A Co-Evolution Algorithm With Dueling Reinforcement Learning Mechanism for the Energy-Aware Distributed Heterogeneous Flexible Flow-Shop Scheduling Problem

Fuqing Zhao, Fumin Yin, Ling Wang, *Member, IEEE*, and Yang Yu, *Member, IEEE*

***Abstract*—The production process of steelmaking continuous casting (SCC) is a typical heterogeneous distributed manufacturing system. The scheduling problem in heterogeneous distributed manufacturing systems is a complex combinatorial optimization problem. In this article, the energy-aware distributed heterogeneous flexible flow shop scheduling problem (EADHFFSP) with variable speed constraints is studied with objectives, including total tardiness (TTD) and total energy consumption (TEC). A mixed-integer linear programming (MILP) model is constructed for the EADHFFSP. A co-evolution algorithm with dueling reinforcement learning mechanism (DRLCEA) is presented to address EADHFFSP. In DRLCEA, a knowledge-based hybrid initialization operation is proposed to generate the initial population of the problem. A global search based on adversarial generative learning is designed to search the solution space. The dueling double deep *Q*-network (DDQN) is applied to select the operator for the local search. A speed adjustment strategy and an energy-saving strategy based on knowledge are proposed to reduce TTD and TEC of the EADHFFSP with regard to the properties of EADHFFSP. The results of experiments show that the performance of DRLCEA is superior to certain state-of-the-art comparison algorithms in solving EADHFFSP.**

***Index Terms*—Co-evolution, distributed heterogeneous flexible flow shop scheduling, dueling double deep *Q* network (DDQN), energy-saving strategy.**

Received 31 August 2024; revised 26 October 2024 and 5 November 2024; accepted 29 November 2024. Date of publication 23 December 2024; date of current version 19 February 2025. This work was supported in part by the National Natural Science Foundation of China under Grant 62473182; in part by the Key Program of National Natural Science Foundation of Gansu Province under Grant 23JRRA784; and in part by the Industry Support Project of Gansu Province College under Grant 2024CYZC-15. This article was recommended by Associate Editor C. Zhang. *(Corresponding authors: Fuqing Zhao; Ling Wang.)*

Fuqing Zhao and Fumin Yin are with the School of Computer and Communication, Lanzhou University of Technology, Lanzhou 730050, China (e-mail: zhaofq@lut.edu.cn; fyin2716@163.com).

Ling Wang is with the Department of Automation, Tsinghua University, Beijing 100084, China (e-mail: wangling@tsinghua.edu.cn).

Yang Yu is with the College of Electrical and Information Engineering, Jiangsu University of Technology, Changzhou 213001, China (e-mail: dxyy@jsut.edu.cn).

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/TSMC.2024.3510384.

Digital Object Identifier 10.1109/TSMC.2024.3510384

I. Introduction

In recent years, green manufacturing has become a central focus for industries due to its environmental impact and the increasing costs of energy. Energy consumption in the industrial sector has doubled over the past 60 years and is expected to continue rising in the next decade [1]. Many efforts have been focused on designing and developing energy-efficient solutions at the machine and product levels to address these challenges, along with the implementation of energy-conscious scheduling methods to achieve considerable decreases in energy usage [2]. Compared to the extensive research and financial investments required for machine or product redesign, energy-conscious scheduling is considered an economical and low-risk approach to improve energy effectiveness and minimize the utilization of resources. Therefore, researching the use of energy consumption as an objective function is a hot topic [3].

The flexible flow shop scheduling problem (FFSP) is a complex decision scheduling problem that is encountered in multiple real-world manufacturing operations, for example, this article, textile, chemical, and computer assembly industries. In addition to the above industries, the steelmaking industry in northwest China is also a FFSP [4]. In the FFSP, an array of n jobs are managed at an array of s stages, a minimum of one stage involves multiple parallel machines, which increases productivity and balances the load on the machines. Each job is managed sequentially from stage 1 to stage s, and every job is managed through a single machine in that stage at every stage [5]. Moreover, the distributed FFSP (DFFSP) has achieved widespread concern [6]. Distributed manufacturing has been widely utilized in industrial production and scheduling [7]. In a distributed system environment, the extension of the FFSP is named as the DFFSP. In this context, multiple factories constitute the production system, and within each factory, there exists a homogeneous FFSP. Specifically, in each factory, the designated jobs proceed through multiple stages consecutively, where each stage comprises several parallel machines. For the DFFSP, three issues are tackled in order to resolve the problem: 1) factory allocation (FA); 2) the determination of job sequence (JS); and 3) selection of machine. A subproblem of FA is added and the decision space of problem is increased in the DFFSP. However, the heterogeneous situation of factories in actual manufacturing

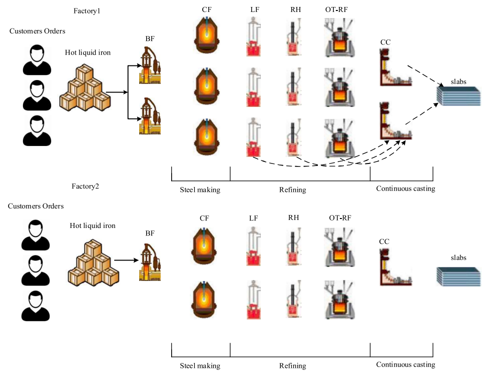


Fig. 1. Steelmaking-refining CC process in heterogeneous factories.

is ignored in the DFFSP. In such a practical production system, specific jobs are processed by factories located in different regions, each with different processing times and machine capacity. In heterogeneous factories, machine numbers, machine power, and machine types are different in each process stage, and this category of problem is named as distributed heterogeneous FFSP (DHFFSP) [8]. Therefore, the research of DHFFSP is of great significance for solving real-world engineering challenges.

This article focuses on investigating the scheduling problem of a distributed heterogeneous flexible flow shop with energy-saving, which arises from the steelmaking process that includes ironmaking, steelmaking continuous casting (SCC), and rolling stages [9]. Among these stages, SCC is considered to be the most complex and critical one as it involves various processes, such as steelmaking, refining, and continuous casting (CC), as depicted in Fig. 1. There are two heterogeneous factories that include different SCC production workshop. Customer orders are assigned to these two factories for processing. In the steelmaking stage, hot liquid iron from the blast furnace (BF) undergoes conversion into molten steel through a converter furnace (CF) before being poured into a ladle in desired quantities. Cranes and trolleys are responsible for transporting the ladle liquid to the refining location where further enhancement of steel quality takes place. This stage contains several substages, including ladle furnace (LF), Ruhstahl–Hausen (RH) vacuum, and other refining furnaces (OT-RF). Finally, during the CC stage, refined liquid steel is cast into slabs. In the manufacturing procedure of SCC, a lot of energy is consumed in the processing of molten steel at high temperature. This research simplifies the SCC process by considering it as an energy-aware DHFFSP (EADHFFSP) with variable machine speed. It is important to note that similar scheduling problems are also encountered in various manufacturing sectors, including but not limited to food processing, pharmaceutical production, and aerospace engineering. The main objective of this study is to minimize both total tardiness (TTD) and total energy consumption (TEC) while solving the EADHFFSP problem due to its practical significance in real-world scenarios prevalent in numerous steel industries.

The complexity of the FFSP has been demonstrated by Gupta [10] to be NP-hard, even in scenarios where there are only two stages with one machine each. In addition, it has been proven that the distributed permutation flow shop scheduling problem (DPFSP) is NP-hard when the total count of jobs exceeds the total count of factories [11]. Furthermore, heterogeneous factories are incorporated into the problem to increase its decision space and add further complexity [12]. Since EADHFFSP is seen as an extension of FFSP and DPFSP, the solution space of EADHFFSP is larger than FFSP and DPFSP. Therefore, EADHFFSP is also proved to be NP-hard.

Since EADHFFSP falls under NP-hard category because of its nature as a distributed flexible flow shop problem, it becomes crucial to develop an optimization method capable of finding approximate solutions within appropriate CPU time constraints. While multiple heuristic and meta-heuristic algorithms are available for solving DFFSP, it is important to note that each algorithm has its own set of strengths or weaknesses depending on specific optimization problems based on the "there is no free lunch" theorem [13]. For solving small-scale problems, the use of analytical algorithms gets the optimal solution, but for solving large-scale problems, it is impossible to get the solution in a short time. When the solution space of the problem is large, more time and difficulty is taken to detect the solution space in the analytical algorithm [14], and the algorithm search is learned according to the strategy learning in the intelligent algorithm with multistrategy, avoid random search, and the intelligent algorithm has the characteristic that as long as the problem is given, a better solution is given than the analytical algorithm. Therefore, meta-heuristic algorithm [15] is used, that is, intelligent algorithm to cope with the problem in this article.

Heuristic algorithms fall into local optima in the process of finding solutions and have high-time complexity, while theoretical guidance is lacked in the metaheuristic algorithms and their performance is greatly impacted by factors, including initialization of the problem and parameter Settings. Heuristic algorithms are suitable for solving problems with clear rules and structure, while incorporating a local search component, the exploration of the solution space during the search process is enhanced in the metaheuristic algorithms, thereby facilitating the identification of optimal solutions compared to heuristic algorithms. Metaheuristics are more suitable for handling combinatorial optimization problems with high complexity and unclear structure, meta-heuristic algorithm [15] is used to solve EADHFFSP in this article.

In recent years, researchers have been focusing on obtaining approximate optimal solutions using metaheuristic algorithms within a finite time. These algorithms have attracted considerable attention for their capacity to tackle a diverse array of optimization problems, without being limited to any specific problem [16]. The co-evolution algorithm is a metaheuristic algorithm that improves solution performance through the interaction between populations. Scholars have recently shown great interest in combining reinforcement learning, known for its strong decision-making and optimization capabilities, with metaheuristic algorithms [17]. One such combination is the dueling double deep Q-networks (DDQN) algorithm, which enhances the learning of state value functions compared to traditional double deep Q networks. This leads to faster convergence of networks and improved efficiency in operator selection [18]. Integrating the dueling DDQN mechanism with the co-evolution algorithm holds significance as it leverages empirical knowledge to prevent ineffective searches. This integration allows for an effective and efficient optimization process.

In this article, a co-evolution algorithm with dueling reinforcement learning mechanism (DRLCEA) is proposed to solve EADHFFSP with the minimization of TTD and TEC. DRLCEA puts the global search and local search into two types of populations, called population1 and population2, respectively. First, in the process of global search, population1 evolves one generation through a global search operator based on adversarial generative learning [19]. Second, population2 absorbs Pareto solutions from population1 to form an elite archive set. The operator selection mechanism based on dueling DDQN is used to update each elite solution and find more potential nondominated solutions locally. Finally, the elite archive set reduces TTD and TEC by using a speed adjustment strategy and energy saving strategy, which outputs the final nondominated solution set. The main contributions of this article are outlined as follows.

1) The EADHFFSP with variable machine speed and the optimization of TTD simultaneously is considered in the production processes of SCC, and a mixed-integer linear programming (MILP) model is constructed to reflect real production requirements.
2) A global search based on adversarial generative learning, which combines the advantages of subpopulations and accelerates the convergence process of the DRLCEA, is proposed for the EADHFFSP.
3) The dueling DDQN, which learns more efficiently than DDQN, is utilized to select the local search operators for each solution to reduce invalid searches of DRLCEA.

