An Application of Reinforcement Learning in Industrial Cyber-Physical Systems

David Heik[1], Fouad Bahrpeyma[1] and Dirk Reichelt[1]

[1]*Faculty of Informatics/Mathematics, University of Applied Sciences Dresden, 01069 Dresden, Germany*

Abstract

Fully automated manufacturing plants are designed to perform the processes effectively and efficiently. During the planning stage, some behavioral controls are already implemented in order to proactively respond to unforeseen outcomes such as a deterioration in the quality of the products, a lack of material supply, or sudden maintenance work. However, not every scenario can be covered, especially if the structure of the production line changes over time or new product variants with different characteristics are introduced. With a particular focus on minimizing the overall completion time (makespan), in this paper we present a simulation environment that mimics an assembly line of the Industrial IoT Test Bed (at HTW Dresden). In this regard we incorporated reinforcement learning techniques such as Deep-Q Networks (DQN), REINFORCE, Advantage Actor Critic (A2C) and Proximal Policy Optimization (PPO) as a mean to bring cost efficiency and productivity in an automated manner. We investigate how fast the trained models converge and how accurately they solve the different problems.

Keywords

Artificial Intelligence, Reinforcement Learning, Manufacturing Systems, Smart Production Systems

1. Introduction

In modern manufacturing systems, job assignment and dynamic scheduling are among the major challenges. Scheduling performance is directly influenced by the design of the corresponding control mechanism and reflects primarily on the production rate and the overall completion time. Nevertheless, the performance of a manufacturing plant depends on a number of challenging factors and, in large part, on the dynamic nature of the manufacturing environment. Dynamic events such as failures or jams pose challenges to traditional static scheduling approaches. As part of an industrial lab at HTW Dresden for smart manufacturing, we are working on a wide range of issues, including control problems to optimize the production process and the corresponding performance. The IIoT Test Bed consists of 13 production stations, where some can perform similar operations. This redundancy allows the physical system to handle problems such as bottlenecks, maintenance or failures and maintain the overall performance. However, the existing control software is quite static and does not take into account dynamic events during the production process, thus preventing the potential from being fully exploited. The challenge is to consider at the conception stage all possible scenarios that could occur during the production process, even though the structure of the system may change. In this paper, Reinforcement

OVERLAY 2022: 4th Workshop on Artificial Intelligence and Formal Verification, Logic, Automata, and Synthesis, November 28, 2022, Udine, Italy

david@heik.science (D. Heik); bahrpeyma@ieee.org (F. Bahrpeyma); dirk.reichelt@htw-dresden.de (D. Reichelt)

Learning (RL) was incorporated to realize dynamic planning for the use of parallel machines to support same operations. Therefore, planning for the allocation of resources is where RL is integrated with the system. While the literature tends to refer to scheduling in a broader sense, job shop scheduling, we have a somewhat different goal of considering redundant capabilities with uncertainty in the operation cycle. The literature reports for the use of algorithms of various types for scheduling including rule-based, heuristics, metaheuristic approaches, supervised learning based methods and also, a field known as reinforcement learning. Boden et al. [1] suggested a rule-based method for a dynamic scheduling of product transportation using heterogeneous automated material handling systems. Svensson et al. [2] presented an offline optimisation method based on a multiple linear regression. Akyol et al. [3] proposed a real time scheduler algorithm based on the neuro-fuzzy network wherein an offline learning procedure was used to develop decision heuristics. RL methods have been used in the literature as alternative problem solvers, especially for dynamic environments. Said et al. [4] used RL to create a dynamic, flexible schedule for job store manufacturing system. In Shafiq et al. [5], the authors introduced a DQN and SARSA to find optimal policies for scheduling jobs of various types. Zadeh et al. [6] introduced a heuristic algorithm that minimises makespan considering the variation in processing time. However, a little attention have been paid to the presence of redundant capabilities as well as sources of uncertainty, for instance the chance of machine failure or varying operation times. This paper incorporates RL to address the mentioned issue in an effective manner.

