Reinforcement Learning for Dynamic Scheduling in Intelligent Manufacturing Systems: Theory, Methods, and Engineering Practice

Dong Yuxin
School of Materials Science and Engineering, Nanjing Institute of Technology, China

ABSTRACT

Dynamic scheduling-constantly changing which machines and resources do what work, unexpected arrival, constantly changing working hours and equipment failures-has always been too difficult to solve with perfect mathematical formulas. Therefore, the manufacturing system uses simple and fixed rules, but when the production environment changes greatly, these rules cannot work normally. Reinforcement learning (RL) regards scheduling as a series of decisions, in which agents learn strategies (state-to-action strategies) by interacting with the production environment, which has become the main AI method to solve this problem. This paper reviews the dynamic scheduling and RL in intelligent manufacturing system, the theory of network physical system (CPS), system engineering and information theory. Scheduling strategy is regarded as a compressed version of production status information, which is enough to make nearly perfect decisions. We follow the theoretical basis of RL-based scheduling, including markov decision processes, state-action representation and reward design. We also analyzed four transformation paths for RL to change the manufacturing industry: from standardized production to customized production, from reactive scheduling to active scheduling, from manual scheduling to autonomous scheduling, and from independent machines to integrated network physical scheduling ecosystem. The specific methods we see include deep Q-network agent, graph-neural-network strategy learning for job shop problems, compound reward shape and explainable RL substitution for production control. We also studied the application of fault detection in planning, optimization of quality-related processes and man-machine allocation, challenging issues, such as sample efficiency, generalization of different problem sizes, and security of deployment. We finally come to a big framework, which says that reward design is the main interface between manufacturing goals and policy learning behavior.

Keywords: Reinforcement Learning; Dynamic Scheduling; Job Shop Scheduling; Cyber-Physical Systems; Smart Manufacturing; Deep Q-Network; Production Control; Markov Decision Process.

1. INTRODUCTION

Production scheduling determines the sequence and timing with which jobs are assigned to machines, and its quality directly governs throughput, due-date performance, and resource utilization across a manufacturing system. Classical scheduling theory addresses this problem through combinatorial optimization—mixed-integer programming, dispatching heuristics, metaheuristic search—approaches that perform well under static, fully known problem instances but degrade sharply when confronted with the stochastic arrivals, machine breakdowns, and processing-time variability characteristic of real production floors. Cyber-physical systems theory frames this degradation as a fundamental mismatch between the static optimization assumptions of classical scheduling and the continuously evolving physical state that a production CPS must actually respond to [1]. Reinforcement learning directly solves this problem: scheduling is not regarded as a one-time optimization problem, but as a series of decisions. The agent looks at the current production status, selects a scheduling operation, gets a reward for displaying the result of the operation, and updates its policy, all of which do not need a clear future work arrival model [2].

This sequential and modeless aspect makes RL very suitable for dynamic scheduling. The traditional method needs to recalculate the optimization when the production state changes, and once RL strategy is trained, it will look at the current state and make scheduling decisions in a constant time, thus meeting the demand of real-time response for production control based on CPS [1]. From the perspective of information theory, the driving strategy can be regarded as a dynamic compressed version of the production environment-transforming a complex random state space into a simple operation suggestion is enough to make a near-optimal decision. Therefore, the biggest technical challenge is not whether RL can learn scheduling strategy, but how to design state representation, action space and reward function, so that the obtained strategy can truly optimize the manufacturing objectives that the system must achieve.

Section 2 explains the theoretical basis of RL-driven scheduling in CPS and system theory. In section 3, four transformation paths brought by RL-based scheduling for intelligent manufacturing system are analyzed. Section 4 introduces the application in fault prediction and scheduling, process optimization and man-machine cooperation. Section 5 reviews current challenges and prospects, and section 6 ends.

2. THE THEORETICAL FOUNDATION OF ARTIFICIAL INTELLIGENCE AND INTELLIGENT ENGINEERING SYSTEMS

2.1 The Connotation and Characteristics of Intelligent Engineering Systems

Within the RL-scheduling paradigm, an intelligent engineering system is best characterized as a cyber-physical production system whose dispatching decisions are generated by a learned policy interacting in closed loop with the physical shop floor [1]. Three characteristics distinguish such systems from conventional rule-based scheduling architectures. First, they exhibit sequential state-dependence: each dispatching decision is conditioned not merely on the current snapshot of machine and queue states but, implicitly, on the policy's learned expectation of how that decision propagates through future states, distinguishing RL scheduling from myopic dispatching rules that consider only immediate priority. Second, they exhibit reward-mediated objective alignment: the system's scheduling behavior is shaped entirely through the reward signal provided during training, meaning the manufacturing objectives—makespan, tardiness, utilization, energy consumption—must be operationalized as a quantitative reward before the system can be said to pursue them at all. Third, they exhibit generalization-bounded adaptivity: a trained policy adapts to novel production states only insofar as those states resemble the distribution of states encountered during training, a property with direct consequences for deployment robustness.

