Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 00 (2019) 000–000

www.elsevier.com/locate/procedia

CIRP Manufacturing Systems Conference 2019

Toward data-driven production simulation modeling: dispatching rule identification by machine learning techniques

Satoshi Nagahara[a]*, Timothy A. Sprock[b], Moneer M. Helu[b]

[a]*Hitachi, Ltd., Research & Development Group, 292 Yoshida-cho, Totsuka-ku, Yokohama-shi, Kanagawa, 244-0817, Japan*
[b]*National Institute of Standards and Technology, 100 Bureau Drive, Gaithersburg, Maryland, 20899, USA*

* Corresponding author. Tel.: +1-50-3135-3415; fax: +1-50-3135-3412. *E-mail address:* satoshi.nagahara.eb@hitachi.com

Abstract

Production simulation is useful to predict and optimize future production. However, it requires effort and expertise to create accurate simulation models. For instance, operational control rules, such as job sequencing rules, are modeled based on interviews with shop-floor managers and some assumptions since those rules are tacit in general. In this paper, we consider a data-driven approach to model operational control rules. We develop job sequencing rule identification methods that model rules from production data using machine learning techniques. These methods are evaluated based on accuracy and robustness against uncertainty in human decision making using virtual and real production data.

Peer-review under responsibility of the scientific committee of the 52nd CIRP Conference on Manufacturing Systems.

Keywords: Production simulation, dicrete event simulation, dispatching rule, job sequencing rule, learning to rank

1. Introduction

Market needs have diversified over the years, making production systems more complex. For complicated production systems, production simulation, an embodiment of discrete event simulation (DES), is a well-known and powerful tool to evaluate, plan, and control production. In that case, it is required to build simulation models that emulates the real system accurately. Simulation inaccuracy directly affects the performance of simulation-based applications such as decision-support systems, scheduling systems, etc. However, building accurate simulation models is a time-consuming task, requiring both production domain knowledge and simulation expertise [1]. It makes the practical use of production simulation difficult. Therefore, automating production simulation modeling is still a challenging and important issue [2].

A general modeling process of production simulation is described as follows (adapted from [1]). First, characteristics of production systems are investigated mainly by field-work and interviews with shop-floor managers. Through the investigation, general requirements, such as scope and granularity of production simulation, are defined. In the second step, data required for simulation is collected and/or generated. In the third step, simulation models are constructed from the data obtained in the second step. Then, simulation accuracy is evaluated by comparing simulation results to historical production data. Lastly, these steps are iterated until the required accuracy is achieved.

To reduce the effort in this modeling process, the automation of the second step (data collection and generation) is crucially important. Barlas et al. [3] state that collecting and generating simulation data is still one of the most time-consuming tasks and a barrier to applying production simulation in practice. They also summarized existing studies on automatic collection and generation of simulation data, classifying the studies into five categories: intermediary database, PLC programs, developed applications, data interfaces / standards translators, and direct integration.

Simulation data is mainly composed with structural information such as process route, time information such as processing time, and information regarding operational control rules. One idea to automate this step is to utilize data stored in

Peer-review under responsibility of the scientific committee of the 52nd CIRP Conference on Manufacturing Systems.

the existing information technology (IT) systems such as enterprise resource planning (ERP), manufacturing execution systems (MES), and hand-held personal digital assistant (PDA) devices. In most cases, the structural information is also stored alongside the production data and can be acquired from these IT systems. For instance, the process route can be determined from Bill of Material (BOM) and Bill of Process (BOP) data, which is often managed in ERP and MES. This structural information can be collected by converting data in IT systems to simulation data using appropriate data interfaces.

Unlike structural data, timing data used to estimate processing times is often difficult to obtain directly from IT systems and rarely produces accurate estimations of actual processing times especially in high-mix and low-volume production. For example, in many systems the time information is collected by workers manually indicating start and stop times of a process. This leaves many opportunities to introduce errors, such as forgetting to start/stop time collection or not appropriately accounting for interruptions in processing. Several researchers have addressed this issue. Hosoe et al. [4] proposed methods to estimate processing time from Point of Production (POP) data using regression techniques for semiconductor production systems. Meidan et al. [5] also reported methods for estimating processing time for semiconductor production systems. They enumerated the explanatory variables such as batch size and extracted the factors with a large influence on processing time using Bayes classifier. Karnok et al. [6] and Nagahara et al. [7] have proposed methods that estimate processing time from noisy (incomplete) POP data of real production systems. Throughout this paper, point of production (POP) data includes all data that is collected from the shop-floor during production, such start/completion times, and can be tied to a specific production process, resource, operator, etc.; the point where production occurred.

