Int J Adv Manuf Technol (2013) 66:1891–1905
DOI 10.1007/s00170-012-4468-6

ORIGINAL ARTICLE

Data-based scheduling framework and adaptive dispatching rule of complex manufacturing systems

Li Li · Sun Zijin · Ni Jiacheng · Qiao Fei

Received: 7 March 2012 / Accepted: 14 August 2012 / Published online: 29 August 2012

Abstract Based on the analysis of the differences and relations between traditional and data-based scheduling methods for complex manufacturing systems, a data-based scheduling framework was proposed and discussed for its implementation into a semiconductor manufacturing system. The state-of-the-art research on the key technologies of data-based scheduling was then introduced together with their development trends. By taking a real wafer fabrication facility (fab) as an example, an adaptive dispatching rule (ADR) was developed. Firstly, a simulation system for the fab was developed, and study samples were generated by simulation. Then, the relations between the parameters of ADR and real-time running state of the fab were obtained by learning with an integration of a binary regression model, backward propagation neuro-network, and particle swarm optimization algorithm from these study samples to realize the adaptive regulations of these parameters of ADR. Finally, ADR was integrated with the simulation system. The simulation results showed that ADR had a positive effect on the operational performance of the fab. Its "move" performance was increased by 2.41 and 7.24 % for the cases of 70 % and 90 % workload, respectively.

Keywords Data-based · Scheduling · Complex manufacturing system · Data mining

1 Introduction

Production scheduling is an important way to increase a factory's productivity and enhance its competitiveness. It has attracted the attention of researchers in a number of fields, e.g., industrial engineering, management engineering, and automation. Generally, it determines the machines to process jobs and their process orders, batch styles, together with assignments of other key resources to them to optimize the operational performance of the factory while meeting the process and resource constraints. The basic requirements on scheduling include the satisfaction of constraints, optimization of operational performance, and the ease of applications. The main tasks of production scheduling can be generalized as modeling and optimization, i.e., the description and solution of the scheduling problems. Since the 1950s, the research on scheduling has made much progress, and some results have been applied to industries successfully. However, the production styles of manufacturing factories have been dramatically changed along with the development of information technologies. The manufacturing processes are becoming increasingly complex, such as large-scale tasks, complicated constraints, coupling of the operational performances, and uncertain scheduling environments. Consequently, considerable existing scheduling modeling and optimization methods are no longer applicable.

The development of information technologies brings not only the challenges on the modeling and optimization of complex manufacturing system scheduling problems but also many other opportunities. The data in information systems of a factory are abundant, including one or many of enterprise resource planning (ERP), manufacturing executive system (MES), advanced production control (APC), and supervisory control and data acquisition (SCADA). These data contain a plentiful of scheduling relevant knowledge. Naturally, a new idea to solve complex scheduling problems is emerging, i.e., to extract useful knowledge from related on- and off-line data to improve the operational performance of the factory.

L. Li (✉) · S. Zijin · N. Jiacheng · Q. Fei
School of Electronics and Information Engineering,
Tongji University,
Shanghai, China
e-mail: lili@mail.tongji.edu.cn

The remainder of this paper is organized as follows. In Section 2, a data-based scheduling framework is designed based upon the discussions on the differences and relations between traditional and data-based scheduling. Then the state-of-the-art research related to the key technologies of data-based scheduling is introduced, with their future trends followed in Section 3. In Section 4, by taking a real semiconductor wafer fabrication facility (fab) as an example, an adaptive dispatch rule (ADR) is proposed to demonstrate the effectiveness of data-based scheduling methods. Section 5 gives conclusions and future works.

2 Data-based scheduling framework

2.1 The challenges facing to traditional scheduling

The motivation of research on data-based scheduling is the limits of traditional scheduling facing to complex manufacturing systems.

In view of modeling, some difficulties are existed to describe, e.g., the time constraints between some steps of some process flows and re-entrant process flows, which make a scheduling model inaccurate or difficult to be optimized and analyzed. Inaccurate models lead to the solutions malfunctioning in the real scheduling environment. Especially, the uncertain events cannot be included in a model in a real-time way. The consequence is that the model lacks accurate and adaptive parameters and is unable to respond dynamically to uncertain environments.

In view of optimization, most of the scheduling optimization problems are NP-hard. It is impossible to obtain an optimal solution in a polynomial time. The satisfactory solutions can be obtained in a reasonable computation time at the sacrifice of the performance. On the other hand, the knowledge-based scheduling methods (such as one-step heuristic rules) are capable to obtain a solution in a short time. However, it is difficult to gain such knowledge. It needs a large amount of simulation experiments. The generalization ability of the knowledge base obtained is relatively low.

On the one hand, more complicated a manufacturing process is, clearer the limit of traditional scheduling will be. On the other hand, there is a huge amount of data containing scheduling related knowledge in ERP, MES, APC, SCADA, and other information systems. Meanwhile, the development on wireless sensor network and radio frequency identification enables a real-time and accurate retrieval of online data. It has drawn great attentions from both academia and industry to the application of data-based methods to the scheduling problems in complex manufacturing systems.

2.2 The relations between data-based and traditional scheduling

The following questions should be answered when one uses data-based scheduling methods to solve corresponding problems in complex manufacturing systems:

- The differences and relations

  Actually, data-based scheduling is not a complete departure from traditional scheduling. On the contrary, they are closely related. The basic tasks of data-based scheduling are the same as those of traditional scheduling, i.e., modeling and optimization. The model is still an important part of data-based scheduling. In addition, the solutions obtained with traditional methods can serve as study samples.

  There are differences between data-based scheduling and traditional scheduling. The former pays more attention to the role of knowledge in scheduling and emphasizes the adaptability to real environments, the operability of the scheduling solutions, and powerful real-time response to the uncertainties in complex manufacturing systems. In addition, its models are not used directly to guide the operations in a production line. Its solutions may be the ideal scheduling performance indicators characterizing its capacity or property to provide an optimal basis for optimization methods.
- The application scope of data-based scheduling

  Obviously, the scheduling problems have no requirements on data-based scheduling if they can be optimally solved with traditional scheduling methods. The systems optimized with data-based scheduling should at least meet one of following conditions. First, the study samples (e.g., on-line, off-line, or simulation data) required by data-based scheduling can be obtained from the systems. Second, without expressions adding high precision on the parameters, the scheduling problems of a system cannot be accurately modeled, or can't be modeled at all. Third, the scheduling problems of a system can be modeled with high precision, but its optimum or satisfaction solution cannot be obtained within a reasonable time span.

2.3 Data-based scheduling framework

According to the above analysis, this work proposes a data-based scheduling framework that is composed of data layer, model layer, and scheduling layer as illustrated in Fig.1.

2.3.1 Data layer

The precondition to implement data-based scheduling is rich data sources related to scheduling, which forms the data