The remaining sections of this article are outlined as follows. Section II offers an overview of the related literature. A comprehensive problem description of EADHFFSP is devised in Section III. In Section IV, the DRLCEA approach is imported. The experimental results and analysis of the DRLCEA are presented in Section V. Finally, the conclusion is summarized and the direction of future work is discussed in Section VI.

II. Literature Review

A. Related Work on the DFFSP

In the past three years, the DFFSP and DHFFSP have garnered a lot of interest, and various mathematical models and optimization algorithms have emerged. The researches about DFFSP and DHFFSP are arranged in Table S-I in the supplementary material. As for DFFSP, Shao et al. [4] designed several constructive heuristics according to the rules of dispatching and proposed a constructive heuristic approach tailored for the no-wait constrained DFFSP, utilizing a variable neighborhood descent strategy. Furthermore, Shao et al. [20] applied some multiple neighborhoods local search operators and an adaptive weight updating method for the DFFSP with the objective of minimizing the maximum completion time, total weighted earliness and tardiness, and total workload. Li et al. [21] designed the DFFSP with variable speed constraints in a prefabricated system and presented a Pareto-based similar job order crossover method to improve search ability which achieve a good result. Lu et al. [22] designed a novel mathematical formulation for the DFFSP, aimed at reducing both the maximum completion time and overall energy consumption to a minimum. Yu et al. [23] presented a DFFSP with assembly and dual-resource constraints. An iterated greedy algorithm was put forward to cope with the problem, combined with the problem knowledge. Liu et al. [24] studied a DFFSP considering blocking constraints with the minimization of the maximum completion time and proposed a strategy to rotate critical factories which achieves a fast convergence speed.

As for DHFFSP, Shao et al. [25] studied the DHFFSP subject to constraints on energy and labor awareness, and designed a network memetic algorithm to cope with the problem. The algorithm takes into account multiple targets, such as total delay, total production cost, and total carbon emissions. Lei and Su [26] solved the DHFFSP with sequence dependence and setup time, and presented a multicourse teaching-based optimization method, where the maximum completion time and the maximum delay were both minimized. Wang and Wang [27] put forward the DFFSP with heterogeneous factories to minimize the maximum completion time. One of the two populations was used for local search, and the other was used to learn the information of the former population. Shao et al. [28] introduced a learning-based selection super-heuristic algorithm to effectively handle situations involving the DHFFSP with blocking constraints. Qin et al. [29] studied the Collaborative Iterative Greedy algorithm as a solution for tackling the DHFFSP with blocking constraints. Shao et al. [30] presented an iterative local search algorithm specifically designed for addressing the issue of DHHFSP with the lot-streaming constraints. To solve the DHFFSP, Xuan et al. [31] put forward an artificial immune differential evolution algorithm, where the maximum completion time and the maximum delay were both minimized. Lu et al. [32] presented a study on the DHFFSP, considering objectives, such as maximum completion time and overall energy consumption, and devised a multiobjective hybrid iterative greedy algorithm in accordance with Pareto principles, which combines genetic operators with iterative greedy heuristic algorithms.

B. Related Work on Meta-Heuristic Algorithm and Dueling Reinforcement Learning

In recent years, meta-heuristic algorithms have demonstrated promising performance in addressing scheduling problems [33], [34]. For example, Lin et al. [35] brought forward a genetic algorithm (GA) to handle the new hospital scheduling problem. Experiments show that the effect is better than the traditional method. Among these algorithms, coevolutionary approaches that optimize solutions through interactions between populations have garnered significant attention. Zhao et al. [36] came up with a two-stage cooperative evolutionary algorithm that incorporates problem-specific knowledge to tackle the flow shop group scheduling problem and optimize the objective function of energy efficiency. Wang and Wang [37] brought forward a collaborative memetic algorithm incorporating learning agents to optimize both the maximum completion time and TEC, ultimately improving the efficiency of convergence for the EADHFFSP. Furthermore, Li et al. [38] put forward a variant of the artificial bee colony (ABC) algorithm that introduces Q-learning mechanism to learn the knowledge of the domain structure to approach the permutation flow-shop scheduling problem.

In recent years, the method of combining reinforcement learning with meta-heuristic algorithm, which is known for its powerful decision-making and optimization ability, is currently a hot topic for researchers [39]. Yu et al. [40] improved the metaheuristic algorithm and designed six local search strategies, and introduced Q-learning mechanism to select the appropriate operator, which achieved good results in optimizing the surgery scheduling problem. Li et al. [41] developed a co-evolution method based upon double deep Q network to solve distributed heterogeneous hybrid flow-shop scheduling problem with multiple job priorities. Combining DDQN with co-evolution improves the performance of the algorithm in distributed job-shop scheduling considering energy consumption constraints.

On the other hand, dueling DDQN is an extension of DDQN algorithm. By combining the ideas of double Q-learning and dueling DQN, it improves the training stability and enhances the performance of learning strategies [18]. Yao et al. [42] developed an XSS attack vector automatic generation method by using the improved Dueling DDQN algorithm to ameliorate algorithm performance and attack vector generation speed. Song et al. [43] proposed a priority replay duel DDQN (PRDDDQN) method for grid edge control in community energy storage systems. In addition, Yan et al. [44] proposed an advanced virtual machine placement (VMP) algorithm based on Dueling DDQN, which solved the problems, such as overestimation of Q value, difficulty in convergence and inability to maximize long-term return in the native DQN algorithm.

C. Research Gaps

According to the Comparative review of the related literatures, the research gaps are described as follows.

1) The previous research primarily concentrates on optimizing the maximum completion time while disregarding the impact of TTD caused by the due date of job on customer satisfaction and core competitiveness.
2) The variable machine speed is seldom considered in DHFFSP. Moreover, considering both variable machine speed and the due date of job helps enterprises deal with real-world problems.
3) The dueling DDQN [44] is a powerful technique and first be introduced to solve the EADHFFSP.

III. Problem Formulation

A. Notation Definition

The symbols defined in EADHFFSP are shown as follows.

F	Number of factories.
n	Number of jobs.
m	Number of stages in every factory.
f	The index of factories. $f \in \{1, 2, \ldots, F\}$.
j	The index of jobs. $j \in \{1, 2, \ldots, n\}$.
k	The index of stages. $k \in \{1, 2, \ldots, m\}$.
$m_{f,k}$	The number of machines at stage k in factory f.
i	Index of machines $i \in \{1, 2, \ldots, m_s\}$.
$v_{f,k}$	Speed of job i at stage k in factory f.
$P_{j,k}$	Standard processing time of job j at stage k.
PP	The power consumption of process when the machine working.
SP	Power consumption of idle when the machine stands by.
D_j	Due date of job j.
U	Very large positive number.
$\mathrm{PEC}_{f,k,i}$	Energy consumption of process for machine i at stage k in factory f during processing mode.
$\mathrm{SEC}_{f,k,i}$	Energy consumption of idle for machine i at stage k in factory f during standby mode.
$\mathrm{ST}_{j,k}$	Start time of job j at stage k.
$\mathrm{FT}_{j,k}$	Completion time of job j at stage k.
TEC	Total energy consumption.
TTD	The total tardiness.
$x_{f,j}$	Binary variable equals 1 if job j is allocated in factory f, otherwise equals 0.
$y_{f,k,j,i}$	Binary variable equals 1 if job j is allocated to machine i in factory f, otherwise equals 0.
$Z_{f,k,j,j'}$	Binary variable equals 1 if job j is allocated to machine i in factory f, otherwise equals 0.
V_j	Due date violation of job j.

B. Problem Definition

The proposed EADHFFSP in this article is abstracted from the SCC. In the production process of steel, hot liquid iron is abstracted the job and the furnaces are abstracted the parallel machines. The machine efficiency of furnaces in different processing stages is heterogeneous, thus abstracting the constraint of variable machine speed. Customers have time deadlines for orders, thus abstractive the goal of TTD delay in order to improve customer satisfaction. A lot of energy consumption is generated in the whole process, introduce the optimization of the TEC is essential in the SCC.

In the proposed EADHFFSP with variable speed constraints, there are n job positions to be allocated among F factories. Every factory consists of m different stages, with $m_{f,k}$ parallel machines in every stage. The processing characteristics vary across factories, with distinct standard processing times and processing speeds of every job in every stage. The practical processing time for a job in every stage is ensured by the standard processing time and processing speed of that stage. Every job has a due date D_i, and the aim is to complete every job before its respective due date. The EADHFFSP involves solving three interconnected subproblems: 1) assigning jobs to specific factories; 2) assigning a machine to every job at every stage; and 3) determining the optimal order of jobs in every heterogeneous factory. For the EADHFFSP, each factory has a different FFSP, which embodies the heterogeneity of the layout. At the same time, each stage has distinct processing varieties of machines, and their processing capabilities are different, which reflects the heterogeneity of processing performance. Specifically, although every factory has the same number of stages, every stage has a different number of machines and the processing speed of each machine is different, and the standard processing time of the job is the same on the machine in one stage and different on the different stages. Hence, an increase in processing speed leads to a decrease in processing time but results in higher-energy consumption by the machine, giving rise to a contradiction between the two objectives within the EADHFFSP.

The following assumptions are made in EADHFFSP: a machine processes one job at one time and one job is managed by one machine. The processing capacity of each scheduling shop is obtained in advance and is different for each scheduling shop. Open when the machine processes the first part assigned, and close when all jobs allocated to the machine are completed. Every job allocated to a factory is not transferred to another one factory, and each machine is not interrupted processing when processing the job. Each machine is continuous and trouble-free. There exists an infinite buffer between adjacent machines, and the setup time is either disregarded or merged into the processing time of every job. The mixed integer linear programming model for EADHFFSP is exhibited as follows:

$$\min\{\mathrm{TTD}, \mathrm{TEC}\} \tag{1}$$

$$\text{Subject to}: \sum_{f\in F} X_{f,j} = 1 \ \ \forall j \tag{2}$$

$$\sum_{i=1}^{m_{f,k}} y_{f,k,j,i} = X_{f,j} \ \ \forall f, j, k \tag{3}$$

$$\mathrm{ST}_{j,1} \geq 0 \ \ \forall j \tag{4}$$

$$\mathrm{FT}_{j,k} = \mathrm{ST}_{j,k} + \sum_{f=1}^{F}\sum_{i=1}^{m_{f,k}} y_{f,k,j,i} \times \frac{p_{j,k}}{v_{f,k,i}} \ \ \forall j, k \tag{5}$$

$$\mathrm{ST}_{j,k+1} \geq \mathrm{FT}_{j,k} \ \ \forall j, k \tag{6}$$

$$z_{f,k,j,j'} + z_{f,k,j',j} \leq 1 \ \ \forall f, k, j, j' \tag{7}$$

$$z_{f,k,j,j'} + z_{f,k,j',j} \geq y_{f,k,j,i} + y_{f,k,j',i} \ \ \forall f, k, j' > j \tag{8}$$

$$\begin{aligned}&\mathrm{ST}_{j',k} - \mathrm{FT}_{j,k} + U\\ &\quad\times\left(3 - y_{f,k,j,i} - y_{f,k,j',i} - z_{f,k,j,j'}\right) \geq 0 \ \ \forall j\\ &\qquad \neq j', k, f, i \in \{1, 2, \ldots, m_{f,k}\}\end{aligned} \tag{9}$$