Operational control rules such as job sequencing rules and resource assignment rules determine dynamic behavior of production systems. Like processing time, operation control rules are rarely managed explicitly in IT systems. In most practical cases, operational control is executed according to human decisions and the rules underlying these decisions are tacit or hidden. Sprock et al. [8] have proposed the Operational Control Model as a standard and comprehensive representation of operational controls. They classified the operational controls into five categories: admission, sequencing, resource assignment, dynamic processing planning, and changing resource states. In production simulation, rules for each control function must be modeled appropriately. Therefore, in general, the rules are determined through by interviewing shop-floor managers.

In contrast to processing time, there is little research using data-driven approaches to modeling operational control rules. Bergman et al. [9, 10] proposed methods for identifying job sequencing rules from POP data using machine learning techniques. However, they evaluated the method through experiments using synthetic data generated from a simple simulation model. The applicability of this data-driven approach for more complex real-world scenarios has not been verified in this field.

Additionally, when operational control is conducted according to human decisions, uncertainty in the decisions will be a critical issue. Consider job sequencing as an example. When workers select the next job from waiting jobs, it is possible that the job with the highest priority may not be selected. This uncertainty in the job sequencing would make the rule identification inaccurate, and so it is important to consider this uncertainty in the rule identification process.

For the above issues, this research verifies the applicability of data-driven job sequencing identification methods through experiments using real production data. In addition, we propose a method that reduces the influence of the uncertainty in the job sequencing.

The remainder of this paper is organized as follows. In section 2, we describe the problem statement of job sequencing rule identification and review some related works. In section 3, we detail the proposed method. In section 4, we first evaluate the proposed methods on virtual production data and then show experimental results of applying the methods to real production data. Finally, in section 5, we discuss conclusions and future work.

2. Job Sequencing Rule Identification Problem

2.1. Problem Statement

Rule identification methods identify the rule mapping between system states and actions. For job sequencing rule identification, the job waiting to be processed corresponds to the state and selecting the next job to be processed corresponds to the action. The objective of the Job Sequencing Rule Identification Problem (JSRIP) is to derive a job ranking model that predicts the highest ranked job among the waiting jobs given the state/action data.

The systems of interest in this research are discrete production systems such as job-shop production systems in which job sequence is decided based on tacit rules. We assume that POP data is available for job sequencing rule identification.In addition, we assume that only one job is selected from waiting jobs and the selected job is processed immediately after selection. Additionally, a list of waiting jobs when each job selection event occurred can be derived from the POP data.

The job ranking model is trained using a training dataset composed with the event data. The accuracy of the trained model is evaluated using a test dataset composed with the other event data. We evaluate the accuracy of job ranking models using two metrics: Top-1 accuracy measures the rate at which the actual selected job (based on POP data) was predicted to be the best (Top 1) by the trained ranking model, and Top-5 accuracy measures the rate at which the rank of the actual selected job is predicted to be among the five best options.

2.2. Related Works on Ranking Method

JSRIP can be thought of as one form of a “Learning to Rank” problem. Learning to Rank problems are actively studied in the information retrieval field. The objective of Learning to Rank in that field is to extract documents with high relevance to a

given query from a document set. A ranking model is derived from a dataset consisting of features of documents and queries. In the dataset, the degree of relevance of each document to the query is given as a label. Representative methods for Learning to Rank problems are divided into three approaches: pointwise, pairwise, and listwise [11].

Pointwise approaches look at a single document. A regressor or classifier is trained to predict the relevance of a particular document to a query. The input variables to the regressor/classifier are features of a single document-query pair, and the output is estimated relevance of the document to the query. This approach assumes that the relevance of each document is independent of the other documents in the document set. General regression and classification techniques can be used for this approach.

