Simulation in Produktion
und Logistik 2021
Jörg Franke & Peter Schuderer (Hrsg.)
Cuvillier Verlag, Göttingen 2021

Simulative Dispatching Optimization of Maintenance Resources in a Semiconductor Use-Case Using Reinforcement Learning

Simulative Optimierung der Planung von Wartungsressourcen in einem Halbleiter-Anwendungsfall mithilfe von Reinforcement Learning

Clara Hoffmann, Thomas Altenmüller, Infineon Technologies AG, Neubiberg (Germany), clara.hoffmann@infineon.com, thomas.altenmueller@infineon.com

Marvin Carl May, Andreas Kuhnle, Gisela Lanza, Karlsruher Institut für Technologie, Karlsruhe (Germany), marvin.may@kit.edu, andreas.kuhnle@kit.edu, gisela.lanza@kit.edu

Abstract: Semiconductors are high-tech products imposing strong requirements on manufacturing processes. To meet these high quality and accuracy standards, semiconductor manufacturing requires complex and reliable processes on high-tech equipment. Due to the high investment costs for machines and thus associated high machine downtime costs and high process dynamics, challenges arise in handling machine maintenance and finding the optimal maintenance resource dispatching for machines requiring repair or preventive maintenance. The interdependency between production control and maintenance resource dispatching can be modelled using a complex manufacturing system simulation. Deep reinforcement learning is promising in handling the ever-increasing complexity in modern production systems and the associated optimization of maintenance resource dispatching.

1 Introduction

Handling the maintenance of machines cannot be studied appropriately without the context of the underlying manufacturing system. The interdependency between production control and maintenance resource dispatching can be modeled using a simulation of a complex manufacturing system. Reinforcement learning (RL), in particular Deep RL, offers promising opportunities to handle ever-increasing complexities in modern manufacturing systems and optimization of maintenance resource dispatching. Recent progress, e.g. RL beating the best human players in complex strategic games, stress this promising development. Therefore, this study investigates whether the application of RL to the problem of maintenance resource dispatching can improve the performance of semiconductor manufacturing systems.

2 Fundamentals and Literature Review

2.1 Production and Maintenance Planning

Production planning and control focuses on organizing and optimizing the internal processes in a manufacturing system (Eversheim & Wiendahl 2000). Maintenance planning is viewed as a special type of production planning (Ben-Daya et al. 2009). Thus, maintenance planning methods are developed in line with production planning methodologies. However, the two systems differ in various aspects. Maintenance jobs have more variability among themselves and maintenance planning requires the coordination with other functional units. In addition, in semiconductor manufacturing the highly stochastic failure pattern of machines lead to a higher variability of demand for maintenance work than for production.

Production planning aims at increasing machine utilization whereas maintenance planning targets guaranteeing high long-term machine readiness levels. Machine interruptions cause delays in production schedules and vice versa. This leads to conflicts in timing, creating negative interactions between the two functional units. To overcome these conflicts, a coordinated planning process and integrated optimization tools for planning and scheduling is proposed. One solution is to prioritize either function and to use the output plan as an input for the second function, i.e. the input from the other function is then taken as constraint (Al-Turki et al. 2014). This is the case when the maintenance schedule in maintenance planning is adjusted to the given WIP schedule in production planning.

