APEM journal
Advances in Production Engineering & Management
Volume 17 | Number 4 | December 2022 | pp 401–412
https://doi.org/10.14743/apem2022.4.444
ISSN 1854-6250
Journal home: apem-journal.org
Original scientific paper

Verification of intelligent scheduling based on deep reinforcement learning for distributed workshops via discrete event simulation

Yang, S.L.[a], **Wang, J.Y.**[b,c,*], **Xin, L.M.**[d], **Xu, Z.G.**[b,c,*]

[a]School of Mechanical Engineering, University of Shanghai for Science and Technology, Shanghai, P.R. China
[b]Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang, P.R. China
[c]Institutes for Robotics and Intelligent Manufacturing, Chinese Academy of Sciences, Shenyang, P.R. China
[d]School of Computer Engineering and Science, Shanghai University, Shanghai, P.R. China

ABSTRACT

Production scheduling, which directly influences the completion time and throughput of workshops, has received extensive research. However, due to the high cost of real-world production verification, most literature did not verify the optimized scheduling scheme in real-world workshops. This paper studied the verification of scheduling schemes and environments, using a discrete event simulation (DES) platform. The aim of this study is to provide an efficient way to verify the correctness of scheduling environments established by programming languages and scheduling results obtained by intelligent algorithms. The system architecture of scheduling verification based on DES is established. The modelling approach via DES is proposed by designing parametric workshop generation, flexible production control, and real-time data processing. The popular distributed permutation flowshop scheduling problem is selected as a case study, where the optimal scheduling scheme obtained by a deep reinforcement learning algorithm is fed into the production simulation model in Plant Simulation software. The experiment results show that the proposed scheduling verification approach can validate the scheduling scheme and environment effectively. The utilization and Gantt charts clearly show the performance of scheduling schemes. This work can help to verify the scheduling schemes and programmed scheduling environment efficiently without costly real-world validation.

ARTICLE INFO

Keywords:
Production scheduling;
Distributed flowshop scheduling;
Discrete event simulation (DES);
Deep reinforcement learning;
Production simulation;
Modelling;
Scheduling verification;
Plant Simulation software

**Corresponding author:*
jywang@sia.cn
(Wang, J.Y.)
zgxu@sia.cn
(Xu, Z.G.)

Article history:
Received 14 November 2022
Revised 12 December 2022
Accepted 15 December 2022

1. Introduction

Production scheduling is an important problem for real-world manufacturers. Effective scheduling schemes can help to reduce production costs, decrease completion time, and increase throughput. For several decades, production scheduling has received extensive research interest. For example, only for the permutation flowshop scheduling problem, more than 100 kinds of heuristic and meta-heuristic algorithms have been proposed according to the reviews of Fernandez-Viagas *et al*. [1].

Under the globalized economy, companies tend to establish production bases in different regions to fulfill the production requirement of customers distributed in different areas. Under distributed manufacturing, the distributed permutation flowshop scheduling problem (DPFSP) was proposed by Naderi, Ruiz [2]. In the DPFSP, a set of factories are distributed in different regions. A set of jobs can be processed in one of the factories. The optimization problem is to minimize the total completion time of all jobs by properly assigning jobs to factories and scheduling jobs in each factory. The DPFSP has been solved by several kinds of intelligent algorithms, such as iterated greedy algorithm [3], artificial bee colony algorithm [4], cooperative memetic algorithm [5], etc. Some realistic characteristics have been considered in the DPFSP, such as sequence-dependent setup times[6, 7], blocking constraint [8, 9], no-wait constraint [10], no-idle constraint [11], hybrid flowshop [12], batch delivery [13], etc.

Most product scheduling is optimized by intelligent algorithms. Usually, intelligent algorithms are written in programming languages, such as C++, Java, MATLAB, Python, etc. At the same time, the production environment for simulating and optimizing the scheduling scheme is also programmed using programming languages. Although the scheduling scheme can be optimized through the programmed scheduling simulation environment, the correctness of the scheduling simulation environment and scheduling scheme remains a problem. Since the verification for an optimized scheduling scheme is crucial for implementation in real-world factories, the verification for the simulation environment and scheduling schemes should be performed.

In current literature, only very few studies applied proposed scheduling models and algorithms to real production workshops, such as applied to the manufacturing execution system (MES) of a real factory. A few studies only used real-world data as production instances. However, most literature did not apply the scheduling algorithms and methods to real-world production cases. For the scheduling application in real-world production, Zhou *et al*. [14] studied the multi-objective flexible job shop scheduling problem using multi-agent-based hyper-heuristics and applied the proposed methods and algorithms to an aero-engine blade manufacturing plant. For the usage of real-world data, Jiang *et al*. [15] studied a real case scheduling problem for aerospace industry components in a flexible job shop. Li *et al*. [16] solved the welding shop scheduling problem using a discrete artificial bee colony algorithm and applied the algorithm in a real-world girder welding shop. Yankai *et al*. [17] studied the hybrid flowshop scheduling problem, in which numerical experiments are carried out based on real-world cases in a hot-rolling workshop. Wang *et al*. [18] studied the energy-aware welding shop scheduling problem using a decomposition-based multi-objective evolutionary algorithm and applied the proposed algorithm to a real-world case. Schumacher, Buchholz [19] studied the hybrid flow shop scheduling problem under uncertainty and applied the proposed methods to a real-world production case. Ojstersek *et al*. [20] studied the multi-objective scheduling problem of flexible job shops using a real-world manufacturing dataset.

Although some literature used real-world data for problem-solving, most scheduling literature did not verify the correctness of scheduling simulation and scheduling results. The reasons may be as follows. Firstly, the implementation of production scheduling in a real-world workshop costs too much with a lot of production resources required. In addition, some researchers do not have suitable industrial cooperation for validation or lack a production line in the laboratory for proper production validation. With the development of simulation software, discrete event simulation (DES) servers as an efficient tool for production optimization and validation.