$$\begin{aligned}&\mathrm{PEC}_{f,k,i} = \sum_{j=1}^{n} y_{f,k,j,i} \times PP \times p_{j,k}/v_{f,k,i}\\ &\qquad \forall f, k, i \in \{1, 2, \ldots, m_{f,k}\}\end{aligned} \tag{10}$$

$$\begin{aligned}&\mathrm{SEC}_{f,k,i} = SP\left\{\begin{matrix}\max_j\left(\mathrm{FT}_{j,k} \times \ y_{f,k,j,i}\right) - \min_j\left(\mathrm{ST}_{j,k} \times \ y_{f,k,j,i}\right)\\ - \sum_j^n y_{f,k,j,i} \ \times \ \frac{p_{j,k}}{v_{f,k,i}}\end{matrix}\right\}\\ &\qquad \forall f, k, i \in \{1, 2, \ldots, m_{f,k}\}\end{aligned} \tag{11}$$

$$\begin{aligned}&V_j \geq \mathrm{FT}_{j,m} - D_j - U\left(1 - y_{f,k,j,i}\right), V_j \leq \mathrm{FT}_{j,m}\\ &\quad - D_j + U\left(1 - y_{f,k,j,i}\right) \ \ \forall f, j, i, k\end{aligned} \tag{12}$$

$$\mathrm{TTD} = \sum_{j=1}^{n} \max\left(0, V_j\right) \tag{13}$$

$$\mathrm{TEC} = \sum_{f=1}^{F}\sum_{k=1}^{m}\sum_{i=1}^{m_s}\left(\mathrm{SEC}_{f,k,i} + \mathrm{PEC}_{f,k,i}\right) \tag{14}$$

$$x_{f,j} \in \{0, 1\} \ \ \forall f, j \tag{15}$$

$$y_{f,k,j,i} \in \{0, 1\} \ \ \forall f, k, j, i \in 1, 2, \ldots, ms \tag{16}$$

$$z_{f,k,j,j'} \in \{0, 1\} \ \ \forall f, k, j, j' \tag{17}$$

where (1) indicates that the goal of the problem is to minimize TTD and TEC. Constraint (2) indicate that every job is guaranteed to be allocated to only one factory. Constraint (3) indicate that every job is guaranteed to be handled by only one machine at every stage. Constraint (4) indicate that the start machining time of any job in the initial stage is greater than or equal to 0. Constraint (5) guarantee the calculation of the completion time of every process. Constraint (6) indicate that the processing time of every process of every job is higher than or equal to the processing completion time of the former stage of the process. Constraints (7)–(9) indicate that every machine processes only one job at a time. Constraints (10) and (11) represent the calculation method of the energy consumption of every machine in the working state and standby state. Constraint (12) represent the expiration date violation for every job. Constraint (13) indicates the calculation method of the TTD of all jobs throughout the machining process. Constraint (14) represents the calculation method of the TEC of all jobs during the entire machining process. Constraints (15)–(17) represent the definition of the decision variables in the entire problem.

C. Example for EADHFFSP

An example is shown in Table I. At the same time, each stage has two parallel machines with differences in processing time and processing speed. Suppose the JS vector is $JS = \{3, 4, 1, 7, 0, 2, 6, 5\}$, and the FA vector is $FA = \{0, 1, 1, 1, 0, 0, 1, 0\}$. Jobs $\{J1, J5, J6, J8\}$ are assigned to $F1$, jobs $\{J2, J3, J4, J7\}$ are allocated to $F2$. The machine with the shortest completion time is selected when the job is scheduled. The completion time of all the work is obtained and it is determined whether each work is past the due date. In Factory $F1$, calculate $\mathrm{TTD}_1 = 1+1+2+1 = 5$, Similarly, in Factory $F2$, calculate $\mathrm{TTD}_2 = 1$, and the total violation is $\mathrm{TTD} = \mathrm{TTD}_1 + \mathrm{TTD}_2 = 5 + 1 = 6$. The power consumption of each machine is determined by the work and idle power rates, which are represented as $4\times v^2$ kW/h and 1 kW/h, respectively. Fig. 2 illustrates the power consumption change chart. The overall energy consumption is determined by measuring the area enclosed between the line and the axis.

$\mathrm{TEC}_1 = 11.6 \times 3 + 12.84 + 20.68 + 3 \times 16.68 + 9.84 + 2 \times 10.76 + 6.76 = 156.48$, $\mathrm{TEC}_2 = 12.52 \times 2 + 21.36 \times 2 + 13.68 \times 2 + 10.68 + 5.84 + 4.00 \times 3 = 123.64$. The TEC $TEC = \mathrm{TEC}_1 + \mathrm{TEC}_2 = 156.48 + 123.64 = 280.12$.

IV. Proposed DRLCEA Algorithm for the EADHFFSP

The DRLCEA is proposed to solve EADHFFSP with TTD criterion and TEC criterion in Algorithm 1.

Inspired by the idea of the co-evolution algorithm, DRLCEA puts the global search and local search into two types of populations, called population1 and population2,

TABLE I
EXAMPLE OF EADHFFSP

Job	Factory 1$(P_{j,k}/v_{f,k,i})$		Factory 2$(P_{j,k}/v_{f,k,i})$		D_j
	Stage 1	Stage2	Stage 1	Stage 2	
$J1$	3.9/1.3	5.0/1.0	4.8/1.2	2.2/1.1	7
$J2$	4.4/1.1	2.4/1.2	2.4/1.2	2.8/1.4	6
$J3$	2.0/1.0	3.0/1.0	4.8/1.2	4.8/1.2	10
$J4$	4.4/1.1	5.6/1.4	5.6/1.4	3.0/1.0	6
$J5$	2.0/1.0	1.0/1.0	5.2/1.3	2.0/1.0	6
$J6$	5.5/1.1	3.9/1.3	3.0/1.0	4.4/1.1	11
$J7$	3.3/1.1	3.9/1.3	4.4/1.1	3.0/1.0	10
$J8$	4.0/1.0	2.8/1.4	2.6/1.3	3.3/1.1	7

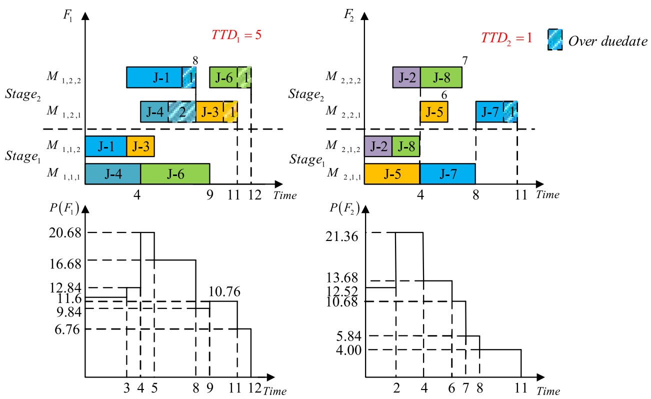


Fig. 2. Gantt chart and power consumption of the example.

respectively. First, the DRLCEA generates the initial population using a knowledge-based hybrid heuristic initialization operation. Second, in the process of global search, population1 which is divided into generative population and discriminant population evolves one generation through a global search operator based on adversarial generative learning. Third, population2 absorbs Pareto solutions from population1 to form an elite archive set. The operator selection mechanism based on dueling DDQN is used to update each elite solution and find more potential nondominated solutions locally. Finally, the elite archive set reduces TTD and TEC by using a speed adjustment strategy and energy saving strategy, which outputs the final Pareto solution set. The DRLCEA is shown in Fig. S-1 in the supplementary material. The pseudocode of the DRLCEA is exhibited as Algorithm 1.

A. Encoding and Decoding

DRLCEA uses a double-layer coding pattern to represent the solution to the problem. The solution representation is shown in Fig. S-2 in the supplementary material, in addition, the encoding and decoding methods are described below.

Coding Mode: The solution to the problem is represented by two vectors, namely, the JS and the FA, and the length of the JS and the FA are both equal to the number of parts. The same job is allocated to the same factories.

Decoding Mode: First, jobs are allocated to heterogeneous factories in accordance with the FA vector. Subsequently, the JS of heterogeneous factories is derived from the JS vector, and machines are assigned to every stage following the first available machine (FAM) rule. Finally, the completion time of each operation in the last stage is determined and subsequently used to calculate both TTD and TEC for obtaining corresponding indices.

Algorithm 1 Procedure of DRLCEA

Input: generative population, discriminant population, Ps, Pc, Pm, Pa
Output: Pareto-Set
generative population, discriminant population ← initialization (generative population, discriminant population);
While the stopping criterion is not satisfied **do**
  generative population, discriminant population ← tournament selection (generative population, discriminant population);
  generative population, discriminant population ← global search based on adversarial generative learning (generative population, discriminant population);
  archive-Set ← nondomination (generative population, discriminant population, Ps);
  use dueling DDQN guides the selection of the operations;
  archive-Set ← Local search(archive-Set);
  train dueling DDQN;
  archive-Set ← Speed adjustment strategy and energy saving(archive-Set);
End while
Pareto-Set ← Get Pareto Front(archive-Set)

B. Initialization Based on the Domain Knowledge

In DRLCEA, two heuristics are presented to produce high-quality and diverse initial solutions. The first heuristic aims to minimize machine load by randomly generating sequences and assigning jobs to factories with lower-machine loads. The Nawaz–Enscore–Ham (NEH) heuristic, known for its efficiency in flow shop scheduling problems (FSP), is utilized in the second heuristic. In this case, priority sequences are obtained based on job delay times, and two NEH-based FA rules are extended to allocate JSs. The first rule, DNEH1, assigns $F(n \gg F)$ activities to F factories in priority order, with one job assigned per factory. The remaining activities are assigned according to the privileged sequence to the factory with the earliest completion time. This rule aims to minimize the maximum completion time and equilibrate the workload across factories.

DNEH2 is an enhanced version of DNEH1, which uses the greedy insertion step to handle the insertion of new jobs at the current plant. In this factory, each job is reinserted into all possible locations and the one that takes the shortest time to complete is selected. Combining the above two heuristics, additional artifact sequences are produced in a random manner to supplement the initial population. The pseudocode of the two NEH-based FA rules is shown in Algorithms 2 and 3, respectively.

C. Global Search Based on Adversarial Generative Learning

A global search based on adversarial generative learning is devised to address EADHFFSP in the DRLCEA.

Algorithm 2 DNEH1

Input: the processing time of each job, a job privilege sequence JS
Output: a feasible solution JS, a feasible solution FA
For $i = 1$ to F do
  insert JS (i) into the proper position by utilizing the machine selection rule
End for
For $i = k + 1$ to n do
  insert JS (i) into all possible positions in all factories and check all maximum completion time values by the machine selection rule.
  place JS (i)in the position resulting in the lowest maximum completion time.
End for

Algorithm 3 DNEH2

Input: the processing time of each job, a job privilege sequence JS
Output: a feasible solution JS, a feasible solution FA
For $i = 1$ to F do
  insert JS (i) into the proper position by utilizing the machine selection rule
End for
For $i = k + 1$ to n do
  insert JS (i) into all possible positions in all factories and check all maximum completion time values by the machine selection rule.
  place JS (i) into the position of the factory (denoted as f') resulting in the lowest maximum completion time.
  For each job in the factory f' do
    take out the job and insert all possible positions in the factory f'
    place the job in the position resulting in the lowest maximum completion time
  End for
End for

Two populations are included in the global search, called generative population and discriminant population, respectively, where they engage in adversarial generative learning [19]. Adversarial Generative Learning is a training process based on Generative Adversarial Network principles and game theory. It involves two neural networks, namely, the generator and discriminator, which compete with each other to generate realistic synthetic data. It combines adversarial training and strategic optimization to achieve high-quality data generation in various domains.