2.2 Maintenance Resource and Task Assignment

In maintenance planning, maintenance resource dispatching can be split up into two categories: (1) maintenance resource assignment, where an available resource pulls work items from a common pool of work items, and (2) task assignment, pushing tasks to resources to queue for handling (Kumar et al. 2002). The decision problems coping with these assignments are either deterministic problems, with a static environment, or stochastic problems, solving assignment problems with tasks arriving in a dynamic, real-time manner. Deterministic problems are mostly solved with classical, mathematical programming techniques, considering different objectives: maximizing a pre-defined score (Kuhn 1995, Martello & Toth 1995), minimizing the job completion time (Arora & Puri 1998, Chauvet, Proth & Soumare 2000), or balancing the resource's workload (Karsu & Azizoglu 2012, Chen et al. 2017), as categorized in Table 1. Deterministic assignment problems can be transformed between categories. In stochastic environments, however, resource assignment is either solved rule-based (Iravani & Krishnamurthy 2007, Arias et al. 2018) or history-based (Liu et al. 2008, Millán-Ruiz & Hidalgo 2010). In comparison, for task assignment mainly rule-based solutions are implemented, either with fixed rules (Stecke & Aronson 1985, Iravani & Krishnamurthy 2007), or adapted rules (Mosley, Teyner & Uzsoy 1998, Langer et al. 2010). Besides, machine learning (ML) is recently used to solve resource and task assignment, using natural language processing (Mo et al. 2020), neural networks (Mao et al. 2016), and RL (Naik, Negi & Sastry 2015). However, so far the focus is on machine reliability and planning or scheduling maintenance service for a single equipment or multiple in a manufacturing system and only limited work was devoted to study the problem in a system-wide environment, integrating maintenance activities and considering machine and maintainer interaction.

Table 1: *Structure of existing literature in this domain*

	Deterministic	Stochastic	
		Resource assignment	Task assignment
Without machine learning	Kuhn (1995), Arora & Puri (1998), Martello & Toth (1995), Chauvet, Proth & Soumare (2000), Karsu & Azizoglu (2012), Chen et al. (2017)	Iravani & Krishnamurthy (2007), Arias et al. (2018), Liu et al. (2008), Millán-Ruiz & Hidalgo (2010)	Stecke & Aronson (1985), Mosley, Teyner & Uzsoy (1998), Iravani & Krishnamurthy (2007), Langer et al. (2010)
With machine learning	Mural, Puri & Prabhakaran (2010)	Mo et al. (2020), Mao et al. (2016)	Naik, Negi & Sastry (2015)

2.3 Basics of Reinforcement Learning

RL is a ML algorithm that can solve problems in dynamic environments. A so-called *RL agent* constantly adapts its strategy in a known or unknown environment through receiving feedback. Based on the state of the environment $\boldsymbol{s_t} \in S$ perceived at each time step t, it selects an action $a_t \in A$. The response of the environment includes the resulting state s_{t+1} and a feedback in form of reward r_t. A Markov decision process describes this iteration formally. The ultimate goal of the RL agent is to optimize its strategy to maximize the cumulated, discounted reward. (Altenmüller et al. 2020)

3 Use Case and Reinforcement Learning Modelling

This section introduces the use case oriented on a real-world wafer fab. The literature analysis findings are incorporated in the simulation set-up. The RL modelling regarding state and action space, reward function and the used optimization algorithm is introduced, answering the following research questions:

1. Can a single-agent-based RL approach learn to optimize global key performance indicators through maintenance resource dispatching?
2. Can the proposed RL agent outperform competitive standard heuristics?
3. Does task assignment perform better than resource assignment regarding defined key performance indicators?

3.1 Description of the Wafer-Fab Use Case in the Simulation

A process-based discrete-event simulation, based on Altenmüller et al. (2020), for production planning and control, is adapted for maintenance resource dispatching in order to train an intelligent and autonomous RL agent for decision making in a complex job shop environment with corrective and preventive maintenance, aiming at outperforming existing approaches. The discrete-event simulation represents the characteristics of typical complex semiconductor-like job shops.

The two approaches, resource and task assignment, are implemented in two independent views, the machine and maintenance view, as shown in Figure 1. The main difference between the two views is, that the maintainer view waiting time results from waiting in the maintenance machine shop, in contrast to the machine view, where the waiting time results from waiting inside the maintainer skill queue.

