Memetic Computing (2021) 13:185–202
https://doi.org/10.1007/s12293-021-00329-6

REGULAR RESEARCH PAPER

Integrated production and distribution scheduling in distributed hybrid flow shops

Hu Qin[1] · Tao Li[1] · Yi Teng[2] · Kai Wang[3]

Received: 10 September 2020 / Accepted: 22 March 2021 / Published online: 23 April 2021

Abstract
To improve the operational efficiency and competitive advantage of supply chains, integrated production and distribution has attracted an increasing attention in recent years. This paper focuses on a novel integrated production and distribution scheduling problem (IPDSP) with consideration of factory eligibility and third-party logistics (3PL). In this problem, products are firstly produced in a number of distributed hybrid flow shops (HFS) and then delivered to a customer in batches. To satisfy the production and distribution practice, some products can only be manufactured in a subset of distributed HFSs, and the transportation of some finished products is outsourced to a 3PL provider. Considering the NP-hardness of IPDSP, three fast heuristics (CR-based heuristic, SLACK-based heuristic, and EDD-based heuristic) and an adaptive human-learning-based genetic algorithm (AHLBGA) are proposed to minimize the sum of earliness, tardiness and delivery costs. Motivated by human learning behaviours, AHLBGA integrates an adaptive learning operator with traditional genetic operators to generate candidate solutions. Such learning operator performs social learning, family learning, and individual random learning to improve offspring individuals. The computational experiments on small-sized and large-sized test problems show the superiority of AHLBGA.

Keywords Production and distribution · Scheduling · Distributed hybrid flow shop · Factory eligibility · Third-party logistics · Genetic algorithm

1 Introduction

Production and distribution are two fundamental and interrelated operations in supply chain management. To improve the operational performance of supply chains, it is critical to schedule these two operations in an integrated manner, which has received much attention from both industrial and academic communities [1, 29, 38]. In the integrated production and distribution systems, the manufacturer firstly processes the customer orders on the machines, and then timely distributes the finished products back to the customers [13, 14, 20]. To reduce the transportation cost from the manufacturer to its customers, the finished products are generally grouped into batches for delivery. A batch is a set of orders or finished products that are delivered in the same vehicle [47, 49]. However, to meet customer's requirements under limited warehouse capacity, products need to arrive at the customers within tight delivery time windows, inevitably requiring frequent delivery of small batches and eventually leading to high transportation cost [22, 42]. Therefore, a trade-off between customer service and delivery performance has to be considered when integrating production and distribution decisions.

The hybrid flow shop (HFS) is a common manufacturing system to produce identical or similar products, especially in industries such as automobile, steel, electronics, semiconductors, textiles and food [39, 41]. In a typical HFS,

✉ Kai Wang
kai.wang@whu.edu.cn

Hu Qin
tigerqin@hust.edu.cn

Tao Li
1065362528@qq.com

Yi Teng
tengyida@126.com

[1] School of Management, Huazhong University of Science and Technology, Wuhan, China

[2] School of Computer Science, Guangdong University of Education, Guangzhou, China

[3] Economics and Management School, Wuhan University, Wuhan 430072, China

machines are arranged into a number of stages in series, each of which has several functionally identical machines. All jobs released to an HFS have to pass through all stages in the same order [44]. To enhance competitiveness and responsiveness in rapidly changing markets, factories have shifted from a centralised to a more decentralized structure [5, 36]. Although an increasing research interest has recently focused on production scheduling in multi-factory environments, relatively few studies have been devoted to addressing the distributed HFS scheduling problems (DHFSSP). Given that the DHFSSP has been proven NP-hard in nature [54], integration of production and distribution scheduling in distributed HFS environments further aggravates its complexity.

As an increasing number of manufacturers adopt the make-to-order (MTO) policy to response to changing market demand, considerable amount of research works have been devoted to the integrated production and distribution scheduling problems (IPDSP) in a variety of manufacturing environments [11, 12, 33], mainly including single machine, parallel machine, and flow shop systems. The earliest research on IPDSP was conducted by Potts [37]. He considered a single machine scheduling problem with non-identical job release times and delivery times. Chang et al. [9] addressed a novel supply chain scheduling problem, in which the production and distribution of products were modelled as an identical parallel machine scheduling problem and a capacitated vehicle routing problem respectively. Considering the speed difference between machines, Guo et al. [21] focused on an unrelated parallel machine scheduling problem with batch delivery. In this problem, several transportation modes with different vehicle capacities and transportation times were considered. Hassanzadeh et al. [24] investigated a multi-objective supply chain scheduling problem, in which jobs are firstly processed in a permutation flow shop and then transported to the customers in batches with unlimited capacity. More recently, to deal with IPDSP in the assembly flow shop environments, Kazemi et al. [30] and Basir et al. [4] developed effective approaches to coordinate production and distribution operations. A search of available literature on IPDSP indicates that no research works have attempted to integrate production and distribution scheduling in a distributed HFS environment.

Exact algorithms, heuristics and meta-heuristics are three commonly used approaches to address the IPDSP in the literature. Exact algorithms, such as the dynamic programming [9, 23], branch-and-bound approach [32, 43], and branch-and-price approach [3], have been applied to generate the optimal solution for small-sized instances. However, owing to high complexity of IPDSP, exact algorithms are inapplicable to deal with medium-sized and large-sized instances in industrial practice. Since heuristics and meta-heuristics are capable of obtaining near-optimal solutions within reasonable computational time [2, 7, 8], recent research efforts have shifted from developing exact algorithms to efficient heuristics and meta-heuristics, mainly including genetic algorithm (GA) [4, 24, 28], artificial immune system (AIS) [26], particle swarm optimization (PSO) [40], ant colony optimization (ACO) [13, 14, 19], imperialist competitive algorithm (ICA) [30, 31], iterated greedy algorithm [52], and hybrid approaches [27, 46, 55].

This paper aims to address a novel IPDSP, in which products are first produced in a number of distributed HFSs and then delivered to a customer in batches. To satisfy the production and distribution practice, factory eligibility and multiple transportation modes are considered in the studied problem. In many real-world manufacturing systems, a job can only be processed on a subset of machines owing to some technical or physical reasons [17, 39]. This constraint, known as machine eligibility in the literature, has recently received much attention in the parallel machine and HFS scheduling problems [6, 56]. As the extension of machine eligibility, factory eligibility is considered in the studied IPDSP, which does not allow a job to be assigned to a factory if it cannot be processed on any parallel machines at some stage in the factory [53]. In addition, as many manufacturers have difficulty in providing sufficient transportation facilities to meet the customer's demand, finished products are allowed to be delivered by both the manufacturer and the third-party logistics (3PL) provider. Therefore, two types of transportation modes, namely manufacturer transportation and 3PL transportation, are considered in the studied problem. The transportation cost by a vehicle of the manufacturer is fixed, whereas the transportation cost by a vehicle of the 3PL provider consists of fixed charge cost and a load-based transportation cost. To our best knowledge, it is the first attempt to integrate production and distribution scheduling in distributed HFSs with consideration of factory eligibility and multiple transportation modes.

Owing to the NP-hardness of the studied IPDSP, three fast heuristics (CR-based, SLACK-based, and EDD-based heuristics) and an adaptive human-learning-based genetic algorithm (AHLBGA) are proposed to obtain sub-optimal solutions in this paper. Each of the three heuristics integrates a simple scheduling rule with a job re-assignment method and a vehicle selection method to generate integrated production and distribution schedules. The uniqueness of the proposed AHLBGA is characterized by hybridizing GA with an adaptive human-learning operator. Apart from commonly used genetic operators of selection, crossover, and mutation, learning is another important evolutionary process to acquire new knowledge for enhancing competitive advantages of species. To improve the performance of offspring individuals after crossover and mutation, AHLBGA applies a novel learning operator to mimic human learning behaviours. This learning operator consists of three typical activities, namely

social learning, family learning, and individual random learning. Furthermore, to provide a better trade-off between exploration and exploitation, the selection probability of different learning activities is dynamically determined by an adaptive learning scheme. A comprehensive search of literature on IPDSP indicates that no research works have attempted to incorporate GA and human learning activities to generate integrated schedules. Since distributed HFSs are typical production systems in many MTO industries, the proposed AHLBGA may help operations managers to better coordinate the scheduling of production and transportation.

The rest of paper is organized as follows. Considering factory eligibility and multiple transportation modes, Sect. 2 formulates a novel IPDSP in a distributed HFS environment. Section 3 presents three fast heuristics and AHLBGA to generate integrated production and distribution schedules. To evaluate the performance of proposed algorithms, experimental results on small-sized and large-sized problems are presented in Sect. 4. Finally, Sect. 5 summarises the research findings and identifies some future research directions.

2 Mathematical model

2.1 Problem description and assumptions

The IPDSP in distributed HFSs is a novel combinatorial optimization problem. As indicated in Fig. 1, this problem consists of two inter-dependent sub-problems, namely distributed HFS scheduling with factory eligibility in the production stage and batch delivery of finished products with two types of transportation modes in the distribution stage.

In the production stage, a set $J=\{1, 2, \ldots, n\}$ of n jobs are released to f distributed identical factories for processing. Each of the factories is arranged as an HFS, which has a set $S=\{1, 2, \ldots, t\}$ of t stages and m_k parallel machines at each stage k. The transportation time between stages in a factory is negligible and therefore not considered in this study. Owing to the technical or physical aspects, factory eligibility is considered in the production stage, in which some jobs can only be processed in a subset of distributed factories.

In the distribution stage, the finished products in each of the distributed factories are grouped into batches and delivered to a customer by the vehicles of the manufacturer and the 3PL provider. The manufacturer only provides a limited number of vehicles for transportation, whereas the 3PL provider offers a sufficient number of vehicles to meet the customer's demand. The transportation cost by a vehicle of the manufacturer is fixed. The transportation cost by a vehicle of 3PL provider consists of a fixed charge cost and a load-based variable cost, which depends on the quantity of products in a batch.