A cooperative precedence operation crossover (CPOX) operator and a competitive uniform crossover (CUX) operator are designed in the global search for JS and FA, where the former is composed of the precedence operation crossover operator [45] and the reverse precedence operation crossover (RPOX) operator and the latter is composed of the uniform crossover operator [45] and the reverse uniform crossover

Algorithm 4 Global Search Based on Adversarial Generative Learning

Input: generative population, discriminant population, Ps
Output: archive-Set
rank1, rank2 ← index of the top Ps/2 individuals from two populations;
If rand > 2 ×pa
  generative population, discriminant population ← crossover and mutation (generative population, discriminant population)
Else
  If rand > pa
    generative population' ← combine (generative population, discriminant population (rank2))
    discriminant population (rank) ← crossover and mutation (discriminant population (rank))
    generative population ← nondomination (generative population', Ps)
  Else
    discriminant population' ← combine (discriminant population, generative population (rank1))
    generative population (rank) ←crossover and mutation (generative population (rank))
    discriminant population ← nondomination (discriminant population', Ps)
  End if
End if
archive-Set ← nondomination (generative population, discriminant population, Ps)

(RUX) operator. A rate of Pa controls which operator is used to update the population. If the probability of randomization is greater than 2 times Pa, the global search utilizes the precedence operation crossover (POX) operator and uniform crossover operator to update the population, otherwise, the global search utilizes the RPOX operator and RUX operator to update the population. Additionally, after the crossover and mutation operators are applied, each offspring undergoes a two-point swap mutation for JS with a mutation rate of Pm, and a one-point swap mutation for FA. The process of environmental selection is conducted, following the approach described in [46]. The pseudocode is exhibited in Algorithm 4.

More details about RPOX and RUX are described in Fig. S-3 in the supplementary material.

RPOX: Fig. S-3 gives an example of RPOX with two JSs JS_1 and JS_2. First, divide set I into two subsets $S_1 = \{5, 2\}$ and $S2 = \{1, 4, 3\}$. Second, copy the jobs belonging to $S_1(S_2)$ into the child $JS_3(JS_4)$. Finally, fill the empty positions of $JS_3(JS_4)$ with jobs belonging to $S_2(S_1)$ from right to left.

RUX: Fig. S-3 gives an example of RUX with two FAs FA_1 and FA_2. First, generate a 0-1 Set S randomly. Second, from right to left, if $S_i = 1$, exchange the value of FA_1 and FA_2 in the same position of S_i. Finally, copy the value of $FA_1(FA_2)$ to $FA_3(FA_4)$ where S_i is 0. The two populations engage in adversarial generative learning, incorporating the strengths of each population and allowing both populations to learn advantageous individuals. Through collaborative evaluation, the final population is obtained.

D. Search Mechanism Based on Dueling DDQN

A search mechanism using dueling DDQN is devised to update the solutions in the elite archive set. The mechanism incorporates thirteen problem-specific local search operators to reduce randomness and enhance efficiency. Dueling DDQN is introduced to determine the optimal search operator for every solution in the elite archive set. The process begins by identifying the critical factory and critical job. The critical factory is the largest factory in TTD (F_d) and the largest factory in TEC (F_c). Second, the job with the largest tardiness (J_d) in the largest factory is defined as the critical job. Finally, thirteen local search operators are described in Table II.

TABLE II
LOCAL SEARCH OPERATORS

Class	Name	Details
Inner-adjustment	N1	Introducing the destruction and reorganization mechanism to obtain two subsequences in F_d . The job with a small deadline in the two subsequences is inserted into the previous position in the original sequence.
	N2	Introducing the destruction and reorganization mechanism to obtain two subsequences in F_c . The job with a small deadline in the two subsequences is inserted into the previous position in the original sequence.
	N3	In F_d , the position of a critical job is identified. The non-critical jobs in the factory are traversed, and a job with a deadline smaller than the critical job is found. The two jobs are exchanged with each other.
	N4	In F_c , the position of a critical job is identified. The non-critical jobs in the factory are traversed, and a job with a deadline smaller than the critical job is found. The two jobs are exchanged with each other.
	N5	In F_c , the job in the last order in the job sequence of the F_c is compared sequentially with the preceding jobs. If a job with a smaller deadline is found, the two jobs are swapped.
	N6	In F_c , the job in the last order in the job sequence of the F_c is compared sequentially with the preceding jobs. If a job with a smaller deadline is found, the last job is inserted in front of that job.
	N7	In F_d , the critical jobs in the job sequence of the F_d are compared sequentially with the preceding jobs. If a job with a smaller deadline is found, the two jobs are swapped.
	N8	In F_d , the critical jobs in the job sequence of the F_d are compared sequentially with the preceding jobs. If a job with a smaller deadline is found, the last job is inserted in front of that job.
Inter-adjustment	N9	In F_d, a random job from the other factories and a job from the F_d are selected, and these two jobs are exchanged.
	N10	In F_c , a random job from the other factories and a job from the F_c are selected, and these two jobs are exchanged.
	N11	In F_d , the deadline of the first job in the other factories is compared with the deadline of the first job in the F_d. If the former is smaller, these two jobs are swapped. The first and last jobs in the F_d are also compared, and if the deadline of the first job is smaller than the deadline of the last job, their positions are swapped.
	N12	In F_d , the deadline of the last job in the other factories is compared with the deadline of the last job in the F_d . If the former is smaller, these two jobs are swapped. The first and last jobs in the F_d are also compared, and if the deadline of the first job is smaller than the deadline of the last job, their positions are swapped.
	N13	Combining N11 and N12, the first and last jobs of the non-critical and critical factories are compared and exchanged.

The sequential decision process between the agent and the environment is the focus of reinforcement learning, which is expressed by the triplet MDP model (s, a, r). The s, a, and r represent status, action, and reward, respectively. In this study, the local search process of an individual is regarded as an MDP, and based on this, the agent, environment, state, action and reward are defined.

Agent: The individual in the elite archive set population for DRLCEA serves as the agent in the MDP model.

Environment: The environment in the DRLCEA is the scheduling optimization problem (EADHFFSP). It is responsible for evaluating the agent's performance during learning.

State: The state is represented by a vector combining the JS and the FA.

Action: The available actions consist of thirteen local search operators (N1–N13).

Reward: The environment assigns a reward when the agent takes an action a_t. In the DRLCEA, the reward (r) is set to 20 when the action a_t leads to an improved solution, and 0 otherwise. If the improvement is solely in terms of TTD, the reward is 15; if only the TEC is improved, the reward is 10.

Network Structure: In this article, a shared network and two branch networks are designed: one is the advantage network (A-network), which is utilized to estimate the advantage value of each action, and the other is the Value network (V-network), which is utilized to estimate the value function of the state. The shared network derives the feature representation of the state using five fully connected layers, and inputs it into the Advantage network and the Value network, respectively, for processing, and obtains the estimated value of each action and the value function estimation of the state, and finally, the action value estimation of each action is obtained by combining the output of the Advantage network and the Value network.

Operator Selection: In this article, the Q network (including the shared network, the separated Advantage network, and the Value network) has been initialized and constructed before the local search. The ϵ-greedy strategy (use the actual Greek alphabet ε) is utilized to equilibrate the exploration and exploitation [47]. First, the current state S_t of the solution in the elite archive set is obtained, and through local search, when the transactions within the experience pool are greater than the threshold specified in advance, the network is trained, and the target network Q-target updates the parameters from the evaluation network Q-evaluate every certain number of steps, and randomly select 16 $(S_t; A_t; R_t; S_{t+1})$. Second, DRLCEA uses the action $A_t$of the current time step and the Q value of all states S_{t+1} of the next time step to predict the Q value of the current time step state S_t. At the same time, the action is chosen with the largest Q value as the Q target. In addition, DRLCEA uses A_{t+1} to predict the Q value of the next time step state S_{t+1} and update the Q value. Finally, DRLCEA uses the Adam optimizer to minimize losses and update Q-evaluate parameters with a learning rate of 0.001. The detailed process of local search based on dueling DDQN is illustrated in Algorithm 5.

E. Speed Adjustment Strategy and Energy Saving

This section focuses on reducing energy consumption and tardiness metrics for nondominated solutions through speed

Algorithm 5 Local Search Based on Dueling DDQN

Input: archive-Set solution, Q-target, Q-evaluate
Output: archive-Set solution
k = size(archive-Set)
For t = 1 to k
  If rand < ε
    Select the action by maximum Q value;
  Else
    Select the action randomly;
  End if
  Switch (action)
    Case 1: N1; Case 2: N2; Case 3: N3; Case 4: N4; Case 5: N5;
    Case 6: N6; Case 7: N7; Case 8: N8; Case 9: N9; Case 10: N10; Case 11: N11; Case 12: N12; Case 13: N13;
  End
  Generate new-Solution in accordance with the operator above.
  Calculate the objective values of the new-Solution (called $TTD_{newsolution}, TEC_{newsolution}$).
  If $TTD_{newsolution} < TTD_{original}$ and $TEC_{newsolution} < TEC_{original}$
    r = 20; archive-Set(k) = new-Solution
  Else if $TTD_{newsolution} < TTD_{original}$
    r = 15; archive-Set(k) = combine (new-Solution, archive-Set(k))
  Else if $TEC_{newsolution} < TEC_{original}$
    r = 10; archive-Set(k) = combine (new-Solution, archive-Set(k))
  Else
    r = 0;
  End if
  update the Q-target and Q-evaluate.
End for

Algorithm 6 Right-Shift Strategy

Input: A original schedule decoded from (JS, FA)
Output: A current schedule with less TEC.
For $f = 1, 2, \ldots, F$ do
  For $k = m, m-1, \ldots, 1$ do
    $JS_{f,k}(n)$ denotes the nth job in $JS_{f,k}$
    For $n = N, N-1, \ldots, 1$ do
      transform the start time of $JS_{f,k}(n)$ on stage k to
      $S_{JS_{f,k}(n)} = \min\left\{S_{next},\ S_{JS_{f,k+1}(n)},\ JobDuedate(n)\right\} - p_{n,k}/v_{f,k,n}$,
      where S_{next} is the start time of the job processed immediately after $JS_{f,k}(n)$ on the same machine, where
      $JobDuedate(n)$ is the duedate of the nth job in $JS_{f,k}$.
    End for
  End for
End for

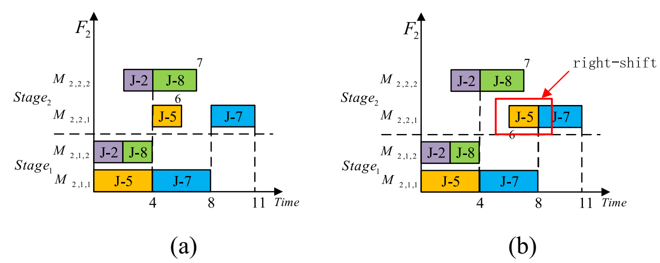


Fig. 3. Example of the right-shift strategy. (a) Original schedule. (b) Current schedule by right shift.

adjustment and energy-saving strategies. First, Critical jobs in critical factories are accelerated, shifting the current frontier toward the energy consumption axis. Noncritical jobs in critical factories are decelerated, moving the current frontier toward the TTD axis. On the other hand, minimizing idle time for each machine helps decrease energy consumption during idle states. A Right-shift strategy is devised to delay certain operations in Algorithm 6, reducing idle time without affecting job completion times.