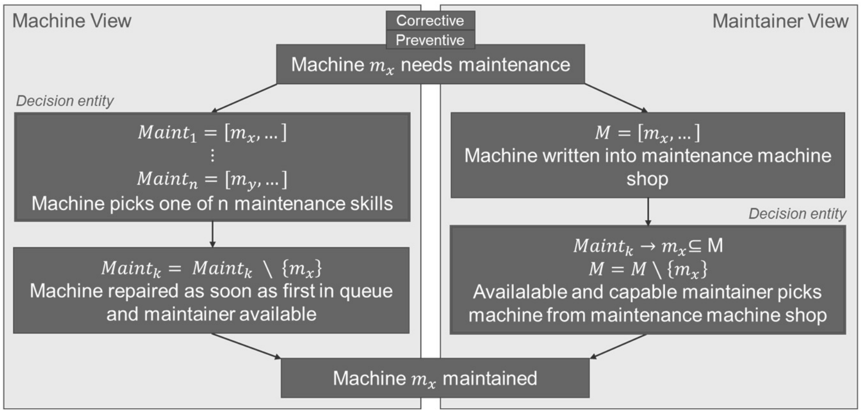
Machine View
Corrective
Preventive
Maintainer View
Machine m_x needs maintenance
Decision entity
$Maint_1 = [m_x, \ldots]$
⋮
$Maint_n = [m_y, \ldots]$
Machine picks one of n maintenance skills
$M = [m_x, \ldots]$
Machine written into maintenance machine shop
Decision entity
$Maint_k = Maint_k \setminus \{m_x\}$
Machine repaired as soon as first in queue and maintainer available
$Maint_k \rightarrow m_x \subseteq M$
$M = M \setminus \{m_x\}$
Availalable and capable maintainer picks machine from maintenance machine shop
Machine m_x maintained

***Figure 1:** Process structure of the machine view compared to the maintainer view*

The considered use case is based on the eWLB line (Altenmüller et al. 2020) and represents a production process with 40 specialized machines organized in five machine groups. Every machine group has a buffer stock with a fixed number of buffer slots. Challenges arise from machines requesting corrective or preventive maintenance which need to be handled by given, sparse maintenance resources. Hereby, the negative influence on the cycle time (CT) of the products being processed within the system shall be reduced to increase speed and lower costs. Furthermore, order flow is re-entrant in earlier visited machine groups. Finally, machines require a product-specific set-up. For the resulting product flow of the used, simplified eWLB line and the parameters used see Altenmüller et al. (2020).

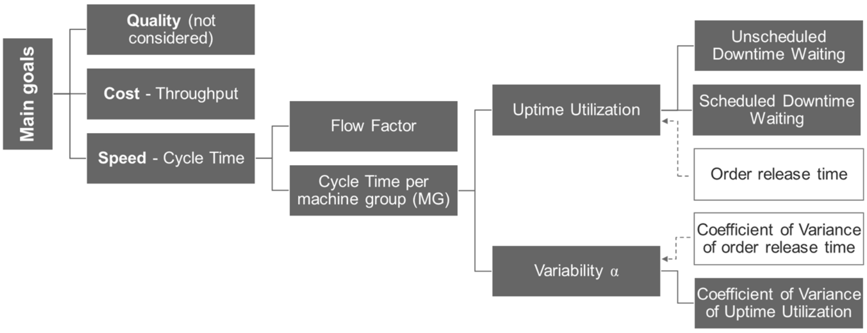
Main goals
Quality (not considered)
Cost - Throughput
Speed - Cycle Time
Flow Factor
Cycle Time per machine group (MG)
Uptime Utilization
Variability α
Unscheduled Downtime Waiting
Scheduled Downtime Waiting
Order release time
Coefficient of Variance of order release time
Coefficient of Variance of Uptime Utilization

***Figure 2:** Hierarchy of the tracked KPIs in the simulation*

3.2 Performance Evaluation

Key performance indicators (KPIs), oriented at SEMI E10-0312 (2012), are used for performance evaluation and, thus, the fulfillment of target performances. The KPIs evaluated in this paper are hierarchically ordered as displayed in Figure 2, showing which KPIs on the lower level (right), influence KPIs on the upper level (left). The overall goals are derived from the overall equipment performance (OEE).

