Proc. of the 9th Intl. Conf. of Asian Society for Precision Engg. and Nanotechnology (ASPEN 2022)
15–18 November 2022, Singapore. Edited by Nai Mui Ling Sharon and A. Senthil Kumar
doi://10.3850/978-981-18-6021-8_OR-08-0197

A Case Study: Dynamic Dispatching Framework for Multi-Product Production System via Reinforcement Learning Approach

Jing Zhuang[1,#] and Guan Leong Tnay[1]

1 Singapore Institute of Manufacturing Technology, 2 Fusionopolis Way, #08-04, Innovis, 138634, Singapore
# Corresponding Author / zhuang_jing@simtech.a-star.edu.sg, TEL: +65-96586948

KEYWORDS: Multi-Product Production System, Reinforcement Learning Controller, Dynamic Dispatching, Multi-Objective Optimization

Contemporary manufacturing systems often exhibit characteristics such as multi-product, multi-objective and increasing types and frequencies of uncertainties during production execution. It is non-trivial to consider a novel dispatching strategy adaptable to the ever-changing environment while maintain satisfactory production performance. In this work, we propose an end-to-end framework which consists of: 1) a simulated multi-product production system that can be configured with comprehensive constraints and uncertainties (the Environment), 2) a dispatcher that makes progressive decisions based on the tracked status of the Environment (the Agent), and 3) a messenger that transmits tracked status, dispatcher's decisions and feedback to decisions in-between the Environment and the Agent. The three-component dynamic dispatching framework, enabled by Reinforcement Learning (RL) approach, is among the first to be demonstrated on such a sophisticated dispatching problem, extracted from a real precision engineering production system. We wish the work would be valuable to advance further development and implementation of RL systems for production control applications.

1. Introduction

Throughout the decades of industry revolution till now, we've seen shopfloors from various manufacturing sectors growing into distinctive levels of maturity: Semiconductor undoubtedly leads the way working towards cyber-physical production system, followed by Tier-2 players such as automobile or pharmaceutical, who strives to achieve line-wise automation with little human intervention. Within Tier 2, it is observed that a group of Small and Medium Enterprises (SME) Precision Engineering (PE) companies, in one hand, rely on the automated solutions (i.e. CNC machines) in some of the standard processes like turning and milling, on the other hand, inevitably retain a great amount of manual processes to facilitate auto processes. Key reason is that customer orders for them can be highly unique with tiny quantity in each (thus multi-product production system, or MPPS), therefore many of the processes are impossible to be standardized but require flexible manual assistance to cater for various order specifications. To our biggest concern, main consequence out of the auto-manual hybrid shopfloor is that production plan and schedule often become unnecessary. Instead, shopfloor supervisors need to adjust shopfloor activities in near real-time manner based on their own observation and judgement. It is therefore hard to guarantee that this on-the-fly job assignment is to the best knowledge of the current state of the MPPS, and such a decision is always aligning with the production goal rather than short-sighted.

To address this challenge, one must understand what we need is *dynamic dispatching* solutions. *Dispatching* differs from planning and scheduling in the way that it rests in production execution phase whereby assigning of order to certain resources (process or transport) in near real-time manner is what it concerns [1]. *Dynamic* denotes that shopfloor environment has its complexity involving multiple work centres, multiple machines of different characteristics and multiple products, coupled with technological and logistic constraints [2]. Key performance indicators (KPIs) for such production system are not only job-oriented, but also need to be economic and sustainable in terms of resource utilization [3]. It is generally agreeable that Reinforcement Learning (RL) is now the leading approach to offer dynamic dispatching solutions compared to rule-based heuristic, thanks to the advancement of sensorisation and connectivity technology in place to support data extraction from real production, to facilitate RL agent training and future estimation [1]. Many research efforts have been reported in the area of designing and modelling RL systems for adaptive dispatching in a simulated environment abstracted from semiconductor use case [4][5][6][7]. To the best of authors' knowledge, such recommendation has not been made available for SME PE industry whereby data availability is lower, constraints are more dominant, and customer orders are of higher variance, therefore sensible dispatching decisions are in greater demand to replace empirical decisions.