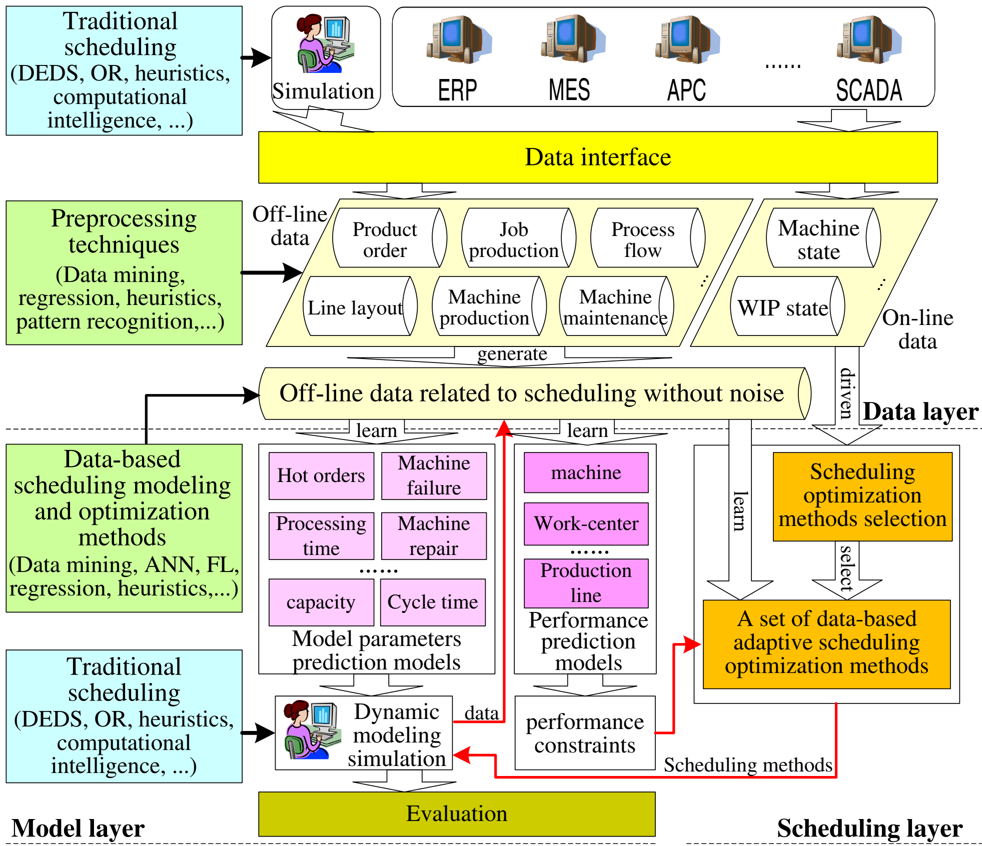

Fig. 1 Data-based scheduling framework

layer of the framework. There are lots of data related to scheduling in information systems of a company, such as ERP, MES, ACP and SCADA. Meanwhile, we can use traditional scheduling methods (such as discrete event simulation system, heuristics, and mathematical programming) to generate a plentiful of data related to scheduling. These data can be classified into off- and on-line ones. The former include the information about production orders, job production, process flows, machine layout, machine production, machine maintenance and etc, while the latter are the real-time information reflecting the working condition of production environments, such as information about machine state (e.g., idle, busy, and occupied) and WIP state (being processed, queuing and hold). Since these data are commonly incomplete and contain noise, the preprocessing techniques are often required to execute filtration, purification, denoising, and optimization. Then these preprocessed data can be taken as study samples to extract useful scheduling-relevant knowledge.

2.3.2 Model layer

There are two kinds of prediction models in the model layer, i.e., parameter and performance prediction models, which are built by learning from the preprocessed data from the data layer. The former is to predict the parameters of scheduling models, such as the possibility of hot orders' arrival, machine failure and machine maintenance, processing time of one step, capacity of a machine and cycle time of a product. These parameters represent the occurrence probability of uncertain events (such as the first four), or the production performance (such as the last two) of a manufacturing system. If they are integrated within a dynamic modeling simulation system, the accuracy and precision of the performance prediction can be improved. Then the dispatching rules mined from these simulation data can be expected a better performance. The latter is to predict the operational performance and constraints of a machine, a work-center or production line in a period, which can provide useful guidance for the scheduling layer to select a suitable dispatching rule and find a near-optimal or satisfactory solution quickly.

2.3.3 Scheduling layer

The scheduling layer is responsible for the job dispatching of a manufacturing system. The adaptive scheduling optimization methods in this layer are learned from the preprocessed data in the data layer. The characteristics of the scheduling environments that adapting to and concerned scheduling performance issues of each method may be different. In a real production environment, a choice of a method is jointly determined by the on-line data (such as machine state and WIP state) together with the ideal performance objectives and constraints obtained by the data-based scheduling models.

2.4 The realization of the data-based scheduling framework

Taking a semiconductor manufacturing system as an example, the realization scheme of the proposed data-based scheduling framework may include the steps as shown in Fig.2:

Step 1: The off-line data in a semiconductor wafer fabrication facility (fab) and the simulation data generated by its dynamic modeling simulation model are taken as study samples. Then, we use various methods (such as data mining, artificial neural-network, support vector machine, and evolutionary algorithm) to learn from these data building the scheduling models of the fab, its work areas (such as photo area, etching area, oxidation area, and implant area), machine groups, and machines. Next, the scheduling model of the fab is used to predict the expected performance of the fab in a certain period (e.g., a month, week, or day), which are further decomposed to work areas, machine groups, and machines. Their performances provide useful guidance to real-time dispatchers.

Step 2: The study samples for mining the adaptive scheduling optimized methods include the offline data of the fab and the scheduling plans generated by traditional scheduling methods (such as computational intelligence, simulation, and heuristic rules). It is notable that those concerned performance issues of each adaptive scheduling optimized method may differ due to its study samples with different characteristics.

Step 3: At the dispatch point (i.e., a machine becomes idle, and there are queuing jobs before it), the machine selects the most suitable one from set of data-based scheduling optimized methods according to its expected performance and on-line data relevant to scheduling.

3 Key technologies of data-based scheduling

3.1 Data preprocessing techniques

If a manufacturing system has a certain scale and its process flows are complex, some problems may appear, such as data abundance, more scheduling attributes to be considered, and data noises. These problems have serious negative effects on the validity of the data-based scheduling. As a result, the data preprocessing is a necessary and key element in the data-based scheduling framework. The existing research results can be classified into attribute selection, clustering and attribute discretization.

3.1.1 Attribute selection

If there is a considerable redundancy in condition attributes, the classification precision may be decreased, the resultant scheduling rules may be invalid and more conflicts between rules may be observed. Therefore, the attribute selection, known as selecting important ones from the condition attributes, is required. Its common methods include fuzzy set, feature selection, and computational intelligence. For

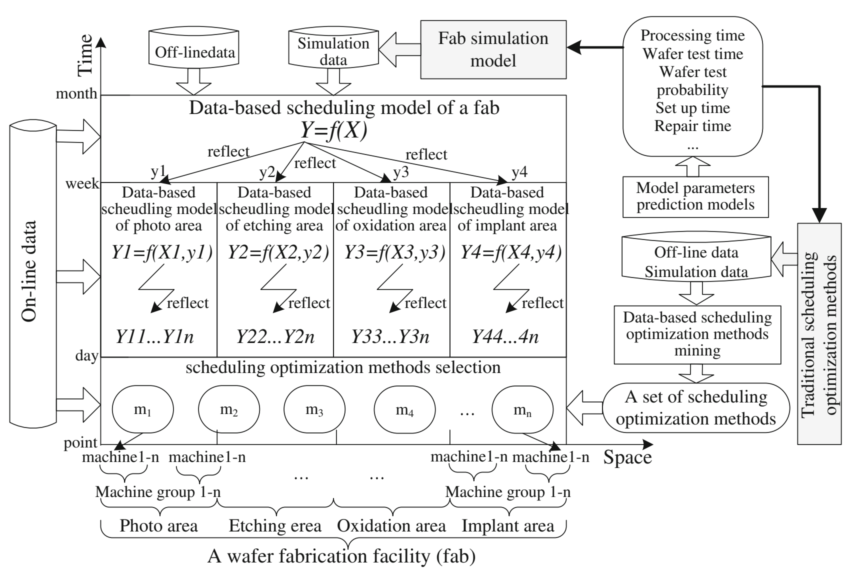

Fig. 2 A realization scheme of the proposed data-based scheduling framework for a fab

example, Kusiak [1–3] proposed a fuzzy-set-based method to obtain an attribute selection rule by learning from samples for semiconductor manufacturing quality problems. He also used feature transfer and data set decomposition techniques to improve the accuracy and efficiency of defect prediction. Chen et al. [4] shrunk the search space with the concept of feature nuclear and used an ant colony algorithm to obtain the reduction of a set of attributes to improve the efficiency of knowledge reduction. Shiue et al. [5–9] developed a two-stage decision tree based adaptive scheduling system and proposed an artificial neural network (ANN) based feature selection algorithm together with a genetic algorithm (GA) for attribute selection, respectively. They applied self-organizing map (SOM) to data clustering and used decision tree, ANN, and support vector machine to learn from each cluster, in order to optimize the parameters of the algorithms. The validity of the proposed methods was shown by simulation.

3.1.2 Data clustering

Data clustering is a technique to classify the samples according to their similarity. The samples with high similarity are included in the same class. This technique can be used to delete the noisy data. Noise has a negative effect on the precision of learning. For example, when one applies C4.5 decision tree model to deal with the samples containing noises, the scale of the generated tree becomes large, the prediction accuracy is reduced, and the pruned processing is needed. The common methods for data clustering include SOM, fuzzy-C means, K-means, and ANN. For example, Hu and Su [10] presented a hierarchical clustering method to find the machines related to the decreasing yield. Chen [11, 12] attempted to integrate fuzzy-C means and K-means with backward propagation neuro-network (BPNN) to cluster the training samples and train an ANN for each clustering data. SOM was used to smooth the noisy data in the samples and improve the learning effects.

