Proceedings of the 2020 Winter Simulation Conference
K.-H. Bae, B. Feng, S. Kim, S. Lazarova-Molnar, Z. Zheng, T. Roeder, and R. Thiesing, eds.

TOWARDS SITUATION AWARE DISPATCHING IN A DYNAMIC AND COMPLEX MANUFACTURING ENVIRONMENT

Chew Wye Chan
Boon Ping Gan

D-SIMLAB Technologies Pte Ltd
8 Jurong Town Hall Road, #23-05
Singapore, 609434, SINGAPORE

Wentong Cai

School of Computer Science and Engineering
Nanyang Technological University
Singapore, 639798, SINGAPORE

ABSTRACT

Dispatch rules are commonly used to schedule lots in the semiconductor industry. Earlier studies have shown that changing dispatch rules that react to a dynamic manufacturing situation improves the overall performance. It is common to use discrete event simulation to evaluate dispatch rules under different manufacturing situations. On the other hand, machine learning method is shown to be useful in learning the relationship of a manufacturing situation and the dispatch rules to generate dispatching knowledge. In this work, we use simulation and machine learning methods to generate dispatching knowledge and define features that are relevant in a dynamic product mix situation. However, more features will increase the risk of overfitting the machine learning model. Hence, dimension reduction methods are explored to reduce overfitting and improve generalization of the model. Simulation results show that this approach can adapt the dispatch rule combination and achieve a comparable factory performance measurement.

1 INTRODUCTION

In a complex manufacturing environment such as the semiconductor industry, scheduling is important to optimize the performance of a factory. However, scheduling in these problems are NP-complete (Garey and Johnson 1979), and in practice, dispatching rules are usually used (Metan and Sabuncuoglu 2005). By applying a dispatch rule, the list of lots in the queue of a machine is given a priority ranking, and the highest priority lot will be the first to dispatch to the machine when the machine becomes available. Global dispatch rules are the same for all work centers, whereas local dispatch rules are specific to a work center aiming at maximizing the capacity utilization. Hence, in a real factory, multiple dispatch rules are used in different work centers.

In order to achieve a better performance in a dynamic manufacturing environment, a multi-pass strategy is proposed (Wu and Wysk 1989; Shiue and Su 2003) as compared to a single-pass approach. A multi-pass approach is to adapt to the best dispatch rules for a scheduling period; whereas a single-pass approach is to use one dispatch rule for all scheduling period. The challenge of the multi-pass approach is the determination of the best dispatch rules for the next scheduling period, and this decision often needs to be made in real-time.

There are various studies to use machine learning techniques to learn the dispatch rules for a manufacturing situation. The idea is to create a set of features that could be best to represent the current manufacturing situation and predict the best dispatch rules to be applied to achieve a category of user-defined performance. In (Shiue and Su 2003), a decision tree learning approach was used to generate the machine learning model for dispatch rule prediction. For the model to work with high prediction accuracy, the training data should represent as many manufacturing situations as possible. Still, it is time-consuming to generate various manufacturing situations. Hence, multiple existing works proposed using the clustering algorithm and reinforcement learning to learn the best dispatch rules for a scheduling period based on a cluster of similar system status (Shiue et al. 2011; Shiue et al. 2018).

It has been shown that using feature selection procedure reduces over-fitting of the training data and delivers better generalization ability (Shiue and Su 2003). On the other hand, omitting important feature will significantly reduce the learning process and affect prediction accuracy. Generating new attributes automatically using frequent itemset and arithmetic operations is also shown to be helpful in classification learning (Olafsson and Li 2010). Although more features will increase the discriminating power of the machine learning algorithm, especially in a dynamic manufacturing situation, it runs the risk of overfitting the machine learning model. Hence, it is essential to generate more features that could capture the manufacturing situations but at the same time, reduce the number of nonimportant features to avoid overfitting and improve generalization of the machine learning model. In a dynamic product mix situation, a high number of features will result in over-fitting, and hence dimension reduction methods are explored to improve the generalization ability of the machine learning model.

This study examines the process of generating a multi-pass system using a machine learning algorithm. The remainder of the paper is organized as follows. In Section 2, we provide an overview of multi-pass systems. In Section 3, we discuss the knowledge base generation of a multi-pass system. In Section 4, we demonstrate the knowledge base generation process using a factory model and show the performance comparison using an on-line simulation. Finally, Section 5 concludes the paper and outlines future research directions.

2 MULTI-PASS SYSTEM

Manufacturing system changes dynamically in real operation, and studies have found that changing dispatching rule results in better factory performance than using a single dispatching rule (Wu and Wysk 1989). However, there are two fundamental requirements of a multi-pass system: i) to interpret real-time manufacturing environment; and ii) to evaluate the dispatch rules to be applied without delaying real operation (Nakasuka and Yoshida 1992). Hence, a knowledge-based system uses the knowledge of the relationship between the manufacturing situations and the dispatch rules to select a dispatch rule (Priore et al. 2014). In order to acquire the knowledge, iterative simulation is often used to explore multiple manufacturing situations to build a knowledge base for dispatch rules selection (Nakasuka and Yoshida 1992). In order to describe the changing of dispatch rules in a specified time horizon, Wu and Wysk (1989) used the term "multi-pass", Nakasuka and Yoshida (1992) used the term "dynamic scheduling", and Su and Shiue (2003) used the term "adaptive scheduling". In this study, the term "multi-pass" is adopted.