The Plant Simulation software is one of the widely used production simulation platforms. Some researchers used the Plant Simulation to optimize production processes and validate the production scenarios. Yang *et al*. [21] proposed a modelling method for workshop modelling in Plant Simulation and optimized the production configurations of an assembly shop. Xu *et al*. [22] studied the optimization problem of multi-stage production scheduling. Wang *et al*. [23] studied the reliability allocation method for a production system using Plant Simulation. Pekarcikova *et al*. [24] studied the simulation testing of the E-Kanban to improve the efficiency of logistics processes. Yang *et al*. [25] optimized the assembly transport optimization problem in a reconfigurable flow shop. Pekarcikova *et al*. [26] studied the bottleneck problem in the logistics flow of a manufacturing company using Plant Simulation software. Li *et al*. [27] dealt with the bottleneck

identification and alleviation in a blocked serial production line via Plant Simulation. Jurczyk-Bunkowska [28] studied the tactical manufacturing capacity planning of a medium-sized production enterprise by Plant Simulation. Gregor *et al*. [29] verified the routes of an automated guided vehicle using Plant Simulation. Li *et al*. [30] studied resource allocation in a production logistics system using Plant Simulation software. Gola *et al*. [31] used the Plant Simulation software to identify the bottlenecks in a reconfigurable manufacturing system.

In addition to optimizing and verifying the production process, Plant Simulation software has also been used to optimize some scheduling problems by using the built-in intelligent algorithm packages. Xu *et al*. [22] optimized a multi-stage production scheduling problem of an automated production system by using Plant Simulation software. Istokovic *et al*. [32] determined the order and size of production batches in a flow shop using the genetic algorithm optimization tool of Plant Simulation. Ferro *et al*. [33] used Plant Simulation to optimize the production planning of the textile industry.

From the above literature review, we can know that the verification of the scheduling scheme has not received adequate research, and Plant Simulation software has been used to optimize and verify some production processes. Verifying the scheduling scheme and programmed environment via the Plant Simulation platform is an efficient way. However, few studies investigated the verification of scheduling schemes and programmed production environment with the help of Plant Simulation.

This paper studied the verification of scheduling schemes and environments based on a discrete event simulation platform—Plant Simulation software. The distributed permutation flowshop scheduling problem is verified in Plant Simulation. The aim of this study is to provide efficient and costless verifications for the programmed scheduling environment and obtained scheduling schemes, before applying those scheduling schemes in real-world workshops. The overall system architecture of verifying scheduling schemes via Plant Simulation platform is established. The distributed permutation flowshops are established in Plant Simulation platform. The simulation results in Plant Simulation validate the correctness of the programmed scheduling environment programmed in Python and the scheduling results obtained by algorithms.

Particularly, the main contributions of this paper are as follows:

- The system architecture of scheduling verification based on scheduling optimization and Plant Simulation is established. The scheduling scheme is optimized by intelligent algorithms programmed in Python language. Then, the optimized scheduling scheme is simulated and verified in the established production model in Plant Simulation platform.
- The modelling approach for production workshops via Plant Simulation platform is proposed. The production simulation model is established by designing parametric modelling of workshops, flexible production control, and real-time data processing.
- The whole verification process is validated by a case study on the distributed permutation flowshop scheduling problem. The videos of scheduling optimization and production simulation are provided.

The rest of the paper is listed as follows. Section 2 illustrates the system architecture of the scheduling verification approach. Section 3 establishes the production workshops using Plant Simulation platform. Section 4 verifies the scheduling environment and scheme using a case study. Section 5 concludes the paper and provides suggestions for future research.

2. System architecture of the scheduling verification approach

The scheduling validation approach contains intelligent scheduling optimization via intelligent algorithms and production simulation via Plant Simulation. The intelligent scheduling optimization generates the scheduling scheme and results. Based on the scheduling scheme, the production model built in Plant Simulation platform executes the production process. The results obtained from Plant Simulation and those obtained from the intelligent scheduling algorithms are compared to verify whether the programmed scheduling environment and scheduling algorithms are correct. The system architecture of the scheduling verification approach is shown in Fig. 1.

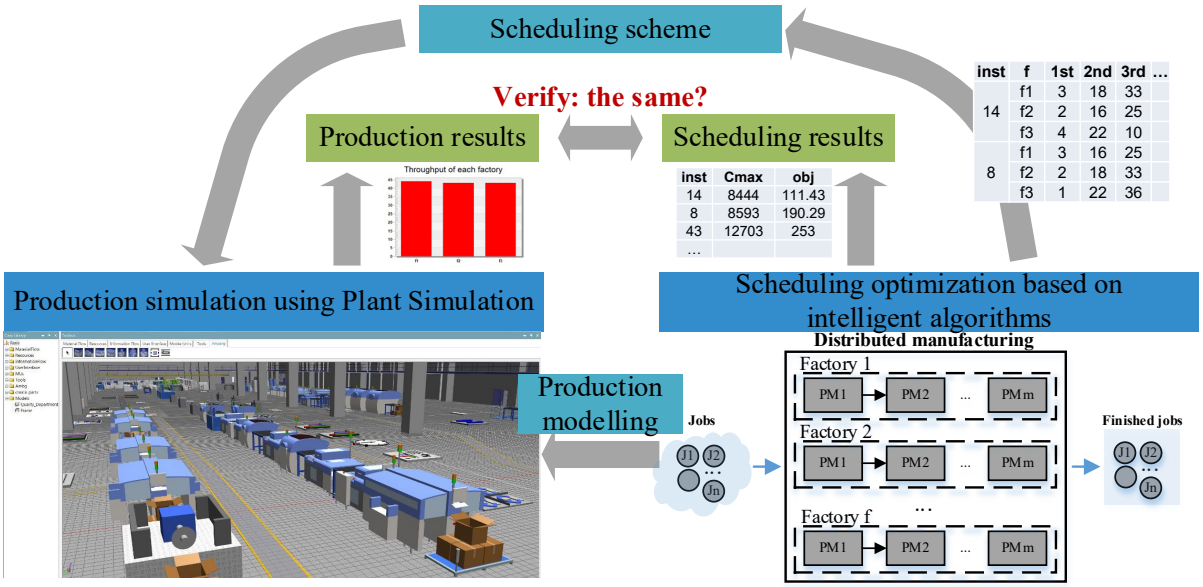

Fig. 1 The system architecture of the proposed scheduling validation approach based on intelligent scheduling optimization and Plant Simulation

2.1 Scheduling optimization based on intelligent algorithms

Most production scheduling in the literature is optimized using intelligent algorithms. Before optimizing the scheduling scheme, the scheduling simulation environment should be programmed. The scheduling simulation environment can process jobs and calculate the objective of candidate schedule schemes. Some popular languages for programming the environment are C++, C#, MATLAB, Java, Python, etc. However, the real-time production interface reflecting the real-time production process is seldom programmed due to the coding complexity. Thus, it is difficult to check the correctness of the production simulation process for the programmed production environment.