2. Methodology

RL is a method for dealing with uncertainty where optimization is the main objective. Within this work, RL was used to find the an optimal and reliable scheduling policy for minimizing makespan in a production system characterized by multiple machines supporting the same operations while experiencing uncertain operating times. This paper, however studies the sole application of RL to the specific architecture of our IoT Test Bed while due to the limitations in the the content space, refers the reader to [7] for detailed explanations of the implementation. As soon as the training process is completed, the policy will serve as a function that observes the state of the machine and suggests decisions on the allocation of jobs. The complete structure of our IIoT Test Bed is shown in Fig. 1. This paper only studies the simplest product variant to be manufactured, which only requires 3 operations, whereby the execution order (A→B→C) must be adhered to. The stations that can be used for this were highlighted in color, and the rest have not been considered in the simulation. They are connected via a conveyor belt and the workpieces are transported using carriers. In our discrete manufacturing system, carriers move one slot further during each time step, and wait for their predecessors to move on or for stations to complete their operations. By increasing the number of carriers and/or parallel stations, the amount of decision combinations for a specific problem also increases exponentially, as shown by equation (1).

$$Decisions = (2^{ParallelOperations})^{Carriers} \tag{1}$$

Since the discovery of the global minimum is an NP-hard problem, it requires a significant amount of computational effort even with a few parallel stations and a small number of carriers. Tbl. 1, represents this correspondence in numbers. In this table, similar to our experiments, a margin of ± 3 seconds was considered as well as the possibility of failure. As a description

Situation Index	Uncertainty (Duration±)	No. of carriers	No. of poss. initial states	No. of poss. decisions	No. of independent experiments to completely represent the solution space
SI=1	1	4	$(3^4) * (24 choose 4) = 860.706$	16	13.771.296
SI=2	3	4	$(7^4) * (24 choose 4) = 25.513.026$	16	408.208.416
SI=3	1	6	$(3^4) * (24 choose 6) = 10.902.276$	64	697.745.664
SI=4	3	6	$(7^4) * (24 choose 6) = 323.164.996$	64	20.682.559.744

Table (1): Complexity

of the methods used to implement the RL algorithm is beyond the scope of this paper, we mention here only these used techniques and refer the reader to further reading: DQN [8, 9, 10], REINFORCE [11], A2C [12, 13] and PPO [14]. In our approach, we used formal verification in the form of an early stopping criteria during the process of reinforcement learning. This formal verification method is repeated for each episode of training as a validation criteria to ensure the performance level of the learning process. In this regard, a test and an evaluation dataset has been incorporated. Due to the largeness of the problem space, it's practically impossible to perform experiments for all the possible combinations. Consequently, we were able to calculate the global makespan minimum for only a fraction of initial states. The experiments were performed in our simulated environment, where all possible decision combinations were evaluated. This type of verification, however, is not common in RL applications as RL techniques are mainly verified by the reward which cannot guarantee the total safety of the policy learnt. The models were trained with randomly generated initial states for which the global minimum is not known. To determine if the models converge or not, one test dataset was used after each training episode for which the global minimum is known already. Then, according to equation (2), the normalized value of the overall completion time was stored in a list. The mean value of this list was then used as a condition to determine the termination of the training process. In other words, this verification was also used as a mean of early stopping, which is kind of reducing the search space.

$$Score = (\frac{OCT_{Worst} - OCT_{RL}}{OCT_{Worst} - OCT_{Best}}); Reward = (OCT_{RL})^3 \tag{2}$$

Where OCT_{Worst}, OCT_{Best} and OCT_{RL} stand for the worst, the best and the rl-output of overall completion times.