To address the studied IPDSP, four operational decisions have to be made: (1) assigning jobs to suitable factories with consideration of factory eligibility; (2) scheduling the assigned jobs for each of the factories; (3) allocating the finished products into an appropriate number of batches; (4) assigning each of the product batches to a vehicle of the manufacturer or 3PL provider for delivery.

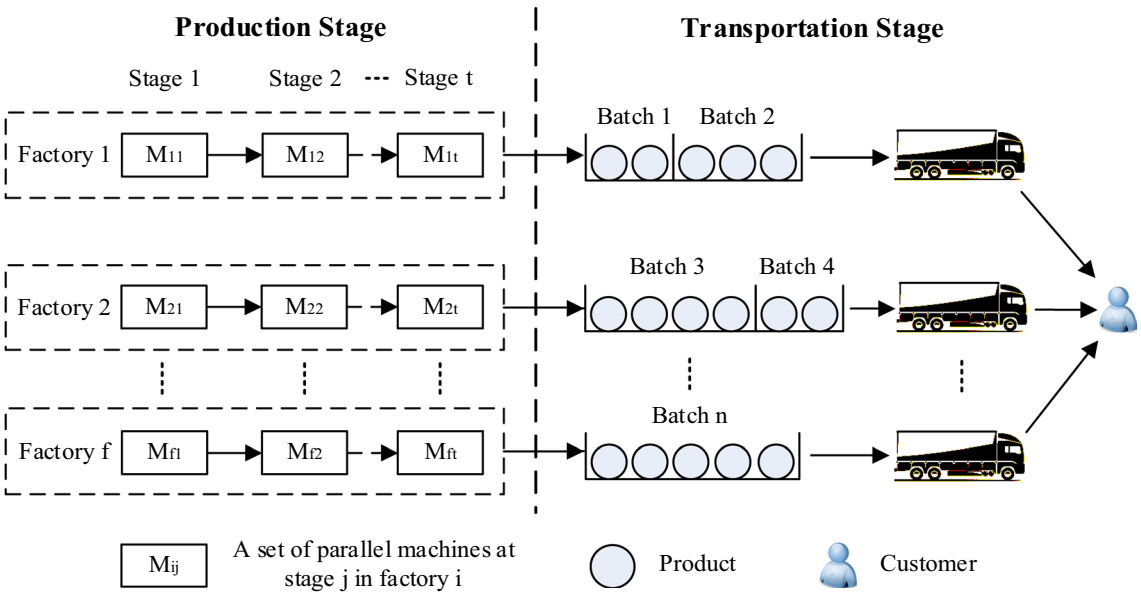


Fig. 1 The distributed HFSs with batch delivery

To simplify the integration of production and distribution, the following assumptions have been made: (1) all jobs are available for processing at time zero; (2) some jobs can only be handled in some specific factories; (3) factory switching is not allowed, that is, once a job is assigned to a certain factory, all its operations have to be performed in this factory; (4) each machine can process only one job at a time, and job preemption is not permitted; (5) the same job processed on any parallel machine at a stage takes equal processing time; (6) a batch is delivered immediately to the customer when all the jobs of the batch are finished; (7) the transportation time from different factories to the customer is fixed.

2.2 Problem formulation

Considering factory eligibility and multiple transportation modes, this section presents the mathematical model of the studied IPDSP. The parameters and decision variables used in the mathematical model are defined as follows:

2.2.1 Parameters

i	index of factories, $1 \le i \le f$.
j	index of jobs, $1 \le j \le n$.
k	index of stages in a factory, $1 \le k \le t$.
l	index of parallel machines at a stage, $1 \le l \le m_t$.
b	index of batches, $1 \le b \le p$.
f	the number of factories.
n	the number of jobs released to distributed HFSs.
n_i	the number of jobs processed in the HFS of factory i.
t	the number of stages in an FFS.
m_t	the number of parallel machines at stage t.
π_i^j	jth job of the job sequence $\pi_i = \{\pi_i^1, \pi_i^2, ..., \pi_i^{n_i}\}$ processed in factory i.
$P(\pi_i^j, k)$	processing time of job π_i^j at stage k in factory i.
TT_i	transportation time between factory i and the customer.
MTC_i	transportation cost of factory i by a vehicle of the manufacturer.
FTC_i	fixed transportation cost of factory i by a vehicle of the 3PL provider.
VTC_i	variable transportation cost of factory i by a vehicle of the 3PL provider.
DC_b	delivery cost of batch b.
d_j^l	the earliness due date of job π_i^j.
d_j^u	the latest due date of job π_i^j.
TC	unit tardiness cost.
EC	unit earliness cost.

2.2.2 Decision variables

$\alpha_{iklj_1j_2}$	1 if job $\pi_i^{j_2}$ is processed immediately after job $\pi_i^{j_1}$ on machine l at stage k in factory i; 0 otherwise.
η_{ib}	1 if the products of batch b are manufactured in factory i; and 0 otherwise.
λ_{ijb}	1 if job π_i^j is assigned to batch b; and 0 otherwise.
μ_b	1 if batch b delivered by a vehicle of the manufacturer; and 0 otherwise.
p	the number of batches delivered to the customer.
$C(\pi_i^j, k)$	completion time of job π_i^j at stage k in factory i.
DB_b	delivery start time of batch b.
DJ_{ij}	delivery start time of job π_i^j processed in factory i.

The paper aims to generate integrated production and distribution schedules in a distributed HFS environment. Based on the above notations, the studied IPDSP is formulated as the following mathematical model:

$$Min \quad \sum_{i=1}^{f}\sum_{j=1}^{n_i} TC \times max\left\{0, DJ_{ij} - d_j^u\right\} + \sum_{i=1}^{f}\sum_{j=1}^{n_i} EC \times max\left\{0, d_j^l - DJ_{ij}\right\} + \sum_{b=1}^{p} DC_b \tag{1}$$

Subject to:

$$C(\pi_i^j, 1) \ge P(\pi_i^j, 1), \quad i = 1, \ldots, f; j = 1, \ldots, m_1 \tag{2}$$

$$C(\pi_i^{j_2}, 1) \ge \sum_{l=1}^{m_1}\sum_{j_1=1}^{n_i} \alpha_{i1lj_1j_2} C(\pi_i^{j_1}, 1) + P(\pi_i^{j_2}, 1), \quad i = 1, \ldots, f; j_2 = m_1 + 1, \ldots, n_i \tag{3}$$

$$C(\pi_i^j, k) \ge C(\pi_i^j, k-1) + P(\pi_i^j, k), \quad i = 1, \ldots, f; j = 1, \ldots, m_1; k = 2, \ldots, t \tag{4}$$

$$C(\pi_i^{j_2}, k) \ge max\left\{\sum_{l=1}^{m_k}\sum_{j_1=1}^{n_i} \alpha_{iklj_1j_2} C(\pi_i^{j_1}, k), C(\pi_i^{j_2}, k-1)\right\} + P(\pi_i^{j_2}, k), \quad i = 1, \ldots, f; j_2 = m_1 + 1, \ldots, n_i; k = 2, \ldots, t \tag{5}$$

$$\sum_{b=1}^{p} \lambda_{ijb} = 1, \quad i = 1, \ldots, f; j = 1, \ldots, n_i \tag{6}$$

$$DC_b = \sum_{i=1}^{f} \eta_{ib} max\left\{ \mu_b MTC_i, (1-\mu_b)(FTC_i + \sum_{j=1}^{n_i} \lambda_{ijb} VTC_i) \right\}, \quad b = 1, \ldots, p \tag{7}$$

$$DB_b = \sum_{i=1}^{f} \eta_{ib} max\left\{ \lambda_{ijb} C(\pi_i^j, k) \right\}, \quad j = 1, \ldots, n_i; k = 1, \ldots, t; b = 1, \ldots, p \tag{8}$$

$$DJ_{ij} = \sum_{b=1}^{p} \lambda_{ijb} DB_b, \quad i = 1, \ldots, f; j = 1, \ldots, n_i \tag{9}$$

$$C(\pi_i^j, k), DC_b, DB_b, DJ_{ij} > 0, \quad i = 1, 2, \ldots, f; j = 1, \ldots, n; k = 1, \ldots, t; b = 1, \ldots, p \tag{10}$$

In the presented model, the objective function (1) is to minimize the sum of earliness, tardiness and delivery costs. For the first stage in a factory, constraints (2) and (3) give the completion time of the first job and the successive jobs on any parallel machine, respectively. Similarly, for all other stages in a factory, constraints (4) and (5) provide the completion time of the first job and the successive jobs on any parallel machine, respectively. Constraint (6) guarantees that each job assigned to a factory is allocated to only one batch for delivery. Considering multiple transportation modes, constraint (7) determines the delivery cost of a batch using the vehicles of the manufacturer or the 3PL provider. Constraint (8) ensures that a batch is delivered immediately after all its jobs have been completed. Constraint (9) determines the delivery start time of any job processed in a factory. Constraint (10) guarantees that all the time-related and cost-related decision variables are positive.

3 Algorithms for IPDSP in distributed HFSs

Considering the NP-hard nature of the studied IPDSP, it is infeasible to generate the optimal schedule in reasonable time, especially for large-sized instances. Therefore, this section presents three simple heuristics (CR-based, SLACK-based, and EDD-based heuristics) and a novel adaptive human-learning-based GA (AHLBGA) to obtain near-optimal solutions for integrated production and distribution scheduling in distributed HFSs.

3.1 Heuristics

Heuristics adopt simple strategies to construct solutions and therefore tend to be much faster but less effective. The most well-known scheduling heuristics are dispatching rules. Among a variety of dispatching rules, the critical ratio (CR), minimum slack (SLACK), and earliest due date (EDD) have been found more effective for earliness and tardiness related criteria [10, 35, 57]. To address the IPDSP with factory eligibility and 3PL distribution, three fast heuristics, namely CR-based, SLACK-based, and EDD-based heuristics, are developed to allocate jobs into an appropriate number of batches for production and distribution.