As the multi-pass approach is the study of changing the dispatch rules in a specific time horizon, the variation of scheduling period on the machine learning model is being studied (Metan et al. 2010). The scheduling period is defined as the period to apply a dispatch rule. It is found that a short scheduling period results in the system switching rule too frequently and affects the long-run performance (Metan and Sabuncuoglu 2005). Hence, in our study, we have fixed the scheduling period to be one day to avoid frequent changes in dispatch rule.

In a real factory, there are varying types of ad-hoc situations that could impact on the factory performance, such as the machine unavailability, rush orders, varying product mix, etc. Applying better dispatch rules can mitigate the effect of randomness in the manufacturing environment (Schoemig 1999). However, for a multi-pass approach to react to this randomness of the manufacturing environment, dispatching knowledge must be acquired first. Simulating all possible scenarios to generate the manufacturing states is time consuming and wasteful. Different randomness might result in the same manufacturing state. Hence, self-organizing map (SOM) is proposed to group different manufacturing states, and only distinct groups are simulated to acquire dispatching knowledge (Shiue et al. 2011).

3 MULTI-PASS SYSTEM KNOWLEDGE BASE GENERATION

A multi-pass system can modify the dispatch rules based on the current manufacturing situation. A knowledge-based approach is used to generate a relationship between a manufacturing situation and the dispatch rules to be applied at the moment using a machine learning algorithm (Priore et al. 2014). However, the manufacturing performance by using a knowledge-based approach would be negatively affected if the training set is insufficient (Priore et al. 2001). Hence, training data generation is important to create different manufacturing situations. A general overview of the self-adaptive dispatching system is shown in Figure 1.

In the training data generator, the factory simulation model is used to evaluate the performance of the dispatch rules with a simulator. The data preprocessing is a list of steps that transform the data generated from the simulator to the machine learning algorithm input. The training data collection is to select the best dispatch rule in a defined period for the training data. After the training data is generated, the machine learning algorithm generates a knowledge base that relates the system state with the dispatch rule. The control system captures the current system state and modifies the dispatch rules for the manufacturing system using dispatch rule knowledge. The performance and manufacturing data from the manufacturing system will then be passed to the knowledge refinement module to decide the generation of more training data.

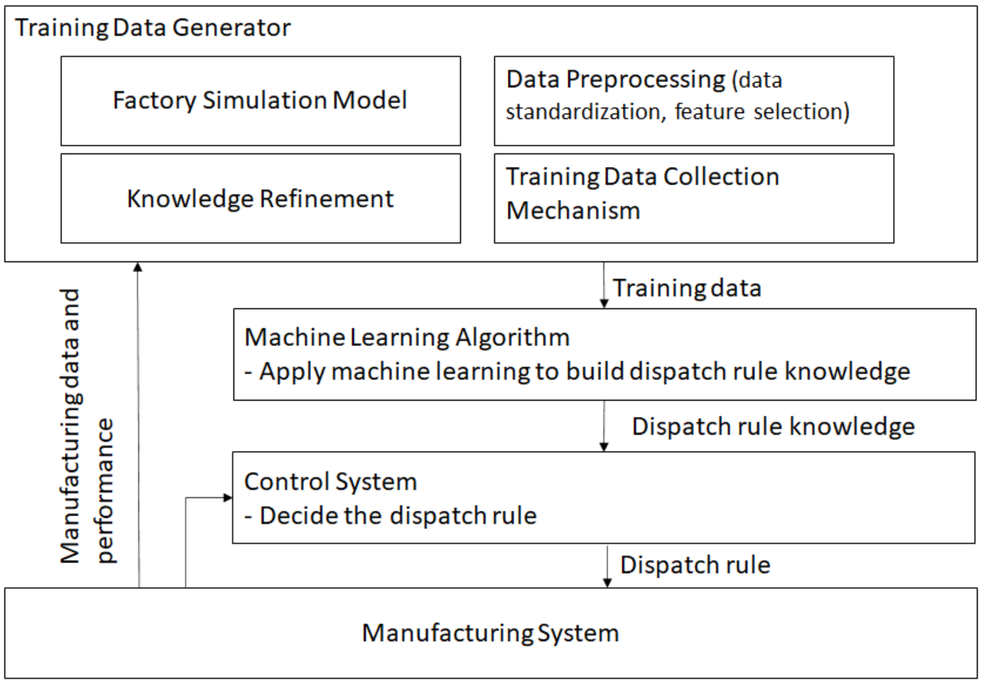
Training Data Generator
Factory Simulation Model
Data Preprocessing (data standardization, feature selection)
Knowledge Refinement
Training Data Collection Mechanism
Training data
Machine Learning Algorithm
- Apply machine learning to build dispatch rule knowledge
Dispatch rule knowledge
Control System
- Decide the dispatch rule
Dispatch rule
Manufacturing data and performance
Manufacturing System

Figure 1: General overview of a self-adaptive dispatch rule system.

3.1 Factory Key Performance Measurement

Factory key performance could be measured in terms of throughput and average cycle time within a specific period. Assuming the set of completed lots within the time period t is $CompleteLot_t$, the cycle time of lot a is $CycleTime_a$. The formula for throughput and average cycle time is given in equation (1) and equation (2) respectively.