The main objective of this work is to illustrate how a three-component dynamic dispatching framework establishes the near real-time shopfloor decision-making capability, using a testbed PE manufacturing scenario as an example. Differing from prior arts that focuses on overall design of RL system with a simplified environment, we have engaged serious investigation in environment modelling and data collection / pre-processing, so that the key constraints and stochastic effects of PE shopfloor can be well represented and considered in the learning process.

2. Dynamic Dispatching Framework

The framework (see Fig 1) consists of three intertwining components: First and foremost is the simulated MPPS. It refers to a

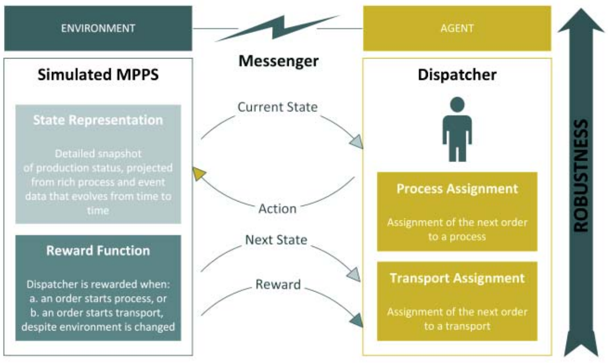
ENVIRONMENT
Simulated MPPS
State Representation
Detailed snapshot of production status, projected from rich process and event data that evolves from time to time
Reward Function
Dispatcher is rewarded when:
a. an order starts process, or
b. an order starts transport,
despite environment is changed
Messenger
Current State
Action
Next State
Reward
AGENT
Dispatcher
Process Assignment
Assignment of the next order to a process
Transport Assignment
Assignment of the next order to a transport
ROBUSTNESS

Fig 1 Three-component dynamic dispatching framework

simulation model built in a professional simulation software - Visual Components. The model replicates the testbed's layout and machine capabilities to the greatest detail. It's easily configurable to feed in process, process flow and product data of the real production, output signal of resources status in real-time. In general, the simulated MPPS enables *State Representation* of the environment. Second component is the dispatcher. Based on state observation at a specific decision point, it either assigns an order to a process, to a transport mean, or does nothing, following the *Policy* which is prescribed in RL algorithm. In short, the dispatcher returns an *Action* of the RL agent. The last component is the messenger, which handles the information transmission in-between the other two components. With the help of the messenger, the Action will trigger the simulated MPPS to move to the next decision point, producing a *Next State* as well as an action evaluation score (namely *Reward)* to feedback to the dispatcher. As this interaction goes on, the dispatcher will improve its action taking strategy (via updating the *policy*) when more and more states have been explored. Theoretically, an optimal policy could be learned by maximizing the total returns received by the dispatcher. To facilitate understanding, the following sub-sections describes the use case and how the proposed framework is applied.

2.1 The Simulated MPPS

The chosen testbed MPPS features sharing of resources by mixed product types as well as by multiple processes, which is usually the case for space-constrained PE shopfloor. Three types of parts are fabricated (namely Part A/B/C) to produce a "set" to be assembled into a "product" at the later stage. The focus of the modelling will be the process flow of Part A/B/C and the maneuvers of materials among production resources as indicated in Fig 2. Basically, all the part types follow the same process flow, except for different choices of machines (MC 1/2 belongs to Work Centre 1, while MC 2/3 belongs to Work Centre 2) depending on part type. The time required for every process step, be it an auto process or manual process, differs in various part types as well. Transportation in-between processes, unless specified in the flow, is via forklift operated by one worker, who is also the only operator (OP 1) for all the manual processes. The production objective is to increase throughput rate for all part types in 1-1-1 ratio, while ensuring machines can be maximally utilized.