Considering factory eligibility and multiple transportation modes, each of the proposed heuristics integrates a simple scheduling rule with a job re-assignment method and a vehicle selection method. In the CR-based heuristic, the jobs released to the manufacturer are firstly sequenced in ascending order of CR, which is defined as the ratio of the remaining time until the due date to the total processing time in a factory. To reduce the search space of feasible solutions, the jobs are then evenly grouped into a number of batches (p) for production and distribution. As a batch contains at least one job, the maximum number of batches is limited to be the number of jobs n. Therefore, all possible numbers of batches, i.e.$1 \leq p \leq n$, need to be examined in this heuristic. Following the obtained job sequence, each of the batches is assigned to a factory with the minimum completion time. For the jobs that cannot be processed in the assigned factory, a job re-assignment method is applied to re-allocate them to eligible factories. In addition, to reduce the transportation cost to the customer, a vehicle selection method is employed to allocate each of the finished batches to a vehicle of the manufacturer or 3PL provider for delivery. Owing to the limited number of manufacturer's vehicles, this method applies the transportation cost difference under multiple transportation modes to determine the batches delivered by the manufacturer's vehicles.

The proposed CR-based heuristic, the job re-assignment method, and the vehicle selection method are detailed as follows. In the CR-based heuristic, $\lfloor x \rfloor$ indicates the greatest integer that is not more than x.

Algorithm I: CR-based heuristic

Step 1: Sequence the jobs in ascending order of their CRs, i.e. $\left(d_j^l + d_j^u\right) \Big/ 2\sum_{k=1}^{t} P(j,k)$.

Step 2: Let $p = 1$ and repeat steps 3-10 until $p > n$.

Step 3: If $n = p \times \lfloor n/p \rfloor$, then evenly group the sequenced jobs into p batches. Each of the batches contains n/p jobs.

Step 4: If $n > p \times \lfloor n/p \rfloor$, then group the sequenced jobs into p batches. Each of the first $p-1$ batches contains $\lfloor n/p \rfloor$ jobs, whereas the last batch has $n-(p-1)\times\lfloor n/p \rfloor$ jobs.

Step 5: Apply the commonly used scheme for distributed scheduling in the literature, namely the earliest completion time (ECT) rule, to determine appropriate factories for job processing. According to the order of batches appeared in the job sequence, this rule allocates each of the batches to a factory with the minimum completion time.

Step 6: Apply the job re-assignment method to ensure all the jobs in a batch are eligible for processing in the assigned factory.

Step 7: Apply the first-come-first-serve (FCFS) rule to schedule the batches assigned to a factory, and compute the delivery start time of each batch, i.e. the completion time of the last job in the batch.

Step 8: Apply the vehicle selection method to allocate the batches to the vehicles of the manufacturer or the 3PL provider for delivery.

Step 9: Compute the sum of earliness, tardiness and delivery costs of batch delivery to the customer using objective function (1).

Step 10: Set $p = p + 1$.

Step 11: Output the best integrated production and distribution schedule.

Algorithm II: job re-assignment method

Step 1: Identify the job set Π_r consisting of n_r jobs that are assigned to ineligible factories.

Step 2: Let $q = 1$ and repeat steps 3-6 until $q > n_r$.

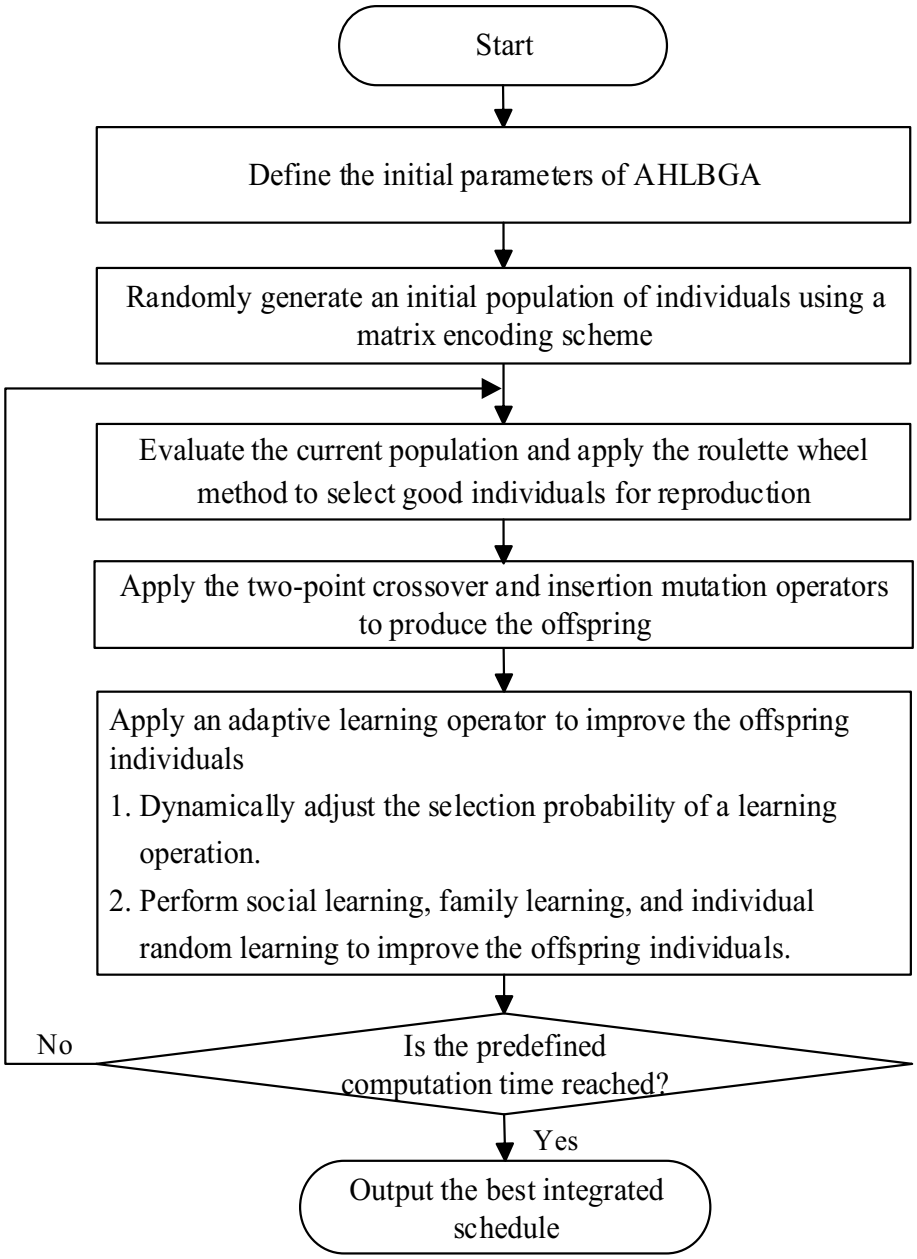
Step 3: Determine the set of available batches, in each of which the q^{th} job of Π_r can be processed in the assigned factory.

Step 4: Assign the q^{th} job of Π_r to the batch with the minimum completion time.

Step 5: Insert the q^{th} job of Π_r to the position that ensures all the jobs in the assigned batch follow the job sequence obtained by the CR-based heuristic.

Step 6: Set $q = q + 1$.

Fig. 2 Framework of AHLBGA



Algorithm III: vehicle selection method

Step 1: For each batch b, compute its transportation cost using the vehicles of the manufacturer and the 3PL provider respectively, i.e. CM_b and CO_b. Accordingly, the transportation cost difference between these two transportation modes $\Delta_b = CO_b - CM_b$ is obtained.

Step 2: Identify the set of batches A that are cheaper to be delivered by the manufacturer ($\Delta_b > 0$), and denote r as the number of batches in this set.

Step 3: If the number of vehicles s provided by the manufacturer is less than r, sort the batches in a descending order of Δ_b, and update A with the first s batches.

Step 4: The batches of A are transported by the vehicles of the manufacturer, whereas the other batches are transported by the vehicles of 3PL provider.

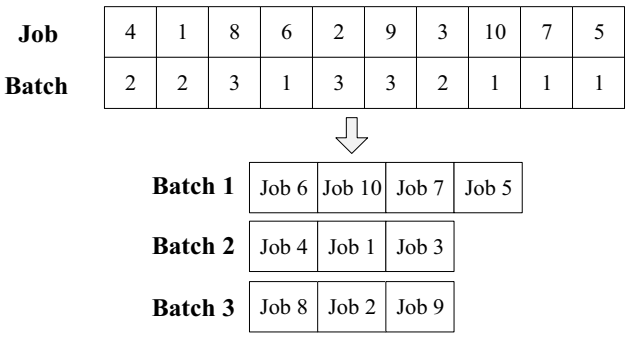


Fig. 3 A chromosome of proposed AHLBGA

To address the IPDSP with factory eligibility and 3PL distribution, the SLACK-based and EDD-based heuristics are also developed. These two heuristics employ the same procedure of the CR-based heuristic to generate the schedules, except that the jobs released to the distributed HFSs are processed in ascending order of $\frac{\left(d_j^l + d_j^u\right)}{2} - \sum_{k=1}^{t} P(j,k)$ and the due time respectively.