In pairwise approaches, a classifier is trained to estimate the magnitude relation of the relevance for a certain document-pair. Given a query and a pair of documents, the classifier is trained to minimize the difference between estimated magnitude relation of the documents and its ground truth. All the general classification techniques can be used for this approach, and the methods using Support Vector Machines, Boosting and Artificial Neural Network (ANN) as the classification technique are called as RankSVM [12], RankBoost [13] and RankNet [14], respectively.

Listwise approaches look at the entire list of documents. In thes approaches, a regressor is trained to estimate the score of a particular document for a query. The score is normalized by the scores of all documents in a query, and the loss is calculated based on the difference between the normalized score and the normalized relevance of each document. The representative method of this approach is ListNet that uses ANN as the classification technique [15]. In ListNet, the top one (Top-1) probability is calculated by normalizing the score/relevance by Soft-max function, and the loss is calculated from Cross Entropy Loss of the top one probability. Top-1 probability of a document is the probability that it is the best, or on top, from the list. Likewise, Top-*k* probability indicates it is ranked among the *k* best.

When formulating JSRIP as a Learning to Rank problem, the job selection event corresponds to the query and waiting jobs correspond to the documents. The characteristics of JSRIP compared to general Learning to Rank problems are as follows.

a) In JSRIP, the complete ranking of waiting jobs is unknown. We know only which job was selected. Therefore, the relevance is determined in two levels (e.g. 0 - 1 value). The relevance of the selected job is 1, and the relevance of the other jobs is 0.
b) In JSRIP, the relevance of each job is not independent of the other jobs. The job selection is decided based on the comparison of waiting jobs. From (b), it is obvious that pointwise approaches are not suitable for JSRIP.

The uncertainty in the job sequencing can be thought of as label noise in Learning to Rank problems. Niu et al. [16] investigated the influence of label noise in Learning to Rank problems. They concluded that fewer relevance levels and greater class imbalances increase the influence of label noise in the learning. From this perspective, JSRIP is a problem that is sensitive to label noise because of the problem characteristics (a). Therefore, it seems critical to consider how to reduce the influence of the uncertainty. Ding et al. [17] proposed a method for Learning to Rank problems with label noise. In their method, the reliability of each data sample is calculated using a generative model, then the loss function is weighted by the reliability. However, this method is not suitable for JSRIP because it is based on the pointwise approach.

As mentioned in the section 1, Bergmann et al. [9, 10] have proposed a job sequencing identification method. Their method is based on the pairwise approach and constructs a classifier that predicts the priority relationship between arbitrary two waiting jobs. They compared classification algorithms and data transformation techniques, and then verified the usefulness of their method through experiments using data generated from production simulation. We've not found any research that applied the listwise approach to JSRIP.

3. Proposed method

3.1. Feature Variables for Job Ranking Model

In most of practical production systems, jobs of similar product type tend to be processed consecutively to reduce setup times. While simple job sequencing rules such as first in first out (FIFO), earliest due date (EDD), shortest processing time (SPT), etc. are useful and well-accepted, the reduction of sequence-dependent setup operations is especially important in high-mix and low-volume production systems. To make these methods, such as Bergmann et al. [9, 10], applicable to more realistic scenarios, it is important to consider this context. Therefore, we include classification features capturing product type differences between waiting jobs and the previous (or in-process) job.

In general, each product type has categorical attributes such as product type name and numerical attributes such as product length. As a result, the features are determined as follows.

$$WT_i := t^e - t_i^a \tag{1}$$

$$DD_i := t_i^d - t^e \tag{2}$$

$$X_{i,k} := \begin{cases} 1 & \text{if } x_{i,k} = x_k^p \\ 0 & \text{otherwise} \end{cases} \quad (k = 1, 2, ..., N_{cate}) \tag{3}$$

$$y_{i,k} \ (k = 1, 2, ..., N_{nume}) \tag{4}$$

$$Y_{i,k} := y_{i,k} - y_k^p \ (k = 1, 2, ..., N_{nume}) \tag{5}$$

where WT_i is the waiting time of *i*-th waiting job in a certain event, and DD_i is the remaining time until due date of *i*-th job. And, t^e is the event occurrence date, t_i^a and t_i^d denote the arrival date and due date of *i*-th job respectively. Furthermore, $x_{i,k}$is *k*-th categorical attribute value of product type of *i*-th job, and x_k^p is that of the job processed before the event. Likewise, $y_{i,k}$ is *k*-th numerical attribute value of product type of *i*-th job, and y_k^p is that of the job processed before the event. And, N_{cate} and N_{nume} denotes the number of categorical and numerical attributes, respectively, of each product type. The features shown in Eq. (3) and (5) will contribute to identify the rules including the setup reduction perspective.

