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.

DYNAMICALLY CHANGING SEQUENCING RULES WITH REINFORCEMENT LEARNING IN A JOB SHOP SYSTEM WITH STOCHASTIC INFLUENCES

Jens Heger
Thomas Voss

Leuphana University
Universitaetsallee 1
Lueneburg, 21335, GERMANY

ABSTRACT

Sequencing operations can be difficult, especially under uncertain conditions. Applying decentral sequencing rules has been a viable option; however, no rule exists that can outperform all other rules under varying system performance. For this reason, reinforcement learning (RL) is used as a hyper heuristic to select a sequencing rule based on the system status. Based on multiple training scenarios considering stochastic influences, such as varying inter arrival time or customers changing the product mix, the advantages of RL are presented. For evaluation, the trained agents are exploited in a generic manufacturing system. The best agent trained is able to dynamically adjust sequencing rules based on system performance, thereby matching and outperforming the presumed best static sequencing rules by $\approx 3\%$. Using the trained policy in an unknown scenario, the RL heuristic is still able to change the sequencing rule according to the system status, thereby providing robust performance.

1 INTRODUCTION

Over the last years, intelligent, adaptive, and autonomous systems have been developed to cope with uncertainties within large manufacturing systems. The usage of reinforcement learning (RL) has proved to be suitable as a heuristic approach to select operations in queues or fix schedules. Even though this method might work for small scenarios, it is extremely difficult to evaluate the impact of selecting a particular order. Hence, this study makes a novel contribution to the literature by applying RL as a hyper heuristic to change the sequencing rules used in the manufacturing system under stochastic influences. The aspects that are being considered include inter arrival times between products being drawn from a given distribution, customers changing the product mix, and reducing the number of setups. Given these variations, the state and action space as well as the training behavior and system performance must be evaluated. The goal of this study is to prove the ability of RL to reduce the mean tardiness of the system by changing the sequencing rule dynamically, trained in a discrete event simulation. In section 2, we provide a summary of the methods implemented in the dynamic selection of sequencing rules in job shop scenarios; furthermore, we present the applications of RL in those scenarios as well as the gaps in the literature. In section 3, we describe in detail the methods deployed, which, in this case, are discrete event simulation and RL. In section 4, we discuss the results evaluating the training of the approach as well as the implications for the manufacturing system being applied. Section 5 provides the conclusion and an overview of future prospects.

2 RELATED WORK

In every manufacturing system, the aspect of sequencing, routing, and dispatching plays an important role. In this study, the dynamic adjustment of sequencing rules is evaluated. For this reason, the routing and dispatching rules were neglected in the state of the art and set consistently throughout the simulation study.

2.1 Sequencing In Flexible Manufacturing Systems

Central heuristics have been developed using mathematical models being solved with particle swarm optimization, genetic algorithms, and simulated annealing. Nevertheless, the problem is NP-hard and cannot be solved optimally in feasible time. Furthermore, the need to reschedule based on unforeseen events, such as machines breaking down or priority orders, makes the problem even more complex. For this reason, decentral sequencing heuristics are applied. This study focuses on the usage of priority-based sequencing rules.

Single attribute sequencing rules, such as the shortest processing time (SPT), assign a priority to the jobs waiting in the queue depending on the given criteria, whenever a machine is ready for operation. Based on the priority assigned, a job is chosen to be processed subsequently. This method of sequencing jobs on a machine is well known and widely applied in the industry owing to its simplicity. Commonly known sequencing rules have been reviewed by Panwalkar and Iskander (1977), who conducted an extensive study presenting more than 100 rules. In our study, we considered the following rules: SPT, first in first out (FIFO), and earliest due deadline (EDD). For scenarios, including setup considerations, special dispatching rules have been developed. A commonly used rule is similar setup preferred (SIMSET). Rules that are more complex and composed of multiple attributes, which evaluate the priority of a job, are called composite rules. The attributes within these rules can be weighted and combined to consider different aspects of the system and jobs. This enables the priority rule, for example, to consider the processing time and deadline for the job. Holthaus and Rajendran (1997) as well as Holthaus and Rajendran (2000) implemented the sequencing rule apparent tardiness cost with setups (ATCS), considering the weighted processing time as well as the slack and duration of setups conducted. These factors have been multiplied with each other applying an exponential function, thereby ensuring that as the factors decrease their weights increase drastically. Riley et al. (2016) stated that, although a large number of attributes can be considered for a composite rule, not all of them are useful. For this reason, determining the best attributes is important in reducing the mean tardiness. Applying genetic algorithms in combination with feature selection, the authors weighted multiple terminals in a composite sequencing rule, thereby improving the system performance. However, the authors stated that their approach is still partially unstable. Pergher and de Almeida (2018) and many others have used weighted composite rules suited and optimized for specific scenarios. Since most of the rules have been optimized to suit a specific situation and scenario, these rules no longer perform well if the situation in the system changes. For this reason, the dynamic adjustment of composite sequencing rules needs to be applied. Heger (2014) as well as Heger et al. (2015) deployed the dynamically adjusted ATCS sequencing rule. The authors exploited a simulation study to generate offline training data and utilized a neural network and Gaussian process regression to estimate the performance of particular parameter combinations. During the online phase, the parameters were adjusted to reduce the mean tardiness. Ma et al. (2017) applied a composite rule considering the real-life attributes of a manufacturing lot, such as its priority, the remaining process steps, and the process time. The linear combination of those terms, including a weighting factor for each one, has been employed along with a support vector regression to adjust the weights in a real-time production system. The parameters used to assess the state of the system included the mean cycle time, total wafer movement amount, work in progress, production rate, and overall equipment efficiency. The authors proved that their solution outperforms other solutions. Nasiri et al. (2017) conducted a simulation-based optimization of real-time scheduling in an open shop using a composite rule with three parameters and weights. Considering the remaining time of the part until completion, number of remaining machines, and processing time, their corresponding weights were optimized with a neural network and simulation study. The results indicated that the second term was no longer necessary during online application and could be neglected. Finally, their approach reduced the mean waiting time. However, all of the above approaches can be applied exclusively to known scenarios.