The shortcoming of using equation (1) and equation (2) to measure throughput and average cycle time is that they only consider the lots that had completed. In a complex wafer manufacturing environment, where the cycle time of a product could be weeks or even months, these key performance measurement formulas do not properly consider the lots that are still work-in-progress. Thus, Hansch (2015) has suggested the use of dynamic daily going rate (*dynDGR*) as a measurement of throughput and dynamic cycle time (*dynCT*) as a measurement of average cycle time. Equation (3) shows the definition of $dynDGR_i$ for product i, where DGR_{li} represents the daily output of operation l for product i and L_i represents the total number of operations for product i. Equation (4) shows that *dynDGR* is the summation of $dynDGR_i$ for all products I. Equation (5) shows the definition of *dynCT*, where WIP is the total number of lots in work. As *dynCT* is derived from *dynDGR*, we will be using *dynCT* as the main key performance measurement for the training data generation.

$$Throughput = |CompleteLot_t| \tag{1}$$

$$AvgCycleTime = \frac{\sum_{a \in CompleteLot_t} CycleTime_a}{|CompleteLot_t|} \tag{2}$$

$$dynDGR_i = \frac{1}{L_i} * \sum_{l=1}^{L_i} DGR_{li} \tag{3}$$

$$dynDGR = \sum_{i=1}^{I} dynDGR_i \tag{4}$$

$$dynCT = \frac{WIP}{dynDGR} \tag{5}$$

3.2 Training Data Representation

A triplet of $\{\mathbf{P}, \mathbf{S}, \mathbf{D}\}$ is used to represent the training data (Shiue and Su 2003). *P* represents the user-defined performance; S represents the features; *D* represents the dispatch rules that perform best under the current system attribute and user-defined performance. Earlier studies have listed the features in generating the dispatching knowledge for the multi-pass system (Park et al. 1997; Su and Shiue 2003; Shiue et al. 2011).

However, the features suggested in the above-mentioned literature are factory level features. In a complex manufacturing environment with different product mixes, using only factory level features is not sufficient to represent the current state of the factory. For example, according to little's law (Ignizio 2009), two different products with the same cycle time but different due-date tightness will have the same level of WIP if the factory throughput rate remains the same for both products. However, due to the difference in due-date tightness, a due date related dispatch rule will be preferable in some cases as compared to other dispatching rules. Hence, product level features are required to differentiate the dispatching requirements under different product mix.

Table 1 shows the different categories of features that could potentially be used in the training data. In this study, only different product mix scenario is generated. Hence, we have chosen the factory and the product category for feature consideration. Table 4 shows the lot related features. The lot related features are used to generate the fab and product features shown in Table 3. Features are further classified into statistics features (e.g., $OprT^{St}$ in Table 3) and summation features (e.g., *SumJ* in Table 3). For each statistics feature, four summary statistics (*St*) are calculated. They are minimum (*Mi*), maximum (*Ma*), average (*Me*), and standard deviation (*Sd*) (that is, $St \in \{Mi, Ma, Me, Sd\}$). Hence, the total number of features can be determined by:

$$NumberofProducts \times NumberofProductStatisticsFeatures \times NumberofSummaryStatistics + NumberofProducts \times NumberofProductSummationFeatures + NumberofFactoryStatisticsFeatures \times NumberofSummaryStatistics + NumberofFactorySummationFeatures$$

For example, for a factory model with two products, there is a total of 87 features.

As we have chosen *dynCT* as the key performance measurement in our study, we selected dispatch rules that will be affected by the lot cycle time as candidate dispatch rules. The selected dispatching rules are shown in Table 2.

Table 1: Category of features for the manufacturing environment.

Feature Category	**Description**
Factory	Features that provide fab level information, such as overall factory WIP.
Product	Features that provide product level information, such as product WIP.
Lot	Features that provide lot level overview, such as lot wait time.
Work center	Features that provide work center level overview, such as uptime.
Equipment	Features that provide equipment level overview, such as equipment uptime.

Table 2: Candidate dispatch rules in the study.

Dispatch Rule	**Description**
SIO	Select the lot with the shortest operation process time.
SPT	Select the lot with the shortest product process time.
CR	Select the lot with minimum ratio between remaining due date time and remaining processing time.

3.3 Training Data Collection Mechanism

The training data collection mechanism as described by (Shiue and Su 2003) is an exhaustive search for every combination of manufacturing situations and dispatch rules. It requires the training data generator to exhaustively explore the combination of dispatch rules with a multi-pass simulation before deciding the training data by observing the performance for the whole simulation period. However, we collect training data by running a single-pass simulation and compare the factory level $dynCT_t$ for each time period t.

Multiple single-pass simulations with different dispatch rules are executed on a steady-state simulation model for a predefined simulation run length to generate the training data. These single-pass simulation models share the same model characteristics such as wafer start pattern and machine availability except only for dispatch rules. After the warm-up time, the simulation period is divided by time period t, as shown in Table 5. For each time period, the features of the model are captured at the start of the time period, and the performance measurement is calculated at the end of the time period. Table 5 shows a numeric example of the features and performance measurement for the training data collection. As wafer start and machine availability are the same between the same time period of different single-pass simulation, it is assumed that $dynCT_t$ of the time period is comparable across the different single-pass simulation. The features corresponding to the best performing dispatch rule in a time period, in terms of dynamic cycle time, are chosen as the training data.