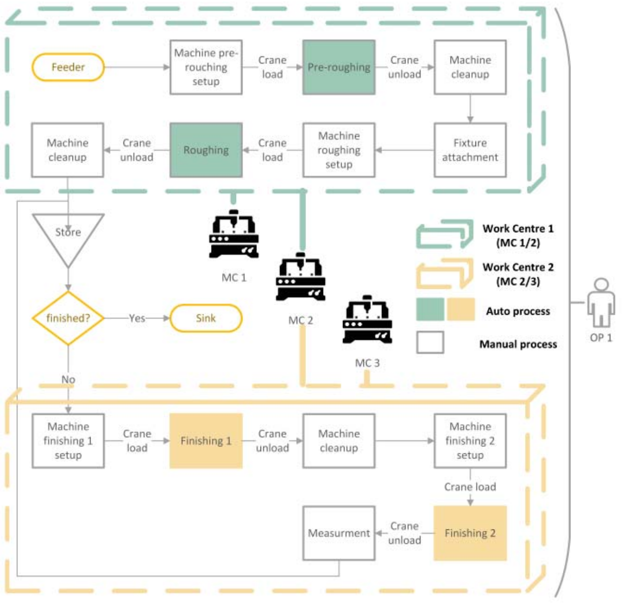

Fig 2 Process flow of the testbed MPPS

2.1.1 Types of constraints & uncertainties

Based on three-month records of production execution via the power meter tracking system, common complexities of testbed MPPS are summarized:

- Manpower constraint

  Manpower is required in all the manual processes and transportation of materials. It has to be concurred that such processes and movements cannot happen concurrently. Observation on MC 2 & 3 has demonstrated that its most frequent "breakdown" is caused by "no manpower". And the main reason to assign only one operator for the testbed is that the nature of the job is not suitable for teamwork – it is the common industrial practice for PE sector that one machinist is favored to take care of a job end-to-end to closely monitor the process and apply contingency in real time. Hence without challenging the rule, we intend to propose dispatching strategy that assigns the operator for various of manual tasks as much as possible when auto process is happening.

- Machine breakdown

For MC 1, the longest "breakdown" is simply "machine breakdown", mainly contributed by "spindle cooler motor" and "pallet changer" malfunctions. Three-month Overall Equipment Effectiveness (OEE) of MC 1 is as low as 41.7%. When it is down, MC 2 is used to work on pre-roughing and roughing operations, but chip removal of MC 2 is much slower compared to MC 1, and more manual adjustment of tools and CNC program is required. To overcome the productivity loss during MC 1's downtime, dispatching strategy should look ahead to trigger a series of required resources (i.e. additional tools, contingency G-code files, backup operator with the knowledge of machine switch etc.) to be standby for the estimated breakdown time.

- Stochastic processing time

Apart from manual processes which are often completely stochastic, auto processes are also never static. The three-month observation illustrates that even within one machining process, multiple stoppages occur for which human intervention is necessary. Examples are tool change, programming, chip removal and dimension verification. The model of testbed does not specify these minor stoppages to maintain its generalizability. However, we aggregate all the contributing factors for each process and turn nominal processing time into stochastic processing time to better represent system dynamics in long run.

2.1.2 State representation

The best form of state representation is a numerical vector, in which each element reflects either job-intrinsic, or job-extrinsic information. Job-intrinsic information could be at any specific decision point for each job, whether it can be assigned, how much time has spent and left over on its current process, how much time has spent and left over for its total completion, how much time for its next required resource to be free, how much idling it has experienced since last process, how much idling it has experienced throughout past processes [8]. Stochastic processing time of job should be reflected in these indicators. Job-extrinsic information is not directly available by tracking just aforementioned attributes. Instead, they are related to system constraints and uncertainties described in Section 2.1.1, i.e. machine breakdown & manpower availability. Machine breakdown could be modelled as stochastic process in the simulated MPPS, and a sampling result could appear into the state representation vector to dictate the temporal status of machine. Manpower availability can be acquired through the combined job specification (i.e. whether or not the operator is occupied by ongoing task) and stochastic sampling (i.e. whether the operator is on or off duty).

