International Journal of Innovative
Computing, Information and Control
Volume **8**, Number **6**, June **2012**

 ISSN 1349-4198
pp. **4025–4044**

# A FUZZY-NEURAL DBD APPROACH FOR JOB SCHEDULING IN A WAFER FABRICATION FACTORY

Toly Chen

Department of Industrial Engineering and Systems Management
Feng Chia University
No. 100, Wenhwa Rd., Seatwen, Taichung 40724, Taiwan
tolychen@ms37.hinet.net

Received March 2011; revised July 2011

Abstract. *Job scheduling is an important but difficult task to wafer fabrication factories. To further improve the performance of job scheduling in a wafer fabrication factory, a fuzzy-neural dynamic-bottleneck-detection (DBD) approach is proposed in this study. The fuzzy-neural DBD approach is modified from the traditional DBD approach after incorporating some major changes. First, taking into account the uncertainty of job classification, fuzzy partition is applied to divide jobs into different categories. Second, the fuzzy c-means and fuzzy back propagation network (FCM-FBPN) approach is applied to estimate the remaining cycle time of a job. Third, we replace the heuristics in the traditional DBD approach, with more advanced and flexible dispatching rules, such as the shortest cycle time until next bottleneck (SCNB) rule and the four-factor bi-criteria nonlinear fluctuation smoothing (4f-biNFS) rule. A real wafer fabrication factory is also simulated as a testing environment for the adoption of several methods. According to the experimental results, the fuzzy-neural DBD approach was better than six existing approaches and their variants in reducing the average cycle time and cycle time standard deviation at the same time.*

**Keywords:** Wafer fabrication, Scheduling, Dispatching rule, Fuzzy neural, Dynamic bottleneck detection

1. **Introduction.** Semiconductor manufacturing process is usually divided into four phases: wafer fabrication, wafer probe, assembly, and final testing. Among them, the longest one is wafer fabrication. The production system required for wafer fabrication is very complex and difficult to control [1]. Every job in a wafer fabrication factory is composed of 20-25 wafers, and has hundreds of steps to undergo. These processing steps can be divided into several categories, such as photolithography, etch, strip. Therefore, the same operation can be performed on the job many times. In other words, a job needs to access the same group of workstations more than once for the same operation, which is typical of the re-entrant production system. The characteristics make job scheduling in a wafer fabrication factory a very challenging task. In addition, the cycle time to complete all operations in a wafer fabrication factory is usually several months, resulting in the accumulation of work-in-progress (WIP). For this reason, cycle time reduction is an important task of job scheduling in a wafer fabrication factory.

To this end, a good release policy has been considered to be the most effective method [2]. However, in most wafer fabrication factories, in particular, foundry factories, the order-related jobs must be released as soon as possible after the receipt of the order. In addition, many studies have confirmed that the use of the existing dispatching rules (such as first-in first out (FIFO), earliest due date (EDD), least slack (LS), shortest processing time (SPT), shortest remaining processing time (SRPT), critical ratio (CR), FIFO+, SRPT+, and SRPT++) to a wafer fabrication factory does not produce very good results. Tie breaking is another issue. Nevertheless, job scheduling in a wafer fabrication factory has become a very important issue [3]. However, Chen [4] pointed out the inadequacies of the existing dispatching rules. First of all, most dispatching rules in this field consider only the attributes of the jobs gathered at the same place, and lack an effective way for taking the conditions of the factory as a whole into consideration. Second, most dispatching rules are not tailored to a particular wafer fabrication factory. Third, most dispatching rules are deterministic and do not reflect the changes in a wafer fabrication factory. Although there are a few dispatching rules incorporating stochastic variables, such as the fluctuation smoothing (FS) rules – fluctuation smoothing policy for variance of cycle time (FSVCT) and fluctuation smoothing policy for mean cycle time (FSMCT), they use the average values for these stochastic variables, and are as such not responsive to environmental changes. Fourth, most of the dispatching rules have not been optimized. Some studies used response surface method (RSM) and the desirability function to handle multiple-factor, multiple-objective optimization [5]. However, most of these studies applied second-order multiple regression, which may not be accurate enough. The desirability function is also a very subjective approach. At last, the dispatching rules are focused on a single performance measure. In theory, single-objective optimization wafer fabrication factory scheduling problem is a strongly NP-hard problem. Nevertheless, optimizing multiple targets at the same time is still being pursued.

Recently, a few studies (e.g., Koonce and Tsai [6]) have suggested data-mining-based approaches that attempt to simulate the best practices in the past for future applications. However, a wafer fabrication factory is a highly dynamic environment in which future conditions might be very different from those in the past. It is also very difficult to find the socalled best practices for such a highly dynamic and complicated production system. Another way is to combine some existing rules, and every time to pick only the most suitable one. For example, Hsieh et al. [7] used five approaches including FSMCT, FSVCT, largest deviation first (LDF), one step ahead (OSA), and FIFO jointly. However, each time an extensive simulation experiment is required to estimate the performance of each candidate in order to determine the most suitable one. Moreover, the transition from one approach to another is radical.

Agent technologies have also been applied. For example, Yoon and Shen [8] constructed a multiple-agent system for scheduling a wafer fabrication factory, in which four types of agents (scheduling agents, work cell agents, machine agents, and product agents) were designed and developed. The optimal scheduling plan was found by the scheduling agent through enumerating some possible scenarios. However, the batch production commonly used in a wafer fabrication factory was not considered in their study and therefore the case might be impractical. Youssef et al. [9] proposed a hybrid genetic algorithm (GA) and data mining approach to determine the best scheduling plan of a jobshop, in which the GA was used to generate a learning population of good solutions. These good solutions were then explored to find a number of decision rules that could be transformed into a meta-heuristic. Koonce and Tsai [6] proposed a similar approach. Sourirajan and Uzsoy [10] proposed a rolling horizon (RH) heuristic that breaks down the factory into smaller sub-problems that can be resolved over time in order using a workcenter-based decomposition heuristic.

This study is dedicated to propose a better dispatching rule. The motivation of this study is explained as follows. Recently, some sophisticated dispatching rules have been proposed. For example, Chen [11] modified the traditional FSMCT rule, and proposed a nonlinear FSMCT (NFSMCT) rule, in which the fluctuation in the remaining cycle time estimate is smoothed, and then its influence is balanced with that of the release time or the mean release rate. Subsequently, the difference in the slack is magnified using the 'division' operator. Subsequently, Chen [4] proposed the one-factor tailored nonlinear FSMCT (1f-TNFSMCT) rule and the one-factor tailored nonlinear FSVCT (1f-TNFSVCT) rule including an adjustable parameter to customize the rules for the wafer fabrication factory. Taking into account two performance measures (average cycle time and cycle time variation) at the same time, Chen and Wang [12] proposed a bi-criteria nonlinear fluctuation smoothing rule that also has an adjustable factor (1f-biNFS). To increase the flexibility of customization, Chen et al. [13] extended the above rules, and proposed the bi-criteria fluctuation smoothing rule with four adjustable factors (4f-biNFS). However, the adjustment factors in these rules are static. In other words, they will not change over time. Chen [14] therefore designed a mechanism to dynamically adjust the values of the factors in Chen and Wang's bi-criteria nonlinear fluctuation smoothing rule (dynamic 1f-biNFS). However, the adjustment of the factors is based on a pre-defined rule. This process is too subjective, and does not also take into account the status of the wafer fabrication factory. In addition, these rules have not been optimized, so there is considerable room for improvement.

To further improve the performance of job scheduling in a wafer fabrication factory, a fuzzy-neural dynamic-bottleneck-detection (DBD) approach is proposed in this study. DBD is very special, because it divides jobs into several categories, and uses four heuristics jointly, which allows considerable space for improvement. The fuzzy-neural DBD approach is modified from the DBD approach proposed by Zhang et al. [5]. The unique features of the proposed methodology include:

(1) Jobs are divided into several categories with fuzzy partition, instead of the tradition crisp partition that may cause some problems.
(2) Some heuristics in the traditional DBD can be replaced with more advanced, flexible dispatching rules including the shortest cycle time until next bottleneck (SCNB) rule and 4f-biNFS. As 4f-biNFS will be used by most job categories in the fuzzy-neural DBD approach, the scheduling performance of the fuzzy-neural DBD approach is expected to be at least close to that of 4f-biNFS.
(3) We estimate the remaining cycle time of a job with the fuzzy c-means and fuzzy back propagation network (FCM-FBPN) approach [15]. According to Chen and Wang [16], with more accurate remaining cycle time estimation, the scheduling performance of a fluctuation smoothing rule can be significantly improved.