3.4 Dimensionality Reduction

High dimensional data is shown to improve the discriminating power of the classifier but increases the processing time (Lee and Landgrebe 1993). Manually generating more features based on domain knowledge and the raw production data is shown to be important in the learning of the relationship between manufacturing situations and dispatch rules (Chen and Yih 1996). Furthermore, new composite features that are automatically generated via frequent itemset have also shown to provide new insight on the dispatch rule (Li and Olafsson 2005). However, performing dimension reduction is important to enhance the generalization ability of the knowledge base and reduce overfitting (Shiue and Su 2003; Priore et al. 2014). Dimension reduction could be achieved by feature selection or projection pursuit.

Table 3: Factory and production features in the study.

Feature Type	Feature	Description
FAB	$OprT^{St}$	Statistic summary of current operation processing time of all lots in the factory.
FAB	$ProT^{St}$	Statistic summary of total processing time of all lots in the factory.
FAB	$RemT^{St}$	Statistic summary of remaining processing time of all lots in the factory.
FAB	$SlkT^{St}$	Statistic summary of slack time of all lots in the factory.
FAB	$LtnA^{St}$	Statistic summary lateness of all lots in the factory.
FAB	$QueT^{St}$	Statistic summary of queue time of all lots in the factory.
FAB	$RmdD^{St}$	Statistic summary of remaining due time of all lots in the factory.
FAB	$SumJ^{St}$	The total number of lots in the factory.
PRODUCT	$OprT^{PdSt}$	Statistic summary of current operation processing time of all lot with product Pd.
PRODUCT	$ProT^{PdSt}$	Statistic summary of total processing time of all lot with product Pd.
PRODUCT	$RemT^{PdSt}$	Statistic summary of remaining processing time of all lot with product Pd.
PRODUCT	Slk^{PdSt}	Statistic summary of slack time of all lot with product Pd.
PRODUCT	$LtnA^{PdSt}$	Statistic summary of lateness of all lot with product Pd.
PRODUCT	$QueT^{PdSt}$	Statistic summary of queue time of all lot with product Pd.
PRODUCT	$RemdD^{PdSt}$	Statistic summary of remaining due time of all lot with product Pd.
PRODUCT	$SumJ^{PdSt}$	The total number by product Pd lots in the system.

Feature selection is to select a subset of features from the original features that yield the minimum generalization error (Vergara and Estévez 2014). The features that are selected still retain the original data and information prior feature selection process. Information retention of the feature is important when new insights are needed to generate new knowledge. However, due to the dynamic nature of the manufacturing environment, feature selection might provide a different set of features under different product mix situations. If the product mix is dominated by a particular group of products in the training data, then feature selection would select the features that are affected by the group of the products due to sparse data from other low volume products. On the other hand, the relationship between the feature and the training label is also important in a supervised learning algorithm. Hence, in this paper, we use mutual information to select the top 20-percentile features (Kraskov et al. 2004).

Projection pursuit is another dimensionality reduction method. Principle Component Analysis (PCA) is a common method for projection pursuit. The goal of the projection pursuit is to find a k orthonormal vector that could best represent the training data and then project the training data with the vector (Han

Table 4: Lot feature.

Lot Attribute	Description
$OprT_i$	Current operation processing time of lot i.
$ProT_i$	Total processing time of lot i.
$RemT_i$	Remaining processing time of lot i.
$SlkT_i$	Slack time of lot i.
$LtnA_i$	Lateness of lot i.
$QueT_i$	Queue time of lot i.
$RmdD_i$	Remaining due time of lot i.

Table 5: Example of training data selection of two dispatch rule.

Time Period, t	Dispatch Rule 1, d_1		Dispatch Rule 2, d_2		Result Min($dynCT_t, d_1$, $dynCT_t, d_2$)	Selected Feature
	Feature (S_t, d_1)	Performance ($dynCT_t, d_1$)	Feature (S_t,d_2)	Performance ($dynCT_t$,d_2)		
1	S_1, d_1	$dynCT_1, d_1$	S_1, d_2	$dynCT_1, d_2$	$dynCT_1, d_1$	S_1, d_1
2	S_2, d_1	$dynCT_2, d_1$	S_2, d_2	$dynCT_2, d_2$	$dynCT_2, d_1$	S_2, d_1
3	S_3, d_1	$dynCT_3, d_1$	S_3, d_2	$dynCT_3, d_2$	$dynCT_3, d_2$	S_3, d_2

et al. 2012). The value k represents the number of significant components that provide information about the variance of the data. By eliminating components that have low variance, the number of the resulting dimensions is reduced. However, PCA knows only about the variance of components and chooses the components that best represent the variance. In a supervised learning algorithm where the training label is provided in the training data, which is the dispatch rule (D), the PCA does not consider the relationship of the projected data with the training label. Hence, Linear Discriminant Analysis (LDA) is used in our attempt to reduce dimensionality by projection pursuit (Hastie et al. 2009). LDA is closely related to PCA but with the additional consideration of promoting separation by the training label. The objective of LDA is to find a vector that could promote the distance between the projected data of each training label and minimize the variance in the projected data of each label. The k value of LDA is bound by the number of dispatch rules in the training data.