3.3 State Space Modelling

The basis for the decision-making by the RL agent is the information in the observation space passed to it about the current status of the system, shown in Table 2. Resource-related information contain information about the decision requesting entity whereas maintenance-related information is made up of information about the decision to take as an action. Production-related information are more global, covering the state of the final products in the exit buffer.

Table 2: *Observation space for the RL agent in the machine and maintainer view*

Information	Machine view	Maintainer view
Resource-related	Active machine Machine group	Active maintainer Maintainer skills
Maintenance-related	Capable maintainer Maintainer shift count Mean time to repair (MTTR) in queue	Possible machines Possible machines for scheduled maintenance Waiting time per machine
Production-related	Waiting time of orders in stock Processing time of orders in stock	

3.4 Action Space Modelling

The action space is discrete, as the RL agent selects a distinct action from a finite action set that depends on the view. The action space of the machine view is smaller, as it contains seven options, one for each maintenance skill, whereas the maintainer view is made up of 40 options, one for each machine. Selecting a maintainer skill not capable of repairing the requesting machine in the machine view is considered as an invalid action, as well as selecting a machine not requesting maintenance in the maintainer view. Invalid actions are not executed (i.e. the simulation state is not changed) but used as feedback for learning: the agent is penalized with a negative reward and requested to select another action.

3.5 Reward Function Modelling

The reward function r is the key element that leads to the RL agent learning a desired behavior. As introduced above, invalid actions are in general rewarded with $r = -1$. A distinction is made between local and global rewards. Local rewards consider the current situation at the machine respectively maintainer requesting a decision, whereas global rewards regard the situation in the exit buffer for finished orders for computing the reward, as a proxy for the entire manufacturing system. The local reward in the machine view is based on a ranking of the length of the sum of the MTTR of machines waiting in the queue of the chosen maintainer skill normalized with the maintainer skill available. In the maintainer view, the ranking is built over the waiting times of the machines requesting maintenance.

The global reward targets the highest through maintenance resource dispatching manipulable level of the hierarchy of KPIs, namely the overall cycle time. For each selected valid action *a*, the RL agent is rewarded based on the cycle time of orders in the exit buffer. As single finished orders in the exit buffer cannot be linked to a certain maintenance resource dispatching decision taken, the cycle time of the last *n* orders is set in ratio to the cycle time of *n * 3* last orders. This is done by z-score normalization, where the value is then inserted in an exponential function such that the RL agent receives exponentially more reward for more desirable behavior than for bad behavior (Bonsai 2017). The resulting episodic global CT reward is made up of a sparse, global reward and a modeled reward, focusing on (in)valid actions.

3.6 Reinforcement Learning Algorithm

A Proximal Policy Optimization (PPO) agent provided by the library Stable Baselines is used as it outperformed Deep Q-Network and Trust Region Policy Optimization agents in several earlier investigations. In general, PPO is a model-free, online, on-policy, policy gradient reinforcement learning method and thus a type of policy gradient training that alternates between sampling data through environmental interaction and optimizing a clipped surrogate objective function with stochastic gradient descent. The clipped surrogate objective function improves training stability by limiting the policy change per step (Schulman et al. 2017).

3.7 Benchmark Heuristics

The benchmark heuristics are compared to the performance of the RL agent in the two views. In machine view, two heuristics are developed. First, the valid heuristic picks maintenance resources that have the skill to handle the machine randomly. Second, the local heuristic picks the maintenance skill with the shortest normalized waiting queue, such that the queues of the maintenance skills are smoothed according to the availability and capability of the maintenance resources, resulting in a smoothing of waiting times between the machines. In the maintainer view, the first in first out (FIFO), longest queue (LQ), longest queue shortest repair time (LQSRT), most wafer (MW) and least remaining capacity (LRC) heuristics from Mosley, Teyner & Uzsoy (1998) are adapted. The difference between these heuristics lies in the way of choosing machines by maintenance resources to maintain next.