Two performance measures, the average cycle time and cycle time standard deviation, are considered. To evaluate the effectiveness of the proposed methodology, production simulation is also applied in this study. The remainder of this paper is organized as follows. Section 2 is divided into two parts. The first part describes the application of FCM-FBPN to estimate the remaining cycle time of a job. The fuzzy-neural DBD approach is then detailed in the second part. To evaluate the effectiveness of the proposed methodology, a real wafer fabrication factory is simulated in Section 3 as a test environment. The fuzzy-neural DBD approach and some existing methods are then applied to schedule jobs in the simulated wafer fabrication factory in Section 4. Some discussion points are also made. Finally, the concluding remarks with a view to the future are given in Section 5.

## 2. **Methodology.** The variables are defined as follows.

(1) $\lambda$: the release rate.
(2) $R_i$: the release time of job $i$.
(3) $U_i$: the average factory utilization at $R_i$.
(4) $UN_i$: the average utilization of the next workstation of job $i$.

(5) $Q_i$: the total queue length on the processing route of job $i$ at $R_i$.
(6) $QN_i$: The average utilization of the next workstation of job $i$.
(7) $BQ_i$: the total queue length before the bottlenecks at $R_i$.
(8) $FQ_i$: the total queue length in the whole factory at $R_i$.
(9) $PT_i$: the processing time of job $i$ at processing step $k$.
(10) $WIP_i$: the factory work-in-progress (WIP) at $R_i$.
(11) $D_i^{(m)}$: the delay of the $m$-th recently completed job at $R_i$, $m = 1 \sim 3$.
(12) $CTE_i$: the estimated cycle time of job $i$.
(13) $CT_i$: the cycle time (actual value) of job $i$.
(14) $ICTE_{ikf}$: the estimated interval cycle time from step $k$ to step $f$ of job $i$.
(15) $SCTE_{ik}$: the estimated step cycle time of job $i$ until processing step $k$.
(16) $SCT_{ik}$: the step cycle time (actual value) of job $i$ until processing step $k$.
(17) $RCTE_{ik}$: the estimated remaining cycle time of job $i$ at processing step $k$.
(18) $RCT_{ik}$: the remaining cycle time (actual value) of job $i$ at processing step $k$.

2.1. **Step 1: estimating the remaining cycle time with FCM-FBPN.** In the proposed methodology, the remaining cycle time of a job needs to be estimated, for this purpose the FCM-FBPN approach proposed by Chen et al. [15] is applied. Unlike [15], the FCM-FBPN approach is applied to estimate both the remaining cycle time and interval cycle time of a job, as the two times are used in the fuzzy-neural DBD approach. With more accurate remaining/interval cycle time estimation, the fuzzy-neural DBD approach is expected to achieve a better scheduling performance.

In the FCM-FBPN approach, jobs (examples) are pre-classified into $K$ categories with FCM before they are fed into the FBPNs. FCM performs classification by minimizing the following objective function:

$$\text{Min} \sum_{k=1}^{K} \sum_{i=1}^{n} \mu_{i(k)}^{m} e_{i(k)}^{2} \tag{1}$$

where $K$ is the required number of categories; $n$ is the number of examples; $\mu_{i(k)}$ represents the membership of example $i$ belonging to category $k$; $e_{i(k)}$ measures the distance from example $i$ to the centroid of category $k$; $m \in (1, \infty)$ is a parameter to increase or decrease the fuzziness. The procedure of applying FCM to classify examples is

(1) Establish an initial classification result.
(2) (Iterations) Obtain the centroid of each category as

$$\bar{x}_{(k)} = \{\bar{x}_{(k)j}\} \tag{2}$$

$$\bar{x}_{(k)j} = \sum_{i=1}^{n} \mu_{i(k)}^{m} x_{ij} \Bigg/ \sum_{i=1}^{n} \mu_{i(k)}^{m} \tag{3}$$

$$\mu_{i(k)} = 1 \Bigg/ \sum_{l=1}^{K} (e_{i(k)}/e_{i(l)})^{2/(m-1)} \tag{4}$$

$$e_{i(k)} = \sqrt{\sum_{all\, j} (x_{ij} - \bar{x}_{(k)j})^2} \tag{5}$$

where $x_{ij}$ indicates the $j$th parameter in $\left\{LS_i, U_i, Q_i, BQ_i, WIP_i, D_i^{(1)}, D_i^{(2)}, D_i^{(3)}\right\}$ of example $i$; $\bar{x}_{(k)}$ is the centroid of category $k$. Note that the buckets of an example are not considered in classifying the example.
(3) Re-measure the distance of each example to the centroid of every category, and then recalculate the corresponding membership.

(4) Stop if the following condition is satisfied. Otherwise, return to Step (2):

$$\max_k \max_i \left| \mu_{i(k)}^{(t)} - \mu_{i(k)}^{(t-1)} \right| < d \tag{6}$$

where $\mu_{i(k)}^{(t)}$ is the membership of example $i$ belonging to category $k$ after the $t$th iteration; $d$ is a real number representing the threshold of membership convergence.

Finally, the separate distance test ($S$ test) proposed by Xie and Beni [17] can be applied to determine the optimal number of categories $K$:

$$\text{Min} \quad S \tag{7}$$

$$\text{s.t.} \quad J_m = \sum_{k=1}^{K} \sum_{i=1}^{n} \mu_{i(k)}^{m} e_{i(k)}^{2}, \tag{8}$$

$$e_{\min}^{2} = \min_{p \neq q} \left( \sum_{all\, j} (\bar{x}_{(p)j} - \bar{x}_{(q)j})^2 \right), \tag{9}$$

$$S = \frac{J_m}{n \times e_{\min}^{2}}, \tag{10}$$

$$K \in Z^+$$

The $K$ value minimizing $S$ determines the optimal number of categories.

Subsequently, examples of different categories are then learned with different FBPNs but with the same topology. The procedure for determining the parameter values of the FBPN is described as follows. The configuration of the FBPN is established as follows:

(1) Inputs: 8 parameters associated with the $n$th example/job including $LS_i$, $U_i$, $Q_i$, $BQ_i$, $WIP_i$, $D_i^{(1)}$, $D_i^{(2)}$, and $D_i^{(3)}$. These parameters have to be normalized so that their values fall within $[0, 1]$. Then some production execution/control experts are requested to express their beliefs (in linguistic terms) about the importance of each input parameter in estimating the cycle time (or step cycle time) of a job. Linguistic assessments for an input parameter are converted into several pre-specified fuzzy numbers. The subjective importance of an input parameter is then obtained by averaging the corresponding fuzzy numbers of the linguistic replies for the input parameter by all experts. The subjective importance obtained for an input parameter is multiplied to the normalized value of the input parameter. After such a treatment, all inputs to the FBPN become fuzzy numbers.

(2) Single hidden layer: Generally one or two hidden layers are beneficial for the convergence property of the FBPN.

(3) Number of neurons in the hidden layer: $1 \sim 16$. The computing efficiency decreases rapidly if the scale of the FBPN (including the number of the hidden-layer nodes) increases. Nevertheless, a large number of the hidden-layer nodes are theoretically beneficial to the estimation accuracy. For these reasons, the optimal number of the hidden-layer nodes in the FBPN is chosen from the interval $[1, 16]$ in the proposed methodology.

(4) Output: The (normalized) estimated cycle time (or step cycle time) of the example.

(5) Network learning rule: Delta rule.

(6) Transformation function: Sigmoid function,

$$f(x) = \frac{1}{1 + e^{-x}} \tag{11}$$

(7) Learning rate ($\eta$): $0.01 \sim 1.0$.

(8) Batch learning.

(9) Number of epochs per replication: 25000 $\sim$ 75000.
(10) Number of initial conditions/replications: 100.

The parameters used in the FBPN are defined:

(1) $\tilde{x}_i$: the input to the $i$th input node.
(2) $\tilde{w}_{ij}^h$: the connection weight between the $i$th input node and the $j$th hidden node.
(3) $\tilde{I}_j^h$: the input to the $j$th hidden node.
(4) $\tilde{\theta}_j^h$: the threshold on the $j$th hidden node.
(5) $\tilde{h}_j$: the output from the $j$th hidden node.
(6) $\tilde{w}_j^o$: the connection weight between the $j$th hidden node and the output node.
(7) $\tilde{I}_o$: the input to the output node.
(8) $\tilde{\theta}_o$: the threshold on the output node.
(9) $\tilde{o}$: the network output (before defuzzification).