3.1.3 Attribute discretization

Some algorithms and models (such as decision tree and fuzzy set) can be used to handle discrete data only. Therefore, attribute discretization is necessary to transfer a continuous attribute to a discrete one. For example, Knooce and Tsai [13] and Li [14] implemented an isometric discrete division on attributes when mining optimized scheduling plans. Rafinejad et al. [15] proposed a fuzzy-C means based attribute discretization method to extract a dispatching rule from optimal scheduling plans with better ability to approximate those optimal scheduling plans.

There is still a long way to go to propose more powerful data preprocessing technologies to find useful knowledge from complicated mass data with noises and deficiencies.

3.2 Data-based scheduling modeling methods

The data-based scheduling modeling of a manufacturing system reflects its off- or on-line data from information systems in the models that describe its production scheduling process or predict its operational performance.

3.2.1 Data-based description model

Data-based description models include Perti net models, discrete event simulation models, and intelligent models. The traditional modeling methods are relatively trivial and rigid. They at first build elements (such as jobs and machines) of a manufacturing system as objects, then add parameters and the relations between these objects. If there are changes on machines or introductions of new processes, the existing model needs to be modified and often requires much work. On the contrary, the data-based modeling methods focus on mapping rules between its data sets and scheduling models of a manufacturing system instead of the tedious work as required by traditional modeling. The data-based scheduling models can be changed conveniently and flexibly by modifying related data.

For example, Gradisar and Music [16, 17] mapped the information related to the factory layout and process flows into a Petri net model describing a production scheduling process. Some heuristic rules were integrated with the Petri net model to evaluate their scheduling performance. A furniture manufacturing process was taken as an example to verify and validate the proposed method. The main shortage of the work was the failure to consider dynamic information in the production environments. Thus, the proposed method cannot be applied to the modeling of some manufacturing systems with nonzero initial state (such as a fab). Mueller and McGinnis [18] proposed a modeling method to reflect the production data of a fab into its object oriented Petri net simulation model. The main elements of the model included machines, tools, process flows, and recipes. The factors, such as batch processing, failure time of machines and tools, and rework jobs, were also considered. The main shortage of the work was the great simplification on the real fab and lacked the consideration on its non-zero initial state as well. Ye et al. [19] developed a dynamic modeling technique to dynamically build a discrete event simulation model of a fab with both off- and on-line data in its information systems. The on-line data made the simulation model capable to reflect the nonzero initial state of the fab. Since the mapping between the data and the simulation model is closely related to the simulation software eM-Plant, the generality of the conversion method requires further improvement.

3.2.2 Data-based prediction model

Many parameters of the models are not constant and do not satisfy some specific distribution due to the complexity and uncertainty of complex manufacturing systems. These parameters play an important role in the scheduling optimization. Consequently, it is an important component of data-based scheduling to build the prediction models of these parameters by learning from related historical data.

Taking the processing time of jobs as an example, it is included in many dispatching rules. The good scheduling results cannot be expected by simply using their theoretical values in process files, their average values, or empirical estimation values. For example, Bagchi et al. [20] pointed out that the processing time of a job in a fab was related to the recipes of machines, the number of the wafers in a lot (i.e., job), the set-up requirements of machines, and the number of parallel loading ports and slots (chambers). Then, they provided a multiple regression analysis method to build the relations between these parameters and the processing time. Hosoe et al. [21] decomposed the processing time into three parts as job loading time, job unloading time, and job transportation time; modified the processing time calculation formula; and used a regression method to obtain the coefficients of the formula by learning the historical data.

In addition, some dispatching rules introduce cycle time and capacity as decision parameters. These parameters have direct effects on the scheduling performance and are difficult to be estimated mathematically. For example, the due date set of the jobs is closely related to the prediction of their cycle time. There are many factors on the cycle time, such as WIP, percentage of hot lots, and processing stages of WIP. Chen [22–24] applied a fuzzy ANN to building the prediction model of cycle time with high precision. Chung and Lai [25] provided a cycle time prediction method for the production-to-inventory environments with hybrid product types and used it to improve the release control method. Peam et al. [26] presented a cycle time prediction method for hybrid product type environments with changeable product requirements. Chang et al., Chen, and Chang and Liao [27–30] integrated SOM, case-based reasoning (CBR), BP, and flow-based programming to predict the cycle time performance with high precision.

Beside of predicting the parameters of scheduling models, the data-based prediction models are also used to predict the occurrence probability of uncertain events (such as hot order arrival and machine failure) and recognize the quality performance. For example, Wang and Ze [31] applied a neural-fuzzy based forecasting approach for rush orders, with a precision being 20 % higher than regression methods. Arrendendo and Martinez [32] proposed a reinforcement learning strategy with local weighted regression to determine whether to accept the hot orders. Shukla et al. [33] developed a bidding based multiagent system for flexible job shops and used a fuzzy decision tree to mine the occurrence probability of tools' failure, and presented a hybrid tabu-simulation annealing (SA)-oriented approach to optimize its planning and scheduling. Li et al., Hsu and Chien, and Rokach and Maimon [34–36] used various methods (such as data mining, decision tree, genetic programming, and SOM) to predict and improve the yield performance of the fab.

Unfortunately, there still lacks the research on the data-based operational performance prediction models. Further study is needed to offer necessary guidance for their utilization.

3.3 Data-based scheduling optimization methods

The main focus of the existing research on data-based scheduling optimization methods is to learn useful dispatching rules from offline data in information systems or the scheduling plans obtained by simulations or intelligent optimization methods.

3.3.1 Learning dispatching rules from scheduling plans obtained by simulations

Much research has shown that there is no optimal dispatching rule that suits all kinds of manufacturing systems. The efficiency of a dispatching rule implemented in a system is closely related to its real-time state. Thus, the better choice is to select the dispatching rule according to real-time operational information. Simulation is one of the important ways to evaluate the performance of various decision methods. Generally, there are two kinds of simulation schemes, i.e., off- and on-line simulations. The former is to simulate with various initial states and various dispatching rules off-line, then save the dispatching rule achieving the best performance to construct a knowledge base. Obviously, the efficiency of off-line simulations is low and the generalization ability of the knowledge base is also quite weak. The latter is to simulate various dispatching rules at a dispatching point online, then select the one with best performance to implement job dispatching. Online simulations have critical requirements on simulation time. Any delay may fail in meeting the requirements of real-time dispatching.

To overcome the deficiencies existing in the simulation-based dispatching methods, machine learning can be used to learn dispatching rules from the scheduling plans generated by simulations. For example, Shaw et al. and Park et al. [37, 38] proposed the concept of adaptive scheduling. They selected the dispatching rule according to the real-time running state and the performance to be optimized at the dispatching point. Lee and Shaw and Lee [39, 40] extracted fuzzy dispatching rules from training samples using an

ANN-based learning algorithm. Priore et al. [41–43] summarized the research on adaptive scheduling systems, proposed a CBR-based learning method and compared the performance of ANN, decision tree, and CBR. Li et al. [44–46] developed an adaptive fuzzy neural-network learning algorithm to obtain the knowledge for selecting adequate dispatching rules suitable to fewer training samples.

The above-mentioned offline learning methods need a large amount of samples gathered by simulations. Reinforcement learning, having the ability to be well combined with online simulations, has been applied to adaptive scheduling. For example, Aydin and Öztemel [47] modeled the job shop scheduling as a multi-agent system and presented a Q-III algorithm to obtain the knowledge for dispatching rules selection. Yang and Yan [48] proposed the concept of multi-agent system-based intelligent manufacturing systems and developed a B-Q learning algorithm with reference to clustering technology to learn how to select dispatching rules.

Machine learning can generalize the selected dispatching rules and play a central role in constructing the knowledge base of adaptive scheduling systems. However, both on- and off-line learning are dependent on the scheduling models of manufacturing systems. The quality of the models directly affects the learning efficiency. In addition, the knowledge bases obtained by offline learning degrade over time and need reasonable update mechanisms. On-line learning strategies have higher robustness, but their optimal effects are not obvious and their learning speed is slow at the beginning stage. It is necessary to provide new methods to combine off-line learning methods with online ones to further expand the set of dispatching rules and select them better.