2.2 Reinforcement Learning

RL has been applied to the sequencing problem over the last couple of years; a collection of such approaches is presented below. Ramirez-Hernández and Fernandez (2009) utilized a dynamic programming approach to select lots waiting to be processed in the MiniFab Scenario. Compared to single attribute sequencing rules, the authors could increase performance. Multiple studies define the sequence of orders by an agent on a machine (Stricker et al. 2018) or create complete production plans (Waschneck et al. 2018). Furthermore, an alternative job can be selected in case of machine failure (Zhao et al. 2019). In all of the above cases, the agent has the option to choose a certain operation or job. The adoption of RL as a hyper heuristic was already considered in Chen et al. (2010). They trained the RL agent to adjust weights on a composite rule for every operation based on the WIP level of the system. Burke et al. (2013) tested and evaluated different heuristics; subsequently, the heuristic with the best values was selected and applied. Shiue et al. (2018) selected priority rules for individual machines statically and applied them. Chen et al. (2019) revealed that the application of allocation strategies in a stochastic environment with the help of intensifying learning can achieve up to 32 % improvement in individual cases compared to conventional methods. Lee et al. (2019) implemented an approach with RL including the priority value calculated based on single attribute sequencing rules into the observation space, which appears to be a viable option. The idea of training the agent in one scenario and applying it to others (called transfer learning) is considered in Zheng et al. (2019). The authors utilized a 2D representation of the shop floor to train their agent; for this purpose, they have a completely different state and action space. Finally, some of the aspects of RL are considered to improve regular genetic programming (Zhang et al.). In particular, the idea that the training agent, in contrast to genetic programming, can store local knowledge from simulation behavior is examined. Based on the literature, this study aims to develop an approach able to adjust the sequencing rule, suitable for the given system performance, and transfer that knowledge to unknown environments, thus harnessing the advantages of RL.

3 METHODS

3.1 Simulation Model

Adopting the notation of Graham et al. (1979), our study considers a flexible job shop with 10 machines ($FJ10$), organized in 5 groups of 2 machines each. The set of all the machines is called M. M_j provides a machine environment where every job j can be processed on a set of alternative machines in the given set of $M_j \in M$. Each of the machines has local unrestricted input and output buffers. The material handling system can be classified R, which accounts for the unspecified number of robots involved in material handling. t_{kl} characterizes the machine pair (k,l) dependent travel time for job j. All the jobs undergo processing in all five groups twice, based on a random sequence, to represent reentrant processes. *prec* represents the preceding constraints of the operations being conducted. The objective function is the mean tardiness $\overline{T_j}$ based on the last 10 000 orders leaving the system.

The processing times in the experimental setup are derived from a uniform distribution ranging from 1 to 99. Our study extends the study conducted by Holthaus and Rajendran (2000) and utilizes the same values. The setup times are fixed asymmetric sequence dependent values considering a product family. The values are provided in Eq. 1 and can be read as: The setup time from product family 1 to product family 2 is 5 min. These values are static and have no variance. Between two machines, the product must be transferred; the transportation time is randomly derived from a normal distribution with a mean equal to 10 and sigma equal to 1. These values, which are chosen based on Kim et al. (1999), lead to a P/T-ratio equal to 5. The impact of larger P/T-ratios has not been evaluated.

$$Setup = \begin{pmatrix} 0 & 5 & 10 & 25 \\ 5 & 0 & 10 & 25 \\ 5 & 5 & 0 & 25 \\ 5 & 5 & 10 & 0 \end{pmatrix} \tag{1}$$