Fig. 3 illustrates a scheduling solution utilizing the Right-shift strategy, resulting in shorter total idle time and optimizing the energy consumption metric. To update the elite archive, the two strategies are sequentially applied to all nondominated solutions in the current elite archive, generating new solutions that offer further improvements in energy consumption and tardiness.

V. Experiment Results and Analysis

To evaluate the effectiveness of the proposed DRLCEA, random test instances are generated with various parameters. The number of factories F is from {2, 3, 5, 6, 7}, the number of jobs n is from {20, 40, 60, 80, 100, 200, 500}, and the number of stages m is from {3, 5}. The processing time $p_{j,k}$ for each job at a specific stage is uniformly sampled from the discrete range [10, 90]. The number of parallel machines $m_{f,k}$ at a stage k in a factory f is randomly chosen from {2, 3}, and the speed $v_{f,k}$ at that stage is selected from {1.0, 1.1, 1.2, 1.3, 1.4}. The power consumption PP is calculated as 4 times the square of the speed $PP = 4 \times {v_{f,k}}^2$, and the power consumption at standby mode is placed as 1. The due date d_j for each job j is generated by (18), where the due date is in the range of $\{d_j - 10, d_j + 20\}$. To ensure fairness in the experiments, thirty-six instances are generated for each combination of (F, n, m). Each instance is run independently twenty times. The termination criterion is placed as Max-NEFs $= 400 \times n$, n, s, f indicate the number of jobs, stages, and factories, respectively. All algorithms are performed in Python 3.9 and executed on a server with a 13th Gen Intel Core i9-13900HX @ 2.20 GHz and 16-GB RAM, running Windows Server 2023 Standard 64-bit operating system. For evaluating the diversity and convergence of Pareto sets in the EADHFFSP, three metrics are used: 1) hyper volume (HV) [21]; 2) inverse generational distance (IGD) [21]; and 3) overall nondominated vector generation (ONVG) [48]. Higher values indicate better performance for HV and ONVG, while lower values indicate better performance for IGD

$$d_j = \sum_{f \in F} \sum_{j \in n} \sum_{k \in m} P_{j,k} / \left(n * n_f\right). \tag{18}$$

A practical case is presented to substantiate the performance of the devised DRLCEA in the steelmaking industry. The case involved multiple steel mills, each with a flexible flow shop that included three processing stages: 1) steelmaking; 2) refining; and 3) CC. These stages consist of unrelated parallel machines. To further substantiate the performance of the devised DRLCEA, a real case is generated from this real production scenario. The case includes two factories and 55 jobs. Every factory has three stages of four parallel machines in every stage. At the same time, the standard processing time of every job in every factory and the processing speed of every job in every stage are different.

A. Control Parameters Analysis

DRLCEA contains four important parameters: 1) population size (*Ps*); 2) the crossover rate (*Pc*); 3) the mutation rate (*Pm*); and 4) the competitive learning rate (*Pa*), population size significantly affects algorithm performance. In DRLCEA, the population size is set to $Ps = \{50, 80, 100\}$. Under the same termination criteria, a smaller population size leads to more iterations of the DRLCEA, while a larger population size implies greater diversity of the DRLCEA. The crossover rate is set to $Pc = \{0.8, 0.9, 1.0\}$. The mutation rate is set to $Pm = \{0.1, 0.2, 0.3\}$. The competitive learning rate has a significant impact on the effectiveness of global search. The competitive learning rate is set to $Pa = \{0.2, 0.25, 0.3\}$. The parameters are calibrated using an orthogonal experimental design and analysis of variance (ANOVA) [49], and 20 independent tests are performed. These tests produce Pareto sets. HV, IGD, and ONVG are the metrics used for parameter calibration. A higher value of HV and ONVG indicates a better-parameter level, while a lower value of IGD represents a better-parameter level. The ANOVA results, specifically focusing on the IGD metric, are presented in Table III.

According to Table III, the p-value of *Pa* is less than 0.05, which represents *Pa* is a critical parameter [48]. The parameter Pa has the highest-*F*-ratio value, indicating that it has the most significant influence on DRLCEA. Fig. 4 presents the main effect plot of parameters to determine their values. According to the plot, the highest-ONVG value occurs when $Ps = 50, Pc = 1.0, Pm = 0.10, Pa = 0.25$. The highest-HV value is achieved when $Ps = 80, Pc = 1.0, Pm = 0.30, Pa = 0.25$. The lowest-IGD value is obtained when $Ps = 100, Pc = 1.0, Pm = 0.20, Pa = 0.30$. Hence, the optimal parameter combination is $Ps = 100, Pc = 1.0, Pm = 0.3, Pa = 0.25$.

B. Evaluating the MILP Model of EADHFFSP

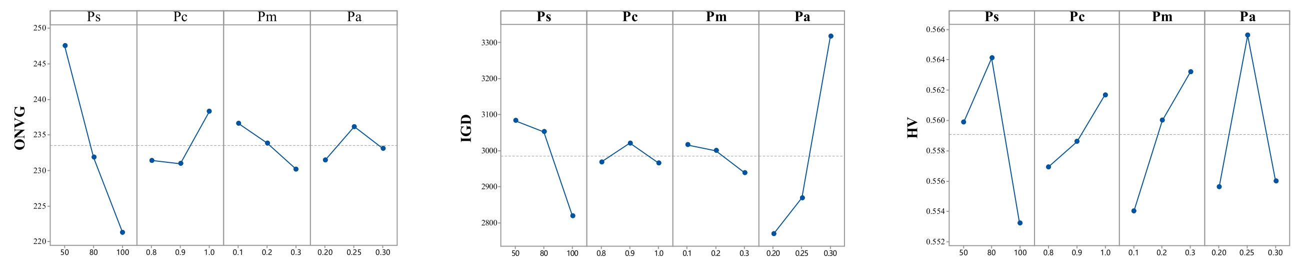
10 small-scale examples are generated to verify the accuracy of the devised EADHFFSP model. CPLEX (version 12.9) is employed to solve the ten small-scale examples. The problem size ranges from six jobs, two factories, and two stages to ten jobs, two factories, and three stages, and each stage consists 2 or 3 machines. Every instance is denoted as F_S_J, indicating the total count of factories, stages, and jobs. The maximum CPU running time of CPLEX is 1000 s, and the stop criteria of DRLCEA is Max-NEFs $= 400 \times n$, n indicates the number of jobs. Compare CPLEX with DRLCEA, the statistical results of the smallest TTD and TEC are shown in the Table S-II in the supplementary material. CPLEX achieves better results than DRLCEA in solving TTD targets because the branch-and-bound algorithm is accurately able to handle small-scale problems. For TTD, the performance of the devised DRLCEA is comparable to that of CPLEX on the first and the third instance, the performance of the proposed DRLCEA is better than CPLEX on the fifth instance. Because of the energy saving strategy devised in this article, DRLCEA gets better results than CPLEX in solving TEC targets. Thus, the results reflect the accuracy of the devised EADHFFSP model. The Table S-II also displays the results of the calculation time. From the result analysis of computation time, CPLEX requires less CPU time than DRLCEA to solve the first four problems, because the branch and bound algorithm is able to handle small-scale problems quickly. But as the size of the problem increases, it takes more time to discover the optimal solution. In addition, the CPU time of DRLCEA is relatively stable, indicating that DRLCEA is able to provide a competitive solution in less time.

TABLE III
ANOVA RESULTS OF PARAMETERS FOR DRLCEA

Source	Sum of Squares	Degress of freedom	Mean Square.	*F* - *ratio*	*p* - *value*
Ps	1127741.1	2	563870.6	1.06	0.3684
Pc	51874.4	2	25937.2	0.05	0.9524
Pm	90202	2	45101	0.09	0.9189
Pa	4628021.7	2	2314010.8	4.36	0.0307
Ps*Pc	2843024.7	4	710756.2	1.34	0.2979
Ps*Pm	1318932.6	4	329733.1	0.62	0.6535
Ps*Pa	1844230.9	4	461057.7	0.87	0.5034
Pc*Pm	2209728.9	4	552432.2	1.04	0.4164
Pc*Pa	1829831.3	4	457457.8	0.86	0.5071
Pm*Pa	376730.1	4	94182.5	0.18	0.9467
Ps*Pc*Pm	3175083.8	8	396885.5	0.75	0.6502
Ps*Pc*Pa	6687857.1	8	835982.1	1.58	0.2084
Ps*Pm*Pa	4307769.6	8	538471.2	1.02	0.4627
Pc*Pm*Pa	2594754.6	8	324344.3	0.61	0.7557
Error	8483116.5	16	530194.8		
Total	41568899.2	80			

C. Effectiveness Analysis of Strategy Composition

Five variant algorithms are designed in this part as follows to evaluate the effectiveness of every improvement part in DRLCEA: 1) DRLCEA-initial means DRLCEA without initialization based on the domain knowledge to demonstrate its effectiveness and 2) DRLCEA-ES means DRLCEA without energy saving and speed adjustment strategy to demonstrate their effectiveness; 3) DRLCEA-LS means DRLCEA without local search operators based on problem knowledge to demonstrate its effectiveness; and 4) DRLCEA-RS and DRLCEA-DDQN mean search mechanism based on random selection and search mechanism based on DDQN, the effectiveness of search mechanism based on dueling DDQN is demonstrated by these two algorithms. The termination criterion is set as Max-NEFs $= 400 \times n$, and each instance is independently executed twenty times. Table IV presents the



(a) (b) (c)

Fig. 4. Main effect plot of parameters. (a) ONVG. (b) IGD. (c) HV.