2.2 The Theoretical Logic of Artificial Intelligence Empowering Engineering Systems

The Markov decision process (MDP) formalism provides the formal bridge between systems engineering's requirement for traceable decision logic and RL's capacity for adaptive, data-driven control. A scheduling MDP defines a state space capturing relevant production information (queue lengths, machine availability, due-date proximity), an action space corresponding to feasible dispatching decisions, a transition function governing how actions change the production state, and a reward function encoding the scheduling objective; the RL agent's task is to learn a policy maximizing expected cumulative reward over this formalism [2]. The deep Q-network architecture, which approximates the action-value function with a neural network trained on temporal-difference error from interaction experience, demonstrated that this formalism could be learned directly from high-dimensional state representations without hand-engineered features, a result that directly motivated its subsequent application to production scheduling [2, 3].

From an information-theoretic standpoint, the state representation chosen for a scheduling MDP determines how much of the production environment's true complexity is preserved versus discarded before the policy ever sees it; representations that omit decision-relevant information impose a hard ceiling on achievable policy performance regardless of training effort, while representations exhibiting strong scale-invariance—such as the disjunctive-graph encodings used in graph-neural-network-based scheduling agents—allow a policy trained on small problem instances to generalize to substantially larger ones [4]. From a systems engineering standpoint, the reward function constitutes the formal specification against which the learned policy's behavior must be verified: because the policy's pursued objective is entirely determined by the reward signal, any divergence between intended manufacturing goals and the engineered reward will manifest as systematically misaligned scheduling behavior, making reward design—not merely model architecture—the central locus of engineering responsibility in RL-driven scheduling systems.

3. THE TRANSFORMATION PATH OF INTELLIGENT ENGINEERING SYSTEMS MODE DRIVEN BY ARTIFICIAL INTELLIGENCE

3.1 Transition from Standardized Production to Personalized and Flexible Manufacturing

Standardized production uses fixed scheduling rules, which are formulated for known and stable product combinations. Flexible customized manufacturing needs planning rules, which can work normally even when the work type and path are constantly changing. The scheduling agent based on graphical neural network can meet this demand. They represent job-shop scheduling problems with branching graphs. The structure of the graph will naturally grow with the size of the problem. This allows the strategies learned from small examples to be used in larger examples that have never been seen before, without having to learn again. This is a generalization attribute that has nothing to do with size, and the rules made

by hand naturally have no [4]. This ability is very important for customized manufacturing, because it allows a single strategy to manage the changes brought about by product customization without setting new rules for each product configuration.

3.2 Transition from Reactive Response to Proactive Prediction and Optimization

Reactive scheduling will only restart scheduling after the problem has slowed down. RL-based active scheduling predicts what will happen after planning decision through the value function learned during training. The compound reward formula integrates multiple targets (machine utilization, delay penalty and workflow inventory) with different weights into one signal, which helps well-trained agents to understand the trade-off between planning priorities, which cannot be displayed by a single target reactivity rule. This provides a planning behavior that can predict and balance these trade-offs throughout the production range, rather than improving individual figures by hurting others [5]. This change from passive to active comes directly from MDP formalism, which emphasizes cumulative reward rather than immediate reward.

3.3 Transition from Manual Operation to Autonomous Intelligent Control

Autonomous scheduling gives human beings the power to plan, or gives fixed rules to always effective learning strategies. This transformation was first proved in a large factory, which trained deep Q network cooperation agents to control the scheduling of multiple workstations in semiconductor manufacturing facilities. Each agent learns how to make local decisions by observing the behavior of neighboring agents, and they try to optimize their production goals without having to re-optimize them every time [3]. Then, other work extends this autonomy to adaptive production control systems with a single agent, which learn scheduling rules directly from workshop simulation. They have shown that compared with conventional scheduling methods, lead time and throughput time can be reduced even if there is realistic random disturbance [6]. But the transition to autonomous control has one limitation: political training distribution. Well-trained agents of interference types cannot guarantee that they will handle other very different types of interference well. This limitation is discussed later in section 5.1.