3.3.2 Learning dispatching rules from optimal scheduling plans obtained by computational intelligence methods

The development of information technologies makes it possible to solve a large-scale scheduling problem optimally. Besides the computation time, the more difficult problem faced by the scheduling optimization algorithms is that the obtained plans cannot be implemented onto the whole scheduling period due to the unpredictable uncertainties in the production environments. It is valuable to discuss how to mine a reasonable dispatching rule from the optimal scheduling plans obtained by an optimization algorithm to generate the plans approximating those generated by its ancestors. Then, we can say that the dispatch rule obtained is near-optimal and suitable for the requirements of real-time dispatching.

For example, Koonce and Gandhi [49] at first generated an optimal plan of a 6×6 job shop scheduling problem with a generic algorithm, then applied an attribute-oriented inductive algorithm and an inductive logic programming method to obtain the relations between jobs' attributes and their relative positions in the optimal scheduling plans, respectively. The simulation results showed that the mined dispatching rule can obtain the dispatching plans closer to the optimal one generated by GA than other common dispatching rules (e.g., shortest remaining processing time rule). Based on this research, Weckman et al. [50] adopted an ANN method to further improve the prediction precision. The obtained ANN had been used to solve a new job shop scheduling problem. The plan obtained with the ANN model was the closest to the optimal plan generated by GA. Shahzad and Mebarki [51] used a tabu-search algorithm finding optimal scheduling plans, then approximated these plans with a decision-tree based learning algorithm. It at last presented a general framework to integrate the optimization, data mining, and simulation. Chaudhuriz and De [52] obtained a black block based scheduler to dispatch jobs by making use of rough fuzzy multilayer perception neural networks to approximate optimal plans obtained by GA. Kumar and Rao [53] optimized the flow shop with an ant colony algorithm and used C4.5 to learn dispatching rules from those optimal plans. Olafsson and Li [54, 55] proposed a GA-based dispatching rule selection method. It had two stages: The first stage was to choose better plans using a GA algorithm, and the second one was to learn the dispatching rules from these plans by using decision-tree based methods.

3.3.3 Learning dispatching rules from off-line data in information systems

The off-line data in information systems of a manufacturing system contain knowledge related to scheduling. One can also learn dispatching rules from these data. For example, Choi et al. [56] used a decision-tree based method to find the knowledge for selecting dispatching rules from offline data in a re-entrant manufacturing system, with consideration of its real-time state. Kwak and Yih [57] used decision-tree based methods to build the relations between dispatching rules and operational performances under the circumstances with various real-time state in a short period by learning from offline data in the manufacturing system. Then, they obtained long-term effective dispatching rules from simulations. Finally, these two methods were integrated for the selection of a good dispatching rule.

Existing research results on data-based scheduling optimization methods have their common deficiency, i.e., lack of flexibility. They select a dispatching rule from a specified set at the dispatching point according to real-time running state or use a learned dispatching rule off-line during the whole decision process without adaptions. The former one's performance is dependent on the specified dispatching rules set, while the latter is dependent on the diversity of study samples. In addition, the considered manufacturing systems are small-scale job shops or flow shops. The research in this field should move forward.

4 An example of data-based scheduling optimization method

Taking a real fab as an example, we demonstrate the effectiveness of a data-based scheduling optimization method. An ADR is proposed in this section. Its adaptability is gained since the parameters of the algorithm are updated along with the real-time running state of the fab to obtain better operational performance.

4.1 Definition of the parameters and variables of ADR

The variables and parameters in ADR are defined as follows:

i	index of the available machine
k	index of the batches composed of the queuing jobs at t before machine i with batch processing style
l	index of the jobs being processed or just finished and their next step will be finished by machine i
m	index of the recipes of machine i
n	index of the queuing jobs before machine i at t
t	dispatching decision point, or time when one machine becomes available
v	index of the recipes of machine d_n
d_n	index of the machine to process the next step of the current step waiting to be processed on machine i of job n at t
q_n	queuing time of job n before machine i at t
x_n	if job n is a hot job (i.e., τ_n=MAX), $x_n=1$ otherwise, $x_n=0$
x_{ib}	if machine i is a bottleneck, $x_{ib}=1$ otherwise, $x_{ib}=0$
x_{d_n}	if downstream machine d_n is idle, $x_{d_n}=1$ otherwise, $x_{d_n}=0$
τ_n	variable of job n at t
τ_{d_n}	variable of machine d_n at t
α_1, β_1	the relative importance parameters to measure the importance level between the on-time
α_2, β_2, γ, σ	the relative importance parameters to measure the importance level between delivery performance, capacity utilization, occupation time, and workload balance performance
B_k	size of batch k, i.e., the number of jobs in it
B_i	capacity of machine i with batch processing style, i.e., the maximum number of jobs in a batch
D_n	due date of job n
FF_n	ratio between the average cycle time (the sum of the process time and queue time) of job n and its raw processing time
M_i	the number of the recipes of machine i
N_i^m	the number of queuing jobs before machine i using its recipe m at t
$N_i^{m'}$	the number of ordinary jobs selected to be processed with hot jobs together as a batch
$N_i^{m''}$	the number of hot jobs selected to be processed together as a batch
$N_i^{m'''}$	the number of ordinary jobs selected to be processed together as a batch
$N_{d_n}^{m'}$	the number of ordinary jobs selected to be processed with the jobs whose downstream machine are idle together as a batch
$N_{d_n}^{m''}$	the number of the jobs whose downstream machines are idle selected to be processed together as a batch
N_{d_n}	the number of jobs queuing before machine d_n
N'_{d_n}	the number of jobs which downstream machines are idle in batch k
P_k	the selected probability of batch k
P_n	the selected probability of job n
PT_k^i	the occupation time of batch k on machine i (including processing time, upload time, download time, set-up time, and qualification time)
PT_m^i	processing time of recipe m of machine i
PT_n^i	occupation time of job n on machine i (including processing time, upload time, download time, set-up time, and qualification time)
$PT_{d_n}^v$	the processing time of recipe v of downstream machine d_n
RPT_n	remaining processing time of job n at t
W_{d_n}	the workload of machine d_n at t that is the total occupation time of its queuing jobs

4.2 Problem assumptions

The following assumptions are made to facilitate the discussion of ADR.

1. The information related to job dispatching (such as processing time of jobs, WIP queuing before a machine, and available time of a machine) can be obtained from MES or other information systems of the fab.
2. The main concerns of the dispatching on the non-BPMs are the on-time delivery performance and fast movement of WIP in a fab.
3. When making dispatching decisions for BPMs, there are two main steps. The first step is to batch the jobs. There are two important constraints: (a) only jobs using the same recipe can be processed together as a batch and (b) the batch size should be no greater than the capacity of the BPMs. In addition, trade-off between the wasted

time and the wasted capacity of a BPM should be made. The second step is to determine the priorities of the batches. The main concerns are same to the dispatching of the non-BPMs.

4. The processing time of one batch on a single machine is independent of the number of jobs in the batch.
5. Once processing begins on one batch, no job can be removed from or added to the machine until the current batch is processed.

4.3 Workflow of ADR

The detailed workflow of ADR is shown in Fig.3.

Step 1: When machine i becomes available at time t, determine whether the machine is a BPM. If yes, go to step 6.

Step 2: Compute the variable of each queuing job before machine i according to

$$\tau_n = \begin{cases} \mathrm{MAX} & \mathrm{RPT}_n \times \mathrm{FF}_n \geq D_n - t \\ \frac{\mathrm{RPT}_n \times \mathrm{FF}_n}{(D_n - t + 1)} - \frac{\mathrm{PT}_n^i}{\sum_n \mathrm{PT}_n^i} & \mathrm{RPT}_n \times \mathrm{FF}_n < D_n - t \end{cases} \tag{1}$$

Eq. (1) assures that a much larger ratio between the theoretical remaining processing time (RPT) and the real RPT of a queuing job is accompanied by a much more urgent on-time delivery requirement. Accordingly, its variable is much larger to have a higher priority to be selected by machine i. It is set to meet the customers' on-time delivery requirements. In addition, if the theoretical RPT of a queuing job is more or equal to its real RPT, the processing may be delayed. The job will be regarded as a hot job with its variable set to MAX. Then, it has the highest priority to be selected by each machine. Furthermore, the occupation time of a queuing job on machine i has a negative impact on the value of its variable. Longer occupation time of a queuing job means the value of its variable should be much lower. In this way, the movement of jobs on machine i can be expedited to improve its utility.