The optimal scheduling scheme is obtained by performing the optimization process using intelligent algorithms. Due to extensive research, many kinds of algorithms have been proposed to solve the scheduling problems, such as heuristics, meta-heuristics [3], deep reinforcement learning algorithms [34], etc.

After the optimization process, the optimal scheduling schemes, including the job sequence for each factory, and the beginning and end processing time, are obtained. Besides, the scheduling results, including the completion time of all factories and the objective value for the scheduling scheme, are returned.

2.2 Production modelling and simulation execution

The production simulation model is established in Plant Simulation platform according to the production resources and processes of the studied scheduling problem. Plant Simulation software can build the production model efficiently by constructing machines, flow lines, jobs, buffers, and source jobs. The production control is programmed by SimTalk language in Plant Simulation platform. The scheduling scheme obtained by intelligent algorithms is fed to the production model in Plant Simulation to get the correct scheduling results.

2.3 Verification by comparing production and scheduling results

After the exact execution of the job sequences in each factory, the production results for the selected scheduling scheme are obtained from Plant Simulation platform. Then, the production results are compared with the scheduling results obtained from the intelligent algorithms. For each production scheme, the maximum completion time $C_{\max}$ and objective value *obj* are compared between the production and scheduling results.

3. Modelling of production workshops via Plant Simulation

The production model can be established efficiently via Plant Simulation platform, which contains powerful production modelling tools. The main components of a production simulation model are the production resources, production control methods, production data, and visualization tools, as shown in Fig. 2. The production resources contain machines, buffers, production lines, etc. The production control is realized by control methods. The production data contains some necessary production and real-time data, such as processing times, arrival time, due date, etc. The visualization tools provide various charts for the real-time production status and workshop performance.

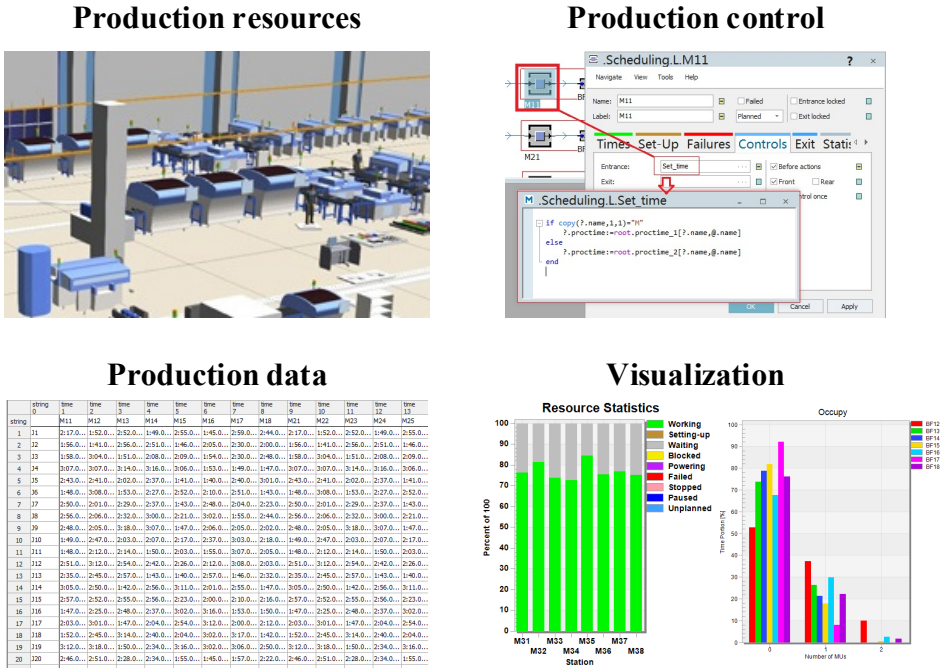

Fig. 2 The main compensates of a production model in Plant Simulation platform

3.1 Parametric modelling of workshops

Workshops are established through parametric modelling. Each workshop contains job storage, buffer, machines, and transfer line between machines. To increase the efficiency of modelling, all of the above production resources are generated by the control method *Start* as shown in Fig. 3. The code for generating machines and buffers is shown in algorithm 1. The production process is modelled exactly as those used in the scheduling algorithms. Thus, the unlimited buffer size is adopted between machines, the transfer time is not considered.

Algorithm 1. Procedure of parametric modelling for generating a workshop using the SimTalk language

```
for j:=1 to n
    for i:=1 to m loop
        if i>1 then
            Name:=Sprint("BF",j,i)
            Obj2:=.Materialflow.Buffer.createObject(current, 40+i*100, 100+(26+60*n)/n*(j+1), Name)
            Obj2.ZoomX:=0.5, Obj2.ZoomY:=0.5, Obj2.Proctime:=0, Obj2.Capacity:=-1
        .Materialflow.Connector.connect(Obj, Obj2), Obj:=Obj2
        end
        Name:=Sprint("M",j,i)
        Obj2:=.Materialflow.Singleproc.createObject(current, 80+i*100,100+(26+60*n)/np*(j+1), Name)
        Obj2.Label := sprint("M",j,i), OBj2.EntranceCtrl:="Set_time"
        obj2.entrancectrlbeforeactions:=true
        if i>1 then
            .Materialflow.Connector.connect(Obj, Obj2)
        end
        Obj:=Obj2
    next
next
```

3.2 Modelling of production control

The production control ensures that workshops produce jobs according to the correct production processes and job sequences. Jobs are released from the source to the workshop when the arrival time is reached. The processing time is set for each machine when a new job arrived at this machine. When a job is finished in a machine, the job is transferred to the buffer of the next machine, and the next job is transferred from the previous buffer. When a job is finished in all machines of a workshop, the job is transferred to the *Drain*.