3.2 Proposed AHLBGA

3.2.1 Framework of AHLBGA

To address the IPDSP with factory eligibility and 3PL distribution, AHLBGA is proposed to generate better solutions. As

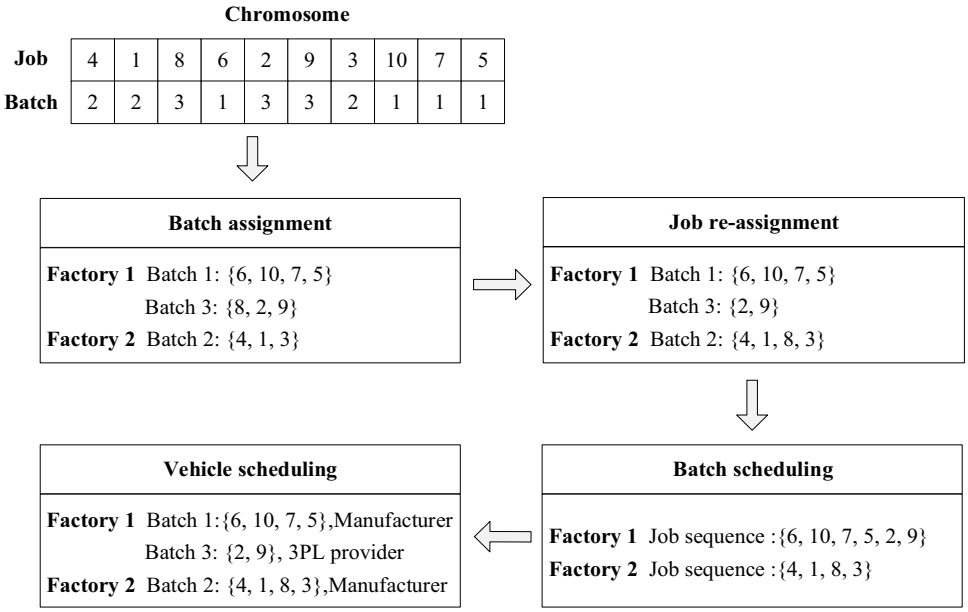


Fig. 4 An example of the decoding procedure

one of the most widely used population-based meta-heuristics, GA has been successfully applied to generate near-optimal solutions to a variety of complex optimization problem [25, 48, 51], including the HFS scheduling problem and IPDSP [47, 49]. It produces the offspring chromosomes by performing three genetic operators, namely selection, crossover, and mutation. Apart from the traditional genetic operators, learning is another crucial biological process that enables all the living organisms to adapt to the changing environment by applying their previous or newly acquired knowledge and skills [34, 47, 49]. However, the existing GAs in the literature seldom perform learning activities to improve the offspring individuals generated after crossover and mutation [16, 18].

Different from traditional GAs, the proposed AHLBGA integrates an adaptive learning operator with traditional genetic operators to obtain better integrated production and distribution schedules. In the AHLBGA, an initial population is firstly randomly generated, and then evolves through selection, crossover, mutation, and adaptive learning until the predefined computation time is reached. To imitate the human learning behaviours, adaptive learning involves three types of learning operations, namely social learning, family learning, and individual random learning. To enhance the learning performance of offspring individuals, the probability of selecting a learning operation is dynamically adjusted by an adaptive learning scheme (to be detailed in Sect. 3.2.5). Such scheme encourages more individual random learning along with population evolution to help escape from local optimum, and therefore leads to potentially better solutions. It is the first attempt in integrating GA with human learning process for the IPDSP. The framework of AHLBGA is shown in Fig. 2 and each part of AHLBGA is detailed in the following subsections.

3.2.2 Encoding and decoding scheme

The encoding scheme plays an important role in designing an effective and efficient meta-heuristic. Different from traditional flow shop scheduling problems, the IPDSP needs to be solved by integrating the decisions of production and distribution. Based on the commonly used permutation encoding scheme for production scheduling, a matrix of two rows and n columns is employed to represent a solution (chromosome) of the IPDSP. In the matrix, the first row is a permutation of n jobs that indicates the processing order in a distributed HFS environment, whereas the second row shows the assigned batches of all jobs for production and distribution. Since a batch contains at least one job, each of the rows is a string of integer numbers from 1 to n.

Figure 3 presents a chromosome of an IPDSP with 10 jobs to be scheduled in the HFSs of two distributed factories. As described in the first row of the chromosome, jobs are suggested to be processed according to the order of {4, 1, 8, 6, 2, 9, 3, 10, 7, 5}. According to the batch numbers in the second row of the chromosome, these jobs are grouped into three batches, namely jobs {6, 10, 7, 5} in batch 1, jobs {4, 1, 3} in batch 2, and jobs {8, 2, 9} in batch 3.

Considering factory eligibility and multiple transportation modes, a new decoding procedure is employed to convert the matrix of a solution into a feasible integrated schedule. As shown in Fig. 4, this decoding procedure mainly consists of four steps: (1) apply the ECT rule to assign each of the batches to a factory with the minimum completion time; (2) identify the jobs that cannot be processed in the assigned factories, and re-assign them to other suitable batches using the job re-assignment method; (3) the batches

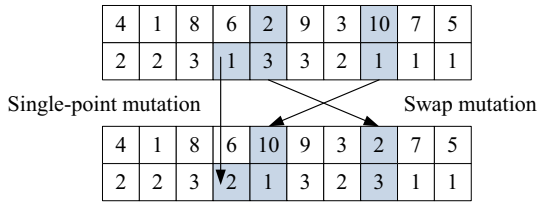


Fig. 6 An example of mutation

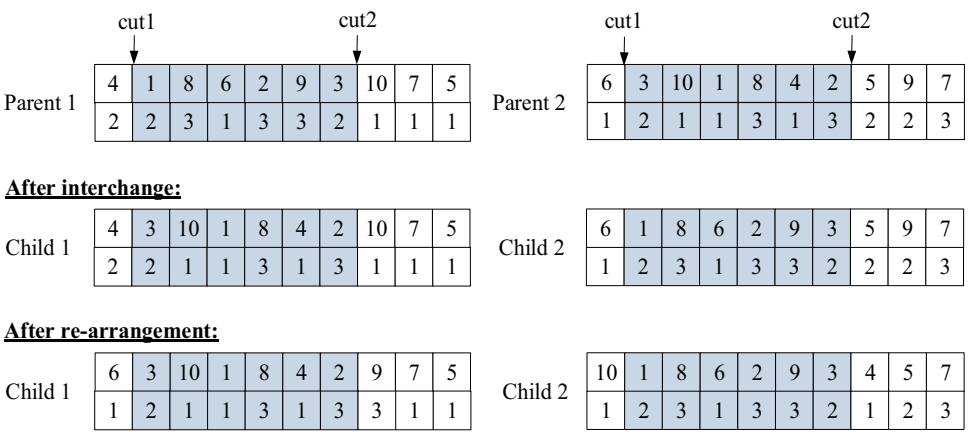


Fig. 5 An example of two-point crossover

assigned to the same factory are scheduled based on the FCFS rule, and the jobs in a batch are processed in parallel based on their order in the batch; (4) apply the vehicle selection method to determine the appropriate vehicles of the manufacturer or 3PL provider to deliver the finished batches in the production stage to the customer. The details of the job re-assignment method and the vehicle selection method have been presented in Sect. 3.1.

Figure 4 describes the decoding process of the chromosomes shown in Fig. 3. According to the ECT rule, batches 1 and 3 are allocated to the FFS of factory 1, and batch 2 is assigned to the FFS of factory 2. Assuming that job 8 in batch 3 can only be processed in the FFS of factory 2, the constraint of factory eligibility is violated and job 8 is therefore re-assigned to batch 2, which is processed in the FFS of factory 2. After the job re-assignment, job processing in factories 1 and 2 follows the job sequence {6, 10, 7, 5, 2, 9} and {4, 1, 8, 3}, respectively. To reduce the transportation cost to the customer, finished batches in the production stage are assigned to suitable vehicles for delivery, i.e. batches 1 and 2 delivered by the manufacturer and batch 2 delivered by the 3PL provider.

3.2.3 Initialisation, evaluation, and selection

The chromosomes in the initial population are randomly generated to cover the entire search space.In each of the generations, the chromosomes are decoded by the proposed decoding procedure and evaluated using the following fitness function:

$$f(x_i) = 1 \Big/ y(x_i), \quad i = 1, 2, \ldots, PS \tag{11}$$

where x_i represents the i^{th} chromosome of a population, $y(x_i)$ denotes the objective value of x_i, and PS is the number of chromosomes in a population.

Based on the above fitness function, the well-known roulette wheel selection method is employed to choose better chromosomes for reproduction. In this method, the probability of a chromosome to participate in the reproduction process is proportional to its fitness. To avoid the loss of the best chromosome, the elitist strategy is applied to preserve the best chromosome to the next generation. In this strategy, the chromosome with the lowest fitness is replaced by the best chromosome found so far.

3.2.4 Crossover and mutation

Once the chromosomes have been selected for reproduction, crossover and mutation are performed to generate the offspring by mimicking the evolution process in the nature. Crossover exchanges and recombines genetic information of parent chromosomes to produce better offspring [15], and mutation slightly perturbs the genes of selected offspring to maintain the diversity of populations [50].

As the two-point crossover has been proven to provide better performance for scheduling problems, it is adopted in the AHLBGA to produce new offspring. As described in Fig. 5, two selected parents are firstly divided into three parts by two random cut-off points. The genes in the second part (middle part) of the two parents are subsequently interchanged. Finally, the jobs and their assigned batches in the first and third parts are re-arranged according to their order in the other parent.

Considering the matrix structure of the chromosomes, a new mutation method is employed to enhance the local search and avoid trapping into local optimum. As shown in Fig. 6, this proposed mutation involves two separate operations, namely the swap mutation and the single-point mutation. The swap mutation chooses two random columns and swaps both the jobs and their assigned batches in these two columns, whereas the single-point mutation only select one random job and randomly re-assigns it to another batch.

3.2.5 Adaptive learning

To improve the offspring individuals after crossover and mutation, AHLBGA applies a novel adaptive learning operator to construct better solutions. By mimicking human learning behaviours, this learning operator consists of the following three types of learning operations.