4 Results

In this chapter, the results of the best performing benchmark heuristic are compared with results of the above explained local and global reward in both views.

4.1 Behavior of the Reinforcement Learning Agent

The agent's learning already converges after a relatively small number of steps. This is shown through the truncation point marked in grey in Figure 3 on the right, evaluating the development of the CT on the left, through using the MSER-5 algorithm, an efficient and effective truncation heuristic. Thus, a stopping criterion is implemented after the learning converged, disrupting the learning of the agent, at a point where the policy shows a good performance.

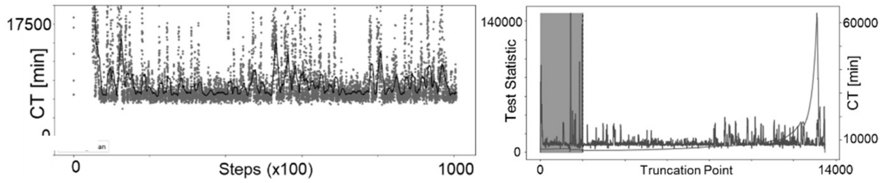

Figure 3: *Instability of the CT during learning and Convergence with MSER-5*

4.2 Comparison of Benchmark Heuristics and Reinforcement Learning Agent

Table 3: *Mean KPI values for machine and maintainer view (high machine failures)*

	TP	CT	CT MG0	CT MG1	CT MG2	CT MG3	CT MG4	UDW	SDW
Machine view	**Heuristic – local**								
	3.79	12700	179	195	4633	501	690	26.27%	4.64%
	Local reward								
	3.72	12744	180	193	4631	495	721	26.45%	4.55%
	Episodic global CT reward								
	3.18	7206	3126	137	375	283	505	33.51%	4.63%
Maintainer view	**Heuristic – LQ**								
	4.36	7748	1887	294	797	700	704	28.95%	1.87%
	Local reward								
	3.76	11995	170	242	4594	412	419	23.07%	5.94%
	Episodic global CT reward								
	4.09	6146	2426	167	381	291	357	22.05%	3.83%

Table 3 presents the results of both the machine and maintainer view with comparably high machine failures. Generally, minimizing the waiting times through local heuristics and rewards for maintenance actions does not lead to the shortest CT, as 90% of the machines are not bottleneck machines, such that waiting for maintenance has a small influence on the total CT. The CT is further reduced using the RL agent with episodic global CT reward compared to the benchmark heuristics. Through the learnt RL policy, the bottleneck is shifted to the first machine group. Thus, the order release into the production system is regulated by high waiting times for maintenance in the first machine group und hence a reduction of availability of machines in the first machine group, also referred to as trumpet planning. The maintainer resources are mainly used to maintain downstream machines. This leads to a high availability of these machines such that orders pass downstream machines as soon as they are released from the first machine group. However, the reduction of the CT happens at the expense of the Throughput (scheduling dilemma). The main mechanism is the implemented simulation logic, reducing order release into the system as soon as buffer and overflow buffer are overloaded. The RL agent utilizes this by filling the buffers in front of the first machine group by disproportionately increased waiting times for maintenance, referred to as reward hacking.

Comparing these results to results with realistic, smaller machine failures in Figure 4 shows, that the throughput is not influenced negatively by the CT reduction, as the agent cannot influence the order release to the previous extent. However, in the maintainer view, the CT performance through using the episodic global CT reward is not further improved in comparison to the benchmark heuristic.