Step 3: Compute the variable of the machine to process the next step of the current step waiting to be processed on machine i of job n at t

$$\tau_{d_n} = \frac{W_{d_n}}{\max_n (W_{d_n})} \tag{2}$$

Eq. (2) means that a much higher workload of a machine results in a larger value of its variable. Its main objective is to obtain the workload balance between consecutive machines.

Step 4: Compute the selected probability of each job queuing before machine i according to

$$P_n = \begin{cases} q_n & \tau_n = \mathrm{MAX} \\ \alpha_1 \tau_n - \beta_1 \tau_{d_n} & \tau_n \neq \mathrm{MAX} \end{cases} \tag{3}$$

Eq. (3) means that dispatching decisions take accounts of the jobs' due date, the

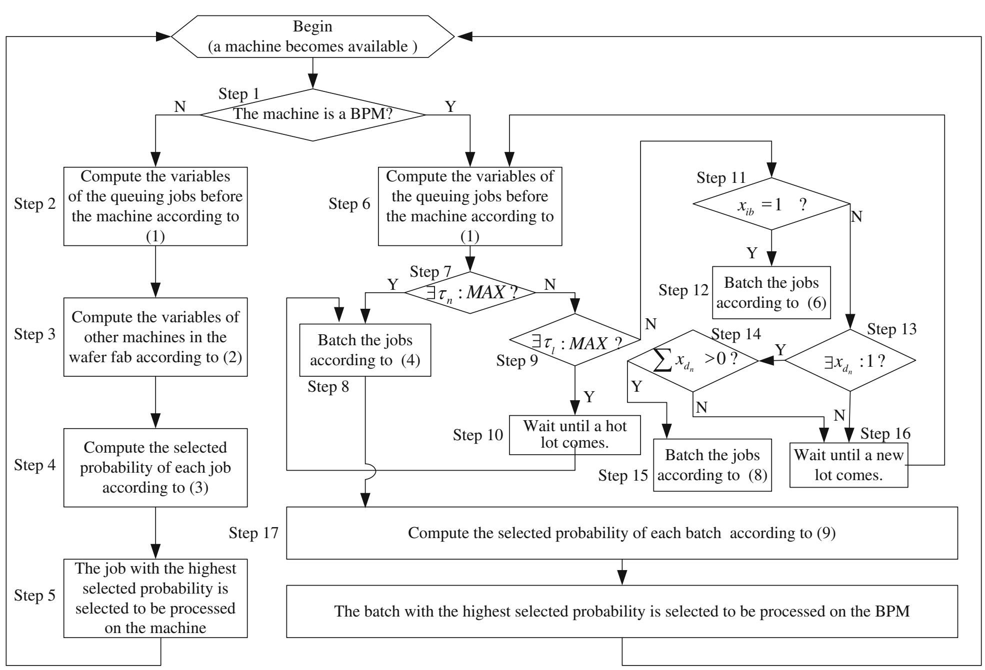

Fig. 3 The workflow of ADR

occupation time of the jobs on the machine, and the bottleneck level of the related downstream machines at the same time to guarantee fast movement and on-time delivery performance. α_1 and β_1 are the relative importance parameters to measure the importance level between the on-time delivery performance and workload balance performance.

Step 5: Select the job with the highest probability of selection to process on machine i.

Step 6: Compute the variables of the queuing jobs before machine i according to Eq. (1).

Step 7: Determine whether there are hot jobs (i.e., $\exists\tau_n$: MAX) queuing before machine i. If not, go to step 9.

Step 8: Batch the jobs queuing before machine i according to

$$\begin{aligned}
&\text{for } m = 1 : M_i\\
&\text{if } 0 \le \sum x_n|_m < B_i\\
&\text{then Select}\left\{\left(N_i^{m'}|_{x_n=0} \le \left(B_i - N_i^m|x_n = 1\right)\right)|_{\max(q_n)}\right\}\\
&\text{else if} \sum x_n|_m \ge B_i\\
&\text{then Select}\left\{\left(N_i^{m''}|_{x_n=1} = B_i\right)|_{\max((RPT_n \times FF_n)-(D_n - t))}\right\}\\
&\text{end}
\end{aligned} \tag{4}$$

The meaning of Eq. (4) is as follows. For each recipe of machine i, if the number of hot jobs using the same recipe is less than B_i, check whether there are ordinary jobs queuing before machine i using the same recipe as these hot jobs. If the number of qualified ordinary jobs is not less than the required number, select the number of jobs (i.e., $N_i^{m'}$) as the required number to compose a batch with maximum batch size according to their queuing time before machine i; otherwise, select the number of jobs (i.e., $N_i^{m'}$) as that of qualified ordinary jobs to compose a batch. If the number of hot jobs using the same recipe is no less than B_i, select the number of hot jobs (i.e.,$N_i^{m''}$) as B_i to compose a batch with the maximum batch size according to their level of urgency for meeting on-time delivery requirements. Then, go to step 17 to compute the priorities of these batches.

Step 9: Determine whether the jobs are being processed or just finished and their next step will be finished by machine i are hot jobs (i.e., $\exists\tau_1$:MAX) according to Eq. (1). If not, go to step 11.

Step 10: Wait for the hot job, then go to step 8 to batch the jobs according to Eq. (4).

Step 11: Determine whether machine i is a bottleneck according to Eq. (5).

$$\begin{aligned}
&\text{If} \quad \sum\nolimits_m N_i^m \ge \left(24B_i/\min\left(PT_m^i\right)\right),\\
&\quad \text{then} \quad x_{ib} = 1.
\end{aligned} \tag{5}$$

If not, go to step 13.

Equation (5) means that machine i is considered as a bottleneck if the number of queuing jobs exceeds its highest capacity per day (i.e., 24 h).

Step 12: Batch jobs according to their required recipe of machine i. If the number of jobs needing the same recipe exceeds the capacity of machine i, select the number of the jobs (i.e., $N_i^{m'''}$) no more than B_i to compose a batch according to their waiting time before machine i. Then, go to step 17.

$$\text{Select}\left\{\left(N_i^{m'''} \le B_i\right)|_{\max(q_n)}\right\} \tag{6}$$

Step 13: Determine whether downstream machine d_n is idle according to

$$\begin{aligned}
&\text{If} \quad N_{d_n} \le \left(24\Big/\max\left(PT_{d_n}^v\right)\right)\\
&\quad \text{then } x_{d_n} = 1.
\end{aligned} \tag{7}$$

If not, go to step 16.

Eq. (7) means that downstream machine d_n is considered as an idle machine if the number of the queuing jobs before it is no more than its lowest capacity per day (i.e., 24 h).

Step 14: Determine whether there are jobs queuing before machine i whose next step will be processed on the idle downstream machine d_n. If not, go to step 16.

Step 15: Batch the jobs according to Eq. (8) below:

$$\begin{aligned}
&\text{for } m = 1 : M_i\\
&\text{if } 0 \le \sum x_{d_n}|_m < B_i\\
&\text{then Select}\left\{(N_{d_n}^{m'} \le \left(B_i - \sum x_{d_n}|_m\right)|_{\max(q_n)}\right\}\\
&\text{else if} \sum x_{d_n}|_m \ge B_i\\
&\text{then Select}\left\{\left(N_{d_n}^{m''} = B_i\right)|_{\max(q_n)}\right\}\\
&\text{end}
\end{aligned} \tag{8}$$

The meaning of Eq. (8) is as follows. For each recipe of machine i, if the number of jobs whose next step is processed on an idle machine is less than B_i, check whether there are other jobs queuing before machine i using the same recipe. If the number of qualified jobs is not less than the required number, select $N_{d_n}^{m'}$ as the required number to compose a batch with the maximum batch size according to their queuing time before machine i; otherwise, select all qualified jobs to compose a batch. If the number of jobs whose next step is processed at an idle machine is no less than B_i, select $N_{d_n}^{m''}$ as B_i to compose a batch with the maximum batch size according to their queuing time before machine i. Then, go to step 17 to compute their priorities.