2.1.3 Reward function

Reward function should be designed in favor of achieving the production objective. In the case study, a positive reward should be given every time when a job is dispatched successfully to a process or transport mean, especially reward more when a job is completed done; a negative reward should be placed to discourage for "doing nothing" at each decision point. This philosophy is an improved version from the previously proposed reward function in [8]. It results in a better performance as can be seen from Section 2.2.2.

2.2 The Dispatcher

To grasp how our dispatcher works, it is important to understand that in this problem specification, "decision point" means "any timestamp that a process is done, or a transport is done, or a machine breaks down / is repaired, or the manpower becomes available / unavailable". Based on each type of decision point, possible actions could be:

- ***Waiting***, in the event process is done but no transport mean is available to ship the job, or transported to the next station but next process is not possible to start,
- ***Process assignment***, in the event when machine is available (either repaired or freed) and manpower stands by (only if required), plus at least one job is waiting to be processed,
- ***Transport assignment***, in the event when process is done and both transportation equipment and manpower are available.

Action space dynamically update itself based on what type of decision point the dispatcher encounters, so that dispatcher will just choose from legal actions to move the environment state forward.

2.2.1 Reinforcement Learning process

Value-based RL method, represented by Q-learning and Deep Q Network (DQN), has been popularly applied to address manufacturing scheduling and dispatching problems since 2016 onwards. But one key advantage of the policy-based RL method against value-based method for the current study is that it allows a non-arbitrary, initial policy to be adopted at the start of training process. From the testbed or any generic PE manufacturing system's perspective, the as-is action selection criteria implicitly developed and used by shopfloor supervisors based on their empirical knowledge of the system behavior is important. It is therefore an obvious option for us to take advantage of the empirical criteria when initializing policy. This does not only save the training efforts, but also introduces useful prior information to the agent before exploration kicks off.

Our RL implementation is firstly demonstrated in full-Python environment, with the simulation model adapted from an OpenAI Gym environment named JJSEnv [8], and policy-based algorithm Proximal Policy Optimization (PPO) implemented by Stable Baselines3 (SB3) in PyTorch [9]. Specific testing instance chosen from Job Shop Scheduling Problem library is Ta01[1], with 15 jobs to be processed over 15 machines.

2.2.2 Performance analysis

[1]http://optimizizer.com/solution.php?name=ta01&UB=1231&problemclass=ta

Since we are using a standard Job Shop Scheduling instance instead of a complex system as we've described in Section 2.1, the main objective of performance analysis is to illustrate on the feasibility of proposed framework, rather than the training outcome. From the Tensorboard chart depicted in Fig 3, it is observed that both

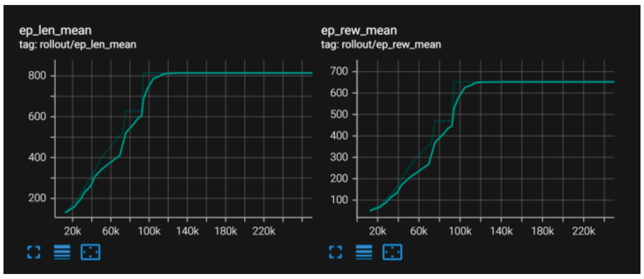

Fig 3 Training progress in Tensorboard

mean episode length (left) and mean episode reward (right) increase along with iterations, meaning the dispatcher learns to schedule more and more jobs to the processes with an enhanced KPI performance. Fig 4 denotes that the best performing dispatcher after training 130k timestamps could successfully schedule 11 out of 15 jobs within the makespan of 5990 timestamps.