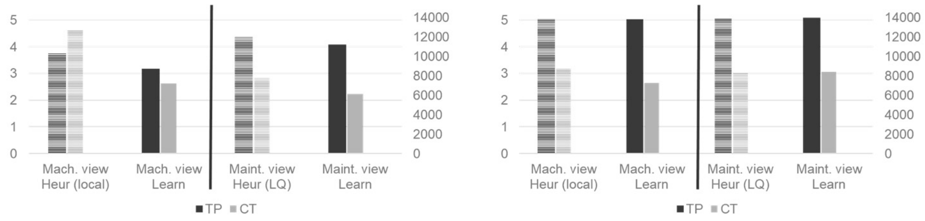

Figure 4: *Result comparison for large (right) and small (left) machine failure rates*

4.3 Discussion

The observation that for higher time to failures (TTFs) the PPO agent in the machine view is performing worse regarding the CT than in the maintainer view can be explained with the different action space size. More differentiated decision-making is possible in the maintainer view due to the bigger action space size. Furthermore, in the machine view, a queue of machines requests in the respective maintenance skill is built up. As the queue is quite long and based on the MTTR of the respective machines, the decision is fraught with uncertainty and not changeable after the initial assignment. In addition, the effect of the decision is more delayed and uncorrelated than within the maintainer view, where the decision is immediately executed.

In general, the large action space is reduced to valid machines, respectively those requesting maintenance. As the total number of machines requesting maintenance is reduced in comparison to lower TTF, the decision options are reduced significantly, such that a differentiated decision might have less influence on the request waiting times. Thus, ad hoc decisions in the maintainer view lack the benefits of planning incorporated in the machine view by filling queues of maintenance resources due to a more short-term view, in a similar setup shown by Millán-Ruiz & Hidalgo (2010).

The PPO agent with episodic global CT reward within the machine view could be used to build up a decision-making according to 'safe AI', which are in general methods working towards a higher safety of AI. The decisions then are short term, with the decision being executed not immediately. Through this, a human control of the resulting maintenance resource dispatching from the PPO agent is possible, increasing the safety aspect and thus leading to a safer AI application in practice.

5 Summary

Semiconductor manufacturing in times of pervasive digitalization opens up new opportunities and challenges, leading to a new era of operations management. The use of RL enhances intelligent and autonomous control approaches. This paper contributes to research that concerns the performance of maintenance resource dispatching in the context of a complex flexible job shop. The paper presents RL utilized to regulate maintenance resource dispatching to achieve CT minimization.

The results of multiple simulation runs show that different RL agents can autonomously develop a policy for solving the maintenance resource dispatching problem. The resulting KPIs, especially the target CT performance of the best PPO agent is superior to the underlying benchmark. If machines request maintenance less often, short-term planning through assigning machines to maintenance resources is outperforming maintenance resource dispatching with ad hoc decisions regarding the CT performance, whereas for a larger number of requests ad hoc decisions excel.

6 Outlook

Based on this results, further research should be conducted on using Deep RL for maintenance resource dispatching, i.e. focusing on further improving the performance of Deep RL agents. Instead of using a single decision-making RL agent, multiple RL agents, respectively a multi-agent system for dispatching, maintenance scheduling and maintenance resource dispatching should be considered. The implemented system architecture offers the potential to conduct more detailed analyses by extending the maintainer skill matrix through further splitting up skills in preventive and corrective skills, as well as incorporate re- prioritization and thus maintenance action disruptions. The parameters for setting up the simulation are oriented at realistic data but adapted and simplified. Through process mining, gaining realistic simulation input data is possible. Thus, simulating production lines in factories realistically becomes possible.

References

Altenmüller, T., Stüker, T., Waschneck, B., Kuhnle, A., & Lanza, G. (2020). Reinforcement learning for an intelligent and autonomous production control of complex job-shops under time constraints. Production Engineering, 14, 319-328.

Al-Turki, U. M., Ayar, T., Yilbas, B. S. & Sahin, A. Z. (2014), Integrated maintenance planning in manufacturing systems, Springer, New York.

Arias, M., Munoz-Gama, J., Sepúlveda, M., & Miranda, J. C. (2018). Human resource allocation or recommendation based on multi-factor criteria in on-demand and batch scenarios. European Journal of Industrial Engineering, 12(3), 364-404.

Arora, S., & Puri, M. C. (1998). A variant of time minimizing assignment problem. European Journal of Operational Research, 110(2), 314-325.