Social learning operation

The humans in the social environment gradually accumulate their knowledge and experience by direct or indirect interactions ([45]; Wang et al., 2017). Social learning from a group of knowledgeable and experienced individuals is an efficient way to develop the learner's ability. To perform the social learning operation, an elite social set Ω_S is firstly established by selecting the best $\alpha \times 100\%$ ($\alpha \in (0, 1)$) of offspring in the current population, and two social knowledge matrices are subsequently obtained by extracting the scheduling and distribution knowledge of Ω_S as follows:

$$SKP = \begin{bmatrix} J_{11} & J_{12} & \cdots & J_{1j} & \cdots & J_{1n} & \\ J_{21} & J_{22} & \cdots & J_{2j} & \cdots & J_{2n} & \\ \vdots & \vdots & & \vdots & & & \vdots \\ J_{i1} & J_{i2} & \cdots & J_{ij} & \cdots & J_{in} & \\ \vdots & \vdots & & \vdots & & & \vdots \\ J_{n1} & J_{n2} & \cdots & J_{nj} & \cdots & J_{nn} & \end{bmatrix}, 1 \le i, j \le n \tag{12}$$

$$SKB = \begin{bmatrix} B_{11} & B_{12} & \cdots & B_{1b} & \cdots & B_{1n} & \\ B_{21} & B_{22} & \cdots & B_{2b} & \cdots & B_{2n} & \\ \vdots & \vdots & & \vdots & & & \vdots \\ B_{j1} & B_{j2} & \cdots & B_{jb} & \cdots & B_{jn} & \\ \vdots & \vdots & & \vdots & & & \vdots \\ B_{n1} & B_{n2} & \cdots & B_{nb} & \cdots & B_{nn} & \end{bmatrix}, 1 \le j, b \le n \tag{13}$$

where J_{ij} denotes the number of times that job j appears on position i in the job sequence, B_{jb} represents the number of times that job j is allocated to batch b, and n is the number of jobs released to distributed HFSs. Therefore, the matrices of *SKP* and *SKB* reserve the knowledge of job processing and batch assignment, respectively.

According to the above two matrices extracted from Ω_s, an offspring individual performs social learning as Eqs. (14) and (15) to generate better production and distribution schedules. In these two equations, Ω_u^i is the set of unassigned jobs until the i^{th} column of an individual, and x_i and y_i indicate the job and its assigned batch in the i^{th} column, which are determined by the job with the largest J_{ij} in *SKP* and the batch with the largest B_{jb} in *SKB* respectively.

$$x_i = \arg \max_{j \in \Omega_u^i} J_{ij}, 1 \le i, j \le n \tag{14}$$

$$y_i = \arg \max_{b=1,2,\ldots,n} B_{x_i b}, \quad 1 \le i, b \le n \tag{15}$$

Family learning operation

As one important form of inter-generational learning, family learning has a major impact on personal development throughout life. Therefore, it is very common that a person uses the knowledge and experience of their ancestors to avoid mistakes and improve decision-making practice. To mimic the family learning process, each of the offspring individual employs two matrices of *FKP* and *FKB,* which have the same structures as *SKP* and *SKB*, to store its family knowledge on scheduling and distribution. Different from the social learning operation, *FKS* and *FKD* of an offspring individual are established on a set of its ancestors rather than an elite social set Ω_S of current population. Similar to social learning, the offspring individuals improves their performance using the knowledge of job processing and batch assignment stored in *FKS* and *FKD*.

Individual random learning operation

A person tends to conduct a random learning for new problem solving, especially in the absence of *priori* knowledge and experience. To imitate the randomness of human learning, an individual random learning operation is adopted to update an offspring as follows:

$$x_i = randsample\left(\Omega_u^i\right), \quad 1 \le i \le n \tag{16}$$

$$y_i = randint(1,\ n), \quad 1 \le i \le n \tag{17}$$

Table 1 Time-related and cost-related parameters

Parameters	Values
Processing time of job π_i^j at stage k in factory i ($P(\pi_i^j, k)$)	*Unidrnd(1, 10)*
Transportation time between factory i and the customer (TT_i)	*Unidrnd(10, 20)*
Transportation cost of factory i by a vehicle of the manufacturer (MTC_i)	3
Fixed transportation cost of factory i by a vehicle of the 3PL provider (FTC_i)	*Unidrnd(4, 4+f)*
Variable transportation cost of factory i by a vehicle of the 3PL provider (VTC_i)	1
The earliness due date of job π_i^j (d_j^l)	*Unidrnd(60,P)-Unidrnd(10,100)/2*, where $P = \sum_{j=1}^{n} \sum_{k=1}^{t} P(j,k)$
The latest due date of job π_i^j (d_j^u)	*Unidrnd(60,P)+Unidrnd(10, 100)/2*, where $P = \sum_{j=1}^{n} \sum_{k=1}^{t} P(j,k)$
Unit earliness cost (EC)	1
Unit tardiness cost (TC)	1

where $randsample(\Omega_u^i)$ is a function that returns a random job from the unsigned job set Ω_u^i, $randint(min, max)$ is a function that returns an integer uniformly distributed in the range of *min* and *max*.

To improve the offspring generated by crossover and mutation, each of the genes in a chromosome performs one type of learning. Based on two predefined parameters P_1 and P_2, the selection probability of social learning, family learning, and individual random learning is determined by the values of P_1, $P_2 - P_1$, and $1 - P_2$.

Since P_1 and P_2 greatly affect the learning performance of offspring individuals, an adaptive learning scheme is applied as Eqs. (18) and (19) to adjust these two parameters throughout the evolution process.

$$P_1 = P_1^{min} + \frac{\left(P_1^{max} - P_1^{min}\right) T_{cur}}{T_{max}} \tag{18}$$

$$P_2 = P_2^{max} - \frac{\left(P_2^{max} - P_2^{min}\right) T_{cur}}{T_{max}} \tag{19}$$

where P_1^{min} and P_1^{max} indicate the minimum and maximum values of P_1, P_2^{min} and P_2^{max} represent the minimum and maximum values of P_2, and T_{cur} and T_{max} are the current time and the maximum computation time of AHLBGA respectively. According to the above two equations, the adaptive learning scheme allows more individual random learning along with population evolution to avoid getting trapped into local optimum, and eventually generates potentially better solutions.

Based on the above details of adaptive learning operator, the complete learning process of an offspring individual is summarised as follows:

Table 2 Factors and their levels of AHLBGA

Parameter level	Parameters			
	PS	P_c	P_m	α
1	50	0.5	0.05	10%
2	100	0.6	0.10	20%
3	150	0.7	0.15	30%
4	200	0.8	0.20	40%

Table 3 Combinations of different factor levels using an Orthogonal array L_{16} (4^4)

Trial	Parameter				ARE
	PS	P_c	P_m	α	
1	50	0.5	0.05	10	0.1238
2	100	0.5	0.10	20	0.1235
3	150	0.5	0.15	30	0.1241
4	200	0.5	0.20	40	0.1229
5	100	0.6	0.05	30	0.1241
6	50	0.6	0.10	40	0.1254
7	200	0.6	0.15	10	0.1256
8	150	0.6	0.20	20	0.1249
9	150	0.7	0.05	40	0.1258
10	200	0.7	0.10	30	0.1239
11	50	0.7	0.15	20	0.1231
12	100	0.7	0.20	10	0.1241
13	200	0.8	0.05	20	0.1222
14	150	0.8	0.10	10	0.1246
15	100	0.8	0.15	40	0.1238
16	50	0.8	0.20	30	0.1240

Algorithm IV: adaptive learning process

Step 1: Initialise the parameters for adaptive learning, including α, P_1^{min}, P_1^{max}, P_2^{min}, and P_2^{max}.

Step 2: Compute the probability of random learning, family learning, and individual random learning based on Eqs. (18) and (19).

Step 3: Determine the type of leaning operation for each of the genes in a chromosome.

Step 4: Update all genes by the assigned learning operations.

Step 5: Output the new generated chromosome.

Table 4 Average ARE at each factor level

Level	Parameter			
	PS	P_c	P_m	α
1	0.1241	0.1236	0.1240	0.1245
2	0.1239	0.1250	0.1243	0.1234
3	0.1249	0.1242	0.1242	0.1241
4	0.1236	0.1236	0.1240	0.1245
Delta (max–min)	0.0012	0.0014	0.0004	0.0011
Rank	2	1	4	3

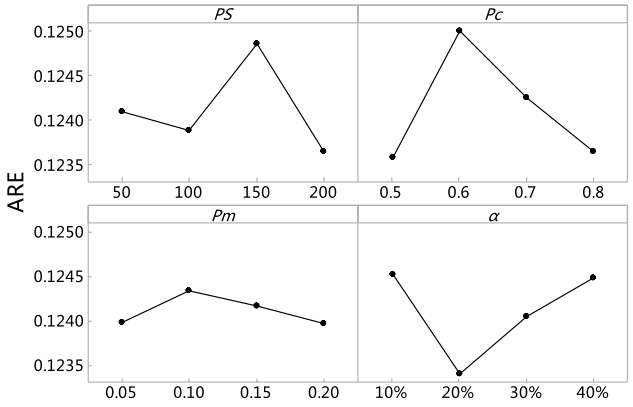


Fig. 7 Factor level trend of AHLBGA

4 Experimental results and discussion

To validate the effectiveness of proposed heuristics and AHLBGA for addressing the IPDSP in distributed HFSs, this section compares them with some competitive algorithms. All the algorithms were developed using C++ in CodeBlocks 16.01 and ran on a PC with Intel® Core™ i5-5200U 2.20 GHz CPU and 4 GB RAM under Microsoft Windows 10 environment. The following subsections present the experiment design, parameter tuning, and experiment results of the compared algorithms, respectively.

4.1 Experiment design

The IPDSP with factory eligibility and 3PL distribution is a new combinatorial scheduling problem and therefore no benchmark instances have been found in the literature. To evaluate the performance of proposed heuristics and AHLBGA, two sets of test problems are randomly generated, namely small-sized problems with $f=\{2, 3, 4, 5\}$, $n=\{20, 50\}$, $t=\{5, 10\}$and large-sized problems with $f=\{2, 3, 4, 5\}$, $n=\{100, 200\}$, $t=\{15, 20\}$. For each of the test problems, the number of parallel machines in a stage is randomly selected in the range [2, 5]. The time-related and cost-related parameters of test problems are shown in Table 1. In this table, *Unidrnd(a, b)* returns a uniform random integer in the interval [a, b].