3.3 Modelling of production data

In addition to the production process, the production data is also fundamental to correctly simulate the production schedule. As shown in Fig. 3, the arrival time and due date of jobs are stored in the table file *Attri*. Jobs are created by the *jobs_create* method based on the arrival time of jobs. The table file *PT* stores the processing times of jobs on machines. The processing time is extracted by the method *Set_time* and set for the corresponding machines.

3.4 Statistical analysis and visualization

One of the advantages of the Plant Simulation platform is its powerful statistical and visualization function. As shown in Fig. 3, in the production model, the variables $C_{\max}$ and *obj* denote the maximum completion time of workshops and the objective value of the current scheduling scheme, respectively. The Gantt chart shows the job sequence and processing durations on machines. The utilization chart reflects the production performance under the current scheduling scheme and production processes. In addition, the occupation of a buffer is provided to show how many jobs exist in buffers before a machine. Since buffer capability is supposed to be infinite, the exact number of existing buffer jobs can help to set buffer sizes in a real production environment.

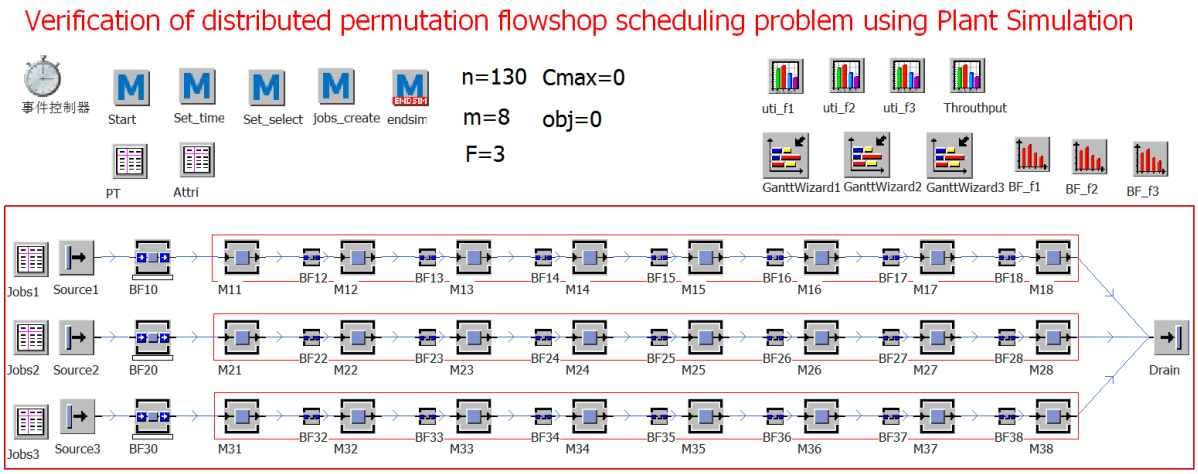

Fig. 3 The production simulation model for the distributed permutation flowshop scheduling problem

4. Verification experiments: A case study

This section verifies the scheduling results obtained by the trained deep reinforcement learning (DRL) algorithm in Plant Simulation software using the modelling and validation approach proposed above. The scheduling results, including the scheduling scheme, total completion time, and objective value, are generated by a DRL algorithm—advanced actor-critic (A2C). Then, the scheduling scheme is executed exactly in the production model in Plant Simulation platform. The production simulation results verified that the scheduling results obtained by the programming environment are correct. In addition, the statistical analysis and visualization were provided via Plant Simulation platform.

4.1 Scheduling scheme obtained by deep reinforcement learning

The distributed permutation flowshop scheduling problem, which has attracted increasing research interest in recent several years, is selected for verification. A production case with 130 jobs, 8 machines, and 3 factories is selected as the case instance by referring to the production characteristics shown in the literature [3]. The scheduling scheme for this production instance is generated by a deep reinforcement learning algorithm A2C. After the dynamic scheduling, the scheduling results of the DRL version are obtained.

Table 1 provides the production data of the selected production instance. Since the production instance has as many as 130 jobs, Table 1 only provides the data of the first 10 jobs. As shown in Table 1, for each job j, the arrival time AT_j, due date d_j, unit tardiness cost α_j, and processing times on all of the 8 machines $p_{M1,j}$-$p_{M8,j}$ are provided.

Table 1 Production data of the studied production instance (only the data of the first 10 jobs are provided)

j	AT_j	d_j	α_j	$p_{M1,j}$	$p_{M2,j}$	$p_{M3,j}$	$p_{M4,j}$	$p_{M5,j}$	$p_{M6,j}$	$p_{M7,j}$	$p_{M8,j}$
1	0.00	2676.00	0.6	137	112	172	109	175	105	179	164
2	0.00	1850.00	1.2	116	101	176	171	106	125	150	120
3	0.00	2711.00	1.3	118	184	111	128	129	114	150	168
4	0.00	2520.00	0.9	187	187	194	196	186	113	109	107
5	5.40	4383.40	1.0	163	161	122	157	101	100	160	181
6	18.14	4528.14	1.2	108	188	113	147	172	130	171	103
7	18.14	2593.14	0.6	170	121	149	157	103	168	124	143
8	21.74	1780.74	1.6	176	126	152	180	141	182	115	164
9	23.33	1767.33	0.1	168	125	198	187	107	126	125	122
10	24.29	4436.29	1.7	109	167	123	127	137	157	183	138

The scheduling scheme and results are obtained using a deep reinforcement learning algorithm by inputting the production data. The scheduling optimization interface in the Python programming environment for the studied case is shown in Fig. 4.

The video of using deep reinforcement learning to solve the studied scheduling problem is provided online. The specific solving procedures are as follows.

- Build the production environment and variable matrix in the Python environment;
- Load the production data of the studied scheduling problem;
- Load the trained model of the A2C algorithm;
- Use the loaded A2C model to make scheduling decisions at each rescheduling point;
- Record the processed jobs sequences of each factory, and calculate the objective value;
- If all jobs are finished in the system, export the total completion time of the system, the objective value of the scheduling plan, and the job sequence of each factory.