4 EXPERIMENT

4.1 Simulation Model Construction and Training Example Generation

The case study uses a four work centers factory model to generate the training data (Ignizio 2009). The model is verified by comparing the simulated throughput of the factory with the theoretical capacities of the factory. Table 6 shows the number of visits to each work center by different products. Table 7 is a summary of the product and product mix used in the factory model. Five different products are created by repeating the operations from PRODUCT_1 by a factor. The model is tuned so that the fab is in steady-state and factory utilization is at 90% without machine interruptions. Mix_Short has a product mix that is dominated by the short process time products; Mix_Equal has a product mix of equal weightage of all products, and Mix_Long has a product mix that is dominated by long process time products.

The lot start rate is exponentially distributed with a mean of 1 day, and 40 different random seeds are used to vary the lot start pattern. Each simulation is run for 30 days, with 10 days of warm-up period. The data collection period is set to be 1 day, and this will result in 20 data points for each arrival pattern. Hence, for a single product mix, we will generate 800 data points and a total of 2400 data points for the 3 different product mix. As there are 5 different products, there is a total of 174 features generated.

Table 6: Work center description.

Work center ID	**Work center Name**	**No. of machine**	**No of visits PID 1**	**No of visits PID 2**	**No of visits PID 3**	**No of visits PID 4**	**No of visits PID 5**	**Mean Process Time (minute per lot)**
1	WC_LITHO	45	3	6	9	12	15	79.3
2	WC_DEP	60	4	8	12	16	20	79.8
3	WC_ETCH	60	4	8	12	16	20	76.4
4	WC_STRIP	30	2	4	6	8	10	74.6

Table 7: Product mix ratio used in this study.

PID	Product Name	Number of Operations	Product Process Time (day)	Product mix ratio (%)		
				Mix_Short	Mix_Equal	Mix_Long
1	PRODUCT_1	13	0.7	84	20	5
2	PRODUCT_2	26	1.4	6	20	4
3	PRODUCT_3	39	2.1	4	20	3
4	PRODUCT_4	52	2.8	3	20	3
5	PRODUCT_5	65	3.5	3	20	85

4.2 Features and Dimension Reduction

Two different types of features are collected during the training data collection process. "FAB" training data features only contain factory level features, while "FAB_PRODUCT" training data features include features that are both product level and factory level. As the product mix ratio changes, it is logical to assume that "FAB_PRODUCT" features are better than "FAB" features because more features increase the discriminating power of the classification data and result in better prediction accuracy. However, as more features are introduced to the training process, the training time will be increased. Furthermore, features that have no data variance such as minimum and maximum of product process time ($ProT^{Mi}$, $ProT^{Ma}$) is also redundant and should be removed. Hence, a feature selection algorithm is important to reduce the unimportant features of the training data and achieved dimension reduction.

In order to select features that have a higher dependence on the classification label, mutual information (MI) score is calculated (Kraskov et al. 2004). A higher MI score means higher dependency between the feature and the training label. Figure 2 shows the feature selection process selecting the top 20% of these features that achieve a high MI score.

Projection pursuit is another dimension reduction method. In this example, we have chosen linear discriminant analysis (LDA) (Hastie et al. 2009). LDA projects the features into a lower dimension to maximize the separation between the classes. The number of dimensions is reduced to k-1 where k is the total number of class in the training data. As we have chosen three dispatch rules to generate our training data, hence, it is possible to visualize the LDA projection in a two-dimensional plot. Figure 3 shows the LDA projection of the training data using "FAB" and "FAB_PRODUCT" features. As more features are provided in "FAB_PRODUCT" the class separation is better than the use of "FAB" features.

Table 8 shows a summary of the total number of features after applying dimension reduction methods. While projection pursuit and features selection are both dimension reduction methods, the result of projection pursuit is the projection of the original feature using the discriminant function. Hence, the feature meaning after the projection pursuit does not carry the same meaning of its original feature.

Table 8: Total number of features after applying dimension reduction on the different feature type.

Feature Type	**Dimension Reduction**	**Total Number of Features**
FAB	No	29
FAB	Feature Selection	6
FAB	Projection Pursuit	2
FAB_PRODUCT	No	174
FAB_PRODUCT	Feature Selection	35
FAB_PRODUCT	Projection Pursuit	2

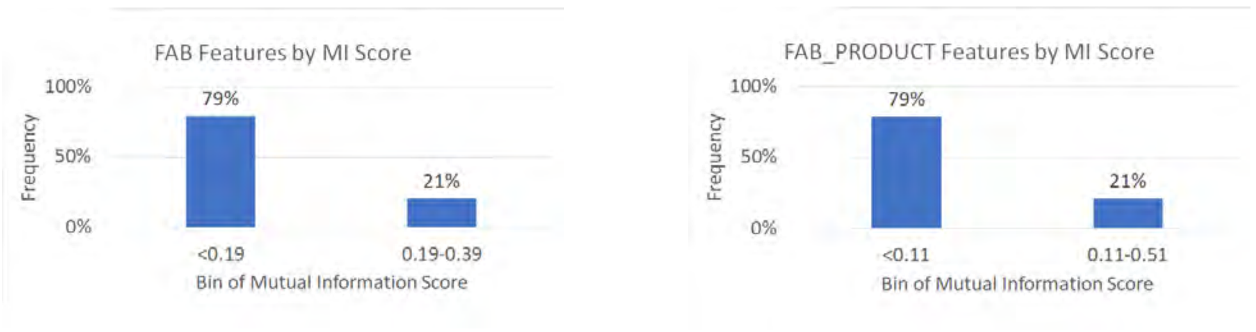

Figure 2: Frequency diagram of FAB and FAB_PRODUCT features by mutual information score.