The distribution of product families being manufactured depends on the product mix chosen. Sixty-six different product mixes could be possibly evaluated, with the first three product types varying by a step of 10 %. The value of the last product was set to 0 throughout the study since the desired effect could be demonstrated with solely three product types. That is, [0.5, 0.2, 0.3, 0] would be read as 50 % of product family 1, 20 % of product family 2, and 30 % of product type 3. All the product mixes result in specific values of setup/processing time ratio; for instance, [0.5, 0.5, 0, 0] results in a setup/processing time ratio equal to 0.1. Notably, these product mix distributions result in different levels of system utilizations due to the different amount of setup times. For the calculation of the due date, the method of total work content is applied. For the calculation of the due dates (ref. Eq. 2), the mean processing time of all operations is multiplied by the number of operations, resulting in the job specific mean processing time $\overline{p_i}$. Due date d_i is the sum of start time s_i and the product of due date factor k multiplied by the sum of mean process times $\overline{p_i}$ of each job i.

$$d_i = s_i + k(\overline{p_i}) \tag{2}$$

Since rules can perform differently if the system performance changes slightly, multiple scenarios must be assessed. The inter arrival time has been chosen to represent a system load level of 85 % and 90 %. For the calculation of the mean lambda value for the Poisson distributed inter arrival times Eq. 3 is used. The impact of the Poisson distribution in combination with the high variance of the processing time in the performance of the manufacturing system will be discussed later. The mean inter arrival time b can be calculated with μ_p being the mean processing time over all the operations and μ_g being the mean number of operations over all the job types. U denotes the utilization and M the number of machines on the shop floor. Owing to waiting times, setups, and transportation, the true machine utilization during the simulation is higher. Based on a short preliminary study, varying the number of AGVs, it can be deduced that three vehicles are enough to supply the manufacturing system. A summary of simulation setup parameters is presented in Table 1.

$$b = \frac{\mu_p \mu_g}{UM} \tag{3}$$

Considering the size of the scenario, it has been pointed out that six machines are sufficient to represent the complexity of a job shop manufacturing system (Poppenborg et al. 2012). The examples with 8 machines (Kumar 2016) and 10 machines (Sharma and Jain 2016) are also valid.

The presented scenario does not include buffer restrictions; a feasible number of AGVs is utilized and their impact for transportation is marginal. Consequently, the dispatching decision can be neglected, and dispatching is set to choose the vehicle with the shortest travel time. The routing of the products is set to choose the machine with the least waiting time in queue.

For the comparison of the simulations, 12 500 job completions per simulation are considered. The jobs are numbered on arrival and simulation outputs from 1 to 2500 are discarded, letting the system reach a steady state behavior (Welch 1983). The following 10 000 jobs are documented and deployed for the calculation of the mean tardiness. To compare the different runs for each rule combination, the ”mean tardiness” has been documented for each run. The values provided are calculated as the mean over all 30 replications, based on different and independent random numbers.

Table 1: Parameters of the simulation experiment

Machine	Number of machines: 10
	Number of AGVs: 3
Job	Job families: 4
	Operations per job: 10
	Utilization levels: 85 % and 90 %
	Inter arrival Time Distr.: Poisson
	Processing time: 1 - 99
	Processing time Dist: Uniform
	Due-date: Total Work Content Method
	Due-date Factor: 2
Seq. rule	SPT, EDD, FIFO, SIMSET
Simulation	Warmup: 2500 jobs
	Run length: 12 500 jobs
	Replications: 30
KPIs	Mean Tardiness

3.2 Reinforcement Agent

When applying the Markov decision process to realize RL, a set of states S (e.g., the situation of a buffer), a set of possible actions (e.g., FIFO), the conditional distribution $P(s'|s,a)$, the reward of transitioning from state s to s': $R(s,s')$, and the discount factor γ are exploited. The cumulative reward in this scenario is the discounted sum of rewards accumulated throughout an episode, as presented in Eq. 4.

$$R = \sum_{t=0}^{n} \gamma^t r_{t+1} \tag{4}$$

In this case, large γ values weight later events more heavily, thus favoring long-term rewards. Conversely, small values shift the focus to a shorter realization of rewards.

$$V_\pi(s) = E\{r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots + \gamma^n r_n \mid s_t = s\} \tag{5}$$

Given the the policy followed at each state is π the expected reward can be calculated. Optimizing the value function $V_{\pi^*}(s)$ (Eq. 5), the optimal policy π^* can be derived using greedy behavior. The agent is trained to maximize the expected reward from a state choosing action π. In this approach, the policies can be regarded as the selection of the sequencing rule.