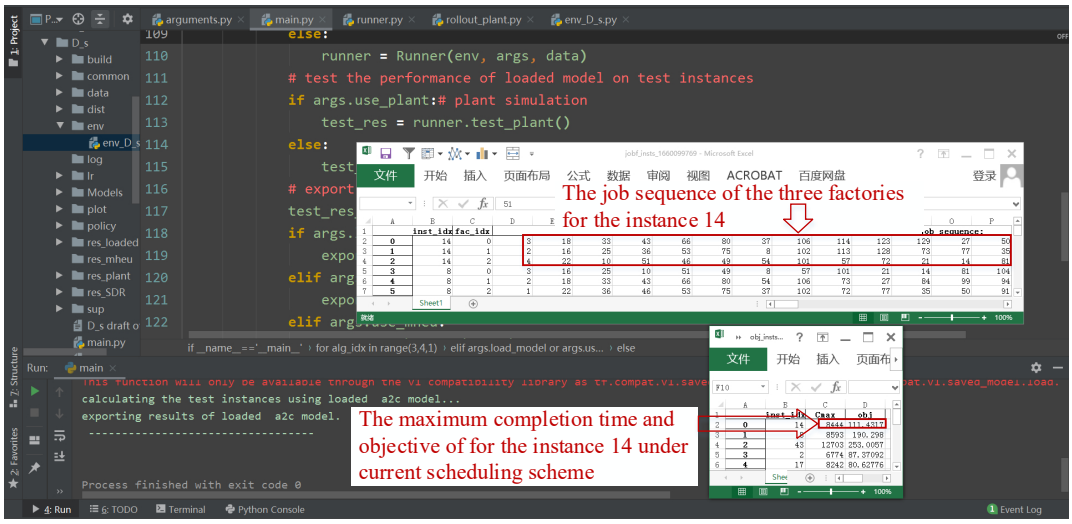

Fig. 4 The scheduling optimization interface in the Python programming environment.
(The video of obtaining the optimal scheduling scheme by loading the DRL model is available at *https://osf.io/qxze5?view_only=e10ee0e3cec44b85ba2f569cd26071f2*)

Table 2 The scheduling results obtained from the DRL algorithm (only the job sequence of the first 10 jobs are presented)

Factory	Completion time	Objective value	Job sequence									
1			3	18	33	43	66	80	37	106	114	123
2	8444	111.43	2	16	25	36	53	75	8	102	113	128
3			4	22	10	51	46	49	54	101	57	72

After the optimization, the scheduling results are provided in Table 2. As shown in Table 2, the total completion time of the system is 8444, the objective value is 111.43. Besides, the job sequence of the three factories is provided. For factory 1, job j3, j18, j33,..., is processed successively. Only the first ten processed jobs of each factory are presented in Table 2.

4.2 Verification using Plant Simulation

The scheduling scheme resulting from the deep reinforcement learning is executed exactly in the production simulation model built in Plant Simulation platform. After production simulation, the scheduling results obtained from Plant Simulation are compared with the results obtained from the Python environment.

Fig. 5 shows the production simulation results in Plant Simulation platform. As shown in Fig. 5, the number of jobs that enter and exit the system is 130, indicating all jobs are processed in the system successfully. The completion time of the system is 8444, and the objective of the scheduling plan is 111.43. The scheduling results obtained in the Plant Simulation platform are the same as those in the Python environment. This verifies the correctness of the simulation environment programmed in Python and the correctness of the scheduling scheme calculation. In addition, Fig. 6-8 show the Gantt chart of the three factories. From Figs. 6-8, we can see that the jobs are processed closely in each factory.

Since production validations in a real production environment cost too much, most scheduling literature did not validate the scheduling results. However, the correctness of the programmed scheduling environment should be checked before performing further scheduling optimizations, and the correctness of the scheduling scheme should be verified before applying the scheduling scheme in real workshops. Our proposed scheduling verification approach provides the necessary support for verifying the scheduling environment and plans. Besides, more statistical analysis and clear visualization can be performed via Plant Simulation platform.

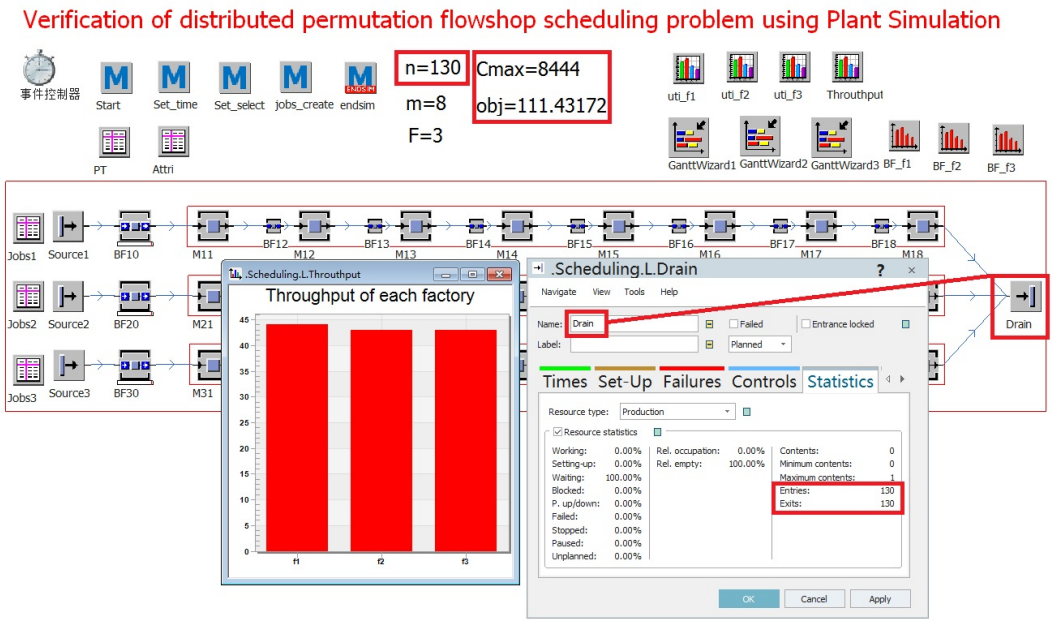

Fig. 5 Production simulation results in Plant Simulation platform (The video of production simulation via Plant Simulation platform is available at *https://osf.io/dzc3u?view_only=e10ee0e3cec44b85ba2f569cd26071f2*)