The procedure for determining the parameter values is now described as follows. After pre-classification, a portion of the adopted examples in each category is fed as "training examples" into the FBPN to determine the parameter values for the category. Two phases are involved at the training stage. At first, in the forward phase, inputs are multiplied with weights, summed, and transferred to the hidden layer. Then activated signals are outputted from the hidden layer as:

$$\tilde{h}_j = \frac{1}{1 + e^{-\tilde{n}_j^h}} \tag{12}$$

where

$$\tilde{n}_j^h = \tilde{I}_j^h(-)\tilde{\theta}_j^h \tag{13}$$

$$\tilde{I}_j^h = \sum_{all\ i} \tilde{w}_{ij}^h(\times)\tilde{x}_{(i)} \tag{14}$$

$\tilde{h}_j$'s are also transferred to the output layer with the same procedure. Finally, the output of the FBPN is generated as:

$$\tilde{o} = \frac{1}{1 + e^{-\tilde{n}^o}} \tag{15}$$

where

$$\tilde{n}^o = \tilde{I}^o(-)\tilde{\theta}^o \tag{16}$$

$$\tilde{I}^o = \sum_{all\ j} \tilde{w}_j^o(\times)\tilde{h}_j \tag{17}$$

To improve the applicability of the FBPN and to facilitate the comparisons with conventional techniques, the fuzzy-valued output $\tilde{o}$ is defuzzified according to Wrather and Yu's formula [19]:

$$d(\tilde{o}) = \int_0^1 E(o^\alpha)d\alpha \tag{18}$$

where $o^\alpha$ is the $\alpha$ cut of $\tilde{o}$. Then the output $o$ is compared with the normalized actual cycle time (or step cycle time) $a$, for which the RMSE is calculated:

$$\text{RMSE} = \sqrt{\sum_{\text{all trained examples}} (o-a)^2 \Big/ \text{number of trained examples}} \tag{19}$$

Subsequently in the backward phase, the deviation between $o$ and $a$ is propagated backward, and the error terms of neurons in the output and hidden layers can be calculated respectively as:

$$\delta^o = o(1-o)(a-o) \tag{20}$$

$$\tilde{\delta}_j^h = \tilde{h}_j(\times)(1-\tilde{h}_j)(\times)\tilde{w}_j^o\delta^o \tag{21}$$

Based on them, adjustments that should be made for connecting weights and thresholds can be obtained as:

$$\Delta\tilde{w}_j^o = \eta\delta^o\tilde{h}_j \tag{22}$$

$$\Delta\tilde{w}_{ij}^h = \eta\tilde{\delta}_j^h(\times)\tilde{x}_i \tag{23}$$

$$\Delta\theta^o = -\eta\delta^o \tag{24}$$

$$\Delta\tilde{\theta}_j^h = -\eta\tilde{\delta}_j^h \tag{25}$$

To accelerate convergence, a momentum can be added to the learning expressions. For example,

$$\Delta\tilde{w}_j^o = \eta\delta^o\tilde{h}_j + \alpha(\tilde{w}_j^o(t) - \tilde{w}_j^o(t-1)) \tag{26}$$

Theoretically, network-learning stops when the RMSE falls below a pre-specified level, or the improvement in the RMSE becomes negligible with more epochs, or a large number of epochs have already been run. Then test examples are fed into the FBPN to evaluate the accuracy of the network that is also measured with the RMSE. However, the accumulation of fuzziness during the training process continuously increases the lower bound, the upper bound, and the spread of the fuzzy-valued output $\tilde{o}$ (and those of many other fuzzy parameters), and might prevent the RMSE (calculated with the defuzzified output $o$) from converging to its minimal value. Conversely, the centers of some fuzzy parameters are becoming smaller and smaller because of network learning. It is possible that a fuzzy parameter becomes invalid in the sense that the lower bound is higher than the center. To deal with this problem, the lower and upper bounds of all fuzzy numbers in the FBPN will no longer be modified if Chen's index [20] converges to a minimal value.

When a new job is released into the factory, the eight parameters associated with the new job are recorded and compared with those of each category center. Then the FBPN with the parameters of the nearest category center is employed to estimate the cycle time (or step cycle time) of the new job.

The remaining cycle time of a job can be calculated as follows:

$$RCTE_{ik} = CTE_i - SCT_{ik} \tag{27}$$

At the same time, the interval cycle time can be expressed as:

$$ICTE_{ikf} = RCTE_{ik} - RCTE_{if} \tag{28}$$

If $f$ is a recent bottleneck step, then $ICTE_{ikf}$ determines the cycle time until the next bottleneck.

2.2. **Step 2: the fuzzy-neural DBD approach.** In the DBD approach proposed by Zhang et al. [5], different heuristics are used for scheduling jobs to non-bottleneck and bottleneck workstations. The traditional DBD approach starts from the division of jobs into several categories:

(1) The first priority category: Jobs will be marked as "hot jobs" and have the highest priority for processing.
(2) The second priority category: including jobs of the following conditions:
   (i) They are not "hot jobs".

(ii) The most recently average utilization of the next workstation ($UN_i$) is greater than $U_b$. $U_b$ is a real-valued constant specified in advance.
(iii) The current queue length before the next workstation ($QN_i$) is shorter than $C_1$ * the capacity of the next workstation per hour. $C_1$ is a positive integer.

According to conditions (ii) and (iii), the next workstation is clearly a weak bottleneck.

(3) The third priority category: including jobs of the following conditions:
(i) and (ii): the same as those in (2).
(iii) $QN_i$ is longer than $C_1$ * the capacity of the next workstation per hour, but shorter than $C_2$ * the capacity of the next workstation per hour. $C_2$ is a positive integer.

In other words, the next workstation is a strong bottleneck.

(4) The fourth priority category: Jobs that are not classified into the first three categories fall into this category.

An example is given in Table 1 to illustrate the classification of DBD. This classification has the following problems:

(1) Many foundry factories work with more than two types of priorities, such as "normal", "hot", "super hot". Such cases need to be expanded.
(2) Some empirical evidences are needed to determine the best values for all three parameters ($U_b$, $C_1$, and $C_2$).
(3) The values of $U_b$, $C_1$, and $C_2$ cannot be automatically adjusted based on the current conditions in the factory.
(4) The classification by DBD is a crisp partitioning. As a result, jobs with similar working conditions may be assigned to different categories. For example, in the previous example if there are two jobs $J_1$ and $J_2$ and $UN_1 = 0.85$, $QN_1 = 8$, $UN_2 = 0.86$, $QN_2 = 7$, then $J_1$ and $J_2$ will be assigned to category 4 and 2, even if their conditions are very similar.

Table 1. The classification of jobs by DBD

| # | Job no. | Priority | $UN_i$ | $QN_i$ | Category |
|---|---|---|---|---|---|
| 1 | 295 | Hot | 88% | 12 | 1 |
| 2 | 198 | Normal | 82% | 9 | 4 |
| 3 | 288 | Hot | 88% | 12 | 1 |
| 4 | 207 | Normal | 92% | 14 | 4 |
| 5 | 128 | Normal | 86% | 7 | 2 |
| 6 | 230 | Normal | 84% | 10 | 4 |
| 7 | 144 | Hot | 83% | 10 | 1 |
| 8 | 256 | Normal | 88% | 11 | 3 |
| 9 | 292 | Normal | 92% | 14 | 4 |

For these reasons, whether there is a more appropriate way to classify jobs for DBD needs to be investigated. In this study, the following fuzzy partition method is proposed instead:

