithms) outperforms the best rule combinations and particularly, gives stable performance. Also, of the two decision tree construction algorithms, the four-stage algorithm suggested in this study was slightly worse than the C5.0 algorithm (0.004%). However, the four-stage algorithm gave lighter decision tree with less number of eligible attributes than the C5.0 algorithm.

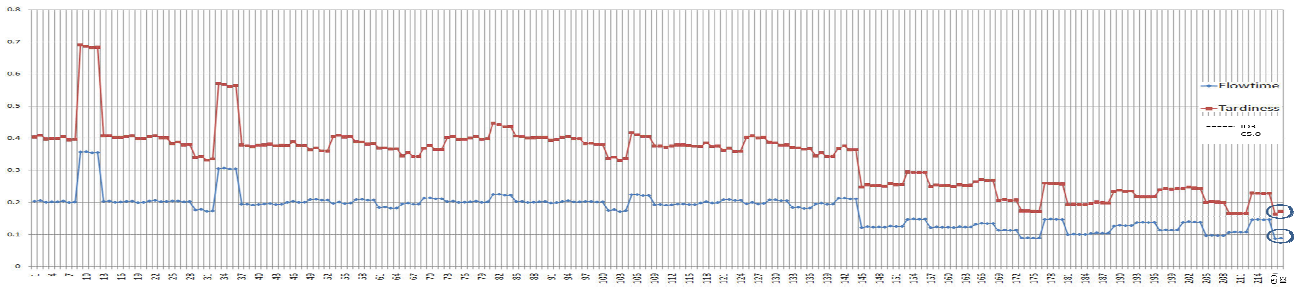

Fig. 4 Performance of decision tree: graphical representation

V. Conclusion

In this study, we suggested a decision tree based approach for flexible job shop scheduling with multiple process plans. The main decisions are selecting operation/machine pair and sequencing the jobs assigned to each machine. The decision tree is used to select a priority rule appropriate for a specific system state and hence the burdens required for developing simulation models and carrying out simulation runs can be eliminated. In the decision tree based scheduling approach, a four-stage algorithm was suggested to construct a decision tree. To show the performance of the decision tree based scheduling approach, a case study was done on a flexible job shop with reconfigurable manufacturing cells and a conventional job shop and the results showed that the decision tree based scheduling approach outperforms the simple priority rule combination based approach that requires simulation runs for each of the objectives of minimizing total flow time and total tardiness.

This study can be extended in several directions. First, it is needed to consider the dynamic flexible job shop scheduling in which jobs arrive over time, not given in advance. In this case, the four-stage decision tree construction algorithm can be used after other dynamic attributes are identified. Second, more case studies for other shop configurations are worth to be performed. Finally, this study can be extended to a decision tree based real-time scheduling mechanism with rescheduling strategies, decision tree update methodologies, etc.

Acknowledgment

This work was supported by the Ministry of Knowledge Economy (MKE) grant funded by Korea government (Grant Code: 10033895-2009-11). This is gratefully acknowledged.

References

[1] C. Ozguven, L. Ozbakir, and Y. Yavuz, "Mathematical models for job-shop scheduling problems with routing and process plan flexibility," *Applied Mathematical Modelling*, vol. 34, pp. 1539–1548, 2010.

[2] H.-H. Doh, J.-M. Yu, J.-S. Kim, D.-H. Lee, and S.-H. Nam, "A priority scheduling approach for flexible job shops with multiple process plans," *International Journal of Production Research.* vol. 51, pp. 3748-3764, 2013.

[3] H. Deng, G. Runger, and E. Tuv, "Bias of importance measures for multi-valued attributes and solutions," *Proceedings of the 21st International Conference on Artificial Neural Networks*, pp.293-300.

[4] N. Shinichi and Y. Taketoshi, "Dynamic scheduling system utilizing matching learning as a knowledge acquisition tool," *International Journal of Production Research*, vol. 30, pp. 411-431, 1992.