3.4 Transition from Isolated Systems to Integrated Cyber-Physical Ecosystems

The latest transformation has changed the way of using RL to arrange work: instead of just controlling a machine or workstation, it is now possible to coordinate multiple stages of production, logistics and maintenance together. Using multiple RL agents, each agent controls a different machine or workstation, and only shares part of the status information of the whole production. It is similar to the main version of deep Q-network, which was used in semiconductor planning before, but now it is used in larger and more diversified production systems [3]. By looking at a lot of research on the deep RL in production, we can see that researchers started with a simple demonstration (single machine, single lens), and now they make a larger deployment through multiple steps. But we also notice that in real industry, there are still few RL systems that can really manage the whole factory, and at the same time carry out planning, logistics and maintenance [7]. We added an idea to this transformation: in order to evaluate whether the integration of the whole ecosystem is good, we can look at the decomposability reward. That is to say, can we break down the big plan goal of the whole factory into small rewards for each agent, and these rewards will not conflict with each other? If possible, then RL agents can work together to achieve a good overall result, instead of everyone doing something beneficial to themselves but unfavorable to everyone.

4. APPLICATIONS OF ARTIFICIAL INTELLIGENCE IN INTELLIGENT ENGINEERING SYSTEMS

4.1 Application in Intelligent Fault Diagnosis and Predictive Maintenance

In the RL-driven scheduling system, fault awareness is mainly manifested in the ability of agents to directly incorporate machine availability issues into their scheduling decisions, rather than treating maintenance and scheduling as separate functions. Adaptive production control systems driven by reinforcement learning have shown that they can dynamically redirect tasks around unavailable machines without special subsystems to predict faults, because the agent state representation already contains information about the availability of machines, and its learned strategies are exposed to interruption scenarios similar to the actual machine downtime during training [6]. This integration shows the broader principles of RL-based scheduling and traditional methods: since the agent strategy depends on the complete production state at each decision moment, the fault information is naturally absorbed into the scheduling behavior without a special interface between diagnosis and scheduling, although the reliability of this integration depends entirely on whether the drive distribution represents the type of fault that the system will see when it is used.

4.2 Application in Intelligent Process Optimization and Quality Control

In RL scheduling, process-level optimization is not just the order of tasks. It also includes the overall optimization of process parameters related to throughput, power consumption and quality using carefully designed composite reward functions. The compound reward formula of production scheduling in smart factories shows that RL agents can be trained to balance multiple competing goals-including indicators related to product quality and resource efficiency-in a single learning strategy. In this way, we avoid the sequential and local tradeoffs when applying a separate optimization routine to each objective [5]. The systematic review of RL documents in production system also shows that the application is not only a simple task sequence, but an integrated control of process parameters. She pointed out that the emphasis on data-driven and real-time response in this field makes RL scheduling a natural complement to the broader quality control architecture, which uses continuous sensors instead of occasional sampling inspection [7].

4.3 Application in Intelligent Human-Machine Collaboration and Operational Efficiency Enhancement

In the production environment of RL programming, the cooperation between man and machine largely depends on the operator's ability to understand and trust the scheduling decision provided by the strategy. This is the demand that deep black box RL policy can't meet well. Explanatory reinforcement learning method is specially used for workshop production control, which meets this demand by transforming the trained scheduling strategy into a small agent in a decision tree that imitates the original strategy behavior and displaying the production state characteristics that drive each scheduling decision. In this way, operators can verify whether the scheduling priority follows legal standards, such as approaching the due date, instead of rewarding shape artifacts [8]. This explainability layer has direct operational consequences: production-control systems whose recommendations can be justified in terms operators already use are markedly more likely to be accepted and acted upon than equivalently performant but opaque scheduling engines, supporting the theoretical claim of Section 2.2 that reward design and policy interpretability jointly determine, rather than merely accompany, the operational efficiency gains RL scheduling delivers in practice [8].

5. CHALLENGES AND PROSPECTS OF ARTIFICIAL INTELLIGENCE EMPOWERING INTELLIGENT ENGINEERING SYSTEMS

5.1 Current Main Challenges

Four challenges are important challenges in current RL-scheduling practice. First of all, sample efficiency is still a problem: in order to drive the scheduling strategy until it runs well, it needs to gain more experience than from the real production line in a short time. Therefore, we must use simulation (discrete event simulation) for training, but there is a risk that the policy may know too much about the details of simulation, which are not in the real machine [7]. Secondly, the generalization varies according to the size and configuration of the problem; Coding based on graphic neural network can generalize the size well in job shop scheduling, but it will not automatically apply to other types of scheduling, and many deployments we have seen are only applicable to production configurations used during training [4, 7]. Thirdly, the nonstandard reward brings continuous risk: because the policy behavior is completely shaped by the constructed reward signal, the ill-defined compound reward may produce planned behavior that meets the literal reward function, and at the same time violate its expected manufacturing goal, and this risk will increase with the increase of the number of goals [5]. Fourthly, compared with the complexity of domain algorithms, deployment security and interpretability are still poor. A systematic review of RL documents in production systems points out that demonstrating safety and reliability under real-world conditions, not just simulation benchmarks, is still the focus of open research in the whole field [7].