(1) High priority categories: Undoubtedly, jobs with different priorities should be treated separately. In other words, there will be at least one category with each priority, such as "the normal priority (N) category", "the hot priority (H) category", "the super hot priority (SH) category", "the super super hot priority ($S^2H$) category".
(2) Normal priority categories: Jobs of "normal priority" are the most, and are further divided into three sub-categories "the first normal priority ($N_1$) category", "the

second normal priority ($N_2$) category", and "the third normal priority ($N_3$) category". The memberships of a job belonging to these sub-categories are calculated as follows (see Figure 1):

$$\mu_{N_1}(J_i) = \frac{UN_iQN_i - 0.5\max_k(UN_kQN_k) - 0.5\min_k(UN_kQN_k)}{0.5\max_k(UN_kQN_k) - 0.5\min_k(UN_kQN_k)} \tag{29}$$

$$\mu_{N_2}(J_i) = \begin{cases} \dfrac{UN_iQN_i - \min_k(UN_kQN_k)}{0.5\max_k(UN_kQN_k) - 0.5\min_k(UN_kQN_k))} \\ \qquad\qquad \text{if } UN_iQN_i \leq 0.5\max_k(UN_kQN_k) + 0.5\min_k(UN_kQN_k) \\ \dfrac{UN_iQN_i - \max_k(UN_kQN_k)}{0.5\min_k(UN_kQN_k) - 0.5\max_k(UN_kQN_k)} \qquad \text{otherwise} \end{cases} \tag{30}$$

$$\mu_{N_3}(J_i) = \frac{UN_iQN_i - 0.5\max_k(UN_kQN_k) - 0.5\min_k(UN_kQN_k)}{0.5\min_k(UN_kQN_k) - 0.5\max_k(UN_kQN_k)} \tag{31}$$

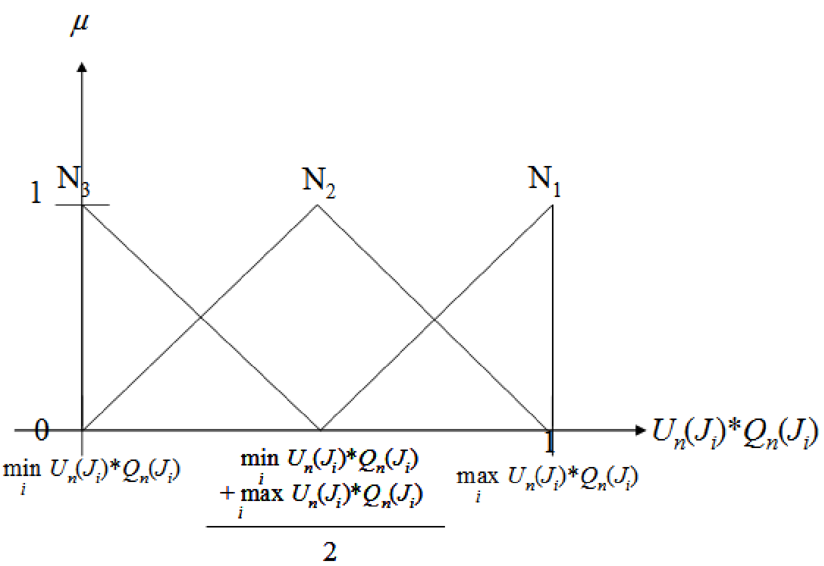


Figure 1. The fuzzy partition by the fuzzy-neural DBD

After the application of the fuzzy partition method to the previous example, the results are shown in Table 2. Thus, there is no need to determine the values of the parameters. In addition, the classification system will be automatically adjusted to reflect the current conditions of the factory. Jobs with similar conditions have similar membership function values, and will be assigned to similar categories.

After job classification, in the traditional DBD the heuristics for different categories are not the same:

(1) The first priority category: The heuristic for this category is a hybrid of CR and FIFO. In other words, CR is first used to dispatch jobs in this category. Then, FIFO is applied to break ties caused by CR.
(2) The second priority category: The shortest processing time until the next bottleneck (SPNB), CR, and FIFO are jointly used for this category. SPNB is first used to dispatch jobs in this category. CR is then applied to break ties caused by SPNB. If there are still some ties not broken, FIFO will be applied to break them. In SPNB,

Table 2. The classification of jobs by the fuzzy partition method

| # | Job no. | Priority | $UN_i$ | $QN_i$ | $\mu_{\mathrm{H}}$ | $\mu_{\mathrm{N1}}$ | $\mu_{\mathrm{N2}}$ | $\mu_{\mathrm{N3}}$ |
|---|---|---|---|---|---|---|---|---|
| 1 | 295 | hot | 88% | 12 | 1.00 | 0.00 | 0.00 | 0.00 |
| 2 | 198 | normal | 82% | 9 | 0.00 | 0.00 | 0.41 | 0.59 |
| 3 | 288 | hot | 88% | 12 | 1.00 | 0.00 | 0.00 | 0.00 |
| 4 | 207 | normal | 92% | 14 | 0.00 | 1.00 | 0.00 | 0.00 |
| 5 | 128 | normal | 86% | 7 | 0.00 | 0.00 | 0.00 | 1.00 |
| 6 | 230 | normal | 84% | 10 | 0.00 | 0.00 | 0.70 | 0.30 |
| 7 | 144 | hot | 83% | 10 | 1.00 | 0.00 | 0.00 | 0.00 |
| 8 | 256 | normal | 88% | 11 | 0.00 | 0.00 | 0.99 | 0.01 |
| 9 | 292 | normal | 92% | 14 | 0.00 | 0.90 | 0.10 | 0.00 |

the next bottleneck is usually a photolithography station. Jobs on it are usually divided into separate wafers that are processed separately. Subsequently, all wafers will be re-incorporated into the original job.

(3) The third priority category: This category uses SPT, CR, and FIFO to dispatch jobs and break ties in order.

(4) The fourth priority category: CR and FIFO are used for this category.

The heuristics used in the traditional DBD have the following problems:

(1) Many ties are formed and need to be broken.

(2) Estimating and considering the future conditions are conducive to the scheduling performance. However, all heuristics in DBD are only based on historical data, and do not take such information into account.

(3) Some advanced dispatching rules have been proposed recently, and can be used to replace the heuristics.

To solve these problems, in the proposed fuzzy-neural DBD approach we use the following dispatching rules instead:

(1) High priority categories (including H, SH, $S^2H$, etc.): The dispatching rule used in these categories is the 4f-biNFS rule:

$$\begin{aligned} SK_i = {} & (R_i - RCTE_{ik} + (RCTE_{ik} - \min(R_j)) \cdot f_1) \cdot \alpha^{-f_2} \cdot \left(\frac{i}{\lambda} - RCTE_{ik}\right. \\ & \left. + \left(RCTE_{ik} - \frac{1}{\lambda}\right) \cdot f_3\right) \cdot \left(\frac{\gamma}{\lambda}\right)^{-f_4} \cdot \left(\frac{(RCTE_{ik} - \min(RCTE_{jl}))}{\beta}\right)^{-(f_2+f_4)} \end{aligned} \tag{32}$$

where $\alpha = \max(R_j) - \min(R_j)$; $\beta = \max(RCTE_{jl}) - \min(RCTE_{jl})$; $\gamma = N - 1$; $f_1 \sim f_4$ are positive real numbers satisfying the following constraints:

$$\text{If } f_1 = 1 \text{ and } f_2 = 1, \text{ then } f_3 = 0, f_4 = 0, \text{ and vice versa} \tag{33}$$

$$\text{If } f_1 = 0 \text{ and } f_2 = 0, \text{ then } f_3 = 1, f_4 = 1, \text{ and vice versa} \tag{34}$$

$$\text{If } f_{1a} \geq f_{1b} \text{ and } f_{2a} \geq f_{2b}, \text{ then } f_{3a} \leq f_{3b} \text{ and } f_{4a} \leq f_{4b} \tag{35}$$

$$\text{If } f_{1a} \leq f_{1b} \text{ and } f_{2a} \leq f_{2b}, \text{ then } f_{3a} \geq f_{3b} \text{ and } f_{4a} \geq f_{4b} \tag{36}$$

where $(f_{1a}, f_{2a}, f_{3a}, f_{4a})$ and $(f_{1b}, f_{2b}, f_{3b}, f_{4b})$ are two different sets of the four adjustable factors. There are many possible models to form such sets. For example,

$$\text{Linear model: } f_1 = f_2, f_3 = f_4, f_1 = 1 - f_3 \tag{37}$$

$$\text{Nonlinear model: } f_1 = f_2^k, f_3 = f_4^k, f_1 = 1/f_3, k \geq 0 \tag{38}$$

$$\text{Logarithmic model: } f_1 = \ln(1 + f_2)/\ln 2, f_3 = \ln(1 + f_4)/\ln 2, f_1 = 1/f_3 \tag{39}$$

The 4f-biNFS rule estimates the remaining cycle time, and is more able to respond to factory conditions. In addition, through the appropriate integration of two nonlinear fluctuation rules NFSMCT and NFSVCT, the 4f-biNFS rule is expected to reduce the average cycle time and cycle time standard deviation at the same time. The four adjustable factors in the 4f-biNFS rule can be customized for a specific factory. In addition, the experimental results in Chen [8] and Chen and Wang [18] showed that it was easy to reduce the number of ties by nonlinear fluctuation smoothing rules.