How do we learn the Q-Values for the system? In some cases, it can be useful to calculate the complete simulation first and evaluate the actions afterwards. In other cases, it is more logical to evaluate the actions according to defined time steps. The former is called Monte Carlo (MC) simulation; the latter is called temporal difference (TD). Thus, it can be assumed that MC simulation has a high variance and low bias, because the total reward of the actions depends on many random actions, transitions, and rewards. For TD, it can be assumed that there is a low variance and a certain bias because the total reward depends on one random action, transition, and reward. The selection has a decisive influence on the convergence behavior. In this particular case, TD has been chosen as follows (Eq. 6).

$$Q_\pi(s_t,a_t) \leftarrow Q_\pi(s_t,a_t) + \alpha[\underbrace{(r_t + \max_a Q_\pi(s_{t+1},a)}_{\text{target}} - Q_\pi(s_t,a_t))]}_{\text{TD-error}} \tag{6}$$

The ε value, which considers the percentage of random actions taken, is set to decrease over the training until it reaches 10 %. Therefore, focusing on the presumed good actions in later training, the reward will converge more smoothly and refine the forecast values for these actions. The γ value, which is described above, is set to 0.99, ensuring that a long-term reduction of the key performance is attained. Preliminary studies reveal that the range from 0.9 to 0.99 provides stable results. Although trainings with 0.5 have converged, they did not lead to significant improvement during evaluation. More precise evaluation is subject to further research. As mentioned above, the Q-values must be predicted for possible strategies. Since the number of possible (state,action) pairs is large, a neural network can be implemented to assess the value of all the possible actions. The number of input neurons is equal to the number of observations; further, the number of outputs is equal to the number of possible actions. This method can be applied since the possible actions are consistent over the full episode. The network is 300 neurons wide and 2 layers deep, using the ReLU activation function. Preliminary tests with fewer neurons reached a lower precision; therefore, they did not result in significant improvement during evaluation. Similar to the tuning of γ and ε values, this is subject to further research. Based on different observations, such as the amount of work in all the queues, the mean tardiness of the products, and the average utilization of the machines, the possible actions are taken. Since the values for "work in queue" can be very large numbers, in contrast to the "average utilization" ranging from zero to one, each observation has been divided by the sum of all the observations. Therefore, each value ranges from 0 to 1, thereby being more suitable for the ReLU activation function of the neural network. It must be clarified that, in this case, the agent chooses not single operations but the sequencing rule to process next, and applies it to every machine in the system. As possible actions, the trained agent can choose any of the four sequencing rules (FIFO, EDD, SPT, and SIMSET). Based on the temporal difference equation above, the reward can be calculated as the difference between the mean tardiness at the time of observation and the mean tardiness after the time step considered divided by the maximum of both values, so that the value is always between -1 and 1 (Eq. 7).

$$Reward = \frac{T_t - T_{t+1}}{max(T_t, T_{t+1})} \tag{7}$$

As mentioned earlier, the mean tardiness is calculated over the last 10 000 orders and the mean utilization over the last week. The amount of work is calculated based on the orders waiting in the queue. In this study, during the training, an action is performed every week (every 10 080 min), according to the observations stated above. Given that the system load varies with respect to the stochastic influences, such as inter arrival time, the time frame used to examine the mean tardiness and machine utilization is subject to further research.

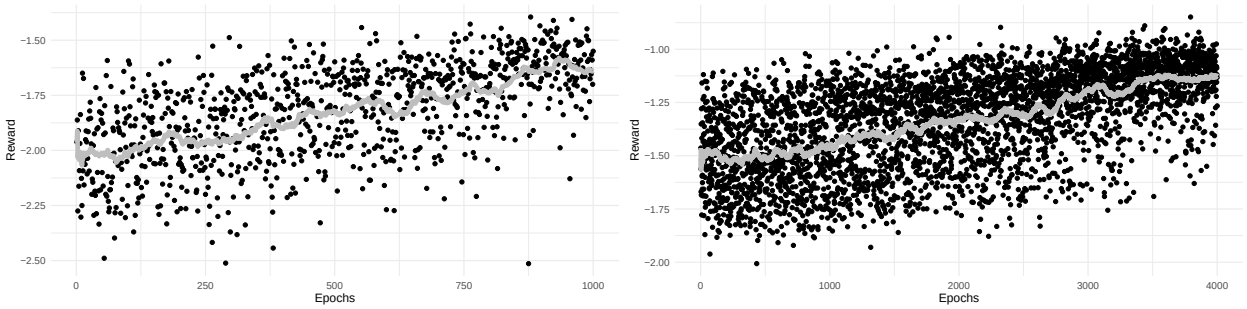
4 RESULTS

In this section, two possible aspects, the training and impact on the production system, are evaluated. For training, the most important feature is the convergence behavior, which indicates that the agent is able to choose good actions. For the evaluation of the production system, the impact of the actions taken on the key performance indicator must be evaluated. The simulation was coded with the Java-based software AnyLogic; moreover, library RL4J was deployed to implement RL. Finally, the following question is answered: Given the stochastic influences, such as random inter arrival times from a given distribution and external inputs, e.g., changing product types, is the system able to select different sequencing rules based on its status?