Step 16: Wait for a new job, go to step 6 to restart the dispatching decision process.

Step 17: Determine the batches' priorities according to

$$p_k = \alpha_2 \frac{\sum_{B_k} x_n}{B_i} + \beta_2 \frac{B_k}{\max_k(B_k)} - \gamma \frac{PT_k^i}{\max_k\left(PT_k^i\right)} - \sigma\left(\sum_{B_k} N'_{d_n} \Big/ \left(\sum_k \sum_{B_k} N'_{d_n} + 1\right)\right) \quad (9)$$

The first item of Eq. (9) is the proportion of hot jobs in batch k, which represents the on-time delivery performance. The second one is the ratio between the size of batch k and the maximum size in each batch, which represents cycle time, movement, and machine's utility. The third one is the ratio of the occupation time of batch k on machine i to the maximum occupation time of the batches, which represents the occupied time of batch k on machine i. The last item is the workload level of machine $i_{\text{downstream}}$, which represents the machine's utility and movement. As a result, the expected performance can be achieved by tuning α_2, β_2, γ, and σ.

4.4 The mining process of the parameters of ADR

The simulation results show that the operational performance (that is set as the number of job moves on the machines; one job finishing one step's processing is counted as a move) of the fab is different when the parameters of ADR (i.e., α_1, β_1, α_2, β_2, γ, and σ) are set differently. To pursue better operational performance, these parameters should be regulated according to the real-time running information. Considering the requirements of the real fabs, we select two kinds of real-time state information: the ratio of hot jobs (denoted as r_h) and the ratio of jobs with less than one third of photo processes left (denoted as r_p) in a fab. Then, we build following formulas for these weighting parameters

$$\begin{aligned}
\alpha_1 &= a_1 r_h + b_1 r_p + c_1 \\
\beta_1 &= a_2 r_h + b_2 r_p + c_2 \\
\alpha_2 &= a_3 r_h + b_3 r_p + c_3 \\
\beta_2 &= a_4 r_h + b_4 r_p + c_4 \\
\gamma &= a_5 r_h + b_5 r_p + c_5 \\
\sigma &= a_6 r_h + b_6 r_p + c_6
\end{aligned} \quad (10)$$

Obviously, the realization of the adaptability of ADR is transferred to a problem of determining the values of a_i, b_i, and c_i (i=1,…, 6). It pursues for the best operational performance, i.e., the largest number of moves. Then the mine workflow to the parameters of ADR is shown in Fig. 4.

Step 1: Generate a number of cases with ADR by simulations with variable values of α_1, β_1, α_2, β_2, γ, and σ. Actually, the machines with less workload have less impact on the operational performance of a fab. Thus, it is unnecessary to apply a complex dispatching rule to it. We only apply ADR to those machines with average utility rate over 60 % in the fab. Other machines are simply subject to FIFO.

Step 2: Obtain better move and bottleneck utility rate performance by training the original cases with a BPNN. Since variable values of α_1, β_1, α_2, β_2, γ, and σ result in changeable operational performance, part of the cases generated in step 1 may have worse operational performance. The parameters with better performance mined from these cases cannot be anticipated. Thus, we should further optimize the operational performance with the original cases. The optimization is realized with a BPNN. There are eight input neurons in BPNN, i.e., α_1, β_1, α_2, β_2, γ, σ, r_h, and r_p. The operational performance issues, i.e., *Move* and *Utility* (bottleneck utility rate), are taken as two output neurons of BPNN. The number of the hidden layers is set to 1 and the number of neurons in it is set to 6. To obtain optimized *Move* and *Utility*, we make some modifications on the basic BPNN training process. The improved BPNN training process is realized by a condition $O_j \geq T_j$ (i.e., output value of a training process is larger than that of samples) to guarantee the objective values (*Move* and *Utility*) are better than those of the cases.

Step 3: Obtain a number of cases of α_1, β_1, α_2, β_2, γ, σ, r_h, and r_p with optimized *Move* and *Utility* performance by training the original cases with BPNN. It means that the optimized *Move* and *Utility* are

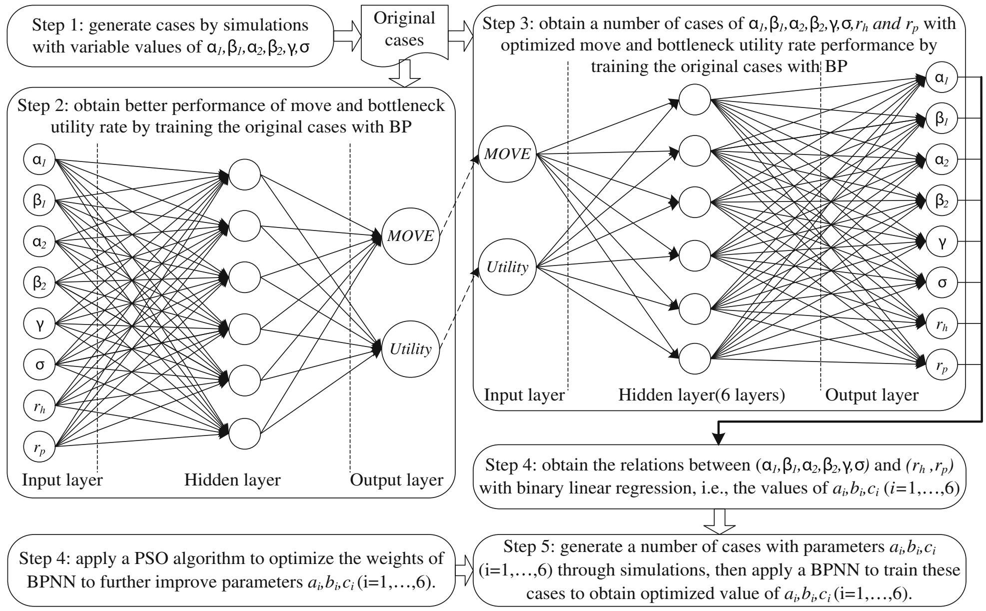

Fig. 4 The mine workflow to the parameters of ADR

set as the input neurons of BPNN, and the parameters α_1, β_1, α_2, β_2, γ, σ, r_h, and r_p are set as the output neutrons. Since the prediction process, which is based on a number of cases, here produces the optimized move-and-bottleneck utilities, the opposite reasoning process is regarded to be reasonable. The simulation results also prove it (see Table 1).

Step 4: Obtain the relations between (α_1, β_1, α_2, β_2, γ, and σ) and (r_h and r_p) with binary linear regression, i.e., the values of a_i, b_i, and c_i (i=1, …, 6). Then the adaptivity of ADR is realized.

Step 5: Generate a number of cases with parameters a_i, b_i, c_i (i=1, …, 6) by implementing ADR with variable weighting parameters during simulation. Here, we consider *Move* only. Then, apply a BPNN to train these cases to obtain optimized value of a_i, b_i, c_i (i=1, …, 6). There is only one neutron (denote move) in its input layer, 18 output neutrons (a_i, b_i, c_i) in its output layer, and a hidden layer with eight neutrons.

Step 6: Apply a PSO algorithm to optimize the weights of BPNN in step 5. The dimensionality of a particle is

$$n = (M + 1) \bullet N + (N + 1) \bullet L$$

where M is the number of input neutrons, N is the number of hidden neutrons, and L is the number of output neutrons. The position of a particle denotes the values of the weights and threshold of the BPNN.

4.5 Simulation results

A simulation model of an industrial fab is used to verify and validate ADR. Its capacity is 7,000 slices of WIP. There are seven machine groups in the fab, i.e., implanter, wet etching, defect scan area, diffusion, spurting, photo, and dry etching. There are bottleneck machines in spurting, diffusion, photo, and dry-etching groups. The ratio of the machines with over 60 % of utility rate in the fab is about 34.8 %.