(2) The third normal priority category ($N_3$): The dispatching rule used for the third normal category is also the 4f-biNFS rule.
(3) The second normal priority category ($N_2$): We use SCNB and 4f-biNFS together for this category. SCNB is modified from the traditional SPNB by replacing the processing time until the next bottleneck with the corresponding interval cycle time. As mentioned earlier, such a treatment is conducive to the scheduling performance.
(4) The first normal priority category ($N_1$): SPT and 4f-biNFS are used for job dispatching and tie breaking.

As a job may be classified into multiple categories with different memberships, the dispatching results by different categories need to be aggregated:

$$OD(i) = \sum_{\text{all } h} OD(i,h)\mu_h(J_i) \tag{40}$$

where $OD(i)$ is the order of processing job $i$; $OD(i,h)$ is the order of processing job $i$ if it belongs to category $h$. $\mu_h(J_i)$ is the membership of job $i$ to category $h$.

After the application of the new rules to the previous example, the sequencing results of all categories are summarized in Table 3. After aggregation, the final sequencing results are shown in Table 4.

Table 3. The sequencing results of all categories

| # | Job no. | Priority | $CTNBE_i$ | $RCTE_{ik}$ | $P_{ik}$ | $R_i$ | $H$ | $N_3$ | $N_2$ | $N_1$ |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 295 | hot | 23 | 110 | 16 | 95 | 3 | | | |
| 2 | 198 | normal | 2 | 155 | 16 | 47 | | 3 | 2 | 5 |
| 3 | 288 | hot | 3 | 112 | 16 | 92 | 2 | | | |
| 4 | 207 | normal | 7 | 156 | 13 | 51 | | 4 | 3 | 2 |
| 5 | 128 | normal | 20 | 192 | 14 | 12 | | 5 | 5 | 3 |
| 6 | 230 | normal | 9 | 146 | 16 | 63 | | 2 | 4 | 4 |
| 7 | 144 | hot | 24 | 187 | 18 | 20 | 1 | | | |
| 8 | 256 | normal | 1 | 128 | 12 | 76 | | 1 | 1 | 1 |
| 9 | 292 | normal | 22 | 116 | 17 | 94 | | 6 | 6 | 6 |

3. **Production Simulation for Generating Test Data.** A simulation system was developed to simulate a wafer fabrication factory with the following assumptions:

(1) Jobs are uniformly released into the wafer fabrication factory.
(2) Distribution of interarrival time of machine breakdown is exponential.
(3) This study considers dynamic-product-mix cases.
(4) The percentage of jobs with different priorities released into the wafer fabrication factory is controlled.
(5) The probabilities of processing a job on alternative machines at any given step are all equal. In advanced wafer fabrication factories, some machines might be dedicated to the operations of certain product types.

Table 4. The final sequencing results

| # | Job no. | Priority | Order |
|---|---|---|---|
| 1 | 295 | hot | 3 |
| 2 | 198 | normal | 6 |
| 3 | 288 | hot | 2 |
| 4 | 207 | normal | 5 |
| 5 | 128 | normal | 8 |
| 6 | 230 | normal | 7 |
| 7 | 144 | hot | 1 |
| 8 | 256 | normal | 4 |
| 9 | 292 | normal | 9 |

(6) A job cannot proceed to the next step until the fabrication on its every wafer has been finished, except when the step is a measurement operation.
(7) No preemption is allowed.

The basic configuration of the simulated wafer fabrication factory is simplified from a real-world wafer fabrication factory which is located in the Science Park of Hsinchu, Taiwan. There are five products (labeled A $\sim$ E) in the simulated wafer factory. The factory has a monthly capacity of 32000 wafers. The average utilization is about 90%. About 43 jobs (1066 wafers) are released into the wafer fabrication factory every day. Three types of priorities (normal, hot, and super hot) are randomly assigned to jobs in the beginning. Percentages of jobs with these priorities released onto the shop floor are restricted to approximately 80%, 18%, and 2%, respectively. Major products require 400 $\sim$ 800 processing steps and 1 $\sim$ 9 reentrances to the machines that are the biggest bottlenecks. In total 500 machines (including alternative machines) are provided to process single-wafer or batch operations in the factory. Fifty replications of the simulation are run successively. The proposed methodology was implemented on a PC with an Intel Dual CPU E2200 2.2 GHz and 1.99G RAM. A horizon of twenty-four months is simulated with the following conditions:

(1) The maximal cycle time is less than three months. Therefore, four months and an initial WIP status (obtained from a pilot simulation run) seemed to be sufficient to drive the simulation into a steady state. The statistical data were collected starting at the end of the fourth month.
(2) For each replication, the data of 1000 jobs was collected and classified by product type and priority. In total, the data of 50000 jobs were collected.
(3) A trace report was generated for every simulation run in order to verify the simulation model.
(4) The simulated average cycle times were compared with the actual values to validate the simulation model.

The FCM-FBPN approach was implemented with the Neural Network Toolbox of MATLAB 2006a with the following conditions:

(1) Number of epochs per replication: 75000.
(2) Number of initial conditions/replications: 100.
(3) Stop training if MSE $< 10^{-5}$ or 75000 epochs have been run.

4. **Results and Discussions.** To evaluate the effectiveness of the fuzzy-neural DBD approach and to compare them with some existing approaches – FIFO, EDD, SRPT, FSMCT, FSVCT, CR, and DBD were applied to schedule the jobs in the simulated wafer

fabrication factory, and to collect the data from 50000 jobs which was then separated by product type and priority. In total, the data of $12 * 50000 = 600000$ jobs were collected. Then the average cycle time and cycle time standard deviation of jobs with every product type and priority were calculated to evaluate the scheduling performance. The results are summarized in Table 5 and Table 6.

TABLE 5. The performances of various approaches in reducing the average cycle time

| Average cycle time (hrs) | A (normal) | A (hot) | A (super hot) | B (normal) | B (hot) | C (normal) | C (hot) |
|---|---|---|---|---|---|---|---|
| FIFO | 1324 | 412 | 334 | 1302 | 460 | 1438 | 580 |
| EDD-5.0 | 1119 | 355 | 317 | 1444 | 480 | 1833 | 602 |
| EDD-5.5 | 1095 | 360 | 305 | 1496 | 485 | 1895 | 632 |
| EDD-6.0 | 1064 | 365 | 307 | 1549 | 501 | 1932 | 604 |
| EDD-6.5 | 1069 | 363 | 309 | 1579 | 493 | 1950 | 586 |
| EDD-7.0 | 1029 | 369 | 308 | 1612 | 513 | 2008 | 595 |
| EDD-7.5 | 1015 | 365 | 310 | 1675 | 503 | 1974 | 609 |
| SRPT | 973 | 368 | 323 | 1800 | 487 | 2021 | 594 |
| CR-5.0 | 1159 | 367 | 306 | 1569 | 481 | 1938 | 562 |
| CR-5.5 | 1210 | 369 | 308 | 1548 | 496 | 1948 | 567 |
| CR-6.0 | 1220 | 375 | 313 | 1619 | 459 | 1964 | 558 |
| CR-6.5 | 1259 | 396 | 313 | 1684 | 497 | 2017 | 561 |
| CR-7.0 | 1346 | 377 | 307 | 1773 | 494 | 1970 | 566 |
| CR-7.5 | 1442 | 393 | 312 | 1911 | 508 | 1831 | 559 |
| FSMCT | 1414 | 407 | 323 | 1438 | 444 | 1366 | 497 |
| FSVCT | 1084 | 392 | 324 | 1774 | 541 | 1913 | 626 |
| DBD-5.0 | 1031 | 349 | 297 | 1501 | 463 | 1709 | 563 |
| DBD-5.5 | 1073 | 357 | 301 | 1537 | 450 | 1731 | 547 |
| DBD-6.0 | 1059 | 355 | 302 | 1524 | 457 | 1799 | 543 |
| DBD-6.5 | 1084 | 358 | 294 | 1566 | 456 | 1729 | 544 |
| DBD-7.0 | 1063 | 362 | 292 | 1570 | 446 | 1750 | 550 |
| DBD-7.5 | 1079 | 358 | 303 | 1587 | 454 | 1740 | 558 |
| The proposed methodology | 848 | 294 | 257 | 1202 | 391 | 938 | 450 |

In FIFO, jobs were sequenced on each machine first by their priorities, then by their arrival times at the machine.