Table 5 Computation results of SLACK, EDD, CR, GA, HLBGA, and AHLBGA for small-sized problems

f	n	t	SLACK	EDD	CR	GA			HLBGA			AHLBGA		
			RE	RE	RE	BRE	ARE	WRE	BRE	ARE	WRE	BRE	ARE	WRE
2	20	5	20.32	20.32	20.32	9.38	13.46	18.22	1.56	4.88	9.30	**0.00**	**1.56**	**3.04**
2	20	10	36.33	26.63	24.31	10.12	13.03	18.32	**0.00**	4.06	8.73	**0.00**	**0.66**	**2.11**
2	50	5	22.99	23.60	21.97	10.56	15.40	17.50	0.90	5.22	14.29	**0.00**	**1.05**	**2.41**
2	50	10	27.09	24.31	21.97	9.42	13.80	16.68	**0.00**	5.41	13.19	**0.00**	**1.55**	**4.86**
3	20	5	18.31	18.31	18.31	3.16	8.33	14.30	1.05	5.93	11.15	**0.00**	**1.97**	**5.74**
3	20	10	21.66	19.33	17.70	2.68	8.42	13.87	**0.00**	4.63	9.05	**0.00**	**1.00**	**2.41**
3	50	5	19.73	17.49	16.78	1.57	8.47	13.68	**0.00**	4.00	7.87	**0.00**	**2.45**	**7.24**
3	50	10	26.92	22.45	22.45	9.44	12.70	18.58	0.30	5.28	9.70	**0.00**	**1.02**	**4.54**
4	20	5	20.42	20.42	21.04	9.13	13.13	16.25	1.17	4.51	9.26	**0.00**	**1.98**	**7.83**
4	20	10	22.88	22.27	22.88	9.93	13.44	17.52	**0.00**	2.66	7.28	**0.00**	**1.42**	**3.28**
4	50	5	16.17	17.09	15.56	10.46	12.82	13.76	**0.00**	5.32	11.21	**0.00**	**0.74**	**1.63**
4	50	10	22.78	16.07	16.07	4.85	9.20	14.74	**0.00**	5.49	12.42	**0.00**	**1.50**	**2.71**
5	20	5	22.99	23.60	21.97	11.53	14.95	20.60	1.91	5.16	8.02	**0.00**	**2.46**	**5.82**
5	20	10	22.88	24.31	22.68	12.34	15.22	17.52	**0.00**	4.41	9.47	**0.00**	**1.67**	**3.12**
5	50	5	24.71	17.21	18.92	11.53	14.14	16.89	**0.00**	4.57	15.90	**0.00**	**1.01**	**2.68**
5	50	10	26.92	22.45	22.45	7.52	12.35	20.47	1.12	5.49	10.44	**0.00**	**0.97**	**1.81**
Average			23.32	20.99	20.34	8.35	12.43	16.81	0.50	4.81	10.46	**0.00**	**1.44**	**3.83**

Table 6 Computation results of SLACK, EDD, CR, GA, HLBGA, and AHLBGA for large-sized problems

f	n	t	SLACK	EDD	CR	GA			HLBGA			AHLBGA		
			RE	RE	RE	BRE	ARE	WRE	BRE	ARE	WRE	BRE	ARE	WRE
2	100	15	29.94	27.34	25.92	15.36	21.02	25.49	2.21	16.51	23.90	**0.00**	**2.69**	**7.32**
2	100	20	30.86	32.71	26.85	4.85	18.32	24.10	**0.00**	6.10	15.78	**0.00**	**2.74**	**6.45**
2	200	15	42.64	34.98	29.93	17.70	22.15	28.21	11.68	18.50	23.98	**0.00**	**4.00**	**8.38**
2	200	20	48.71	51.49	39.12	11.17	18.64	23.49	2.61	11.08	20.92	**0.00**	**3.02**	**7.17**
3	100	15	68.02	58.45	55.30	16.78	27.39	38.91	7.66	10.72	12.57	**0.00**	**7.33**	**9.95**
3	100	20	35.74	36.33	26.63	11.17	19.08	24.31	5.08	8.87	18.11	**0.00**	**3.89**	**5.32**
3	200	15	28.52	29.94	30.29	15.26	20.55	26.77	11.24	16.35	19.55	**0.00**	**5.41**	**10.38**
3	200	20	35.68	30.41	26.03	11.80	19.42	24.92	4.12	10.56	20.46	**0.00**	**2.88**	**5.90**
4	100	15	48.71	51.49	39.56	16.99	27.66	39.49	12.63	15.20	20.82	**0.00**	**6.09**	**9.91**
4	100	20	36.21	35.74	32.90	9.71	23.50	31.15	**0.00**	6.44	18.06	**0.00**	**6.09**	**9.85**
4	200	15	36.33	36.33	35.74	16.17	25.48	33.83	10.32	14.42	21.91	**0.00**	**4.43**	**9.63**
4	200	20	32.02	30.17	28.97	11.36	17.25	24.92	10.77	13.44	19.35	**0.00**	**3.67**	**7.27**
5	100	15	58.45	51.49	39.12	16.78	28.07	38.86	2.83	15.81	22.35	**0.00**	**8.07**	**12.03**
5	100	20	29.94	27.34	25.92	11.36	19.05	23.60	0.53	8.48	23.23	**0.00**	**2.88**	**9.07**
5	200	15	31.60	35.03	34.32	16.27	22.60	30.66	9.12	12.04	15.90	**0.00**	**4.94**	**10.74**
5	200	20	25.68	30.41	26.03	15.36	18.01	24.10	3.06	13.61	18.04	**0.00**	**4.10**	**7.45**
Average			38.69	37.48	32.66	13.63	21.76	28.93	5.87	12.38	19.68	**0.00**	**4.51**	**8.55**

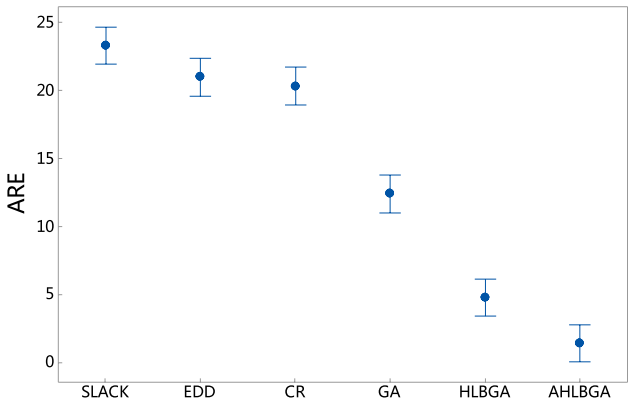


Fig. 8 Means plot and LSD intervals of different algorithms for small-sized problems

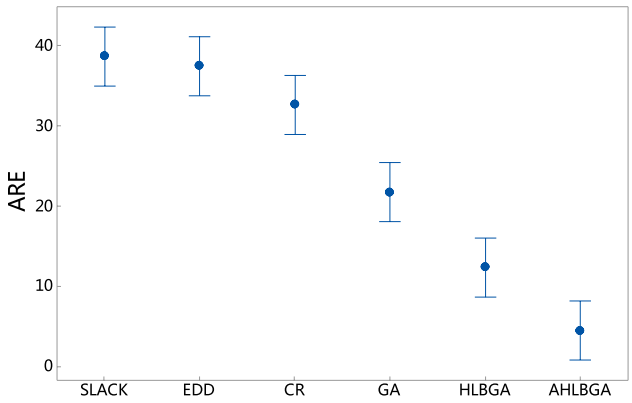


Fig. 9 Means plot and LSD intervals of different algorithms for large-sized problems

As a new combinatorial optimization problem, there are no scheduling algorithms specifically designed for the IPDSP with factory eligibility and 3PL distribution. Instead of using some existing algorithms, the proposed three fast heuristics (CR-based, EDD-based, and SLACK-based heuristics) and AHLBGA are compared with the following two competitive algorithms:

- GA: As a standard GA, this algorithm does not employ the proposed learning operator to improve the offspring individuals. It starts with a random initial population, and adopts the same selection, crossover and mutation operators as used in the proposed AHLBGA for reproduction.
- Human-learning-based GA (HLBGA): This algorithm is the same as AHLBGA, except for determining the selection probability of learning activities using $P_1 = 1/3$ and $P_2 = 2/3$ throughout the evolution process.