We set 110 sets of (α_1, β_1, α_2, β_2, γ, and σ) randomly to run the simulations (the simulation period is set to 5 days) with

Table 1 An example demonstrating the optimized move and bottleneck machine utility

	α_1	β_1	α_2	β_2	γ	σ	r_h	r_p	Move (slice)	Utility (%)	WIP (slice)
Sample	0.85	0.15	0.7	0.1	0.1	0.1	0.2414	0.1862	10,889	0.6543	5,671
	0.05	0.95	0.01	0.01	0.97	0.01	0.3035	0.1900	6,552	0.5737	5,994
	0.35	0.65	0.4	0.2	0.2	0.2	0.3111	0.1714	12,224	0.7577	6,323
BPNN	0.49	0.51	0.22	0.36	0.15	0.27	0.2713	0.2168	12,118	0.6891	5,671
	0.47	0.53	0.25	0.27	0.26	0.22	0.2955	0.1985	7,188	0.6378	5,994
	0.42	0.58	0.17	0.44	0.13	0.26	0.3097	0.1632	12,622	0.7677	6,323

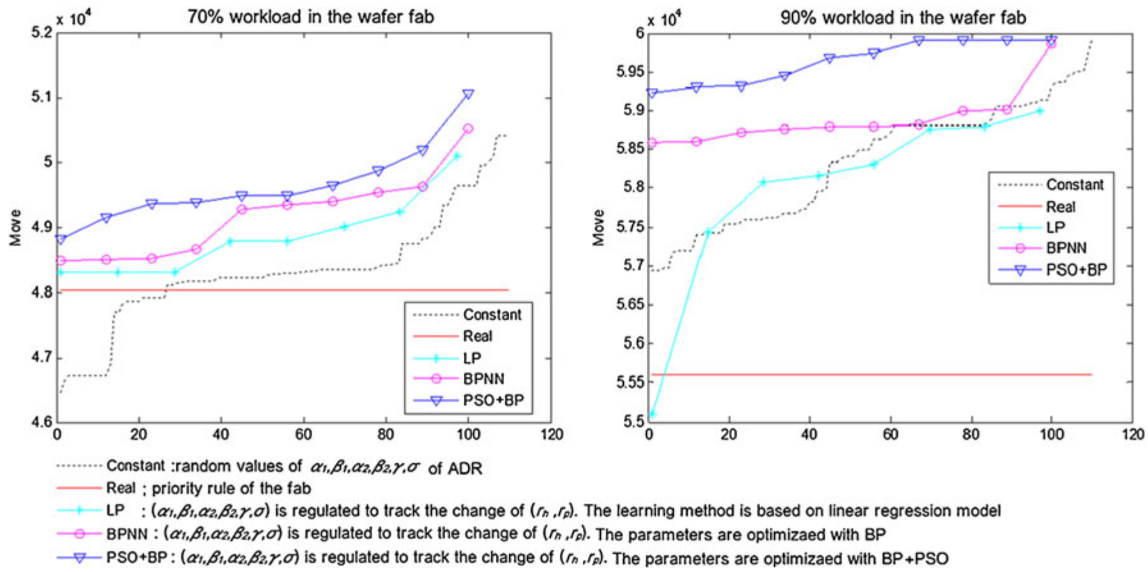

Fig. 5 Simulation results

ADR. Record the information about r_h, r_p, *Move* and *Utility* per 12 h. Then the number of study samples is 10×110.

The simulation results are shown in Fig.5 and Table 2.

We can obtain following conclusions from the simulation results.

1. The workload of the fab is about 70 %.

- The values of α_1, β_1, α_2, β_2, γ, and σ have serious impact on the operational performance of the fab. The least *Move* obtained by ADR may be lower than that of the existing rule of the fab (*Real*) by 3 %. However, the best *Move* obtained by ADR is better than that of *Real* by 13 %. The average improvement on *Move* of ADR of the 10×110 cases is about 0.64 %. Obviously, it is very important to set the values of α_1, β_1, α_2, β_2, γ, and σ reasonably.
- After building the relations between (α_1, β_1, α_2, β_2, γ, and σ) and (r_h and r_p) with a binary regression based learning algorithm, i.e., a set of constant values of a_i, b_i, and c_i (i=1, …, 6) are determined, we run eight times of simulations randomly with these 18 constant parameters. It can be seen from the simulation results that the move performance of each case is better than that of *Real*, independent to the original state of the simulations. In addition, the average *Move* is improved by 1.08 % compared to ADR with constant parameters.
- The linear regression method is not precise enough. We further improve the parameters of a_i, b_i, and c_i (i=1, …, 6) with a BPNN. The training time of BPNN is 358 s. Then we randomly run the simulations 10 times with these improved 18 constant parameters. It is shown that the *Move* performance with them is improved by 0.25 % than that of parameter values without optimization. In addition, the best *Move* is better that the best *Move* obtained with constant parameters of α_1, β_1, α_2, β_2, γ, and σ with 0.2 % improvement.
- Although the BPNN can optimize the coefficients a_i, b_i, and c_i (i=1, …, 6) of Eq. (10), the training process is a little longer and may be not convergent. To overcome these advantages, we apply a PSO algorithm to optimize the weights and thresholds of the BPNN in

Table 2 Simulation results

70 % workload in the fab			90 % workload in the fab		
Rule	Move (slice)	Improvement (%)	Rule	Move (slice)	Improvement (%)
Real	48,027	–	Real	55,607	–
Constant	48,335	+0.64	Constant	58,340	+4.91
LP	48,853	+1.72	LP	57,956	+4.22
BPNN	48,975	+1.97	BPNN	58,894	+5.91
PSO + BP	49,184	+2.41	PSO + NN	59,633	+7.24

step 5. The parameters c_1 and c_2 of the PSO are set to 1.494. Then, we run 10 times of simulations randomly with these further improved 18 constant parameters. The simulation results show that the *Move* performance with them is improved by 0.44 % than that of optimized parameter values obtained with BPNN only. In addition, the best *Move* is better that the best *Move* obtained with constant parameters of α_1, β_1, α_2, β_2, γ, and σ with 1.3 % improvement. The training time is shortened to 208 s, too.

2. The workload of the fab is about 90 %.

- On this occasion, the binary regression method failed to reach a full performance. The *Move* performance is a little less than the average *Move* obtained by ADR with constant parameters.
- After the parameters of a_i, b_i, and c_i (i=1, …, 6) is improved by BPNN and BPNN+PSO, the further improvements on *Move* are 1.69 and 1.33 %, respectively, much higher than those of 70 % workload.
- The data-based scheduling optimization method can achieve a better performance under heavier workload environments. It is consisted with the expectation from industries.

5 Conclusion

The development of information technology helps advancing the automation level of complex manufacturing systems. There are much more on- and off-line data in their information systems, which are useful knowledge, rules and optimal decisions for solving their scheduling problems. A data-based scheduling framework is proposed together with its utilization introduced by taking a semiconductor manufacturing system as an example. In addition, an example is given from a real fab to demonstrate the potentials of data-based scheduling optimization methods. The future work is set to further improve them and apply the research results to the actual production environments.

Acknowledgment This work was supported in part by National Natural Science Foundation of China (numbers 50905129 and 61034004) and the Fundamental Research Funds for the Central Universities.