In EDD, jobs were sequenced first by their priorities, then by their due dates. The performance of EDD is dependent on the way of determining the due date of a job. In the experiment, the due date of a job was determined as follows:

$$\text{Due date} = \text{release time} + (\psi - 1.5 * \text{priority}) * \text{ total processing time} \tag{41}$$

where $\psi$ indicates the cycle time multiplier. Although EDD is aimed at improving due-date related performance, it is also investigated because the due date of a job places a threshold on the cycle time of the job. Nevertheless, a tighter due date does not guarantee a shorter cycle time. For this reason, trying various approaches of determining the due-date to optimize cycle-time related performance is a reasonable attempt.

In FSMCT, there were two stages. First, jobs were scheduled with the FIFO policy, for which the remaining cycle times at each step of all jobs were recorded and averaged. Then, the FSMCT policy was applied to schedule jobs based on the average remaining

Table 6. The performance of various approaches in reducing cycle time variation

| Cycle time standard deviation (hrs) | A (normal) | A (hot) | A (super hot) | B (normal) | B (hot) | C (normal) | C (hot) |
|---|---|---|---|---|---|---|---|
| FSVCT | 324 | 35 | 28 | 227 | 55 | 295 | 54 |
| FIFO | 56 | 24 | 23 | 88 | 40 | 74 | 31 |
| EDD-5.0 | 133 | 26 | 24 | 51 | 40 | 134 | 23 |
| EDD-5.5 | 105 | 35 | 17 | 60 | 28 | 149 | 61 |
| EDD-6.0 | 103 | 32 | 22 | 42 | 50 | 146 | 34 |
| EDD-6.5 | 90 | 26 | 20 | 38 | 53 | 143 | 37 |
| EDD-7.0 | 85 | 24 | 13 | 35 | 48 | 144 | 34 |
| EDD-7.5 | 75 | 30 | 17 | 43 | 42 | 154 | 34 |
| SRPT | 249 | 33 | 23 | 108 | 30 | 253 | 38 |
| CR-5.0 | 69 | 30 | 19 | 58 | 38 | 147 | 38 |
| CR-5.5 | 64 | 26 | 15 | 54 | 51 | 160 | 53 |
| CR-6.0 | 63 | 37 | 16 | 50 | 34 | 138 | 55 |
| CR-6.5 | 65 | 43 | 16 | 35 | 52 | 145 | 70 |
| CR-7.0 | 79 | 47 | 14 | 16 | 42 | 169 | 43 |
| CR-7.5 | 98 | 50 | 12 | 25 | 35 | 193 | 49 |
| FSMCT | 42 | 44 | 23 | 35 | 29 | 81 | 34 |
| DBD-5.0 | 136 | 25 | 19 | 77 | 29 | 156 | 30 |
| DBD-5.5 | 134 | 24 | 18 | 76 | 32 | 156 | 33 |
| DBD-6.0 | 134 | 26 | 18 | 75 | 28 | 154 | 34 |
| DBD-6.5 | 133 | 28 | 18 | 72 | 32 | 155 | 38 |
| DBD-7.0 | 137 | 29 | 18 | 68 | 31 | 158 | 31 |
| DBD-7.5 | 141 | 29 | 18 | 70 | 30 | 164 | 33 |
| The proposed methodology | 68 | 25 | 14 | 41 | 13 | 117 | 24 |

cycle times obtained previously. In other words, jobs were sequenced on each machine first by their priorities, then by their slack values, which was equal to their release times minus the average remaining cycle times.

In CR, jobs were sequenced first by their priorities, then by their critical ratios. The critical ratio of a job is calculated as follows:

$$\text{Critical ratio} = (\text{time} - \text{due date})/\text{remaining processing time} \tag{42}$$

The performance of CR is dependent on the way of determining the due date of a job. Equation (41) is applied for this purpose.

In the proposed methodology, various combinations of the four parameters were tried to optimize the performance.

With respect to the average cycle time, the FIFO policy was adopted as the basis for comparison. On the other hand, the FSVCT policy was adopted as the comparison basis with respect to cycle time standard deviation.

The effectiveness of the proposed methodology with respect to various performance measures is illustrated in Figures 2 and 3. According to the experimental results, the following points can be made:

(1) An example is used to illustrate the effects of replacing SPNB by SCNB in the fuzzy-neural DBD approach. The results are shown in Figure 4 and Figure 5. Obviously, this attempt was successful, especially when it came to the average cycle time.

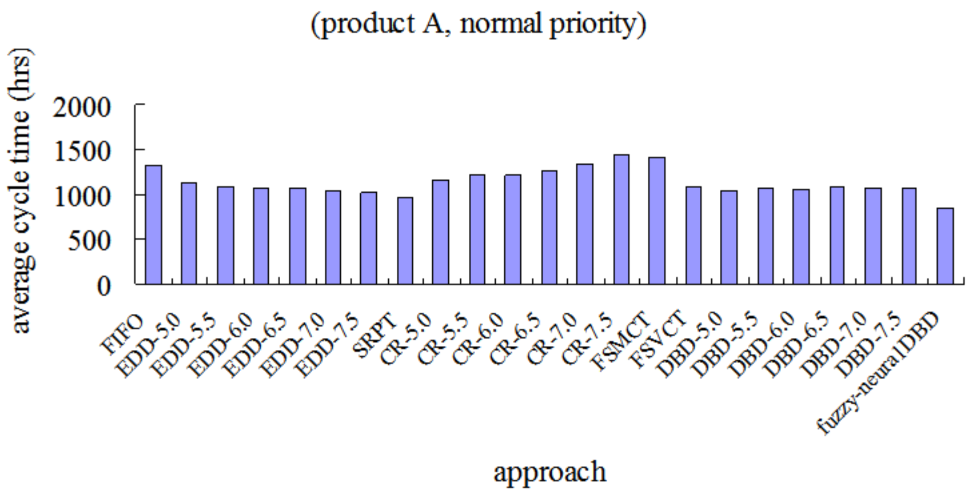


Figure 2. The performances of various approaches in reducing the average cycle time

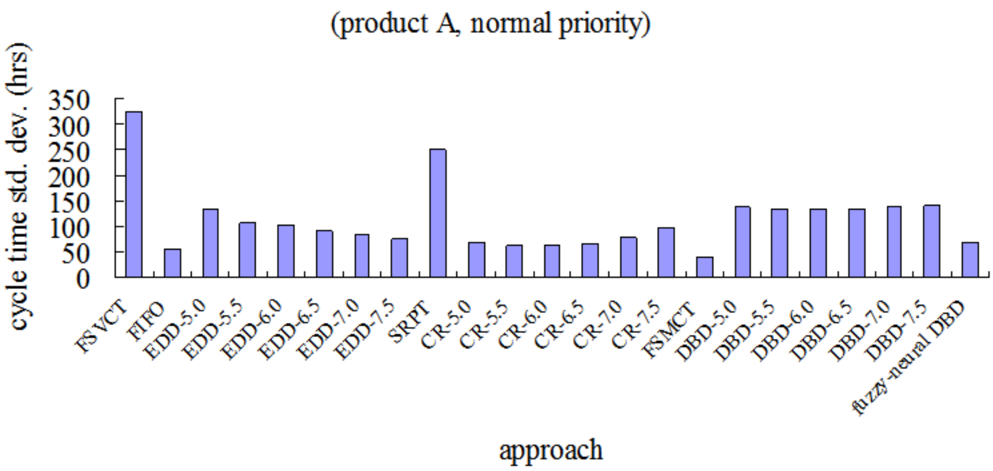


Figure 3. The performances of various approaches in reducing cycle time standard deviation

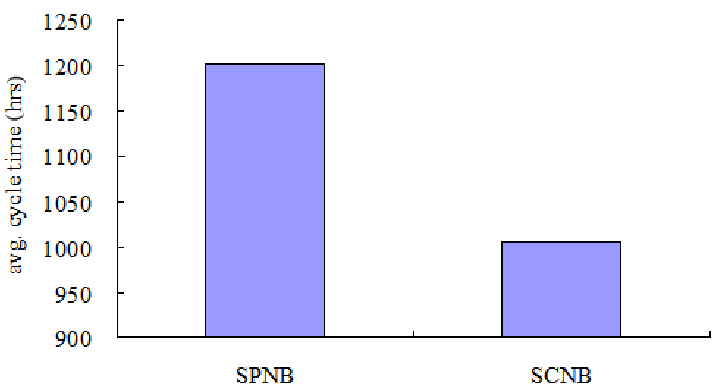


Figure 4. The effects of replacing SPNB by SCNB (with respect to the average cycle time)


FIGURE 5. The effects of replacing SPNB by SCNB (with respect to cycle time standard deviation)

(2) Through the use of 4f-biNFS instead of CR + FIFO, the improvements in both performance measures were also significant, due to the bi-criteria nature. Figure 6 and Figure 7 illustrate this fact. However, the advantage of 4f-biNFS over CR + FIFO seemed more obvious when cycle time standard deviation was optimized, which was more or less in line with Chen et al.'s study.