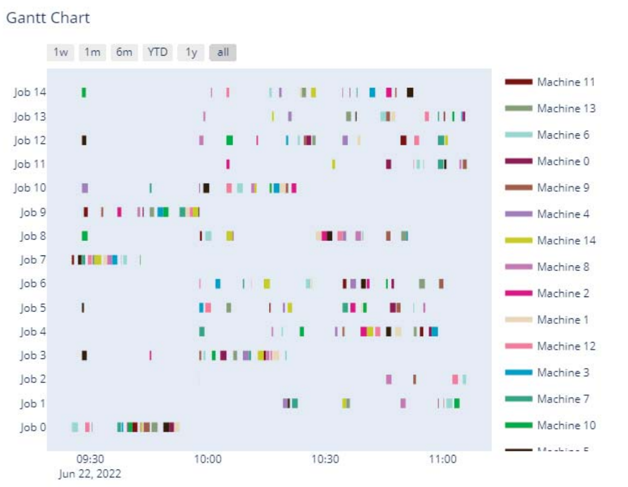

Fig 4 Gantt Chart of 11 scheduled jobs with makespan of 5990

2.3 The Messenger

The anticipated outcome of current study aims for real-world implementation, in which the messenger will be playing an important role. To link up with the latest industrial digitalization efforts, as well as the near real-time cyber-physical interaction requirement enforced by the proposed framework, one would recommend that the messenger to perform its information transmission function via OPC-UA communication protocol[2]. Fortunately, Visual Components software supports the OPC-UA plug-in for us to investigate the performance of the messenger in bridging the simulated MPPS and the dispatcher. This will be the focus of upcoming development.

[2] https://opcfoundation.org/about/opc-technologies/opc-ua/

3. Conclusions and Future Work

In this work, a dynamic dispatching framework (DDF) is proposed with a detailed description of each contributing component. The framework has demonstrated its feasibility in Python environment, to be followed by further adjustment into a sophisticated simulated MPPS. Upon next milestone, a successful use case of DDF would be developed to achieve multi-objective optimization in cyber world. Ultimately, this study would be extended to the real manufacturing environment to examine DDF's real-world production control capability.

ACKNOWLEDGEMENT

This work was supported by the 1st A*STAR Career Development Award Grant (No. 202D800040).

REFERENCES

1. Kuhnle, A., "Adaptive Order Dispatching based on Reinforcement Learning: Application in a Complex Job Shop in the Semiconductor Industry." DOI: 10.5445/IR/1000127740, 2020.
2. Waschneck, B. et al., "Deep reinforcement learning for semiconductor production scheduling," 29th Annual SEMI Advanced Semiconductor Manufacturing Conference (ASMC), pp. 301-306, doi: 10.1109/ASMC.2018.8373191, 2018.
3. Rinciog, A. and Meyer, A., "Towards Standardizing Reinforcement Learning Approaches for Stochastic Production Scheduling," arXiv:2104.08196, 2021.
4. Nicole, S. et al., "Reinforcement learning for adaptive order dispatching in the semiconductor industry," CIRP Annals, Vol. 67, Issue 1, pp. 511-514, 2018.
5. Kuhnle, A. et al., "Design, Implementation and Evaluation of Reinforcement Learning for an Adaptive Order Dispatching in Job Shop Manufacturing Systems," Procedia CIRP, Vol. 81, pp. 234-239, 2019.
6. Kuhnle, A., Röhrig, N. and Lanza, G., "Autonomous order dispatching in the semiconductor industry using reinforcement learning," Procedia CIRP, Vol. 79, pp. 391-396, 2019.
7. Kuhnle, A. et al. "Designing an adaptive production control system using reinforcement learning," J Intell Manuf, Vol. 32, pp. 855–876, 2021.
8. Pierre, T. et al., "A Reinforcement Learning Environment For Job-Shop Scheduling," arXiv:2104.03760, 2021.
9. Antonin, R. et al., "Stable-Baselines3: Reliable Reinforcement Learning Implementations," Journal of Machine Learning Research, Vol. 22, No. 268, pp. 1-8, 2021.