References

1. Kusiak A (2001) Rough set theory a data mining tool for semiconductor manufacturing. IEEE Trans Electron Packing Manuf 24 (1):44–50
2. Kusiak A (2000) Decomposition in data mining an industrial case study. IEEE Trans Electron packing Manuf 23(4):345–353
3. Kusiak A (2001) Feature transformation methods in data mining. IEEE Trans Electron Packing Manuf 24(4):214–221
4. Chen YM, Miao DQ, Wang RZ (2010) A rough set approach to feature selection based on ant colony optimization. Pattern Recognit Lett 31(3):226–233
5. Shine YR, Su CT (2002) Attribute selection for neural network based adaptive scheduling systems in flexible manufacturing systems. Int J Adv Manuf Technol 20(7):532–544
6. Shiue YR, Guh RS (2006) The optimization of attribute selection in decision tree-based production control systems. Int J Adv Manuf Technol 28(7–8):737–746
7. Shiue YR, Guh RS (2006) Learning based multi pass adaptive scheduling for a dynamic manufacturing cell environment. Rob Comput Integr Manuf 22(3):203–216
8. Shiue YR (2009) Development of two-level decision tree-based real-time scheduling system under product mix variety environment. Robot Comput-Integr Manuf 25(4-5):709–720
9. Shiue YR, Guh RS, Tseng TY (2009) GA based learning bias selection mechanism for real time scheduling systems. Expert Syst Appl 36(9):11451–11460
10. Hu CH, Su SF (2004) Hierarchical clustering methods for semiconductor manufacturing data. Proceedings of the 2004 IEEE International Conference on Networking, Sensing Control, Taiwan, pp 1063–1068
11. Chen T (2007) Predicting wafer-lot output time with a hybrid FCM–FBPN approach. IEEE Trans Dystem Man Cybern B Cybern 37(4):784–793
12. Chen T (2007) An intelligent hybrid system for wafer lot output time prediction. Adv Eng Inform 21(1):55–65
13. Koonce DA, Tsai SC (2000) Using data mining to find patterns in genetic algorithm solutions to a job shop schedule. Comput Ind Eng 38(3):361–374
14. Li XN (2006) Application of data mining in scheduling of single machine system. Ph.D. dissertation, Iowa State University
15. Rafinejad SN, Ramtin F, Arabani AB (2009) A new approach to generate rules in genetic algorithm solution to a job shop schedule by fuzzy clustering. Proceedings of the World Congress on Engineering and Computer Science, USA
16. Gradisar D, Music G (2007) Automated Petri-net modeling based on production management data. Math Comput Model DynSyst 13 (3):267–290
17. Gradisar D, Music G (2007) Production-process modeling based on production-management data: a Petri–Net approach. Int J Comput Integr Manuf 20(8):794–810
18. Mueller R, McGinnis LF (2007) Automatic generation of simulation models for semiconductor manufacturing. Proceedings of the 2007 Winter Simulation Conference. Washington, pp 648–657
19. Ye K, Qiao F, Ma YM (2010) General structure of the semiconductor production scheduling model. Appl Mech Mater 20–23:465–469
20. Bagchi S, Baseman RJ, Davenport A, Natarajan R, Slonim N, Weiss S (2010) Data analytics and stochastic modeling in a semiconductor fab. Appl Stoch Model Bus Ind 26(1):1–27
21. Hosoe H, Knanamori N, Yoshida K (2007) The methods of data collection and tool processing time estimation in lot processing. In: International Symposium on Semiconductor Manufacturing. Santa Clara, Institute of Electrical and Electronics Engineers Inc., Piscataway, pp 241–244
22. Chen T (2003) A fuzzy back propagation network for output time prediction in a wafer fab. Appl Soft Comput 2(3):211–222
23. Chen T (2010) Intelligent scheduling approaches for a wafer fabrication factory. J Intell Manuf 23(3):897–911
24. Chen T, Wang YC (2009) A nonlinear scheduling rule incorporating fuzzy-neural remaining cycle time estimator for scheduling a semiconductor manufacturing factory—a simulation study. Int J Adv Manuf Technol 45(1–2):110–121

25. Chung SH, Lai CM (2006) Job releasing and throughput planning for wafer fabrication under demand fluctuating make-to stock environment. Int J Adv Manuf Technol 31(3–4):316–327
26. Pearn WL, Chung SH, Lai CM (2007) Due date assignment for wafer fabrication under demand variate environment. IEEE Trans Semicond Manuf 20(2):165–175
27. Chang PC, Wang YW, Liu CH (2006) Combining SOM and GA-CBR for flow time prediction in semiconductor manufacturing factory. Lect Notes Comput Sci 4259:767–775
28. Chen T (2006) A hybrid SOM-BPN approach to lot output time prediction in a wafer fab. Neural Process Lett 24(3):271–288
29. Chen T (2007) A hybrid look-ahead SOM-FBPN and FIR system for wafer-lot-output time prediction and achievability evaluation. Int J Adv Manuf Technol 35(5–6):575–586
30. Chang PC, Liao TW (2006) Combining SOM and fuzzy rule base for flow time prediction in semiconductor manufacturing factory. Appl Soft Comput 6(2):198–206
31. Wang WP, Ze C (2008) A neuro-fuzzy based forecasting approach for rush order control applications. Expert Syst Appl 35(1–2):223–234
32. Arredondo F, Martinez E (2010) Learning and adaptation of a policy for dynamic order acceptance in make-to-order manufacturing. Comput Ind Eng 58(1):70–83
33. Shukla KS, Tiwari MK, Son YJ (2008) Bidding-based multi-agent system for integrated process planning and scheduling a data-mining and hybrid tabu-SA algorithm-oriented approach. Int J Adv Manuf Technol 38(1–2):163–175
34. Li TS, Huang CL, Wu ZY (2006) Data mining using genetic programming for construction of a semiconductor manufacturing yield rate prediction system. J Intell Manuf 17(3):355–361
35. Hsu SC, Chien CF (2007) Hybrid data mining approach for pattern extraction from wafer bin map to improve yield in semiconductor manufacturing. Int J Prod Econ 107:88–103
36. Rokach L, Maimon O (2006) Data mining for improving the quality of manufacturing: a feature decomposition approach. J Intell Manuf 17(3):285–299
37. Shaw MJ, Park S, Raman N (1992) Intelligent scheduling with machine learning capabilities the induction of scheduling knowledge. IIE Trans 24(2):156–168
38. Park SC, Raman N, Shaw MJ (1997) Adaptive scheduling in dynamic flexible manufacturing systems. IEEE Trans Robot Autom 13(4):486–502
39. Lee I, Shaw MJ (2000) A neural-net approach to real time flow-shop sequencing. Comput Ind Eng 38(1):125–147
40. Lee KK (2008) Fuzzy rule generation for adaptive scheduling in a dynamic manufacturing. Appl Soft Comput 8(4):1295–1304
41. Priore P, Fuente D, Pino R (2001) Learning-based scheduling of flexible manufacturing systems using case-based reasoning. Appl Artif Intell 15(10):949–963
42. Priore P, Fuente D, Puente J, Parreno J (2006) A comparison of machine-learning algorithms for dynamic scheduling of flexible manufacturing systems. Eng Appl Artif Intel 19(3):247–255
43. Priore P, Fuente D, Gomez A, Puente J (2001) A review of machine learning in dynamic scheduling of flexible manufacturing systems. Artif Intell Eng Design Anal Manuf 15(3):251–263
44. Li DC, Wu CH, Chang FM (2005) Using data-fuzzification technology in small data set learning to improve FMS. Int J Adv Manuf Technol 27(3–4):321–328
45. Li DC, Chen LS, Lin YS (2003) Using functional virtual population as assistance to learn scheduling knowledge in dynamic manufacturing environments. Int J Prod Res 41(17):4011–4024
46. Li DC, Wu CH, Tsa TI, Fengming MC (2006) Using mega-fuzzification and data trend estimation in small data set learning for early FMS scheduling knowledge. Comput Oper Res 33 (6):1857–1869
47. Aydin ME, Öztemel E (2000) Dynamic job-shop scheduling using reinforcement learning agents. Robot Auton Syst 33(2–3):169–178
48. Yang HB, Yan HS (2009) An adaptive approach to dynamic scheduling in knowledgeable manufacturing cell. Int J Adv Manuf Technol 42(3–4):312–320
49. Koonce DA, Gandhi SA (2004) Applying inductive logic programming for knowledge discovery in genetic algorithms solutions to a job shop schedule. Proceedings of the IIE 2004 Annual Conference, Houston
50. Weckman GR, Ganduri CV, Koonce DA (2008) A neural network job-shop scheduler. J Intell Manuf 19(2):191–201
51. Shahzad A, Mebarki N (2010) Discovering dispatching rules for job shop scheduling problem through data mining. The 8th International Conference of Modeling and Simulation, Hammamet
52. Chaudhuri A, De K (2010) Job scheduling problem using rough fuzzy multilayer perception neural networks. J Artif Intell Theory Appl 1(1):4–19
53. Kumar S, Rao CSP (2009) Application of ant colony, genetic algorithm and data mining-based techniques for scheduling. Robot Comput-Integr Manuf 25(6):901–908
54. Li XN, Olafsson S (2005) Discovering dispatching rules using data mining. J Sched 8(6):515–527
55. Olafsson S, Li XN (2010) Learning effective new single machine dispatching rules from optimal scheduling data. Int J Prod Econ 128:118–126
56. Choi HS, Kim JS, Lee DH (2011) Real-time scheduling for reentrant hybrid flow shops: a decision tree based mechanism and its application to a TFT-LCD line. Expert Syst Appl 38 (4):3514–3521
57. Kwak C, Yih Y (2004) Data-mining approach to production control in the computer-integrated testing cell. IEEE Trans Robot Autom 20(1):107–116