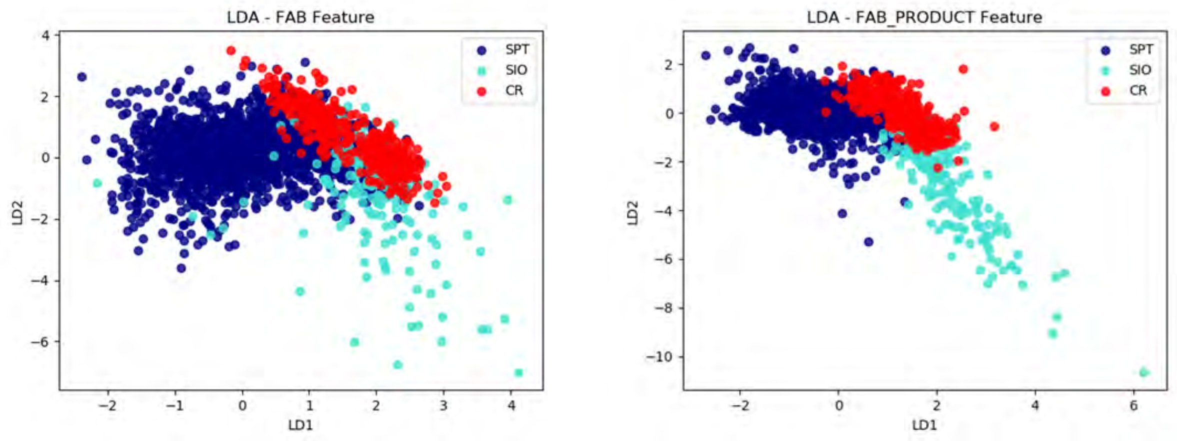

Figure 3: LDA data projection of FAB and FAB_PRODUCT features.

4.3 Cross-Validation of Machine Learning Model

In this paper, we used the support vector classification (SVC) to generate the machine learning model (Chang and Lin 2011). We use 10-fold cross-validation to validate the machine learning model where training and testing are done in 10 iterations. The training data is first randomly partitioned into 10 partitions. During each iteration of validation, 9 partitions of data are retained for training, and 1 partition is used for testing. The accuracy of the evaluation is measured by the number of an exact match between the predictions label and the testing label.

Figure 4 shows the average accuracy of the validation scenarios with different feature types. The result has shown the "FAB_PRODUCT" feature type improves accuracy. This is because the variation of the product-related features is affected by the product mix used in the training data generation. Hence, by including the product-related features into the training data, it improves the discriminating power of the learning algorithm. The result also shows that the dimension reduction method is effective in improving accuracy as compared to no dimension reduction.

4.4 On-line Simulation Verification

In this study, the learning model is used to rerun the same product mix with 40 replications, using the same wafer start pattern as the training data. The objective of the study is to determine whether the learning model can adapt the right dispatch rule based on the different manufacturing situations. For each replication, we set 10 days of warm-up period that uses the FIFO dispatch rule and run the on-line decision model for the next 20 days. The decision period is set to one day, and the learning model adapts a global dispatch

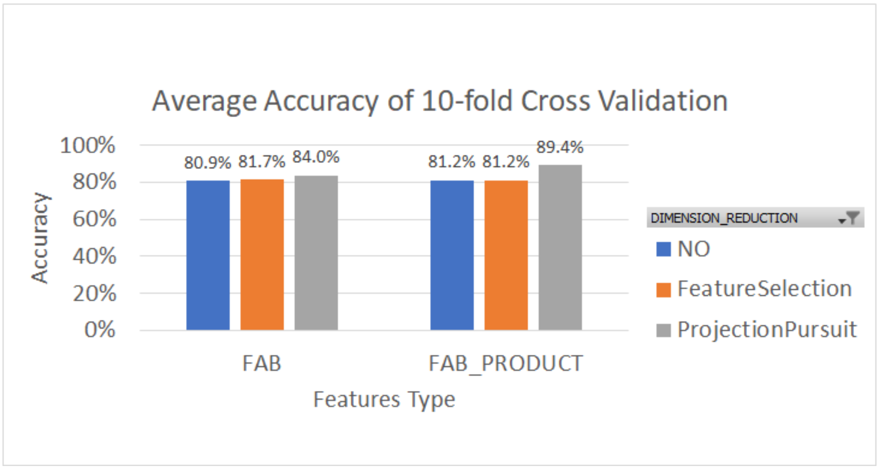

Figure 4: Average accuracy of 10-fold cross-validation across different feature with all data points from product mix Mix_Short, Mix_Equal and Mix_Long.

rule based on the current manufacturing simulation at the beginning of every simulation day. As *dynCT* is used as the factory performance in our study, it is reasonable to expect that the SPT dispatch rule is the best long-term dispatch rule to be applied in Mix_Equal and Mix_Long scenario because SPT would result in high fab level *dynDGR* and in return reduces *dynCT*. We also observed SPT rule as the best dispatch rule for Mix_Short scenarios. This is because fab level *dynDGR* is the aggregation of all products *dynDGR* without weightage of the individual product.