Ben-Daya, M., Duffuaa, S. O., Raouf, A., Knezevic, J. & Ait-Kadi, D. (2009), Handbook of maintenance management and engineering, vol. 7, Springer.

Bonsai (2017), Deep Reinforcement Learning Models: Tips and Tricks for Writing Reward Functions. https://medium.com/@BonsaiAI/deep-reinforcement-learning-models-tipstricks-for-writing-reward-functions-a84fe525e8e0.

Chauvet, F., Proth, J. M., & Soumare, A. (2000). The simple and multiple job assignment problems. International journal of production research, 38(14), 3165-3179.

Chen, G., He, W., Leung, L. C., Lan, T., & Han, Y. (2017). Assigning licenced technicians to maintenance tasks at aircraft maintenance base: a bi-objective approach and a Chinese airline application. International Journal of Production Research, 55(19), 5550-5563.

Eversheim W., Wiendahl HP. (eds) (2000). Wörterbuch der PPS—Dictionary of PPC: German–English/English–German. Springer, Berlin

Iravani, S. M., & Krishnamurthy, V. (2007). Workforce agility in repair and maintenance environments. Manufacturing & Service Operations Management, 9(2), 168-184.

Karsu, Ö., & Azizoğlu, M. (2012). The multi-resource agent bottleneck generalised assignment problem. International Journal of Production Research, 50(2), 309-324.

Kuhn, H. W. (1955). The Hungarian method for the assignment problem. Naval research logistics quarterly, 2(1-2), 83-97.

Kumar, A., Van Der Aalst, W. M., & Verbeek, E. M. (2002). Dynamic work distribution in workflow management systems: How to balance quality and performance. Journal of Management Information Systems, 18(3), 157-193.

Langer, R., Li, J., Biller, S., Chang, Q., Huang, N., & Xiao, G. (2010). Simulation study of a bottleneck-based dispatching policy for a maintenance workforce. International Journal of Production Research, 48(6), 1745-1763.

Liu, Y., Wang, J., Yang, Y., & Sun, J. (2008). A semi-automatic approach for workflow staff assignment. Computers in Industry, 59(5), 463-476.

Mao, H., Alizadeh, M., Menache, I., & Kandula, S. (2016, November). Resource management with deep reinforcement learning. In Proceedings of the 15th ACM workshop on hot topics in networks (pp. 50-56).

Martello, S., & Toth, P. (1995). The bottleneck generalized assignment problem. European journal of operational research, 83(3), 621-638.

Millán-Ruiz, D., & Hidalgo, J. I. (2010, April). A memetic algorithm for workforce distribution in dynamic multi-skill call centres. In European Conference on Evolutionary Computation in Combinatorial Optimization (pp. 178-189). Springer, Berlin, Heidelberg.

Mo, Y., Zhao, D., Du, J., Syal, M., Aziz, A., & Li, H. (2020). Automated staff assignment for building maintenance using natural language processing. Automation in Construction, 113, 103150.

Mosley, S. A., Teyner, T., & Uzsoy, R. M. (1998). Maintenance scheduling and staffing policies in a wafer fabrication facility. IEEE Transactions on Semiconductor Manufacturing, 11(2), 316-323.

Mural, R. V., Puri, A. B., & Prabhakaran, G. (2010). Artificial neural networks based predictive model for worker assignment into virtual cells. International Journal of Engineering, Science and Technology, 2(1), 163-174.

Naik, N. S., Negi, A., & Sastry, V. N. (2015). Performance improvement of MapReduce framework in heterogeneous context using reinforcement learning. Procedia Computer Science, 50, 169-175.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A. & Klimov, O. (2017), „Proximal policy optimization algorithms“, in: arXiv preprint arXiv:1707.06347.

SEMI E10-0312 (2012), Specification for definition and measurement of equipment reliability, availability, and maintainability (RAM) and utilization.

Stecke, K. E., & Aronson, J. E. (1985). Review of operator/machine interference models. International Journal of Production Research, 23(1), 129-151.