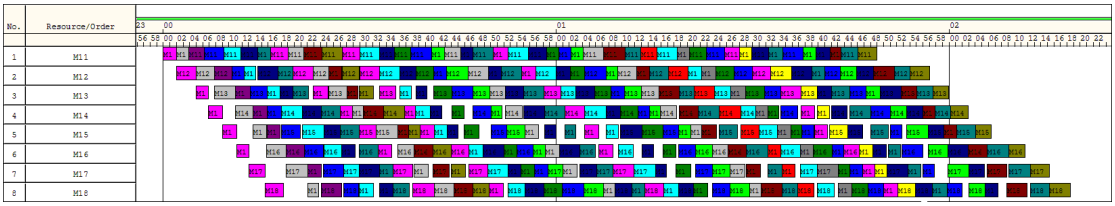

Fig. 6 The Gantt chart of factory 1 for the studied scheduling problem

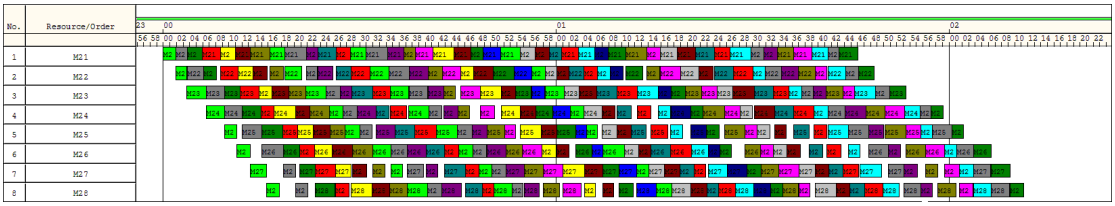

Fig. 7 The Gantt chart of factory 2 for the studied scheduling problem

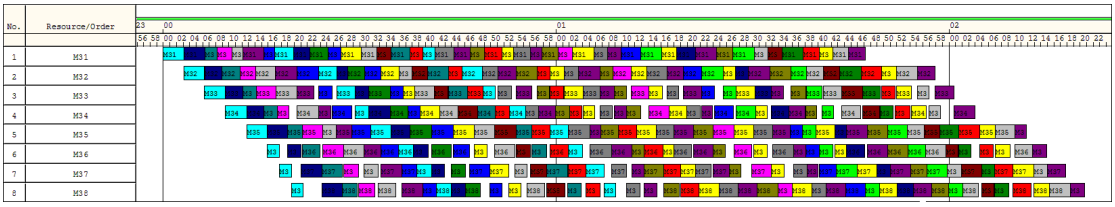

Fig. 8 The Gantt chart of factory 3 for the studied scheduling problem

4.3 Statistical analysis and visualization

The Plant Simulation platform can provide powerful statistical analysis and visualization. In addition to the completion time and objective value, some production indicators are also important for assessing the efficiency and production status of the system. The utilization of machines and the occupation of buffers are analyzed.

Fig. 9 shows the real-time utilization of all machines for the three factories. As shown in Fig. 9, under the optimized scheduling plan, the utilization of machines is approximately 75 %, which is relatively high. Besides, the utilization between machines differs little, indicating that the workloads between those workshops and machines are balanced.

Fig. 10 provides the occupation of buffers in the three factories. Since infinite buffer capability is adapted as used in the scheduling literature, the number of jobs in a buffer can be 0-*inf*. As shown in Fig. 10, the most portion occurs when the number of jobs is 0. This indicates that for most cases, approximately 70-90 %, no jobs exist in the buffers. The number of jobs in all buffers can be 0,1,2,3, and 4. The second most frequent portion occurs when the number of jobs is 1. For factories 1 and 2, only a 5-10 % portion occurs when the number of jobs in buffers is 2. Factory 3 requires more buffers in BF32 and BF36. From the above analysis, we can know that for each buffer the buffer size set to 2 can fulfil most production requirements.

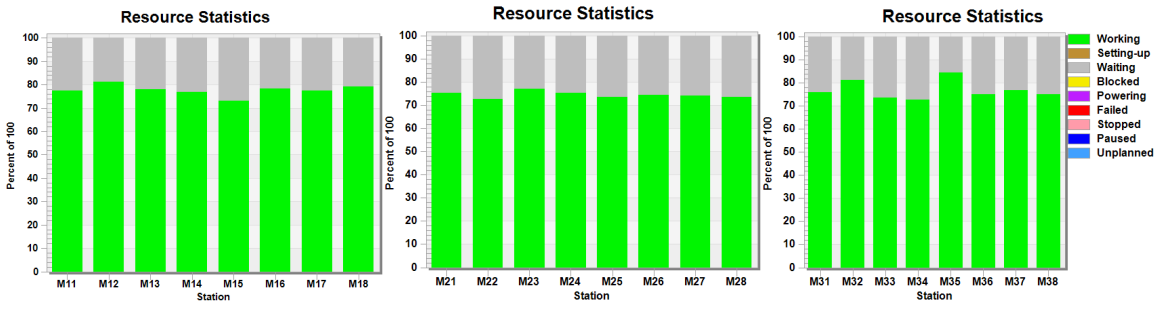

Fig. 9 The utilization of machines in the three factories

The analysis for buffer occupation is very necessary for efficient production, since if no buffers are set the blocking will occur during production and therefore increase the completion time of the system. Most traditional scheduling literature simply assumes that infinite buffers exist between machines. However, in reality, it is impossible to set an infinite buffer size because the area between machines is limited. The buffer occupation analysis can help to determine the best buffer size to support efficient production.