The training data is potentially not generated from a continuous manufacturing situation because each period is treated independently during *dynCT* comparison. However, using the discriminating power of the "FAB_PRODUCT" feature type, the on-line simulation is able to adapt the best dispatch rule most of the time and resulted in a comparable long term *dynCT* when compared with SPT dispatch rule. Figure 5 shows the average *dynCT* across the 40 replications between SPT and on-line simulation.

In the future, it will be interesting to see how the on-line simulation will perform in a varying product mix wafer start. As the wafer start mix is not the same as the line mix, the varying product mix wafer start will have an intermediate effect on the line mix before the line reaches a steady state. In order to adapt to the right dispatch rule, it is essential to have a knowledge refinement module to generate a sufficient manufacturing situation as the training data for the machine learning algorithm. Furthermore, as *dynCT* does not consider individual product performance, other performance measurements should also be explored to account for product-level performance.

5 CONCLUSIONS AND FUTURE WORK

In a situation-aware dispatching system, selecting a good dispatch rule will help increase factory performance. Hence, forming a relationship between the manufacturing situations and best dispatch rules is important for situation aware dispatching system.

In this study, we have analyzed the features of three different product mix. By including features that are product-related, we noticed accuracy improvement of the prediction model. However, using a large number of features will result in overfitting and reduce the generalization of the machine learning model. Hence, we proposed using LDA as a dimension reduction method. This resulted in higher prediction accuracy and better generalization, as shown in the experimental results.

We then used the machine learning model to predict the dispatch rule in multiple on-line simulation runs. The results show that the factory performance using the machine learning model is comparable with that of using the best-known dispatch rule. This shows the dispatching knowledge is acquired, and the relationship between manufacturing situations and best dispatch rules is created. However, the multi-pass

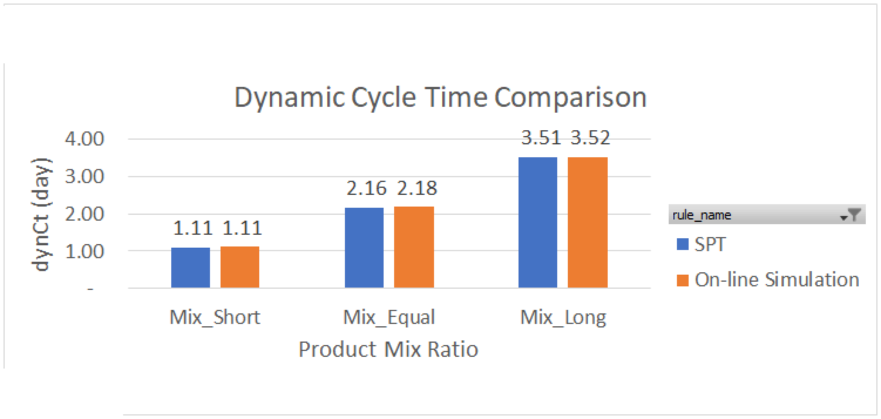

Figure 5: Dynamic cycle time comparison between SPT and on-line simulation.

system did not outperform the best-known dispatch rule in our scenario because we use only fab level dynCT in this study which does not account for product-level performance. Furthermore, the factory model used in the study is simplified and does not consider other variability in a manufacturing environment, such as the machine interruption.

This paper represents a step towards the direction of realizing situation aware dispatching in a dynamic and complex manufacturing system. Further studies need to be carried out to include other manufacturing situations such as machine interruptions into the machine learning model. Besides, techniques to explore distinct manufacturing situations so as to reduce the time to acquire the knowledge will be investigated. Finally, it would be interesting to apply the situation aware dispatching to the real-world scenarios with more complex factory models.

REFERENCES

Chang, C. C., and C. J. Lin. 2011. “LIBSVM: A Library for Support Vector Machines”. *ACM Transactions on Intelligent Systems and Technology* 2(3):1–27.

Chen, C. C., and Y. Yih. 1996. “Indentifying Attributes for Knowledge-based Development in Dynamic Scheduling Environments”. *International Journal of Production Research* 34(6):1739–1755.

Garey, M. R., and D. S. Johnson. 1979. *Computers and Intractability: A Guide to The Theory of NP-completeness*. New York: W. H. Freeman and Company.

Han, J., J. Pei, and M. Kamber. 2012. *Data Mining: Concepts and Techniques*. 3rd ed. Massachusetts: Morgan Kaufmann Publishers.

Hansch, W. 2015. *Factory Dynamics*. Universität der Bundeswehr München. https://dokumente.unibw.de/pub/bscw.cgi/d9324722/07_Factorydynamics_I.pdf, accessed 16th May 2020.

Hastie, T., R. Tibshirani, and J. Friedman. 2009. *The Elements of Statistical Learning*. Springer Series in Statistics. New York: Springer.

Ignizio, J. P. 2009. “Reducing Complexity”. In *Optimizing Factory Performance: Cost-Effective Ways to Achieve Significant and Sustainable Improvement*, Chapter 10, 231–234. New York: McGraw-Hill.