[5] M. J. Shaw, S. Park and N. Raman, "Intelligent scheduling with machine learning capabilities: the induction of scheduling knowledge," *IIE Transactions*, vol. 24, pp. 156-168, 1992.

[6] S. Piramuthu, N. Raman and M. J. Shaw, "Learning-based scheduling in a flexible manufacturing flow line," *IEEE Transactions on Engineering Management*, vol. 41, pp. 172-182, 1994.

[7] S.-C. Park, N. Raman, and M. J. Shaw, "Adaptive scheduling in dynamic flexible manufacturing systems: a dynamic rule selection approach," *IEEE Transactions on Robotics and Automation*, vol. 13, pp.486-502, 1997.

[8] C.-Y. Lee, S. Piramuthu, and Y.-K. Tsai, "Job shop scheduling with a genetic algorithm and machine learning," *International Journal of Production Research*, vol. 35, pp.1171–1191, 1997.

[9] Y. Arzi and L. Iaroslavitz, "Operating an FMC by a decision-tree-based adaptive production control system," *International Journal of Production Research*, vol. 38, pp.675-697, 2000.

[10] C. T. Su and Y. R. Shiue, "Intelligent scheduling controller for shop floor control systems: a hybrid genetic algorithm/decision tree learning approach," *International Journal of Production Research*, vol. 41, pp.2619-2641, 2003.

[11] C. Kwak and Y. Yie, "Data mining approach to production control in the computer integrated testing cell," *IEEE Transactions on Robotics and Automation*, vol. 20, pp.107-116, 2004.

[12] P. Priore, D. De La Fuente, J. Puente, and J. Parreno, "A comparison of machine learning algorithms for dynamic scheduling of flexible manufacturing systems, *Engineering Applications of Artificial Intelligence*, vol. 19, pp.247-255, 2006.
[13] Y.R. Shiue, R. S. Guh, and T. Y. Tseng, "GA-based learning bias selection mechanism for real-time scheduling systems," *Expert Systems with Applications*, vol. 36, pp.11451-11460, 2009.
[14] H.-S. Choi, J.-S. Kim, and D.-H. Lee, "Real-time scheduling for reentrant hybrid flow shops: a decision tree based mechanism and its application to a TFT-LCD line," *Expert Systems with Applications*, Vol. 38, pp. 3514-3521, 2011.
[15] Y.-J. Kwon, J.-M. Yu, H.-H. Doh, S.-H. Nam and D.-H. Lee, "Decision tree based real-time scheduling: a decision tree construction method," *Proceedings of the International Symposium on Green Manufacturing and Applications, Honolulu, USA,* 2013.
[16] Quinlan, J. R. "Introduction of decision trees," *Machine Learning*, Vol. 1, pp. 80-106, 1986.
[17] Y. Koren, U. Heisel, F. Jovane, T. Moriwaki, G. Pritschow, G. Ulsoy, and H. Brussel, "Reconfigurable manufacturing systems," *Annals of the CIRP*, vol.48, pp.527–540, 1999.
[18] H.-W. Kim, J.-M. Yu, J.-S. Kim, H.-H. Doh, D.-H. Lee, and S.-H Nam, "Loading algorithms for flexible manufacturing systems with partially grouped unrelated machines and additional tooling constraints," *International Journal of Advanced Manufacturing Technology*, vol. 58, pp. 683-691, 2012.
[19] H.-H. Doh, J.-M. Yu, J.-S. Kim, D.-H. Lee, and S.-H. Nam, "A Priority Scheduling Approach for Flexible Job Shops with Multiple Process Plans," *International Journal of Production Research*, vol. 51, pp. 3748-3764, 2013.
[20] J.-M. Yu, H.-H. Doh, J.-S. Kim, Y.-J. Kwon, D.-H. Lee, and S.-H. Nam, "Input Sequencing and Scheduling for a Reconfigurable Manufacturing System with a Limited Number of Fixtures," *International Journal of Advanced Manufacturing Technology*, vol. 67, pp. 157-169, 2013.