As a commonly used performance measure, the relative error (RE) is applied to evaluate the effectiveness of scheduling algorithm as follows:

$$RE = \frac{C_i - C_{min}}{C_{min}} \times 100 \tag{20}$$

where C_{min} represents the minimum total cost of earliness, tardiness and delivery obtained by all the compared

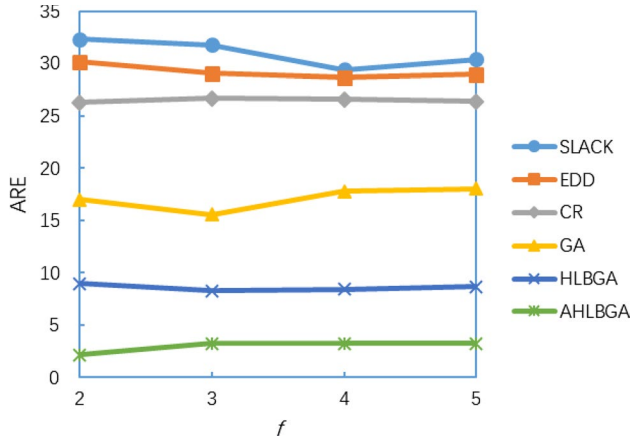


Fig. 10 ARE with different numbers of factories (f)

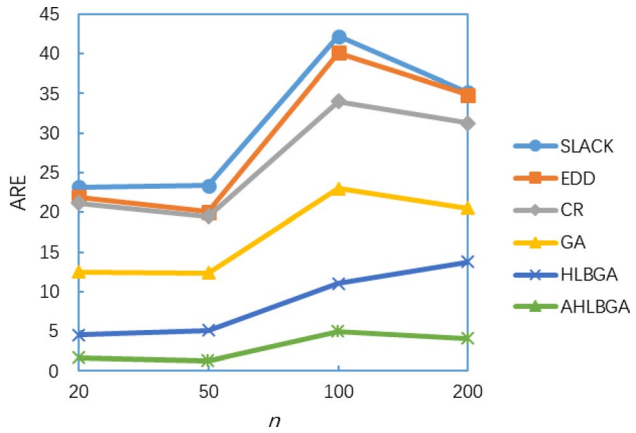


Fig. 11 ARE with different numbers of jobs (n)

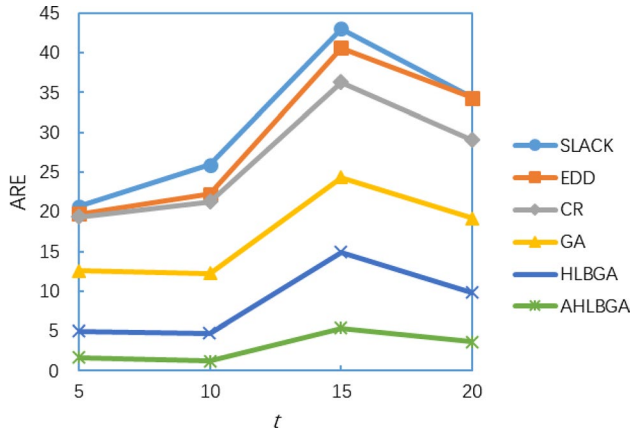


Fig. 12 ARE with different stages in a factory (t)

algorithms, and C_i indicates the total cost of earliness, tardiness and delivery obtained by a specific algorithm i.

Among all the compared algorithms, RE is directly used to evaluate the performance of CR-based, EDD-based, and SLACK-based heuristics. To obtain more reliable experimental results of GA, HLBGA, and AHLBGA, we ran these three meta-heuristics 20 independent times for each of the test problems and measured their performance by the best RE (BRE), the average RE (ARE), and the worst RE (WRE). Accordingly, a meta-heuristic with lower BRE, ARE, and WRE is capable of generating better integrated production and distribution schedules. For all the compared meta-heuristics, the maximum computation time required to solve each of the test problems is set to $0.1 \times n \times t$ (s).

4.2 Parameter tuning with Taguchi method

The appropriate parameter setting plays a significant role in the effectiveness of meta-heuristics. The performance of AHLBGA greatly depends on four key parameters, namely population size (PS), crossover rate (P_c), mutation rate (P_m), and the population ratio to establish the social knowledge matrices (α). The Taguchi method provides a systematic and efficient way to determine near optimum design parameters. Compared with a full factorial design, this method can significantly reduce the number of required experiments by using an orthogonal array and is therefore adopted to tune the parameters of AHLBGA.

The Taguchi method is implemented on a moderate-sized IPDSP with $f=4$, $n=50$ and $t=10$. Table 2 describes the key parameters and their levels considered in the Taguchi method. Based on an orthogonal array L_{16} (4^4), different parameter combinations shown in Table 3 are examined. AHLBGA with a specific parameter combination firstly runs 10 times to address the selected IPDSP, and ARE is subsequently obtained as the responsible variable (RV) value. Instead of conducting $4^4=256$ experiments in the full factorial design, the Taguchi method only performs 16 experiments to determine the parameters of AHLBGA.

According to the ARE of Table 3, Table 4 ranks the four factors based on their Delta statistics, each of which is the difference of maximum and minimum values of average ARE for a specific factor. From the ranks in this table, it is clear that P_c has the most significant effect on the performance of AHLBGA.

Based on the results of Table 4, the trend of each parameter is depicted in Fig. 7. Since the studied IPDSP is a minimization type problem, AHLBGA tends to perform better when these parameters are set to the values with the minimum average ARE. A larger value of PS allows to explore more solutions of the search space and the value of PS is set to 200 according to Fig. 7. In addition, a moderate is capable of maintaining a better balance between population diversity and convergence performance, and therefore the best 20% of offspring individuals are chosen to extract social

knowledge in this study. Furthermore, the values of and can be easily determined by Fig. 7. From the above analysis, a good choice of parameter setting is suggested as,,, and. For a fair performance comparison among the scheduling algorithms,,, and of GA and HLBGA are set to the same values as used in AHLBGA.

4.3 Experiment results and discussion

To evaluate the performance of the proposed fast heuristics and AHLBGA, both small-sized and large-sized test problems shown in Tables 5 and 6 are tested. In these two tables, f, n, and t indicate the numbers of factories, jobs, and stages in the distributed FFS environment, respectively. For each of the compared meta-heuristics, the maximum running time is set to $0.1 \times \mathrm{n} \times \mathrm{t}$ (s) in this study and the CPU time is therefore in the range between 10 and 400 s. Compared with the meta-heuristics, the three heuristics are much faster and give a solution within less than 1.9 s. For better performance comparison, the best BRE, ARE, and WRE for each of the test problems in these two tables are highlighted in bold.

According to the experiment results on the two sets of test problems, the following are observed:

1. Among of the six scheduling algorithms, ARE of GA, HLBGA, and AHLBGA is much less than RE of the three fast heuristics. The better performance of the three meta-heuristics results from the stochastic search of solution space, which is capable of offering a good balance between exploration and exploitation.
2. Compared with the SLACK and EDD heuristics, the CR heuristic provides better results on average, indicating that the CR rule is more effective to reduce the total cost of earliness, tardiness and delivery.
3. Since HLBGA and AHLBGA further improve the offspring individuals by a learning operator after crossover and mutation, they perform better than GA in terms of BRE, ARE, and WRE over all the test problems. Furthermore, AHLBGA outperforms HLBGA when generating integrated production and distribution schedules. The superiority of AHLBGA lies in the effectiveness of the adaptive learning operator to guide the learning process of offspring individuals.

Based on the experiment results of the above two tables, the analysis of variance (ANOVA) and the least significant difference (LSD) test are conducted to validate whether the performance differences among the compared algorithms are statistically significant. With regard to small-sized and large-sized test problems, Figs. 8 and 9 show the means plots with LSD intervals (95% confidence level), respectively. As shown in these two figures, AHLBGA provides the statistically best performance when generating the integrated production and distribution schedules in distributed HFS environments.

To further evaluate the performance of SLACK, EDD, CR, GA, HLBGA, and AHLBGA, the trends of ARE for different sizes of f, n, and t are presented in Figs. 10, 11 and 12, respectively. From these figures, AHLBGA provides the least ARE with slight fluctuation for all sizes of f, n, and t, indicating its effectiveness and robustness for solving the IPDSP with factory eligibility and 3PL distribution. As described in Fig. 10, ARE obtained using these six algorithms is not significantly changed, implying that the number of factories (f) has little influence on the performance of all the compared algorithms. In addition, the overall ARE values of the six algorithms in Figs. 11 and 12 show an increasing trend when addressing large-sized test problems ($n = 100, 200$; $t = 15, 20$).

5 Conclusion

This paper focuses on a new integrated production and distribution scheduling problem (IPDSP), in which jobs are firstly processed in distributed hybrid flow shops (HFS) and subsequently delivered to a customer in batches. To satisfy the industrial practice of production and distribution, both factory eligibility and multiple transportation modes are considered. Accordingly, in the studied IPDSP, some products can only be manufactured in a subset of distributed factories, and the transportation of finished products is conducted by the manufacturer or the 3PL provider.

To minimize the sum of earliness, tardiness and delivery costs, three fast heuristics (CR-based, SLACK-based, and EDD-based heuristics) and an adaptive human-learning-based genetic algorithm (AHLBGA) are developed to generate integrated production and distribution schedules. These three heuristics employ simple scheduling rules to sequence the released jobs and evenly group them into batches for production and distribution. Different from the traditional GA, AHLBGA hybridizes an adaptive learning operator with crossover and mutation to enhance the global and local search ability. Motivated by human learning behaviours, the learning operator involves social learning, family learning, and individual random learning operations. To provide a better trade-off between search exploration and exploitation, the selection probability of a learning operation is dynamically adjusted throughout the evolution process. To evaluate the effectiveness of proposed algorithms, two sets of randomly generated IPDSPs are established. Experimental results indicate that AHLBGA offers the best performance among all the compared scheduling algorithms. The superiority of AHLBGA results from incorporating three types of human learning operations into the evolution process of GA. Accordingly, AHLBGA has the potential to provide MTO

manufacturers with better operational decisions in distributed HFS environments.

Future research may focus on integrated production and distribution scheduling in more realistic industrial environments. It would be interesting to extend the proposed algorithms to solve the IPDSP with batch delivery to multiple customers at different locations, such that vehicle routing decisions have to be made. Since inventory buffers exist between the production and distribution stages, another interesting extension of IPDSP is to take into account inventory characteristics, such as holding cost and inventory capacity. Furthermore, apart from the proposed three simple learning operations of AHLBGA, more sophisticated human learning mechanisms may be developed to enhance the search ability of GA.

Acknowledgements We would like to thank the anonymous reviewers for their constructive comments. This research is supported by National Key R&D Program of China (No. 2018YFB1700600), National Natural Science Foundation of China (No. 71671131), Natural Science Foundation of Hubei Province (No. 2019CFB487), and the Fundamental Research Funds for the Central Universities.