Kraskov, A., H. Stögbauer, and P. Grassberger. 2004. “Estimating Mutual Information”. *Physical Review E* 69(6):066138.

Lee, C., and D. Landgrebe. 1993. “Feature Extraction and Classification Algorithms for High Dimensional Data”. Technical report, School of Electrical Engineering, Purdue University.

Li, X., and S. Olafsson. 2005. “Discovering Dispatching Rules Using Data Mining”. *Journal of Scheduling* 8(6):515–527.

Metan, G., and I. Sabuncuoglu. 2005. “A Simulation Based Learning Meachanism for Scheduling Systems with Continuous Control and Update Structure”. In *Proceedings of the 2005 Winter Simulation Conference*, edited by M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines, 2148–2156. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Metan, G., I. Sabuncuoglu, and H. Pierreval. 2010. “Real Time Selection of Scheduling Rules and Knowledge Extraction via Dynamically Controlled Data Mining”. *International Journal of Production Research* 48(23):6909–6938.

Nakasuka, S., and T. Yoshida. 1992. “Dynamic Scheduling System Utilizing Machine Learning as A Knowledge Acquisition Tool”. *International Journal of Production Research* 30(2):411–431.

Olafsson, S., and X. Li. 2010. “Learning Effective New Single Machine Dispatching Rules from Optimal Scheduling Data”. *International Journal of Production Economics* 128(1):118–126.

Park, S. C., N. Raman, and M. J. Shaw. 1997. “Adaptive Scheduling in Dynamic Flexible Manufacturing Systems: A Dynamic Rule Selection Approach”. *IEEE Transactions on Robotics and Automation* 13(4):486–502.

Priore, P., D. de la Fuente, and R. Pino. 2001. “Learning-based Scheduling of Flexible Manufacturing Systems Using Case-based Reasoning”. *Applied Artificial Intelligence* 15(10):949–963.

Priore, P., A. Gómez, R. Pino, and R. Rosillo. 2014. “Dynamic Scheduling of Manufacturing Systems Using Machine Learning: An Updated Review”. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* 28(1):83–97.

Schoemig, A. K. 1999. “On The Corrupting Influence of Variability in Semiconductor Manufacturing”. In *Proceedings of the 1999 Winter Simulation Conference*, edited by P. A. Farrington, H. B. Nembhard, D. T. Sturrock, and G. W. Evans, Volume 1, 837–842. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Shiue, Y. R., R. S. Guh, and K. C. Lee. 2011. “Study of SOM-based Intelligent Multi-controller for Real-time Scheduling”. *Applied Soft Computing* 11(8):4569–4580.

Shiue, Y. R., K. C. Lee, and C. T. Su. 2018. “Real-time Scheduling for A Smart Factory Using A Reinforcement Learning Approach”. *Computers & Industrial Engineering* 125:604–614.

Shiue, Y. R., and C. T. Su. 2003. “An Enhanced Knowledge Representation for Decision-tree Based Learning Adaptive Scheduling”. *International Journal of Computer Integrated Manufacturing* 16(1):48–60.

Su, C. T., and Y. R. Shiue. 2003. “Intelligent Scheduling Controller for Shop Floor Control Systems: A Hybrid Genetic Algorithm/Decision Tree Learning Approach”. *International Journal of Production Research* 41(12):2619–2641.

Vergara, J. R., and P. A. Estévez. 2014. “A Review of Feature Selection Methods Based on Mutual Information”. *Neural Computing and Applications* 24(1):175–186.

Wu, S. D., and R. A. Wysk. 1989. “An Application of Discrete-event Simulation to On-line Control and Scheduling in Flexible Manufacturing”. *International Journal of Production Research* 27(9):1603–1623.

AUTHOR BIOGRAPHIES

CHEW WYE CHAN is a Software Engineer of D-SIMLAB Technologies (Singapore). He holds a Master of Computing degree from the National University of Singapore. He is currently working as a doctoral student at the School of Computer Engineering at Nanyang Technological University, Singapore. His research interests include machine learning, data science, and simulation-based optimization. His email address is chew.wye@d-simlab.com.

BOON PING GAN is the CEO of D-SIMLAB Technologies (Singapore). He has been involved in simulation technology application and development since 1995, with a primary focus on developing parallel and distributed simulation technology for complex systems such as semiconductor manufacturing and aviation spare inventory management. He was also responsible for several operations improvement projects with wafer fabrication clients which concluded with multi-million dollar savings. He holds a Master of Applied Science degree, specializing in Computer Engineering. His email address is boonping@d-simlab.com.

WENTONG CAI is a Professor in the School of Computer Science and Engineering at Nanyang Technological University, Singapore. He received his Ph.D. in Computer Science from University of Exeter (UK) in 1991. His expertise is mainly in the areas of Modeling and Simulation and Parallel and Distributed Computing. He has published extensively in these areas and has received a number of best paper awards at the international conferences for his research in distributed simulation. He is an Associate Editor of the ACM Transactions on Modeling and Computer Simulation (TOMACS), an Editor of the Future Generation Computer Systems (FGCS), and in the Editorial Board of the Journal of Simulation. His email address is aswtcai@ntu.edu.sg.