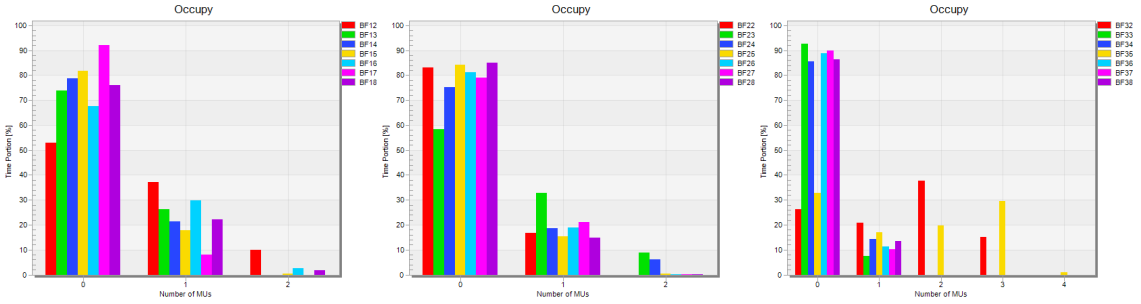

Fig. 10 The occupation of buffers in the three factories

5. Conclusion

This paper studied the verification of production scheduling via discrete event simulation. Since most scheduling literature did not consider the verification of programmed scheduling environment and optimized scheduling schemes, this paper proposed a verification approach to validate the correctness of programmed scheduling environment and intelligent algorithms, by establishing the production simulation model in a Plant Simulation platform. The system architecture for validating the programmed production environment and optimized scheduling schemes are proposed. The parametric modelling method for production workshops with correct production processes is proposed. The verification experiment is carried out by taking the distributed permutation flowshop scheduling problem as a case study. Experimental results show that the scheduling results obtained by a deep reinforcement learning algorithm programmed in Python language are the same as results obtained in Plant Simulation platform. This verifies the correctness of the programmed production environment and scheduling algorithms. Besides, the utilization and Gantt charts clearly show the production efficiency under the selected scheduling scheme. The occupancy of buffers helps to determine the best buffer size before each machine. The proposed scheduling verification approach can help to verify the production environments programmed by researchers and the scheduling results obtained by intelligent algorithms.

In the future, more realistic production resources, such as automated guided vehicles, production lines, and workers, can be considered with the help of Plant Simulation platform after the verification stage to further optimize the scheduling scheme obtained by programmed scheduling algorithms. In addition, the real-time interaction between Python and Plant Simulation can be studied to check the problems of the programmed scheduling environment when the programmed environment is not correct.

Acknowledgement

This work was supported by the National Defense Basic Scientific Research Program of China (Grant No. JCKY2021208B003), the National Key Research and Development Program of China (Grant No. 2022YFB3306000), and the National Natural Science Foundation of China (Grant No. 62073211).

References

[1] Fernandez-Viagas, V., Ruiz, R., Framinan, J.M. (2017). A new vision of approximate methods for the permutation flowshop to minimise makespan: State-of-the-art and computational evaluation, *European Journal of Operational Research,* Vol. 257, No. 3, 707-721, doi: 10.1016/j.ejor.2016.09.055.

[2] Naderi, B., Ruiz, R. (2010). The distributed permutation flowshop scheduling problem, *Computers & Operations Research,* Vol. 37, No. 4, 754-768, doi: 10.1016/j.cor.2009.06.019.

[3] Ruiz, R., Pan, Q.-K., Naderi, B. (2019). Iterated Greedy methods for the distributed permutation flowshop scheduling problem, *Omega,* Vol. 83, No. 213-222, doi: 10.1016/j.omega.2018.03.004.

[4] Tao, X.-R., Pan, Q.-K., Gao, L. (2022). An efficient self-adaptive artificial bee colony algorithm for the distributed resource-constrained hybrid flowshop problem, *Computers & Industrial Engineering,* Vol. 169, Article No. 108200, doi: 10.1016/j.cie.2022.108200.

[5] Wang, J.-J., Wang, L. (2022). A cooperative memetic algorithm with learning-based agent for energy-aware distributed hybrid flow-shop scheduling, *IEEE Transactions on Evolutionary Computation,* Vol. 26, No. 3, 461-475, doi: 10.1109/tevc.2021.3106168.

[6] Huang, J.-P., Pan, Q.-K., Miao, Z.-H., Gao, L. (2021). Effective constructive heuristics and discrete bee colony optimization for distributed flowshop with setup times, *Engineering Applications of Artificial Intelligence,* Vol. 97, Article No. 104016, doi: 10.1016/j.engappai.2020.104016.

[7] Karabulut, K., Öztop, H., Kizilay, D., Tasgetiren, M.F., Kandiller, L. (2022). An evolution strategy approach for the distributed permutation flowshop scheduling problem with sequence-dependent setup times, *Computers & Operations Research,* Vol. 142, Article No. 105733, doi: 10.1016/j.cor.2022.105733.

[8] Shao, Z., Pi, D., Shao, W. (2020). Hybrid enhanced discrete fruit fly optimization algorithm for scheduling blocking flow-shop in distributed environment, *Expert Systems with Applications,* Vol. 145, Article No. 113147, doi: 10.1016/j.eswa.2019.113147.

[9] Chen, S., Pan, Q.-K., Gao, L. (2021). Production scheduling for blocking flowshop in distributed environment using effective heuristics and iterated greedy algorithm, *Robotics and Computer -- Integrated Manufacturing,* Vol. 71, Article No. 102155, doi: 10.1016/j.rcim.2021.102155.

[10] Li, H., Li, X., Gao, L. (2021). A discrete artificial bee colony algorithm for the distributed heterogeneous no-wait flowshop scheduling problem, *Applied Soft Computing,* Vol. 100, Article No. 106946, doi: 10.1016/j.asoc.2020.106946.

[11] Rossi, F.L., Nagano, M.S. (2021). Heuristics and iterated greedy algorithms for the distributed mixed no-idle flowshop with sequence-dependent setup times, *Computers & Industrial Engineering,* Vol. 157, Article No. 107337, doi: 10.1016/j.cie.2021.107337.

[12] Li, J.-Q., Chen, X.-L., Duan, P.-Y., Mou, J.-H. (2022). KMOEA: A knowledge-based multiobjective algorithm for distributed hybrid flow shop in a prefabricated system, *IEEE Transactions on Industrial Informatics,* Vol. 18, No. 8, 5318-5329, doi: 10.1109/tii.2021.3128405.

[13] Yang, S., Xu, Z. (2020). The distributed assembly permutation flowshop scheduling problem with flexible assembly and batch delivery, *International Journal of Production Research,* Vol. 59, No. 13, 4053-4071, doi: 10.1080/00207543.2020.1757174.