4.1 Training Of The RL Agent

During the first training, only one product mix at planned high load has been trained (Scenario 1). Since the reward does not necessarily relate to the actual performance of the system, these plots only prove that the agent learns to choose an action that increases the reward, thereby reducing the mean tardiness. The

dots in figure 1a and 1b represent single epochs; the line represents the simple moving average over the last 100 values. In the left figure, it can be noticed that the reward converges slowly from a mean of -2 to around -1.6. During the later epochs, rewards with a negative value around -2 can be observed. This may be owing to the random action choice during the training. Although the variance of the reward should decrease when ε decreases, some variation remains, given the 10 % random actions. Furthermore, the variance of the different epochs is a consequence of the stochastic inputs, such as the Poisson distributed inter arrival time. Considering the duration of 1000 epochs, the training is comparatively short. It can be assumed that longer trainings may improve the results. For this reason, Scenario 2 has been retrained as 1000 epochs per product mix. In other words, in the second training (Scenario 2), four different product mixes have been presented to the agent, and the training behavior is plotted in 1b. Notably, two out of four product mixes have lower setup ratios resulting in reduced mean tardiness during the simulation. For this reason, the first epochs range from -1.8 to about -1.1 with a mean around -1.5. The mean settles around -1.1. Just like before, the convergence of the reward is noticed up to a certain level of variance.



(a) Training the agent with a single product mix (Scenario 1) reveals a convergence behavior at the reward around -1.6.

(b) Training the agent with multiple product mixes (Scenario 2) leads to a higher reward, which seems to converge around -1.1.

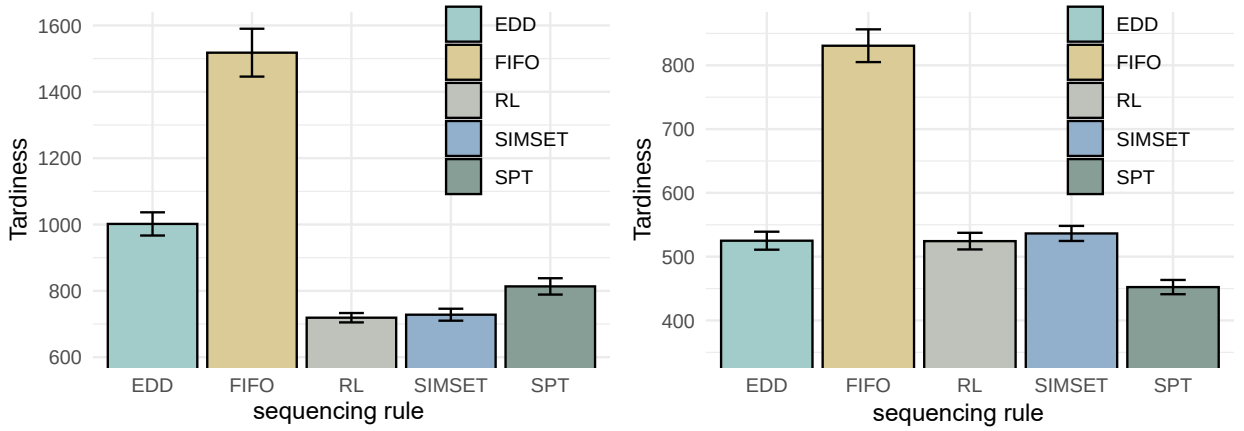
Figure 1: Based on the data presented during the training, the reward can be different.

During training, a few aspects were observed as follows. First, training the agent in the selection of a sequencing rule needs a highly utilized manufacturing system, since the difference between the rules must be significant to be detected by the agent. Second, the reward is bound by the worst and best sequencing rules, assuring that all the solutions are feasible. Furthermore, this ensures that the values are in a narrow range, leaving the solution corridor being narrow. Additionally, shorter inter arrival time leads to SPT being used more often than any other rule. For this reason, it can be assumed that the agent was biased based on the scenario presented during the training and evaluated as very good for some scenarios but poor for others.

4.2 Evaluation Of The Approach In The Theoretical Manufacturing System

In Figure 2a, it can be recognized that the performance of the RL agent trained in Scenario 1 is just as good as that of the best single attribute priority sequencing rule. Concerning the agents ability to choose exclusively from a given set of sequencing rules, this is reasonable since choosing a sequencing rule other than the best would result in lower performance on high system load.

Presenting the agent with an unknown product mix, e.g., [70, 20, 10, 0], which has a significantly lower number of setups, it can be noticed that the RL agent is able to achieve results close to these of SIMSET, thereby being second in line. The fact that the mean tardiness is about half the value in which the agent was trained originally, and the queues as well as the utilization levels will be completely different, this does not come as a surprise. Still, Figures 2a and 2b show, that the agent was able to learn the strategy to reduce the mean tardiness in the first scenario and partially transfer it to another.