3.2. Combination with Voting Filter

One idea to reduce the influence of the uncertainty is filtering unreliable samples from the training dataset. From this point of view, we propose a method that combines the ranking algorithms with Voting filter. Voting filter is one of the countermeasures for label noise in classification problems [18]. In this method, several weak classifiers are trained using different training datasets and/or different classification algorithms. Then, the reliability of each data sample is evaluated based on the predictions by those weak classifiers, and unreliable samples are filtered from the original dataset. There are two major voting algorithms to judge the reliability of each sample. One is *Majority* voting that judges the sample as reliable for which half or more weak classifiers correctly predicted the ground truth class. Second is *Consensus* voting that judges the sample as reliable for which all weak classifiers correctly predicted the ground truth class. Majority voting is generally considered better than Consensus voting [18].

In the listwise approach, the reliability judgement is conducted for each event data since the listwise approach looks at the entire list of waiting jobs. On the other hand, in the pairwise approach, the reliability judgement is conducted for each pairwise comparison data.

4. Experiment and Discussion

4.1. Experiment using Virtual Production Data

Since the ground truth rule is unknown in the real scenario, we start evaluating the proposed methods by conducting experiments using virtual production data. The virtual production data is created from a simulation model consisting of one machine. The machine processes 2,000 jobs sequentially. The attributes of each job such as product type, due date, arrival date and processing time are randomly set. At the machine, job selection is conducted based on the priority score shown in Eq. (6).

$$S_i = a \cdot WT_i + b \cdot \exp\left(-DD_i\right) + c \cdot \delta_i \tag{6}$$

where S_i, WT_i and DD_i denotes the priority score, waiting time and remaining time until due date of waiting job i respectively. If the product type of job i is the same as that of the job processed before, $\delta_i = 1$. Otherwise, $\delta_i = 0$. The first, second, and third terms corresponds to FIFO, EDD and setup-reduction rule respectively, and a, b and c are the weighting coefficients for each term. The value of these coefficients is randomly set so that each rule influences job selection in some extent.

In this experiment, two scenarios for job sequencing are considered. One scenario selects the job with the maximum score (maximum score selection scenario), and the second selects a job based on the probability shown in Eq. (7) (stochastic selection scenario).

$$P_i = \exp\left(S_i\right) \Big/ \sum_{k=1}^{M} \exp\left(S_k\right) \tag{7}$$

where P_i is the probability that job i is selected, and M denotes the number of waiting jobs. This probability is introduced to express the uncertainty in the job sequencing.

The flowchart of the experiments for the proposed method is shown in Fig. 1. In the experiments, we use RankNet and ListNet to compare the pairwise and listwise approaches. In these methods, ANN is used as a classifier/regressor. In addition, the methods with/without Voting Filter are also compared. In the training of ANN, the Cross Entropy Loss with the L2 regularization term is applied as the loss function [19]. The event data is divided into training, validation, and test datasets. To prevent data leak, the event data are arranged in order of event occurrence date, and then divided into the three datasets. The rate of events in each dataset is 40% for training, 20% for validation, and 40% for test. The validation dataset is used for tuning hyper-parameters of the L2 regularization term.

The experimental results using RankNet and ListNet are shown in Table 1. Cases 1 and 2 are the results from the maximum score selection and stochastic selection scenarios, respectively. Case 3 denotes the results when the training and validation dataset are generated from the stochastic selection scenario and the test dataset is generated from the maximum score selection scenario. In cases 1 and 2, there is no significant difference between RankNet and ListNet. However, in case 3, RankNet outperforms ListNet. This result means that in our case RankNet is more robust to the uncertainty.