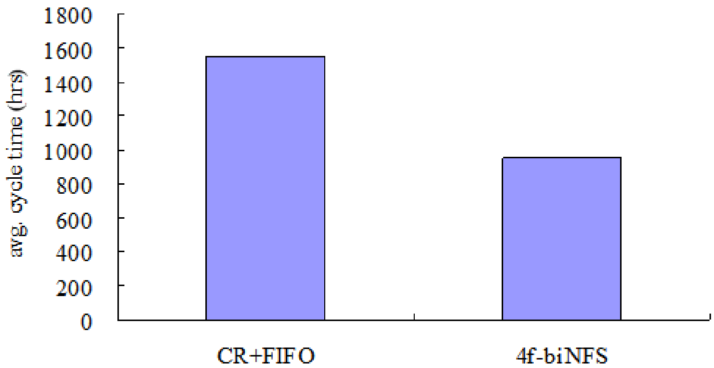


FIGURE 6. The effects of using 4f-biNFS instead of CR + FIFO (with respect to the average cycle time)

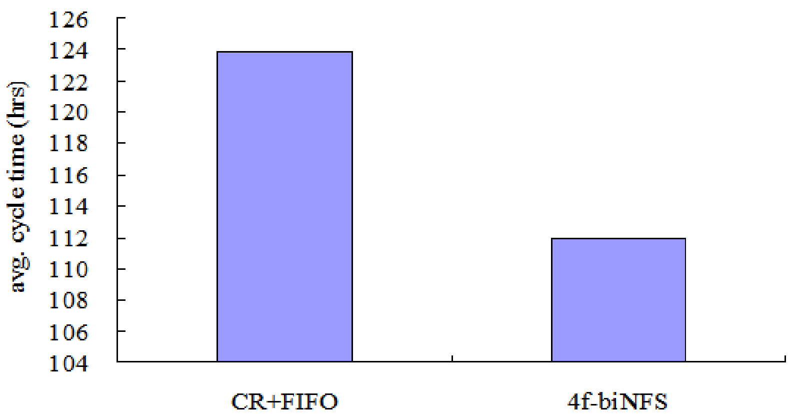


FIGURE 7. The effects of using 4f-biNFS instead of CR + FIFO (with respect to cycle time standard deviation)

(3) With respect to the average cycle time, the fuzzy-neural DBD approach was better than the baseline approach, the FIFO policy, in all cases with an average advantage of 24%. In Chen et al.'s study, nonlinear fluctuation smoothing rules reduced the average cycle time more than the traditional fluctuation smoothing rules. It is reasonable to believe that the same improvement is also possible to achieve by the proposed fuzzy-neural DBD approach.
(4) As in the traditional DBD approach, controlling the flow of jobs into the bottleneck workstations were proven to be very effective in reducing the average cycle times in the fuzzy-neural DBD approach.
(5) At the same time, the fuzzy-neural DBD approach has also made a very good performance in reducing cycle time standard deviation. The fuzzy-neural DBD approach surpassed the baseline p-FSVCT policy clearly in all cases with an average advantage of 62%, indicating that the same treatments also reduced the fluctuation in cycle time and improved the performance of the traditional DBD policy.
(6) Due date determination is required in the traditional DBD approach, but the algorithm presented here does not require such a step.
(7) As expected, SRPT performed also well in reducing the average cycle times, but might give an exceedingly bad performance with respect to cycle time standard deviation. Among various EDD rules, the performance of EDD-5.0 was the best in reducing the average cycle times, while EDD-7.0 was the best choice if cycle time standard deviation was to be minimized.

To ascertain whether there were significant differences between the performance of the proposed methodology and those of the existing approaches, a Wilcoxon sign-rank test was applied to test the following hypotheses:

$H_{a0}$: The performance of the fuzzy-neural DBD approach is the same as those of the traditional approaches with respect to the average cycle time.
$H_{a1}$: The performance of the fuzzy-neural DBD approach is better than those of the traditional approaches with respect to the average cycle time.
$H_{b0}$: The performance of the fuzzy-neural DBD approach is the same as those of the traditional approaches with respect to cycle time standard deviation.
$H_{b1}$: The performance of the fuzzy-neural DBD approach is better than those of the traditional approaches with respect to cycle time standard deviation.

The results of hypothesis testing are summarized in Table 7. With respect to the average cycle time, the performance of the fuzzy-neural DBD approach was significantly better than those of the existing approaches. On the other hand, its advantage over sixteen existing approaches on reducing cycle time standard deviation was also statistically significant.

## 5. Conclusion and Directions for Future Research.

This study presents a fuzzy-neural DBD approach to further improve the performance of job scheduling in a wafer fabrication factory. The fuzzy-neural DBD approach is modified from the well-known DBD approach after making some important changes. First, in the fuzzy-neural DBD approach the boundaries of job categories are no longer rigid and inflexible by fuzzy partition. Second, in the traditional DBD approach the remaining cycle time of a job is usually estimated with the average historical value, while we apply the FCM-FBPN approach to improve the accuracy of estimation, which has been shown to be conducive to the performance of job scheduling [12]. Third, some of the heuristics in the traditional DBD approach have been replaced by more advanced and flexible dispatching rules, including SCNB and 4f-biNFS.

TABLE 7. The results of testing hypotheses using Wilcoxon sign-rank test

| Approach | $H_{a0}$ | $H_{b0}$ |
|---|---|---|
| FIFO | $Z = 2.37^{***}$ | $Z = 0.51$ |
| EDD-5.0 | $2.37^{***}$ | $2.03^{**}$ |
| EDD-5.5 | $2.37^{***}$ | $2.28^{**}$ |
| EDD-6.0 | $2.37^{***}$ | $2.37^{***}$ |
| EDD-6.5 | $2.37^{***}$ | $2.03^{**}$ |
| EDD-7.0 | $2.37^{***}$ | 1.44 |
| EDD-7.5 | $2.37^{***}$ | $2.37^{***}$ |
| SRPT | $2.37^{***}$ | $2.37^{***}$ |
| CR-5.0 | $2.37^{***}$ | $2.28^{**}$ |
| CR-5.5 | $2.37^{***}$ | $1.77^{*}$ |
| CR-6.0 | $2.37^{***}$ | $1.94^{*}$ |
| CR-6.5 | $2.37^{***}$ | 1.52 |
| CR-7.0 | $2.37^{***}$ | 1.52 |
| CR-7.5 | $2.37^{***}$ | $1.77^{*}$ |
| FSMCT | $2.37^{***}$ | 0.00 |
| FSVCT | $2.37^{***}$ | $2.37^{***}$ |
| DBD-5.0 | $2.37^{***}$ | $2.37^{***}$ |
| DBD-5.5 | $2.37^{***}$ | $2.20^{**}$ |
| DBD-6.0 | $2.37^{***}$ | $2.37^{***}$ |
| DBD-6.5 | $2.37^{***}$ | $2.37^{***}$ |
| DBD-7.0 | $2.37^{***}$ | $2.28^{**}$ |
| DBD-7.5 | $2.37^{***}$ | $2.28^{**}$ |

To assess the effectiveness of the fuzzy-neural DBD approach, and compare it with some existing methods, a real wafer fabrication factory was also simulated, and then the proposed methodology and seven existing approaches (with their variants) were all applied to job scheduling in the simulated wafer fabrication factory. According to the experimental results, some remarkable conclusions are mentioned as follows:

(1) The scheduling performance (measured in terms of the average cycle time) of the proposed methodology was significantly better than that of some existing approaches.
(2) At the same time, the proposed methodology also outperformed these existing approaches in cycle time standard deviation.
(3) Controlling the flow of jobs into bottleneck workstations once again proved to be very important to the performance of job scheduling in both the average cycle time and cycle time standard deviation.