TABLE IV
AVERAGE ONVG OF FIVE VARIANTS

$n/s/f$	DRLCEA-LS	DRLCEA-DDQN	DRLCEA-ES	DRLCEA-Initial	DRLCEA-RS	DRLCEA
20/3/2	197-	315-	233-	517-	281-	**667**
20/3/3	146-	**795+**	755+	371-	772+	402
20/3/5	75-	146-	125-	143-	**162+**	155
20/5/2	248-	730-	691-	**800+**	706-	773
20/5/3	209-	425-	389-	587-	390-	**740**
20/5/5	39-	61-	53-	58-	60-	**65**
40/3/2	362-	495-	593-	776-	525-	**1204**
40/3/3	193-	633+	**641+**	456-	597-	606
40/3/5	198-	383-	385-	451-	373-	**540**
40/5/2	280-	494-	487-	**995+**	451-	926
40/5/3	245-	630-	639-	653-	630-	**666**
40/5/5	253-	365-	297-	**671+**	313-	636
60/3/2	331-	464-	451-	974-	473-	**1063**
60/3/3	209-	552-	452-	566-	431-	**759**
60/3/5	168-	495-	315-	497-	316-	**510**
60/5/2	327-	935-	439-	978-	426-	**987**
60/5/3	211-	**1075+**	508-	952-	556-	1001
60/5/5	153-	637+	340-	**735+**	507-	607
80/3/2	364-	760-	422-	**804+**	562-	795
80/3/3	287-	965-	297-	**1007+**	740-	995
80/3/5	157-	637+	276-	**796+**	400-	544
80/5/2	279-	1360-	243-	1230-	949-	**1389**
80/5/3	260-	**1034+**	361-	941-	625-	954
80/5/5	185-	744-	256-	678-	526-	**763**
100/3/2	420-	**1203+**	445-	913-	845-	1146
100/3/3	205-	699-	526-	698-	487-	**730**
100/3/5	190-	**766+**	357-	650-	587-	729
100/5/2	344-	**1455+**	389-	1409+	1040-	1336
100/5/3	217-	799-	376-	927-	606-	**971**
100/5/5	186-	817-	397-	**1173+**	605-	874
200/3/2	423-	462-	444-	1594-	478-	**2061**
200/3/3	423-	334-	370-	1501-	352-	**2071**
200/3/5	235-	916-	422-	784-	579-	**971**
200/5/2	483-	435-	450-	**1814+**	398-	1589
200/5/3	343-	410-	407-	**1347+**	382-	1144
200/5/5	208-	337-	374-	**1067+**	320-	995
realworld	109-	224-	**310+**	197-	193-	232
-/=/+	37/0/0	28/0/9	34/0/3	25/0/12	35/0/2	

TABLE V
RESULTS OF THE FRIEDMAN TEST

strategy	ONVG		HV		IGD	
	rank	p-value	rank	p-value	rank	p-value
DRLCEA-DDQN	4.20		4.42		2.79	
DRLCEA-ES	2.65		2.64		4.33	
DRLCEA-Initial	4.76	6.138E-23	1.48	6.992E-18	5.45	1.222E-16
DRLCEA-RS	2.91		4.39		2.76	
DRLCEA-LS	1.19		2.91		3.94	
DRLCEA	**5.30**		**5.15**		**1.73**	

statistical results of each algorithm. The symbols "$-$" and "$+$" indicate significantly worse and better-performance relative to DRLCEA, respectively. The symbol "$\approx$" indicates that there is no significant difference between the variant algorithm and DRLCEA. The best value for each indicator is highlighted in bold. Table V shows the results of the Friedman rank sum test with a confidence level of $\alpha = 0.05$.

The experimental results lead to the following conclusions: 1) the p-value < 0.05 indicates the superiority of DRLCEA over all variant algorithms; 2) the effectiveness of the initialization based on the knowledge is evaluated through comparing DRLCEA and DRLCEA-initial; and 3) comparing DRLCEA with DRLCEA-LS and DRLCEA-ES evaluates the effectiveness of the local search operators based on problem knowledge, energy-saving operators, and speed adjustment strategy; and 4) comparing DRLCEA with DRLCEA-DDQN and DRLCEA-RS ensure the effectiveness of the search mechanism based on dueling DDQN.

D. Comparison Experiment and Discussions

In this section, three state-of-the-art algorithms, including D2QCE [41], NSGA2-LS [37], and NSGA2 [46], are chosen as the comparison algorithms to substantiate the performance of DRLCEA in handling the problem in this article. The operators of NSGA2 come from a research work on solving multiobjective optimization problem [45]. NSGA2 is a classical multiobjective optimization algorithm. The universal crossover (UX) and POX are utilized as crossover operators. The randomly mutation and two-point swap mutation are utilized as mutation operators. The termination criteria of NSGA2 in the original paper is a maximum of 25 000 function evaluations. D2QCE and NSGA2-LS are proposed as solutions for minimizing two objectives in distributed scheduling problems. These algorithms have demonstrated superior performance compared to some state-of-the-art methods. Local intensification is added to the NSGA2 algorithm in the NSGA2-LS. The UX operator, two-point swap mutation operator, and randomly mutation operator is used as crossover and mutation operator. The termination criteria of NSGA2-LS are running time of $0.1 \times n \times m$ (seconds) in the original paper. D2QCE is a recent algorithm that combines reinforcement learning mechanism and meta-heuristic algorithm. UX and POX are used as crossover operators. The randomly mutation and two-point swap mutation are utilized as mutation operators. The termination criteria of D2QCE in the original paper is MaxNFEs $= 400 \times n$. The parameter settings for the comparison algorithms align with those specified in [37] and [41]. To

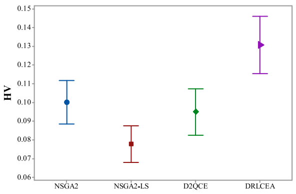


Fig. 5. Interval plot of HV of four algorithms in 95% confidence intervals.

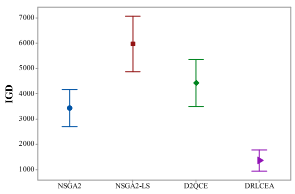


Fig. 6. Interval plot of IGD of four algorithms in 95% confidence intervals.

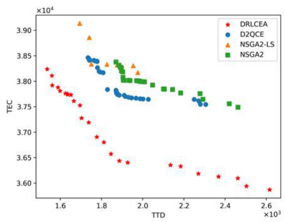


Fig. 7. Pareto front on instance $f = 3$, $s = 5$, and $n = 40$.

ensure fairness, the termination criteria is Max-NEFs $= 400 \times n$, and each instance is run twenty times independently.

Table S-III in the supplementary material presents the mean values of HV, IGD, and ONVG for all instances, with the optimal values highlighted in bold. According to the table, DRLCEA consistently achieves higher-HV values than the comparison algorithms across all instances. HV serves as a comprehensive indicator, where a higher value signifies a Pareto set with better convergence and diversity. The interval plot in Fig. 5 displays the 95% confidence interval for the four algorithms. It is observed that the HV values of DRLCEA, across different factory numbers, are generally larger than those of all the comparison algorithms.

Table S-III reveals that DRLCEA consistently achieves lower-IGD values than the comparison algorithms across all instances. Additionally, the confidence interval span of DRLCEA is smaller than that of all the comparison algorithms. A low value of IGD indicates that the Pareto set obtained by DRLCEA exhibits better convergence, diversity, spread, uniformity, and cardinality. Fig. 6 depicts the interval plot of the four algorithms based on the 95% confidence interval. It is discovered that DRLCEA consistently outperforms the comparison algorithms in terms of IGD. Furthermore, Table S-III indicates that DRLCEA attains higher-ONVG values than the comparison algorithms in approximately all instances. A larger ONVG value for DRLCEA signifies a higher number of nondominated solutions compared to the other algorithms. This implies that DRLCEA exhibits better effectiveness and stability than all the comparison algorithms.

According to the results shown in Figs. 7–12, the approximate distribution of the nondominated solutions is acquired

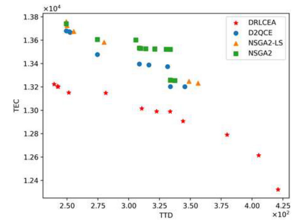


Fig. 8. Pareto front on instance $f = 2$, $s = 3$, and $n = 55$.

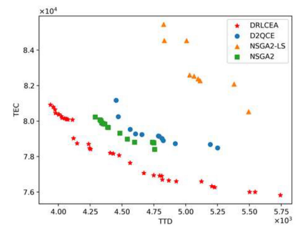


Fig. 9. Pareto front on instance $f = 5$, $s = 5$, and $n = 80$.

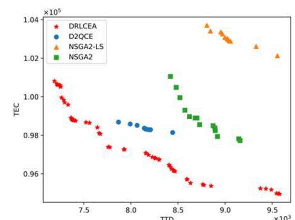


Fig. 10. Pareto front on instance $f = 5$, $s = 5$, and $n = 100$.

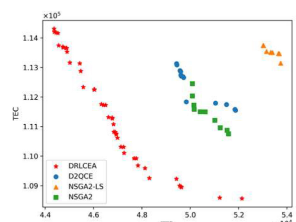


Fig. 11. Pareto front on instance $f = 5$, $s = 3$, and $n = 200$.

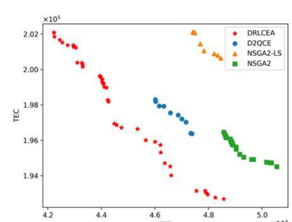


Fig. 12. Pareto front on instance $f = 5$, $s = 5$, and $n = 200$.

by the four algorithms in all cases. The experimental results are also shown in Figs. 7–12, and it is discovered that the Pareto optimal solution acquired by DRLCEA is superior to the comparison algorithm. These results clearly show that DRLCEA is an effective algorithm for solving EADHFFSP.

Discussions: From Table S-III and Figs. 5–12, DRLCEA exhibits better than all the comparison algorithms in all metrics. Comparing to NSGA2, D2QCE, and NSGA2-LS, the reason why DRLCEA performs better in HV and ONVG metric is that many computation resources are allocated to global search to obtain high-diversity solutions in the proposed co-evolution algorithm. Comparing to NSGA2, D2QCE, and NSGA2-LS, the reason why DRLCEA performs better in IGD metric is that thirteen problem-specific local search operators are devised to search elite solutions to obtain high-quality local searching. The efficiency of local search is enhanced and invalid search is avoided because of the search mechanism using dueling DDQN. The speed adjustment and energy-saving strategies is reducing TTD and TEC to enhance the convergence of DRLCEA. Compared to NSGA2 and NSGA2-LS, DRLCEA performs better because many computation resources are allocated to global search and thirteen problem-specific local search operators are devised to search elite solutions, and two heuristic initialization methods of DRLCEA obtain high-quality than NSGA2 and NSGA2-LS. However, NSGA2 and NSGA2-LS lack these two strategies. Compared to D2QCE, DRLCEA performs better because D2QCE utilizes the search mechanism using DDQN and the search mechanism using dueling DDQN of DRLCEA learns more efficiently than DDQN, and the speed adjustment and energy-saving strategies of DRLCEA obtain better solutions as Pareto solutions than D2QCE. Thus, DRLCEA is an effective algorithm for solving EADHFFSP.

E. Nonparametric Tests

By using Friedman test, the statistical influence of the DRLCEA algorithm is presented in this article. The results of the Friedman test are displayed in Figs. S-4–S-6 in the supplementary material. and analyzed using the confidence level $\alpha = 0.05$. ONVG, HV and IGD metrics are utilized to evaluate the algorithm performance. In Figs. S-4-S-6, the solid and dashed lines represent the critical difference (CD) between the 95% and 90% confidence intervals. Compared to the other three algorithms, DRLCEA is the only algorithm whose rank sum is above the real and dashed lines, which means that DRLCEA significantly outperforms these three comparison algorithms on the 95% and 90% confidence intervals. In summary, the experimental results clearly display that DRLCEA is superior to the comparison algorithms in handling EADHFFSP problem.