One reason why RankNet is better may be related to the rate of samples that comply with the ground truth rule, i.e. correct samples. Consider an event where there are N jobs waiting and the job with the second largest score is selected. This event is regarded as an incorrect sample in the listwise approach. On the other hand, in the pairwise approach, N-2 samples among N-1 samples generated from the pairwise comparisons of the waiting jobs hold correctness. The actual rate of correct samples in the training dataset is 64 % and 15 % in RankNet and ListNet, respectively. These results suggest that the pairwise approach is suitable for JSRIP with the uncertainty compared to the listwise approach.

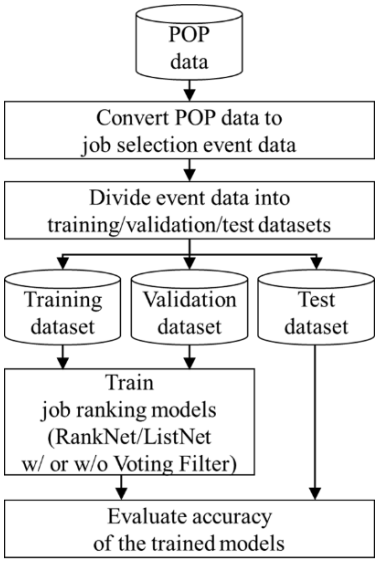

Fig. 1. Flowchart of the experiments for the proposed method

Table 1. Top-1 accuracy for virtual production data.

	Case	RankNet	ListNet
1	Maximum score selection	98.0 %	98.1 %
2	Stochastic selection	15.4 %	15.4 %
3	Training, Validation: Stochastic selection Test: Maximum score selection	88.9 %	71.5 %

Table 2. Result of reliable/unreliable judgement by Voting filter.

(i) RankNet

		Result of Voting Filter	
		Reliable	Unreliable
Correctness of data sample	Correct	(a) 62.6 %	(b) 1.6 % (2.4 %)
	Incorrect	(c) 1.5 % (4.1 %)	(d) 34.4 %

(ii) ListNet

		Result of Voting Filter	
		Reliable	Unreliable
Correctness of data sample	Correct	(a) 12.6 %	(b) 2.7 % (17.9 %)
	Incorrect	(c) 2.0 % (2.4 %)	(d) 82.7 %

Table 3. Top-1 accuracy of RankNet/ListNet for different sample sets.

	Case	RankNet	ListNet
1	All samples (a)(b)(c)(d)	88.9 %	71.5 %
2	Correct samples (a)(b)	97.5 %	91.8 %
3	Correct or reliable samples (a)(b)(c)	92.5 %	86.6 %
4	Correct and reliable samples (a)	92.5 %	81.9 %
5	Reliable samples (a)(c)	90.6 %	64.9 %

Next, the experimental results using RankNet/ListNet with Voting filter is shown in Table 2 and 3. Table 2 shows the results of reliable / unreliable judgement by Voting filter. The percentage values in this table denotes the rate of samples in the training dataset. For instance, the value in (a) denotes the rate of samples which are correct and judged as reliable by Voting filter. The values in the parentheses in (b) and (c) denote the false detection rate and overlooking rate respectively. The false detection rate is the rate of the samples judged as unreliable among all correct samples, and the overlooking rate is the rate of samples judged as reliable among all incorrect samples.

The reliable / unreliable judgment is made with a low false detection rate and overlooking rate in RankNet. On the other hand, the false detection rate is high in ListNet. This is because ListNet is a multi-class classification method, which is difficult to obtain the same prediction result among the weak classifiers, while RankNet is a 0-1 classification method. Table 3 shows the Top-1 accuracy of RankNet / ListNet for five cases in which the samples used for training are different. The case 5 corresponds to the result of the proposed method (RankNet / ListNet with Voting Filter). In the case 2 that only correct samples are used for the training, high accuracy is realized. And, the accuracy degrades due to increase of incorrect data (case 3), decrease of correct data (case 4), and both of them (case 5). This degradation is more pronounced in ListNet, and ListNet with Voting filter (case 5) is inferior to the original ListNet (case 1). On the other hand, RankNet with Voting filter shows the improvement compared to the original RankNet because the false detection rate and the overlooking rate are low as described above.