3. Evaluation

The experimental results for the simulated experiments are presented in Fig. 2, the 4 sections each representing the results for one RL method. Each section shows 4 experiments (SI=1,2,3 or 4) detailed in Tbl. 1. For each experiment, we applied 4 different early stopping condition that differ by their color listed in the legend (e.g. $\overline{OCT_{RL}} \geq 0.92; m = 10$), where m represents the horizon over which $\overline{OCT_{RL}}$ is calculated. The example means that early stopping is activated when $\overline{OCT_{RL}} \geq 0.92$ over the last $m = 10$ episodes. These experiments are repeated for the remaining conditions mentioned. In Fig. 2, boxplots each represents the results of multiple runs of the corresponding method with the same parameters. The trained models were applied to each of 1000 evaluation datasets. The numerical results for the experiments evaluated in Fig. 2 are provided in Tbl. 2. As shown in the results, PPO was able to outperform DQN, A2C and REINFORCE in terms of the average score, duration and the number of episodes required for training with rising uncertainty. Therefore, PPO is the recommended method for our IIoT Test Bed to realize dynamic scheduling.

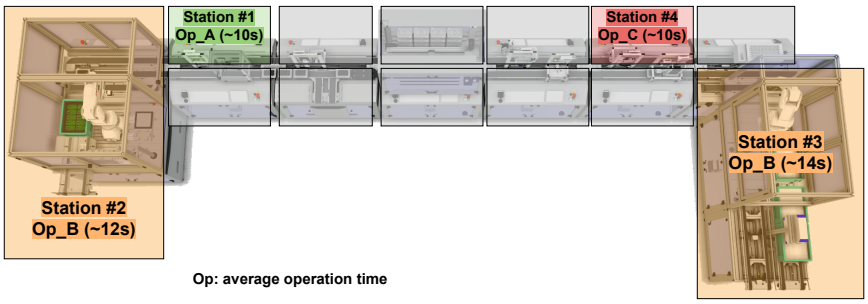

Figure (1): Structure of our IIoT Test Bed with highlighted stations used in the simulation

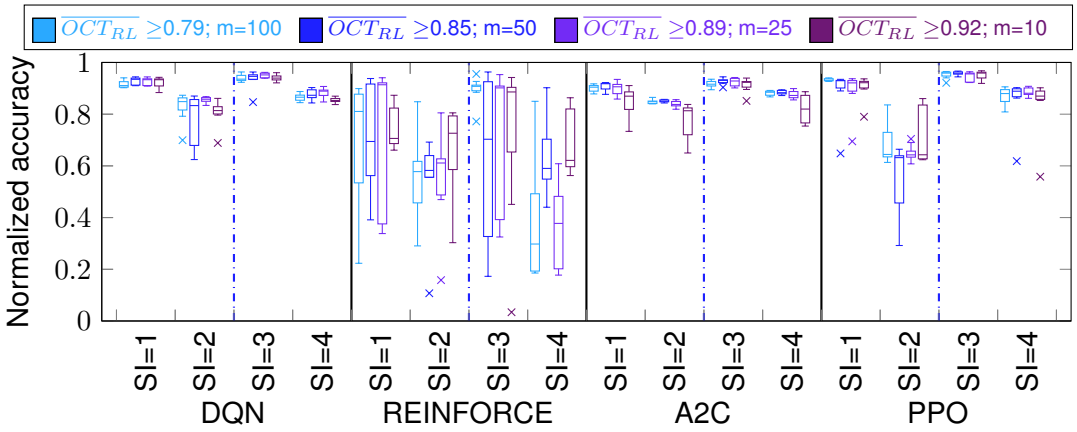

Figure (2): Results in Boxplots

Situation Index	best alg.	best e.s.m.	av score (best)	av no. of ep. (best)	fastest alg.	fastest e.s.m.	av score (fastest)	av no. of ep. (fastest)	av score (Δ)	av no. of ep. (Δ)
SI=1	PPO	0.85-50	0.928	307	DQN	0.85-50	0.915	273	-0,013	-34
SI=2	DQN	0.89-25	0.861	24781	DQN	0.89-25	0.819	799	-0,042	-23982
SI=3	PPO	0.89-25	0.957	321	PPO	0.79-100	0.951	202	-0,006	-119
SI=4	PPO	0.79-100	0.890	5563	PPO	0.92-10	0.869	524	-0,021	-5039

Table (2): Best and fastest converging models

4. Conclusion

In this paper, we proposed the use of RL for dynamic job scheduling in a virtual environment that has the behaviour of our IIoT Test Bed. For different scenarios, we illustrated how the corresponding NP-hard scheduling problem can be addressed using RL without the need for examining an enormous number of experiments.