F. Comparisons on Instances Under Production Scenario of Aluminum Industry

Nonferrous metallurgy industry is the pillar industry in Northwest China. Fig. S-7 in the supplementary material shows the common technological processes of aluminum production in nonferrous metallurgy industry, including electrolysis, casting, cold rolling, and hot rolling. During the electrolysis process, pure aluminum is extracted from alumina in multiple parallel electrolytic cells. In the CC process, liquid aluminum is processed on multiple parallel CC machines, and each order needs to be completed as early as possible in order to avoid solidification. The liquid aluminum of the rolling process is also processed on multiple parallel machines. In the production process, there are usually two conflicting objectives involved, equipment utilization and energy efficiency. Minimizing total delay is similar to improving equipment utilization and customer satisfaction, and minimizing TEC is similar to improving energy efficiency [50]. Therefore, DRLCEA is used to solve the scheduling flow of aluminum production with the objective of minimizing the total delay and the TEC.

Eighteen instances are generated to further verify the stability and generality of DRLCEA. The number of factories F is from $\{2, 3, 5\}$, the number of jobs n is from $\{20, 40, 60, 80, 100, 200\}$, and the number of stages m is from $\{4\}$. The rest of the settings are the same as before. A real case realworld1 is generated from the aluminum production. The case includes two factories and 55 jobs. Every factory has four stages with four parallel machines in every stage. At the same time, the standard processing time of every job in every factory and the processing speed of every job in every stage are different.

As in the above experiment, comparative experiments are conducted on these instances. The interval plot of HV, IGD, and ONVG metric are shown in Fig. S-8–S-10 in the supplementary material. The average HV, IGD, and ONVG metric of four algorithms are shown in Table S-IV–S-VI in the supplementary material. The Pareto front on realworld1 instance is shown in Fig. S-11. In the HV, IGD, and ONVG metric, the results of Friedman test are given in Fig. S-12-S-14. These results clearly show that DRLCEA is an effective algorithm for handle the aluminum production.

G. Robustness Experiment and Resilience Experiment

In order to verify the robustness [51] of the algorithm, experimental tests are carried out on all instances. The IGD is selected as the evaluation metric. The result is shown in Fig. S-15 in the supplementary material. Compared with the other three algorithms, the mean value of the IGD obtained by DRLCEA is lower. In DRLCEA, the appearance of a single peak indicates that the results are concentrated and uniform, thus verifying the robustness of the algorithm. Resilience embodies a fundamental attribute of a system, signifying its ability to endure and persist amidst adversity or damage [52]. The resilience ratio is imported and systematically measured in the following manner:

$$r_A\left(I, I^D\right) = \frac{\left\|\mathrm{ob}\left(s(A, I^D), I^D\right)\right\|}{\left\|\mathrm{ob}(s(A, I), I)\right\|} \tag{19}$$

where $\mathrm{ob}(s(A, I^D), I^D)$ represents the objective outcome achieved when action A is reimplemented under the influence of disruptive scenario I^D. Subsequently, the resilience ratio of algorithm A for instance I, in the context of damage within the range $X(I)$, is formally clarified as follows:

$$\begin{aligned} R_A(I) = \max\bigl\{ r_A\left(I, I^D\right) | \text{ for all } I^D \text{ with} \\ \left(P(I, I^D), E(I, I^D), S(I, I^D)\right) \in X(I) \bigr\} \end{aligned} \tag{20}$$

Wherein the modifications stemming from variations in parameters, the number of variables, and the inherent structure of the problem are denoted, respectively, as $P(I, I^D), E(I, I^D), S(I, I^D)$.

Static scheduling refers to the situation where it is already known that the factory will be unable to operate normally prior to the scheduling process. The resilience experiment is designed without considering dynamic events. The factory will be brought to a standstill by the failure of any machine. The resilience of the algorithm is impacted by the occurrence of a failure in a particular factory. The destruction is caused by the number of variables. To maintain the heterogeneity of the EADHFFSP, the single-plant case is not considered. There are eighteen cases caused by damage in one factory. The numbers of instance are listed in order F_S_J from smallest to largest. HV is the metric result of algorithm DRLCEA considering EADHFFSP. The experimental results are shown in Fig. S-16 in the supplementary material.

$R_{\text{DRLCEA}}(I) = \max\{r_{\text{DRLCEA}}(I, I^D)\} = 1.30403$ is shown in Fig. S-16. In addition, most of the resilience ratios tend to plateau. Fig. S-16 shows that DRLCEA is resilient to the damage of one factory in a heterogeneous plant system. Results below the 1.0 line indicate a factory fault causing a small HV value. However, the minimum value of $r_{\text{DRLCEA}} = 0.5947$ indicates that DRLCEA is able to reduce HV by 40.53% at most in the case of one factory damage in the factory under the heterogeneous system.

VI. Conclusion and Future Work

In this article, DRLCEA is devised to minimize TTD and TEC for EADHFFSP in industrial-scale steelmaking and CC. The experimental results display that compared withNSGA2, NSGA2-LS and D2QCE, DRLCEA has significant advantages in solving EADHFFSP problems. The hybrid heuristic initialization operation is introduced as a presented initialization method, which produces better-initial candidate solutions than random initialization. Through global search based on adversarial generative learning, DRLCEA effectively improves the diversity and quality of understanding. In addition, DRLCEA adopts dueling DDQN as a proposed operator selection method, which gets the coevolutionary algorithm to select the best operator and effectively accelerate the convergence speed. Dueling DDQN provides effective empirical guidance for DRLCEA, which selects local search operators to avoid invalid searches accurately. Finally, knowledge-based speed regulation strategies and energy saving strategies further optimize the TTD and TEC goals. In summary, DRLCEA algorithm has great application potential in industrial scale steelmaking, CC process and similar practical scheduling problems.

In the future, the policy gradient and the learning mechanism of the graph neural network will be combined with the co-evolutionary meta-heuristic at the algorithm level. Further research will be conducted on related energy indicators at the problem level, including electricity costs, carbon footprint, and pollutant emissions, among other objectives. Redesign the appropriate strategy to apply co-evolution algorithm to the study of supply chain systems [53] in the era of globalization and big data [54]. Moreover, considering hybrid power supply system [55] is a future research work.

References

[1] M. Li and G.-G. Wang, "A review of green shop scheduling problem," *Inf. Sci.*, vol. 589, pp. 478–496, Apr. 2022.

[2] Y. Fu, M. Zhou, X. Guo, L. Qi, K. Gao, and A. Albeshri, "Multiobjective scheduling of energy-efficient stochastic hybrid open shop with brain storm optimization and simulation evaluation," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 54, no. 7, pp. 4260–4272, Jul. 2024.

[3] W. Wang, G. Tian, H. Zhang, Z. Li, and L. Zhang, "A hybrid genetic algorithm with multiple decoding methods for energy-aware remanufacturing system scheduling problem," *Robot. Comput.-Integr. Manuf.*, vol. 81, Jun. 2023, Art. no. 102509.

[4] W. Shao, Z. Shao, and D. Pi, "Effective constructive heuristics for distributed no-wait flexible flow shop scheduling problem," *Comput. Oper. Res.*, vol. 136, Dec. 2021, Art. no. 105482.

[5] R. Ruiz and J. A. Vázquez-Rodriguez, "The hybrid flow shop scheduling problem," *Eur. J. Oper. Res.*, vol. 205, no. 1, pp. 1–18, 2010.

[6] W. Shao, Z. Shao, and D. Pi, "A multi-neighborhood-based multi-objective memetic algorithm for the energy-efficient distributed flexible flow shop scheduling problem," *Neural Comput. Appl.*, vol. 34, no. 24, pp. 22303–22330, 2022.

[7] Y. Pan, K. Gao, Z. Li, and N. Wu, "A novel evolutionary algorithm for scheduling distributed no-wait flow shop problems," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 54, no. 6, pp. 3694–3704, Jun. 2024.

[8] W. Shao, Z. Shao, and D. Pi, "An ant colony optimization behavior-based MOEA/D for distributed heterogeneous hybrid flow shop scheduling problem under nonidentical time-of-use electricity tariffs," *IEEE Trans. Autom. Sci. Eng.*, vol. 19, no. 4, pp. 3379–3394, Oct. 2022.

[9] S.-L. Jiang, C. Xu, L. Zhang, and Y. Ma, "A decomposition-based two-stage online scheduling approach and its integrated system in the hybrid flow shop of steel industry," *Expert Syst. Appl.*, vol. 213, Mar. 2023, Art. no. 119200.

[10] J. N. D. Gupta, "TWO-stage, hybrid flowshop scheduling problem," *J. Oper. Res. Soc.*, vol. 39, no. 4, pp. 359–364, 1988.

[11] B. Naderi and R. Ruiz, "The distributed permutation flowshop scheduling problem," *Comput. Oper. Res.*, vol. 37, no. 4, pp. 754–768, 2010.

[12] J. Duan, M. Feng, and Q. Zhang, "Energy-efficient collaborative scheduling of heterogeneous multi-stage hybrid flowshop for large metallic component manufacturing," *J. Clean. Prod.*, vol. 375, Nov. 2022, Art. no. 134148.

[13] D. H. Wolpert and W. G. Macready, "No free lunch theorems for search," Santa Fe Inst., Santa Fe, NM, USA, Rep. SFI-TR-95-02-010, 1995.

[14] B. Han, W. Zhang, X. Lu, and Y. Lin, "Online supply chain scheduling for single-machine and parallel-machine configurations with a single customer: Minimizing the makespan and delivery cost," *Eur. J. Oper. Res.*, vol. 244, no. 3, pp. 704–714, 2015.

[15] R. Marti, M. Sevaux, and K. Sorensen, "Fifty years of metaheuristics," *Eur. J. Oper. Res.*, vol. 321, no. 2, pp. 345–362, 2024.

[16] Y. Li, X. Li, and L. Gao, "An effective solution space clipping-based algorithm for large-scale permutation flow shop scheduling problem," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 53, no. 1, pp. 635–646, Jan. 2023.

[17] F. Zhao, G. Zhou, and L. Wang, "A cooperative scatter search with reinforcement learning mechanism for the distributed permutation flowshop scheduling problem with sequence-dependent setup times," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 53, no. 8, pp. 4899–4911, Aug. 2023.

[18] Y. Huang, G. Wei, and Y. Wang, "V-D D3QN: The variant of double deep *Q*-learning network with dueling architecture," in *Proc. 37th Chin. Control Conf. (CCC)*, Wuhan, China, 2018, pp. 9130–9135.

[19] H. E. Nouri, A. Ghandri, O. B. Driss, and K. Ghedira, "Bi-EvoGAN: Bi-level evolutionary approach for generative adversarial networks," *Appl. Soft Comput.*, vol. 147, Nov. 2023, Art. no. 110738.

[20] W. Shao, Z. Shao, and D. Pi, "Multi-objective evolutionary algorithm based on multiple neighborhoods local search for multi-objective distributed hybrid flow shop scheduling problem," *Expert Syst. Appl.*, vol. 183, Nov. 2021, Art. no. 115453.

[21] J.-Q. Li, X.-L. Chen, P.-Y. Duan, and J.-H. Mou, "KMOEA: A knowledge-based multiobjective algorithm for distributed hybrid flow shop in a prefabricated system," *IEEE Trans. Ind. Informat.*, vol. 18, no. 8, pp. 5318–5329, Aug. 2022.

[22] C. Lu, J. Zhou, L. Gao, X. Li, and J. Wang, "Modeling and multi-objective optimization for energy-aware scheduling of distributed hybrid flow-shop," *Appl. Soft Comput.*, vol. 156, May 2024, Art. no. 111508.