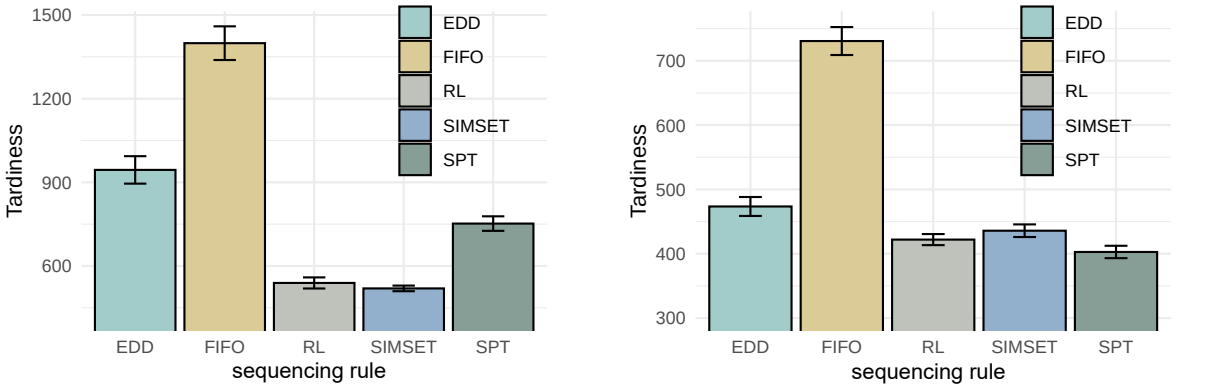


(a) The trained agent in Scenario 1 achieves the same performance as that of the best sequencing rule given the product mix [50, 30, 20, 0].

(b) The trained agent in Scenario 1 achieves bad performance on the sequencing given the product mix [70, 20, 10, 0]

Figure 2: The agent trained in Scenario 1 is able to adjust the sequencing rules in that particular scenario. Applying the strategy to another scenario, which the agent has not seen before, does not lead to feasible results.

The agent trained in Scenario 2, achieves better results regarding the different product mixes. In Figure 3a and 3b, the resulting mean tardiness for two different product mixes is demonstrated. The trained agent is able to achieve similar results to these of the best sequencing rules. By examining the plots, it becomes evident that the two different product mixes have contradicting rule behaviors. This may be owing to the fact that some of the states and the presumed best sequencing rule may be similar regarding high system load. In this case, although the mean tardiness was different by a factor 2, the agent was able to adjust the sequencing rule and produce feasible behavior. In case of product mix [0, 50, 50, 0], the case behavior is comprehensible. Choosing a rule other than the best would result in significant loss of reward.



(a) The trained agent in Scenario 2 achieves the same performance as that of the best sequencing rule given the product mix [0, 50, 50, 0].

(b) The trained agent in Scenario 2 achieves a good performance on the sequencing given the product mix [50, 50, 0, 0].

Figure 3: Training in Scenario 2 leads to a more stable behavior over different product mixes.

During the evaluation, some aspects emerged and should be highlighted for further training. Given that the mean tardiness over all the products during the simulation run is a sluggish indicator for the system performance, a new variable considering the tardiness of the last 200 orders, representing about one week of work, has been introduced to the model. The variation of this variable is much higher and shows the wavering of the performance indicators better than before.

Since the agent checks the system every week and changes the sequencing rule with respect to the observations, tracking the sequencing rule and considering the correlation with the mean tardiness can offer some insight into the agents behavior making it comprehensible to humans.
This also leads to the idea that the agent should be able to detect a change in product mixes over simulation time. As mentioned before, no sequencing rule is superior to all the others under varying system performance. Since the rules are categorical, they are represented by the values 0 to 3 in Figure 4. The encoding is the following: 0 - SPT, 1 - EDD, 2 - FIFO, and 3 - SIMSET. Similar to the plots above, the agent varies between 0 and 3, which are the best rules in this scenario. On multiple occasions, it can be noticed that, if the mean tardiness (solid black line) rises slightly, the agent decides to apply the sequencing rule SIMSET. This becomes more apparent when examining the short mean tardiness over the last 200 orders (dotted black line). As the value rises, the agent, who usually utilizes SPT as a sequencing rule, starts to use SIMSET to compensate for the peaks of tardiness. After three quarters of a year (388 880 min), the product mix changes and, therefore, the mean tardiness rises slightly yet more constantly. The agent tries to compensate this by changing to a particular sequencing rule, in this case, rule no. 3 SIMSET. Using the same sequence of random numbers, the RL approach can achieve a mean tardiness of 458 min, which is $\approx 3\%$ lower than that of the best sequencing rule, i.e., SPT. Using SIMSET throughout the procedure would result in a mean tardiness of 479 min.