The advantages of the proposed methodology over the existing approaches include:

(1) The fuzzy-neural DBD approach outperformed the seven existing approaches in reducing the average cycle time and cycle time standard deviation at the same time, for which the bi-criteria dispatching rule 4f-biNFS played a key role.
(2) The traditional DBD approach incorporates CR, and therefore needs to determine the due date of each job, which is not necessary for the fuzzy-neural DBD approach, because it replaces CR by 4f-biNFS. The performance of the traditional DBD is also limited by the suitability of the due date setting method. In the fuzzy-neural DBD approach, the drawback does not exist.

Conversely, the possible deficiencies of the proposed methodology include:

(1) Long time is required for estimating the remaining cycle time in the proposed methodology. To tackle this problem, a dedicated software package can be developed in the future for implementing the proposed methodology.
(2) Even the linguistic set for forming fuzzy partition should be selected carefully [21-31]. Lack of a better way to pick up the linguistic set may harm the performance of scheduling.

Replacing the parts in the fuzzy-neural DBD approach to further improve the scheduling performance can be tried in future research.

**Acknowledgment.** This work is partially supported by the National Science Council of Taiwan.

## REFERENCES

[1] T. Chen and Y.-C. Lin, A fuzzy-neural fluctuation smoothing rule for scheduling jobs with various priorities a semiconductor manufacturing factory, *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol.17, no.3, pp.397-417, 2009.
[2] L. M. Wein, Scheduling semiconductor wafer fabrication, *IEEE Transactions on Semiconductor Manufacturing*, vol.1, pp.115-130, 1998.
[3] A. K. Gupta and A. I. Sivakumar, Job shop scheduling techniques in semiconductor manufacturing, *International Journal of Advanced Manufacturing Technology*, vol.27, pp.1163-1169, 2006.
[4] T. Chen, A tailored nonlinear fluctuation smoothing rule for semiconductor manufacturing factory scheduling, *Proc. of the Institution of Mechanical Engineers, Part I, Journal of Systems and Control Engineering*, vol.223, pp.149-160, 2009.
[5] H. Zhang, Z. Jiang and C. Guo, Simulation-based optimization of dispatching rules for semiconductor wafer fabrication system scheduling by the response surface methodology, *International Journal of Advanced Manufacturing Technology*, vol.41, no.1-2, pp.110-121, 2009.
[6] D. A. Koonce and S.-C. Tsai, Using data mining to find patterns in genetic algorithm solutions to a job shop schedule, *Computers and Industrial Engineering*, vol.38, no.3, pp.361-374, 2000.
[7] B.-W. Hsieh, C.-H. Chen and S.-C. Chang, Scheduling semiconductor wafer fabrication by using ordinal optimization-based simulation, *IEEE Transactions on Robotics and Automation*, vol.17, no.5, pp.599-608, 2001.
[8] H. J. Yoon and W. Shen, A multiagent-based decision-making system for semiconductor wafer fabrication with hard temporal constraints, *IEEE Transactions on Semiconductor Manufacturing*, vol.21, no.1, pp.83-91, 2008.
[9] H. Youssef, C.-M. Brigitte and Z. Noureddine, A genetic algorithm and data mining based meta-heuristic for job shop scheduling problem, *Proc. of the IEEE International Conference on Systems, Man and Cybernetics*, vol.7, pp.280-285, 2002.
[10] K. Sourirajan and R. Uzsoy, Hybrid decomposition heuristics for solving large-scale scheduling problems in semiconductor wafer fabrication, *Journal of Scheduling*, vol.10, no.1, pp.41-65, 2007.
[11] T. Chen, Fuzzy-neural-network-based fluctuation smoothing rule for reducing the cycle times of jobs with various priorities in a wafer fabrication factory – A simulation study, *Proc. of the Institution of Mechanical Engineers, Part B, Journal of Engineering Manufacture*, vol.223, pp.1033-1044, 2009.
[12] T. Chen and Y. C. Wang, A bi-criteria nonlinear fluctuation smoothing rule incorporating the SOM-FBPN remaining cycle time estimator for scheduling a wafer fab – A simulation study, *International Journal of Advanced Manufacturing Technology*, vol.49, pp.709-721, 2009.
[13] T. Chen, Y. C. Wang and Y. C. Lin, A bi-criteria four-factor fluctuation smoothing rule for scheduling jobs in a wafer fabrication factory, *International Journal of Innovative Computing, Information and Control*, vol.6, no.10, pp.4289-4304, 2009.
[14] T. Chen, Dynamic fuzzy-neural fluctuation smoothing rule for jobs scheduling in a wafer fabrication factory, *Proc. of the Institution of Mechanical Engineers, Part I, Journal of Systems and Control Engineering*, vol.223, pp.1081-1094, 2009.
[15] T. Chen, Y. C. Wang and H. C. Wu, A fuzzy-neural approach for remaining cycle time estimation in a semiconductor manufacturing factory – A simulation study, *International Journal of Innovative Computing, Information and Control*, vol.5, no.8, pp.2125-2139, 2009.

[16] T. Chen and Y.-C. Wang, A nonlinear scheduling rule incorporating fuzzy-neural remaining cycle time estimator for scheduling a semiconductor manufacturing factory, *International Journal of Advanced Manufacturing Technology*, vol.45, pp.110-121, 2009.
[17] X. L. Xie and G. Beni, A validity measure for fuzzy clustering, *IEEE Transactions of Pattern Analysis and Machine Intelligence*, vol.13, pp.841-847, 1991.
[18] T. Chen, Y. C. Wang and H. R. Tsai, Lot cycle time prediction in a ramping-up semiconductor manufacturing factory with a SOM-FBPN-ensemble approach with multiple buckets and partial normalization, *International Journal of Advanced Manufacturing Technology*, vol.42, no.11-12, pp.1206-1216, 2009.
[19] C. Wrather and P. L. Yu, Probability dominance in random outcomes, *Journal of Optimization Theory and Applications*, vol.3, pp.315-334, 1982.
[20] T. Chen and Y. C. Lin, A fuzzy back-propagation-network ensemble with example classification for lot output time prediction in a wafer fab, *Applied Soft Computing*, vol.9, no.2, pp.658-666, 2009.
[21] T.-S. Shih, J.-S. Su and H.-M. Lee, Fuzzy estimation of one vague missing value in two-factor experiments, *International Journal of Innovative Computing, Information and Control*, vol.5, no.12(B), pp.4971-4980, 2009.
[22] K. Kato and M. Sakawa, An interactive fuzzy satisficing method based on simple recourse model for multiobjective linear programming problems involving random variable coefficients, *International Journal of Innovative Computing, Information and Control*, vol.5, no.7, pp.1997-2010, 2009.
[23] T. Wang, Y. Chen and S. Tong, Fuzzy improved interpolative reasoning methods for the sparse fuzzy rule, *ICIC Express Letters*, vol.3, no.3(A), pp.313-318, 2009.
[24] T. Chen, Applying a fuzzy and neural approach for forecasting the foreign exchange rate, *International Journal of Fuzzy System Applications*, vol.1, no.1, pp.36-48, 2011.
[25] T. Chen, A hybrid fuzzy and neural approach for DRAM price forecasting, *Computers in Industry*, vol.62, pp.196-204, 2011.
[26] T. Chen, Evaluating the mid-term competitiveness of a product in a semiconductor fabrication factory with a systematic procedure, *Computers & Industrial Engineering*, vol.53, pp.499-513, 2007.
[27] T. Chen, An intelligent mechanism for lot output time prediction and achievability evaluation in a wafer fab, *Computers and Industrial Engineering*, vol.54, pp.77-94, 2008.
[28] T. Chen, Incorporating fuzzy c-means and a back-propagation network ensemble to job completion time prediction in a semiconductor fabrication factory, *Fuzzy Sets and Systems*, vol.158, pp.2153-2168, 2007.
[29] T. Chen, A SOM-FBPN-ensemble approach with error feedback to adjust classification for wafer-lot completion time prediction, *International Journal of Advanced Manufacturing Technology*, vol.37, pp.782-792, 2008.
[30] T. Chen, An intelligent hybrid system for wafer lot output time prediction, *Advanced Engineering Informatics*, vol.21, pp.55-65, 2007.
[31] T. Chen, A. Jeang and Y. C. Wang, A hybrid neural network and selective allowance approach for internal due date assignment in a wafer fabrication plant, *International Journal of Advanced Manufacturing Technology*, vol.36, pp.570-581, 2008.