References

1. Agnetis A, Aloulou MA, Fu LL (2014) Coordination of production and interstage batch delivery with outsourced distribution. Eur J Oper Res 238(1):130–142
2. Amaya JE, Cotta C, Fernández-Leiva AJ, García-Sánchez P (2020) Deep memetic models for combinatorial optimization problems: application to the tool switching problem. Memet Comput 12(1):3–22
3. Bard JF, Nananukul N (2010) A branch-and-price algorithm for an integrated production and inventory routing problem. Comput Oper Res 37(12):2202–2217
4. Basir SA, Mazdeh MM, Namakshenas M (2018) Bi-level genetic algorithms for a two-stage assembly flow-shop scheduling problem with batch delivery system. Comput Ind Eng 126:217–231
5. Behnamian J, Ghomi SF (2016) A survey of multi-factory scheduling. J Intell Manuf 27(1):231–249
6. Bektur G, Saraç T (2019) A mathematical model and heuristic algorithms for an unrelated parallel machine scheduling problem with sequence-dependent setup times, machine eligibility restrictions and a common server. Comput Oper Res 103:46–63
7. Cai X, Sun Y, Cui Z, Zhang W, Chen J (2019) Optimal LEACH protocol with improved bat algorithm in wireless sensor networks. KSII Trans Internet Inf Syst 13(5):2469–2490
8. Cai X, Zhang J, Liang H, Wang L, Wu Q (2019) An ensemble bat algorithm for large-scale optimization. Int J Mach Learn Cybern 10(11):3099–3113
9. Chang YC, Chang KH, Chang TK (2013) Applied column generation-based approach to solve supply chain scheduling problems. Int J Prod Res 51(13):4070–4086
10. Chen SJ, Lin LI (1999) Reducing total tardiness cost in manufacturing cell scheduling by a multi-factor priority rule. Int J Prod Res 37(13):2939–2956
11. Chen ZL, Vairaktarakis GL (2005) Integrated scheduling of production and distribution operations. Manag Sci 51(4):614–628
12. Chen ZL (2010) Integrated production and outbound distribution scheduling: review and extensions. Oper Res 58(1):130–148
13. Cheng BY, Leung JT, Li K, Yang SL (2015) Single batch machine scheduling with deliveries. Nav Res Logist 62(6):470–482
14. Cheng BY, Leung JYT, Li K (2015) Integrated scheduling of production and distribution to minimize total cost using an improved ant colony optimization method. Comput Ind Eng 83:217–225
15. Cheng T, Zhong J (2020) An efficient memetic genetic programming framework for symbolic regression. Memet Comput 12(4):299–315
16. Chou FD (2009) An experienced learning genetic algorithm to solve the single machine total weighted tardiness scheduling problem. Expert Syst Appl 36(2):3857–3865
17. Edis EB, Oguz C, Ozkarahan I (2013) Parallel machine scheduling with additional resources: notation, classification, models and solution methods. Eur J Oper Res 230(3):449–463
18. Engin O, Günaydin C (2011) An adaptive learning approach for no-wait flowshop scheduling problems to minimize makespan. Int J Comput Intell Syst 4(4):521–529
19. Ganji M, Kazemipoor H, Molana SMH, Sajadi SM (2020) A green multi-objective integrated scheduling of production and distribution with heterogeneous fleet vehicle routing and time windows. J Clean Prod 259:120824
20. Gao S, Qi L, Lei L (2015) Integrated batch production and distribution scheduling with limited vehicle capacity. Int J Prod Econ 160:13–25
21. Guo Z, Zhang D, Leung SYS, Shi L (2016) A bi-level evolutionary optimization approach for integrated production and transportation scheduling. Appl Soft Comput 42:215–228
22. Guo Z, Shi L, Chen L, Liang Y (2017) A harmony search-based memetic optimization model for integrated production and transportation scheduling in MTO manufacturing. Omega 66:327–343
23. Hall NG, Potts CN (2003) Supply chain scheduling: batching and delivery. Oper Res 51(4):566–584
24. Hassanzadeh A, Rasti-Barzoki M, Khosroshahi H (2016) Two new meta-heuristics for a bi-objective supply chain scheduling problem in flow-shop environment. Appl Soft Comput 49:335–351
25. Hong TY, Chien CF, Wang HK, Guo HZ (2018) A two-phase decoding genetic algorithm for TFT-LCD array photolithography stage scheduling problem with constrained waiting time. Comput Ind Eng 125:200–211
26. Liu X, Chung TP (2017) An outsourcing-scheduling problem in a two-stage supply chain via improved immunoglobulin-based artificial immune system. Comput Ind Eng 113:819–830
27. Liu X, Lu S, Pei J, Pardalos PM (2018) A hybrid VNS-HS algorithm for a supply chain scheduling problem with deteriorating jobs. Int J Prod Res 56(17):5758–5775
28. Liu H, Guo Z, Zhang Z (2020) A hybrid multi-level optimisation framework for integrated production scheduling and vehicle routing with flexible departure time. Int J Prod Res. https://doi.org/10.1080/00207543.2020.1821927
29. Jamili N, Ranjbar M, Salari M (2016) A bi-objective model for integrated scheduling of production and distribution in a supply chain with order release date restrictions. J Manuf Syst 40:105–118
30. Kazemi H, Mazdeh MM, Rostami M (2017) The two stage assembly flow-shop scheduling problem with batching and delivery. Eng Appl Artif Intell 63:98–107
31. Marandi F, Fatemi Ghomi SMT (2018) Integrated multi-factory production and distribution scheduling applying vehicle routing approach. Int J Prod Res 57:722–748
32. Mazdeh MM, Sarhadi M, Hindi KS (2008) A branch-and-bound algorithm for single-machine scheduling with batch delivery and job release times. Comput Oper Res 35(4):1099–1111
33. Moons S, Ramaekers K, Caris A, Arda Y (2017) Integrating production scheduling and vehicle routing decisions at the operational decision level: a review and discussion. Comput Ind Eng 104:224–245

34. Mujtaba H, Kendall G, Baig AR, Özcan E (2015) Detecting change and dealing with uncertainty in imperfect evolutionary environments. Inf Sci 302:33–49
35. Pan QK, Suganthan PN, Liang JJ, Tasgetiren MF (2011) A local-best harmony search algorithm with dynamic sub-harmony memories for lot-streaming flow shop scheduling problem. Expert Syst Appl 38(4):3252–3259
36. Pan QK, Gao L, Wang L, Liang J, Li XY (2019) Effective heuristics and metaheuristics to minimize total flowtime for the distributed permutation flowshop problem. Expert Syst Appl 124:309–324
37. Potts CN (1980) Analysis of a heuristic for one machine sequencing with release dates and delivery times. Oper Res 28(6):1436–1441
38. Pundoor G, Chen ZL (2005) Scheduling a production–distribution system to optimize the tradeoff between delivery tardiness and distribution cost. Nav Res Logist 52(6):571–589
39. Ribas I, Leisten R, Framiñan JM (2010) Review and classification of hybrid flow shop scheduling problems from a production system and a solutions procedure perspective. Comput Oper Res 37(8):1439–1454
40. Rostami M, Kheirandish O, Ansari N (2015) Minimizing maximum tardiness and delivery costs with batch delivery and job release times. Appl Math Model 39(16):4909–4927
41. Ruiz R, Vázquez-Rodríguez JA (2010) The hybrid flow shop scheduling problem. Eur J Oper Res 205(1):1–18
42. Selvarajah E, Zhang R (2014) Supply chain scheduling at the manufacturer to minimize inventory holding and delivery costs. Int J Prod Econ 147:117–124
43. Wang H, Lee CY (2005) Production and transport logistics scheduling with two transport mode choices. Nav Res Logist 52(8):796–809
44. Wang K, Choi SH, Qin H (2014) An estimation of distribution algorithm for hybrid flow shop scheduling under stochastic processing times. Int J Prod Res 52(24):7360–7376
45. Wang K, Choi SH, Lu H (2015) A hybrid estimation of distribution algorithm for simulation-based scheduling in a stochastic permutation flowshop. Comput Ind Eng 90:186–196
46. Wang K, Ma WQ, Luo H, Qin H (2016) Coordinated scheduling of production and transportation in a two-stage assembly flowshop. Int J Prod Res 54(22):6891–6911
47. Wang K, Luo H, Liu F, Yue X (2018) Permutation flow shop scheduling with batch delivery to multiple customers in supply chains. IEEE Trans Syst Man Cybern Syst 48(10):1826–1837
48. Wang K, Qin H, Huang Y, Luo M, Zhou L (2021) Surgery scheduling in outpatient procedure centre with re-entrant patient flow and fuzzy service times. Omega 102:102350
49. Wang L, Pei J, Wen Y, Pi J, Fei M, Pardalos PM (2018) An improved adaptive human learning algorithm for engineering optimization. Appl Soft Comput 71:894–904
50. Deng L, Zhang L, Sun H, Qiao L (2020) DSM-DE: a differential evolution with dynamic speciation-based mutation for single-objective optimization. Memet Comput 12(1):73–86
51. Wu X, Che A (2019) A memetic differential evolution algorithm for energy-efficient parallel machine scheduling. Omega 82:155–165
52. Yang S, Xu Z (2020) The distributed assembly permutation flowshop scheduling problem with flexible assembly and batch delivery. Int J Prod Res. https://doi.org/10.1080/00207543.2020.1757174
53. Yazdani M, Gohari S, Naderi B (2015) Multi-factory parallel machine problems: improved mathematical models and artificial bee colony algorithm. Comput Ind Eng 81:36–45
54. Ying KC, Lin SW (2018) Minimizing makespan for the distributed hybrid flowshop scheduling problem with multiprocessor tasks. Expert Syst Appl 92:132–141
55. Yilmaz OF, Pardalos PM (2017) Minimizing average lead time for the coordinated scheduling problem in a two-stage supply chain with multiple customers and multiple manufacturers. Comput Ind Eng 114:244–257
56. Yu C, Semeraro Q, Matta A (2018) A genetic algorithm for the hybrid flow shop scheduling with unrelated machines and machine eligibility. Comput Oper Res 100:211–229
57. Zhang Z, Zheng L, Li N, Wang W, Zhong S, Hu K (2012) Minimizing mean weighted tardiness in unrelated parallel machine scheduling with reinforcement learning. Comput Oper Res 39(7):1315–1324