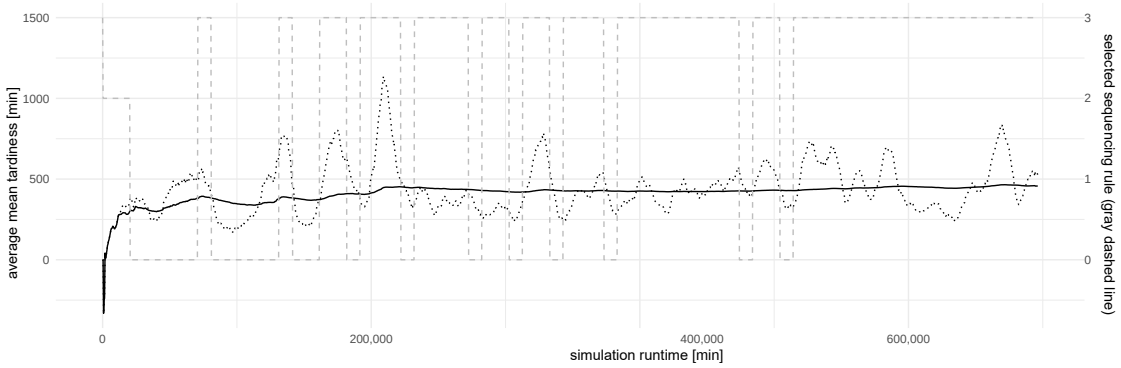


Figure 4: The agent adjusts the sequencing rule based on the change of the mean and short tardiness.

4.3 Implementation Of The Approach In A Real-World Scenario

Since no real-life data has been provided by project or associated partners, the evaluation in this context must still be carried out. Based on the presented approach, possible users should schedule their production with sequencing rules and record basic production data, such as the amount of work in queues on machines, applying real-time data acquisition tools. Given these prerequisites, it can be assumed that larger manufacturing firms with multiple machines can reduce their mean tardiness by $\approx 5\%$. Additionally, a reduction in peak loads and a more stable production process can be expected. A proof of concept that presents the dynamic adjustment of parameters in a live system with RL will be set up in the authors learning factory. The learning factory represents a complex manufacturing system with multiple stations and material movement between these stations. The data from the orders within the manufacturing process, such as the due date or position, are tracked with radio frequency identification and stored in a SQL-database. The

web-based production and planning system evaluates the situation, and periodically submits the observation state, which is aggregated from the SQL-database, to the agent via a REST-API. The agent responds with an action to choose, in this case, the presumed best rule to reduce the mean tardiness.

5 CONCLUSION

The use of decentral priority based sequencing rules was proved to be a simple and effective method for handling NP-hard problems. Given that no rule is superior to others under varying system performance, a dynamic change of sequencing rules was proposed. Different scenarios to train the RL agent have been presented. The RL approach was able to find the best sequencing rule and apply it to the system, after training in a particular scenario. Consequently, providing the agent with more scenarios increased the potential to determine the best sequencing rule. Finally, the agent was able to apply the knowledge within a simulation run, where a product mix changed. Regardless the fact, that this was a situation which has not been observed before, the agent was able to achieve feasible results. Hence, it can be concluded that RL can be exploited to dynamically change the sequencing rules in this flexible job shop scenario. As mentioned above, multiple aspects that should be evaluated more thoroughly exist, such as the hyper-parameters of the RL approach as well as the observation space in the manufacturing scenario.

ACKNOWLEDGMENTS

The authors would like to thank the reviewers for their critical feedback and suggestions to improve the quality of the paper. Furthermore, the authors would like to thank the team from Pathmind for their insightful discussion. This contribution has not received any funding from organizations.

REFERENCES

Burke, E. K., M. Gendreau, M. Hyde, G. Kendall, G. Ochoa, E. Özcan, and R. Qu. 2013. "Hyper-heuristics: A survey of the state of the art". *Journal of the Operational Research Society* 64(12):1695–1724.

Chen, J. Y., M. E. Pfund, J. W. Fowler, D. C. Montgomery, and T. E. Callarman. 2010. "Robust scaling parameters for composite dispatching rules". *IIE Transactions* 42(11):842–853.

Chen, S., S. Fang, and R. Tang. 2019. "A reinforcement learning based approach for multi-projects scheduling in cloud manufacturing". *International Journal of Production Research* 57(10):3080–3098.

Graham, R. L., E. L. Lawler, J. K. Lenstra, and A. R. Kan. 1979. "Optimization and approximation in deterministic sequencing and scheduling: a survey". *Annals of Discrete Mathematics* 5:287–326.

Heger, J. 2014. *Dynamische Regelselektion in der Reihenfolgeplanung*. Wiesbaden: Springer Fachmedien Wiesbaden.

Heger, J., T. Hildebrandt, and B. Scholz-Reiter. 2015. "Dispatching rule selection with Gaussian processes". *Central European Journal of Operations Research (CEJOR)* 23(1):235–249.

Holthaus, O., and C. Rajendran. 1997. "Efficient dispatching rules for scheduling in a job shop". *International Journal of Production Economics* 48(1):87–105.

Holthaus, O., and C. Rajendran. 2000. "Efficient jobshop dispatching rules: Further developments". *Production Planning & Control* 11(2):171–178.

Kim, C. W., J. A. Tanchoco, and P.-H. Koo. 1999. "AGV dispatching based on workload balancing". *International Journal of Production Research* 37(17):4053–4066.