Acknowledgments

This work has been supported and funded by the ESF (European Social Fund) as part of the REACT research group "Wandlungsfähige Produktionsumgebungen" (WaPro, application number: 100602780). REACT-EU: Funded as part of the EU reaction to the COVID-19 pandemic.

References

[1] P. Boden, S. Rank, T. Schmidt, Control of heterogenous AMHS in semiconductor industry under consideration of dynamic transport carrier transfers, in: 2021 22nd IEEE International Conference on Industrial Technology (ICIT), IEEE, 2021. doi:10.1109/icit46573.2021.9453585.

[2] D. T. Pham, A. A. Afify, Machine-learning techniques and their applications in manufacturing, Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture 219 (2005) 395–412. doi:10.1243/095440505X32274. arXiv:https://doi.org/10.1243/095440505X32274.

[3] D. E. Akyol, G. M. Bayhan, A review on evolution of production scheduling with neural networks, Computers & Industrial Engineering 53 (2007) 95–122. URL: https://www.sciencedirect.com/science/article/pii/S0360835207000666. doi:https://doi.org/10.1016/j.cie.2007.04.006.

[4] N. E.-D. A. Said, Y. Samaha, E. Azab, L. A. Shihata, M. Mashaly, An online reinforcement learning approach for solving the dynamic flexible job-shop scheduling problem for multiple products and constraints, in: 2021 International Conference on Computational Science and Computational Intelligence (CSCI), IEEE, 2021. doi:10.1109/csci54926.2021.00095.

[5] S. Shafiq, C. Mayr-Dorn, A. Mashkoor, A. Egyed, Towards optimal assembly line order sequencing with reinforcement learning: A case study, in: 2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), IEEE, 2020. doi:10.1109/etfa46521.2020.9211982.

[6] M. S. Zadeh, Y. Katebi, A. Doniavi, A heuristic model for dynamic flexible job shop scheduling problem considering variable processing times, International Journal of Production Research 57 (2018) 3020–3035. doi:10.1080/00207543.2018.1524165.

[7] D. Heik, Discrete manufacturing simulation environment (Version_0.0.1), 2022. URL: https://doi.org/10.5281/zenodo.7214999. doi:10.5281/zenodo.7214999.

[8] J. Peng, R. J. Williams, Incremental multi-step q-learning, Machine Learning 22 (1996) 283–290. doi:10.1007/bf00114731.

[9] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, M. Riedmiller, Playing atari with deep reinforcement learning, 2013. doi:10.48550/ARXIV.1312.5602.

[10] S. Kapturowski, G. Ostrovski, J. Quan, R. Munos, W. Dabney, Recurrent experience replay in distributed reinforcement learning, in: International conference on learning representations, 2018.

[11] R. J. Williams, Simple statistical gradient-following algorithms for connectionist reinforcement learning, Machine Learning 8 (1992) 229–256. doi:10.1007/bf00992696.

[12] R. S. Sutton, A. G. Barto, Reinforcement learning: An introduction, MIT press, 2018.

[13] T. Degris, P. M. Pilarski, R. S. Sutton, Model-free reinforcement learning with continuous action in practice, in: 2012 American Control Conference (ACC), IEEE, 2012, pp. 2177–2182.

[14] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov, Proximal policy optimization algorithms, arXiv preprint arXiv:1707.06347 (2017).