4.2. Experiment using Real Production Data

To evaluate the feasibility of the proposed method for realistic scenarios, we then conducted experiments using POP data collected from a real production plant. In this plant, over one hundred product types of industrial equipment are produced in mixed flow. The attributes of the product type are composed with eight categorical attributes such as product model name and five numerical attributes such as product length. From the POP data, we selected one process as a target and extracted about 2,000 job selection events in the process. In this process, jobs are processed by a machine, and a operator of the machine selects a next job from waiting jobs.

Table 4 shows the Top-1 and Top-5 accuracy of RankNet and ListNet. For the comparison, the accuracy when the rankings are predicted by general rules such as FIFO and EDD is also shown. As shown in the table, there is no significant difference in the prediction performance between RankNet and ListNet. In both methods, the Top-1 accuracy is not very high, but the Top-5 accuracy reaches about 90%. From this result, it can be considered that the trained ranking model can express the major tendency of the ground truth rule. One reason for the degraded Top-1 accuracy could be the uncertainty in the actual job sequencing, i.e. deviation from the ground truth rule.

Additionally, we investigate the feature importance to find the key features in the job sequencing rule. By using Random Forest as the classification method in the pairwise approach, the out-of-bag feature importance is calculated as shown in Fig. 2. The result shows that the key features are the waiting time (WT_A an WT_B in Fig.2) and the difference of 3rd and 4th categorical product type attributes between the waiting job and the job processed before (X_A3, X_B3, X_A4 and X_B4 in Fig.2). Here we have censored the actual attributes to protect the production data. This result indicates that the ground truth rule in this scenario includes the setup reduction perspective,

Table 4. Top-1 and Top-5 accuracy for real production data.

Ranking model	Top-1 accuracy	Top-5 accuracy
RankNet	61 %	90 %
ListNet	62 %	90 %
FIFO	5 %	36 %
EDD	15 %	49 %

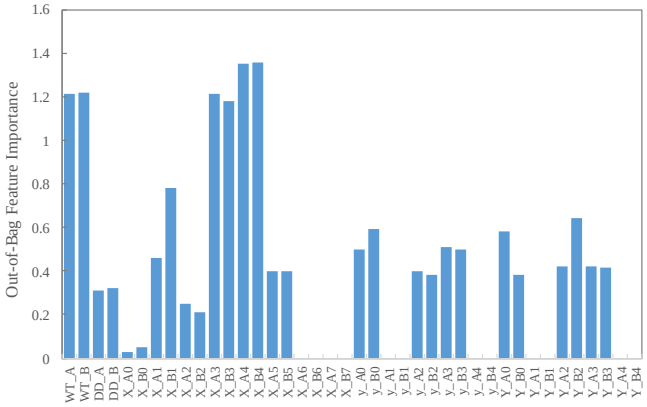

Fig. 2. Feature importance for real production data

and the features shown in Eq. (3), which describe the difference of categorical product type attributes between the waiting jobs and the job processed before, should be considered to identify the ground truth rule.

5. Summary and Conclusion

In this paper, we have proposed a job sequencing rule identification method to realize automated data-driven modeling of operational control rules for production simulation. At first, we organized the job sequencing rule identification problem (JSRIP) as a form of Learning to Rank problem. We clarified the difference between the two problems and investigated the applicability of Learning to Rank methods to JSRIP. Through the experiments using real production data, it was found that considering sequence-dependent setup operations is important for JSRIP in practical cases. By introducing the difference of product type attributes between the waiting job and the jobs processed before as features, the methods show good prediction accuracy for the real scenarios.

In addition, we compared the pairwise and listwise approaches for JSRIP with the uncertainty in the job sequencing. We found that the pairwise approach (RankNet) is more robust to uncertainty than the listwise approach (ListNet) and the pairwise approach can identify the ground truth rule accurately even in the existence of the uncertainty. Furthermore, we proposed a novel method utilizing Voting filter to reduce the influence of the uncertainty. Voting filter did not work well for the listwise approach. On the other hand, in the pairwise approach, the training samples were filtered with low false detection rate and low overlooking rate by Voting filter and the proposed method (RankNet with Voting filter) showed the improvement in the accuracy compared to the original RankNet.

The rules discovered by the proposed methodology are inputs into constructing production line simulation models. The rules are implemented by simulation queues that use the rules to sequence jobs to be processed. These simulations are useful for optimizing production planning and operations management decisions.