Kumar, R. 2016. "Simulation of Manufacturing System at Different Part Mix Ratio and Routing Flexibility". *Global Journal of Enterprise Information System* 8(1):10–14.

Lee, W.-J., B.-H. Kim, K. Ko, and H. Shin. 2019. "Simulation Based Multi-Objective Fab Scheduling by Using Reinforcement Learning". In *Proceedings of the 2019 Winter Simulation Conference*, edited by N. Mustafee, K.-H. G. Bae, S. Lazarova-Molnar, M. Rabe, C. Szabo, P. Haas, and Y.-J. Son, 2236–2247. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Ma, Y., F. Qiao, F. Zhao, and J. Sutherland. 2017. "Dynamic Scheduling of a Semiconductor Production Line Based on a Composite Rule Set". *Applied Sciences* 7(12):1052.

Nasiri, M. M., R. Yazdanparast, and F. Jolai. 2017. "A simulation optimisation approach for real-time scheduling in an open shop environment using a composite dispatching rule". *International journal of computer integrated manufacturing* 30(12):1239–1252.

Panwalkar, S. S., and W. Iskander. 1977. "A Survey of Scheduling Rules". *Operations Research* 25(1):45–61.

Pergher, I., and A. T. de Almeida. 2018. "A multi-attribute, rank-dependent utility model for selecting dispatching rules". *Journal of Manufacturing Systems* 46:264–271.

Poppenborg, J., S. Knust, and J. Hertzberg. 2012. "Online scheduling of flexible job-shops with blocking and transportation". *European Journal of Industrial Engineering (EJIE)* 6(4):497–518.

Ramirez-Hernández, J. A., and E. Fernandez. 2009. "A simulation-based approximate dynamic programming approach for the control of the intel mini-fab benchmark model". In *Proceedings of the 2009 Winter Simulation Conference*, edited by M. D. Rossetti, R. R. Hill, B. Johansson, A. Dunkin, and R. G. Ingalls, 1634–1645. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Riley, M., Y. Mei, and M. Zhang. 2016. "Improving job shop dispatching rules via terminal weighting and adaptive mutation in genetic programming". In *2016 IEEE Congress on Evolutionary Computation (CEC)*, 3362–3369. Piscataway, NJ: Institute of Electrical and Electronics Engineers.

Sharma, P., and A. Jain. 2016. "Effect of routing flexibility and sequencing rules on performance of stochastic flexible job shop manufacturing system with setup times: Simulation approach". *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture* 231(2):329–345.

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.

Stricker, N., A. Kuhnle, R. Sturm, and S. Friess. 2018. "Reinforcement learning for adaptive order dispatching in the semiconductor industry". *CIRP Annals* 67(1):511–514.

Waschneck, B., A. Reichstaller, L. Belzner, T. Altenmüller, T. Bauernhansl, A. Knapp, and A. Kyek. 2018. "Optimization of global production scheduling with deep reinforcement learning". *Procedia CIRP* 72(1):1264–1269.

Welch, P. D. 1983. "The statistical analysis of simulation results". *The computer performance modeling handbook* 22:268–328.

Zhang, F., Y. Mei, and M. Zhang. "Can Stochastic Dispatching Rules Evolved by Genetic Programming Hyper-heuristics Help in Dynamic Flexible Job Shop Scheduling?". In *2019 IEEE Congress on Evolutionary Computation (CEC)*, 41–48. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Zhao, M., X. Li, L. Gao, L. Wang, and M. Xiao. 2019. "An improved Q-learning based rescheduling method for flexible job-shops with machine failures". In *2019 IEEE 15th International Conference on Automation Science and Engineering (CASE)*, 331–337. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Zheng, S., C. Gupta, and S. Serita. 2019. "Manufacturing Dispatching using Reinforcement and Transfer Learning". *arXiv preprint arXiv:1910.02035*.

AUTHOR BIOGRAPHIES

JENS HEGER is a professor at Leuphana University in Lueneburg. He has been leading the work group "modeling and simulation of technical systems and processes" at the Institute for Product and Process Innovation. Furthermore, he is a speaker at the research center for digital transformation at Leuphana University. During his doctorate in the field of production engineering, he worked on various research projects as a research associate. Since 2015, he has been a junior professor at Leuphana University in Lueneburg. His research focuses on the dynamic adjustment of parameters based on system states. He is leading various projects with different funding agencies. His e-mail address is jens.heger@leuphana.de

THOMAS VOSS is a research associate at Leuphana University. He completed his studies as an industrial engineer with a focus on production engineering in 2016. He graduated with a master thesis on the optimal scheduling of autonomous guided vehicles in a blocking job shop environment. His Ph.D. project is a follow up on this topic and includes the sequencing and routing of operations in a manufacturing system. He has been able to gain experience in simulation and optimization through various projects. His e-mail address is thomas.voss@leuphana.de.