[23] F. Yu, C. Lu, J. Zhou, and L. Yin, "Mathematical model and knowledge-based iterated greedy algorithm for distributed assembly hybrid flow shop scheduling problem with dual-resource constraints," *Expert Syst. Appl.*, vol. 239, Apr. 2024, Art. no. 122434.

[24] F. Liu, G. Li, C. Lu, L. Yin, and J. Zhou, "A tri-individual iterated greedy algorithm for the distributed hybrid flow shop with blocking," *Expert Syst. Appl.*, vol. 237, Mar. 2024, Art. no. 121667.

[25] W. Shao, Z. Shao, and D. Pi, "A network memetic algorithm for energy and labor-aware distributed heterogeneous hybrid flow shop scheduling problem," *Swarm Evol. Comput.*, vol. 75, Dec. 2022, Art. no. 101190.
[26] D. Lei and B. Su, "A multi-class teaching–learning-based optimization for multi-objective distributed hybrid flow shop scheduling," *Knowl. Based Syst.*, vol. 263, Mar. 2023, Art. no. 110252.
[27] J.-j. Wang and L. Wang, "A bi-population cooperative memetic algorithm for distributed hybrid flow-shop scheduling," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 5, no. 6, pp. 947–961, Dec. 2021.
[28] Z. Shao, W. Shao, and D. Pi, "LS-HH: A learning-based selection hyper-heuristic for distributed heterogeneous hybrid blocking flow-shop scheduling," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 7, no. 1, pp. 111–127, Feb. 2023.
[29] H.-X. Qin, Y.-Y. Han, Y.-P. Liu, J.-Q. Li, Q.-K. Pan, and H. Xue, "A collaborative iterative greedy algorithm for the scheduling of distributed heterogeneous hybrid flow shop with blocking constraints," *Expert Syst. Appl.*, vol. 201, Sep. 2022, Art. no. 117256.
[30] W. Shao, Z. Shao, and D. Pi, "Modelling and optimization of distributed heterogeneous hybrid flow shop lot-streaming scheduling problem," *Expert Syst. Appl.*, vol. 214, Mar. 2023, Art. no. 119151.
[31] H. Xuan, W. Li, and B. Li, "An artificial immune differential evolution algorithm for scheduling a distributed heterogeneous flexible flowshop," *Appl. Soft Comput.*, vol. 145, Sep. 2023, Art. no. 110563.
[32] C. Lu, Q. Liu, B. Zhang, and L. Yin, "A Pareto-based hybrid iterated greedy algorithm for energy-efficient scheduling of distributed hybrid flowshop," *Expert Syst. Appl.*, vol. 204, Oct. 2022, Art. no. 117555.
[33] W. Zhang, J. Shi, S. Zhang, and M. Chen, "Scenario-based robust remanufacturing scheduling problem using improved biogeography-based optimization algorithm," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 53, no. 6, pp. 3414–3427, Jun. 2023.
[34] L. Wang, K. Gao, Z. Lin, W. Huang, and P. N. Suganthan, "Problem feature based meta-heuristics with *Q*-learning for solving urban traffic light scheduling problems," *Appl. Soft Comput.*, vol. 147, Nov. 2023, Art. no. 110714.
[35] W. Lin, P. Babyn, Y. yan, and W. Zhang, "A novel scheduling method for reduction of both waiting time and travel time of patients to visit health care units in the case of mobile communication," *Enterprise Inf. Syst.*, vol. 17, no. 12, 2023, Art. no. 2188124.
[36] F. Zhao, X. He, and L. Wang, "A two-stage cooperative evolutionary algorithm with problem-specific knowledge for energy-efficient scheduling of no-wait flow-shop problem," *IEEE Trans. Cybern.*, vol. 51, no. 11, pp. 5291–5303, Nov. 2021.
[37] J.-J. Wang and L. Wang, "A cooperative memetic algorithm with learning-based agent for energy-aware distributed hybrid flow-shop scheduling," *IEEE Trans. Evol. Comput.*, vol. 26, no. 3, pp. 461–475, Jun. 2022.
[38] H. Li, K. Gao, P.-Y. Duan, J.-Q. Li, and L. Zhang, "An improved artificial bee colony algorithm with *Q*-learning for solving permutation flow-shop scheduling problems," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 53, no. 5, pp. 2684–2693, May 2023.
[39] M. Karimi-Mamaghan, M. Mohammadi, P. Meyer, A. M. Karimi-Mamaghan, and E.-G. Talbi, "Machine learning at the service of meta-heuristics for solving combinatorial optimization problems: A state-of-the-art," *Eur. J. Oper. Res.*, vol. 296, no. 2, pp. 393–422, 2022.
[40] H. Yu, K. Gao, N. Wu, M. Zhou, P. N. Suganthan, and S. Wang, "Scheduling multiobjective dynamic surgery problems via *Q*-learning-based meta-heuristics," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 54, no. 6, pp. 3321–3333, Jun. 2024.
[41] R. Li, W. Gong, L. Wang, C. Lu, Z. Pan, and X. Zhuang, "Double DQN-based coevolution for green distributed heterogeneous hybrid flowshop scheduling with multiple priorities of jobs," *IEEE Trans. Autom. Sci. Eng.*, vol. 21, no. 4, pp. 6550–6562, Oct. 2024.
[42] Y. Yao, J. He, T. Li, Y. Wang, X. Lan, and Y. Li, "An automatic XSS attack vector generation method based on the improved dueling DDQN algorithm," *IEEE Trans. Depend. Secure Comput.*, vol. 21, no. 4, pp. 2852–2868, Jul./Aug. 2024.
[43] H. Song, Y. Liu, J. Zhao, J. Liu, and G. Wu, "Prioritized replay dueling DDQN based grid-edge control of community energy storage system," *IEEE Trans. Smart Grid*, vol. 12, no. 6, pp. 4950–4961, Nov. 2021.
[44] J. Yan, J. Xiao, and X. Hong, "Dueling-DDQN based virtual machine placement algorithm for cloud computing systems," in *Proc. IEEE/CIC Int. Conf. Commun. China, (ICCC)*, Xiamen, China, 2021, pp. 294–299.
[45] R. Li, W. Gong, L. Wang, C. Lu, and C. Dong, "Co-evolution with deep reinforcement learning for energy-aware distributed heterogeneous flexible job shop scheduling," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 54, no. 1, pp. 201–211, Jan. 2024.
[46] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.
[47] J. Lin, Y.-Y. Li, and H.-B. Song, "Semiconductor final testing scheduling using *Q*-learning based hyper-heuristic," *Expert Syst. Appl.*, vol. 187, Jan, 2022, Art. no. 115978.
[48] F. Zhao, S. Di, and L. Wang, "A hyperheuristic with *Q*-learning for the multiobjective energy-efficient distributed blocking flow shop scheduling problem," *IEEE Trans. Cybern.*, vol. 53, no. 5, pp. 3337–3350, May 2023.
[49] W.-H. Chen et al., "A comprehensive review of thermoelectric generation optimization by statistical approach: Taguchi method, analysis of variance (ANOVA), and Response Surface Methodology (RSM)," *Renew. Sustain. Energy Rev.*, vol. 169, Nov. 2022, Art. no. 112917.
[50] R. Li, W. Gong, C. Lu, and L. Wang, "A learning-based memetic algorithm for energy-efficient flexible job-shop scheduling with type-2 fuzzy processing time," *IEEE Trans. Evol. Comput.*, vol. 27, no. 3, pp. 610–620, Jun. 2023.
[51] Y. Han, D. Gong, Y. Jin, and Q. Pan, "Evolutionary multiobjective blocking lot-streaming flow shop scheduling with machine breakdowns," *IEEE Trans. Cybern.*, vol. 49, no. 1, pp. 184–197, Jan. 2019.
[52] B. Han, C. L. Liu, and W. J. Zhang, "A method to measure the resilience of algorithm for operation management," *IFAC-PapersOnLine*, vol. 49, no. 12, pp. 1442–1447, 2016.
[53] J. Wang, R. Dou, R. R. Muddada, and W. Zhang, "Management of a holistic supply chain network for proactive resilience: Theory and case study," *Comput. Ind. Eng.*, vol. 125, pp. 668–677, Nov. 2018.
[54] H. Y. Yu, A. Ogbeyemi, W. J. Lin, J. He, W. Sun, and W. J. Zhang, "A semantic model for enterprise application integration in the era of data explosion and globalisation," *Enterprise Inf. Syst.*, vol. 17, no. 4, 2023, Art. no. 1989495.
[55] W. J. Zhang, P. R. Ouyang, and Z. H. Sun, "A novel hybridization design principle for intelligent mechatronics systems," in *Proc. 5th Int. Conf. Adv. Mechatronics (ICAM)*, 2010, pp. 67–74.

Fuqing Zhao received the B.Sc. and Ph.D. degrees in control theory and control engineering from Lanzhou University of Technology, Lanzhou, China, in 1994 and 2006, respectively.

Since 1998, he has been with the School of Computer Science Department, Lanzhou University of Technology, where he became a Full Professor in 2012. He has been as the Postdoctoral Fellow with the State Key Laboratory of Manufacturing System Engineering, Xi'an Jiaotong University, Xi'an, China, in 2009. He has been a Visiting Scholar with Exeter Manufacturing Enterprise Center, Exeter University, Exeter, U.K., in 2008 and 2019, respectively, and with Georgia Tech Manufacturing Institute, Atlanta, GA, USA, and Georgia Institute of Technology, Atlanta, in 2014 and 2015, respectively. He has authored two academic book and over 90 refereed papers. His current research interests include intelligent optimization and scheduling.

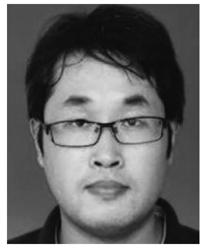
Fumin Yin received the B.Eng. degree in computer science and engineering from Northwest Minzu University, Lanzhou, China, in 2022. He is currently pursuing the M.Eng. degree with Lanzhou University of Technology, Lanzhou.

His research interests include intelligent optimization, manufacturing scheduling, and reinforcement learning.

Ling Wang (Member, IEEE) received the B.Sc. degree in automation and the Ph.D. degree in control theory and control engineering from Tsinghua University, Beijing, China, in 1995 and 1999, respectively.

Since 1999, he has been with the Department of Automation, Tsinghua University, where he became a Full Professor in 2008. He has authored five academic books and more than 300 refereed papers. His current research interests include intelligent optimization and production scheduling.

Prof. Wang is a recipient of the National Natural Science Fund for Distinguished Young Scholars of China, the National Natural Science Award (Second Place) in 2014, the Science and Technology Award of Beijing City in 2008, the Natural Science Award (First Place in 2003, and the Second Place in 2007) nominated by the Ministry of Education of China. He is currently the Editor-in-Chief of *International Journal of Automation and Control*, and the Associate Editor of IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION and *Swarm and Evolutionary Computation*.



Yang Yu (Member, IEEE) received the B.S. degree in control theory and control engineering from Jiangsu Normal University, Xuzhou, China, in 2003, the M.S. degree from Lanzhou University, Lanzhou, China, in 2006, and the Ph.D. degree from Shinshu University, Matsumoto, Japan, in 2013.

Since 2013, he has been an Associate Professor with the Department of Electric Information Engineering, Jiangsu University of Technology, Changzhou, China. His current research interests include coding and modulation for mobile communication systems, wireless sensor networks, and signal processing.