[14] Zhou, Y., Yang, J.-J., Zheng, L.-Y. (2019). Multi-agent based hyper-heuristics for multi-objective flexible job shop scheduling: A case study in an aero-engine blade manufacturing plant, *IEEE Access,* Vol. 7, 21147-21176, doi: 10.1109/access.2019.2897603.

[15] Jiang, X., Tian, Z., Liu, W., Suo, Y., Chen, K., Xu, X., Li, Z. (2022). Energy-efficient scheduling of flexible job shops with complex processes: A case study for the aerospace industry complex components in China, *Journal of Industrial Information Integration,* Vol. 27, Article No. 100293, doi: 10.1016/j.jii.2021.100293.

[16] Li, X., Xiao, S., Wang, C., Yi, J. (2019). Mathematical modeling and a discrete artificial bee colony algorithm for the welding shop scheduling problem, *Memetic Computing,* Vol. 11, No. 4, 371-389, doi: 10.1007/s12293-019-00283-4.

[17] Wang, Y., Wang, S., Li, D., Shen, C., Yang, B. (2021). An improved multi-objective whale optimization algorithm for the hybrid flow shop scheduling problem considering device dynamic reconfiguration processes, *Expert Systems with Applications,* Vol. 174, Article No. 114793, doi: 10.1016/j.eswa.2021.114793.

[18] Wang, L., Wang, J.-J., Jiang, E. (2021). Decomposition based multiobjective evolutionary algorithm with adaptive resource allocation for energy-aware welding shop scheduling problem, *Computers & Industrial Engineering,* Vol. 162, Article No. 107778, doi: 10.1016/j.cie.2021.107778.

[19] Schumacher, C., Buchholz, P. (2020). Scheduling algorithms for a hybrid flow shop under uncertainty, *Algorithms,* Vol. 13, No. 11, Article No. 277, doi: 10.3390/a13110277.

[20] Ojstersek, R., Tang, M., Buchmeister, B. (2020). Due date optimization in multi-objective scheduling of flexible job shop production, *Advances in Production Engineering & Management,* Vol. 15, No. 4, 481-492, doi: 10.14743/apem2020.4.380.

[21] Yang, S.L., Xu, Z.G., Wang, J.Y. (2019). Modelling and production configuration optimization for an assembly shop, *International Journal of Simulation Modelling,* Vol. 18, No. 2, 366-377, doi: 10.2507/ijsimm18(2)co10.

[22] Xu, N., Hou, X.Y., Jia, N. (2022). Optimization of multi-stage production scheduling of automated production, *International Journal of Simulation Modelling,* Vol. 21, No. 1, 160-171, doi: 10.2507/ijsimm21-1-co3.

[23] Wang, Y., Xu, B., Ma, T., Wang, Z. (2020). Research on the reliability allocation method for a production system based on availability, *Mathematical Problems in Engineering,* Vol. 2020, Article ID 6159462, doi: 10.1155/2020/6159462.

[24] Pekarcikova, M., Trebuna, P., Kliment, M., Mizerak, M., Kral, S. (2021). Simulation testing of the e-Kanban to increase the efficiency of logistics processes, *International Journal of Simulation Modelling,* Vol. 20, No. 1, 134-145, doi: 10.2507/ijsimm20-1-551.
[25] Yang, S.L., Xu, Z.G., Li, G.Z., Wang, J.Y. (2020). Assembly transport optimization for a reconfigurable flow shop based on a discrete event simulation, *Advances in Production Engineering & Management,* Vol. 15, No. 1, 69-80, doi: 10.14743/apem2020.1.350.
[26] Pekarcikova, M., Trebuna, P., Kliment, M., Dic, M. (2021). Solution of bottlenecks in the logistics flow by applying the Kanban module in the Tecnomatix plant simulation software, *Sustainability,* Vol. 13, No. 14, Article No. 7989, doi: 10.3390/su13147989.
[27] Li, G.Z., Xu, Z.G., Yang, S.L., Wang, H.Y., Bai, X.L., Ren, Z.H. (2020). Bottleneck identification and alleviation in a blocked serial production line with discrete event simulation: A case study, *Advances in Production Engineering & Management,* Vol. 15, No. 2, 125-136, doi: 10.14743/apem2020.2.353.
[28] Jurczyk-Bunkowska, M. (2021). Tactical manufacturing capacity planning based on discrete event simulation and throughput accounting: A case study of medium sized production enterprise, *Advances in Production Engineering & Management,* Vol. 16, No. 3, 335-347, doi: 10.14743/apem2021.3.404.
[29] Gregor, M., Hodoň, R., Grznár, P., Mozol, Š. (2022). Design of a system for verification of automatic guided vehicle routes using computer emulation, *Applied Sciences,* Vol. 12, No. 7, Article No. 3397, doi: 10.3390/app12073397.
[30] Li, G., Yang, S., Xu, Z., Wang, J., Ren, Z., Li, G. (2020). Resource allocation methodology based on object-oriented discrete event simulation: A production logistics system case study, *CIRP Journal of Manufacturing Science and Technology,* Vol. 31, 394-405, doi: 10.1016/j.cirpj.2020.07.001.
[31] Gola, A., Pastuszak, Z., Relich, M., Sobaszek, Ł., Szwarc, E. (2021). Scalability analysis of selected structures of a reconfigurable manufacturing system taking into account a reduction in machine tools reliability, *Eksploatacja i Niezawodnosc - Maintenance and Reliability,* Vol. 23, No. 2, 242-252, doi: 10.17531/ein.2021.2.4.
[32] Istokovic, D., Perinic, M., Dobovicek, S., Bazina, T. (2019). Simulation framework for determining the order and size of the product batches in the flow shop: A case study, *Advances in Production Engineering & Management,* Vol. 14, No. 2, 166-176, doi: 10.14743/apem2019.2.319.
[33] Ferro, R., Cordeiro, G.A., Ordóñez, R.E.C., Beydoun, G., Shukla, N. (2021). An optimization tool for production planning: A case study in a textile industry, *Applied Sciences,* Vol. 11, No. 18, Article No. 8312, doi: 10.3390/app11188312.
[34] Yang, S., Xu, Z. (2022). Intelligent scheduling and reconfiguration via deep reinforcement learning in smart manufacturing, *International Journal of Production Research,* Vol. 60, No. 16, 4936-4953, doi: 10.1080/00207543.2021.1943037.