5.2 Prospects for Future Development Trends

Three routes look promising. Sim-to-real transfer method mainly learns rules in simulation, and uses skills to narrow the difference between simulation and actual production, which provides a method to solve the efficiency problem of examples without long training in real life. Standardized and size-invariant state representation, using the disjunctive-graph method already shown for job-shop scheduling, may be used for other types of scheduling, extending the size-generalization attribute, which is now only displayed in smaller problems [4]. Finally, explainable RL surrogates, already validated for production control, are likely to mature from post-hoc decision-tree distillation toward natively interpretable policy architectures designed jointly for performance and operator trust from the outset, rather than as an explanatory layer appended after training [8]. Building on the reward-decomposability extension proposed in Section 3.4, we further suggest that future multi-agent scheduling architectures treat reward decomposition explicitly as a design-time verification step—checking, before deployment, whether locally optimal agent behavior under the decomposed reward structure is provably

consistent with the global scheduling objective—rather than discovering misalignment only through observed production-floor behavior after deployment.

6. CONCLUSION

This review has examined reinforcement learning as the principal mechanism through which artificial intelligence operationalizes dynamic scheduling within intelligent, cyber-physical manufacturing systems, grounding the analysis in CPS theory, systems engineering, and information-theoretic reasoning about state representation and reward design. The Markov decision process formalism proves central throughout: it is the mechanism by which sequential, reward-mediated decision-making is rendered both learnable from data and traceable to explicit manufacturing objectives. Across the four transformation paths examined, increasing scheduling sophistication consistently depends on more carefully engineered state representations and reward functions, met by complementary methodological advances including graph-neural-network policy architectures, composite reward shaping, and explainable policy distillation. Applications in fault-aware scheduling, process optimization, and human-machine collaboration demonstrate that RL scheduling's operational value depends as much on the fidelity of its reward specification and the interpretability of its decisions as on raw policy performance. Persistent challenges in sample efficiency, generalization, reward misspecification, and deployment safety suggest that RL-driven scheduling remains an active engineering frontier, but the proposed framing of reward decomposability as a design-time verification criterion offers a concrete path toward scheduling systems whose autonomous behavior can be specified, verified, and trusted before, rather than only after, physical deployment.

REFERENCES

[1] Monostori, L., Kádár, B., Bauernhansl, T., Kondoh, S., Kumara, S., Reinhart, G., Sauer, O., Schuh, G., Sihn, W., & Ueda, K. (2016). Cyber-physical systems in manufacturing. CIRP Annals, 65(2), 621-641. https://doi.org/10.1016/j.cirp.2016.06.005

[2] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., & Hassabis, D. (2015). Human-level control through deep reinforcement learning. Nature, 518(7540), 529-533. https://doi.org/10.1038/nature14236

[3] Waschneck, B., Reichstaller, A., Belzner, L., Altenmüller, T., Bauernhansl, T., Knapp, A., & Kyek, A. (2018). Optimization of global production scheduling with deep reinforcement learning. Procedia CIRP, 72, 1264-1269. https://doi.org/10.1016/j.procir.2018.03.212

[4] Zhang, C., Song, W., Cao, Z., Zhang, J., Tan, P. S., & Xu, C. (2020). Learning to dispatch for job shop scheduling via deep reinforcement learning. Advances in Neural Information Processing Systems, 33, 1621-1632. https://proceedings.neurips.cc/paper/2020/hash/11958dfee29b6709f48a9ba0387a2431-Abstract.html

[5] Zhou, T., Tang, D., Zhu, H., & Wang, L. (2021). Reinforcement learning with composite rewards for production scheduling in a smart factory. IEEE Access, 9, 752-766. https://doi.org/10.1109/ACCESS.2020.3046784

[6] Kuhnle, A., Kaiser, J.-P., Theiß, F., Stricker, N., & Lanza, G. (2021). Designing an adaptive production control system using reinforcement learning. Journal of Intelligent Manufacturing, 32(3), 855-876. https://doi.org/10.1007/s10845-020-01612-y

[7] Panzer, M., & Bender, B. (2022). Deep reinforcement learning in production systems: a systematic literature review. International Journal of Production Research, 60(13), 4316-4341. https://doi.org/10.1080/00207543.2021.1973138

[8] Kuhnle, A., May, M. C., Schäfer, L., & Lanza, G. (2022). Explainable reinforcement learning in production control of job shop manufacturing system. International Journal of Production Research, 60(19), 5812-5834. https://doi.org/10.1080/00207543.2021.1972179