One future works is to evaluate using other ranking methods and filtering methods. It would be also interesting to investigate what kind of classification / regression algorithm is suitable for JSRIP with uncertainty. In addition, it is important to investigate how the prediction error of the job ranking model affects the accuracy of production simulation. Furthermore, rule identification methods for the other operation control rules such as resource assignment rules should be developed to automate the whole production simulation modeling process.

Disclaimer

Commercial equipment and materials might be identified to adequately specify certain procedures. In no case does such identification imply recommendation or endorsement by the U.S. National Institute of Standards and Technology, nor does it imply that the materials or equipment identified are necessarily the best available for the purpose.

References

[1] Law, Averill M.. How to build valid and credible simulation models. Proceedings of the 2009 Winter Simulation Conference. 2009. pp. 24-33

[2] Fowler, J. W., Rose, O.. Grand challenges in modeling and simulation of complex manufacturing systems. Simulation. Vol. 80. Issue 9. pp.469-476.

[3] Barlas, P., Heavey, C.. Automation of input data to discrete event simulation for manufacturing: A review. Int. Journal of Modeling, Simulation and Scientific Computing. Vol.7. No.1. 1630001. 2016.

[4] Hosoe, H., Kanamori, N., Yoshida, K.. The methods of data collection and tool processing time estimation in lot processing. ISSM Paper. MC-P-075. 2007.

[5] Meidan, Y., Lerner, B., Rabinowitz, G., Hassoun, M.. Cycle-time key factor identification and prediction in semiconductor manufacturing using machine learning and data mining. IEEE transactions on semiconductor manufacturing. Vol. 24. No. 2. 2011. pp.237-248.

[6] Karnok, D., Monostori, L.. Extracting process time information from large-scale noisy manufacturing event logs. 44th CIRP Conference on Manufacturing Systems. 2011.

[7] Naghaara, S., Nonaka, Y.. Product-specific process time estimation from incomplete point of production data for mass customization. Procedia CIRP 67. 2018. pp.558-562.

[8] Sprock, T., McGinnis, L. F.. Simulation optimization in discrete event logistics systems: the challenge of operational control，Proceedings of the 2016 Winter Simulation Conference. 2016. pp.1170-1181.

[9] Bergmann, S., Feldkamp, N., Strassburger, S.. Approximation of dispatching rules for manufacturing simulation using data mining methods. Proceedings of the 2015 Winter Simulation Conference. 2015. pp.2329-2340.

[10] Bergmann, S., Feldkamp, N., Strassburger, S.. Emulation of control strategies through machine learning in manufacturing simulations. Journal of Simulation. 11. 2017. pp.38-50.

[11] Jagerma, R., Kiseleva, J., Rijke, M.. Modeling label ambiguity for neural list-wise learning to rank, arXiv:1707.07493v1. 2017.

[12] Herbrich, R., Graspel, T., Obermayer, K.. Support vector lerning for ordinal regression. Proceedings of ICANN. 1999. pp.97-102.

[13] Freund, Y., Iyer, R., Schapire, R. E., Singer, Y.. An efficient boosting algorithm for combining preferences. Proceedings of ICML. 1998. pp.170-178.

[14] Burges, C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N., Hunllender, G.. Learning to rank using gradient descent. Proceedings of ICML. 2005. pp.89-96

[15] Cao, Z., Qin, T., Liu, T., Tsai, M., Li, H.. Learning to rank: from pairwise approach to listwise approach. Microsoft TechReport. MSR-TR-2007-40. 2007.

[16] Niu, S., Lan, Y., Guo, J., Wan, S., Cheng, X.. Which noise affects algorithm robustness for learning to rank, Information Retrieval Journal. 2015. pp.215-245

[17] Ding, W., Geng, X., Zhang, X.. Learning to Rank from Noisy Data. ACM Transactions on Intelligent Systems and Technology. 2015

[18] Frenay, B., Verleysen, M.. Classification in the presence of label noise: a survey, IEEE transactions on neural networks and learning systems, Vol. 25, No. 5. 2014. pp.845-869

[19] Krogh, A., Hertz, J. A.. A simple weight decay can improve generalization. Advances in neural information processing systems. Vol. 4. 1992. pp. 950-957