Information Sciences 753 (2026) 123602

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

# A Bi-Level structured decomposition algorithm for intelligent Multi-Factory scheduling with batch delivery

Fateme Marandi [a], Madjid Tavana [b,c,*], Ying Xie [d]

[a] Faculty of Business, Law and Social Sciences, Birmingham City University, Birmingham, UK
[b] Business Systems and Analytics Department, Distinguished Chair of Business Analytics, La Salle University, Philadelphia, USA
[c] Business Information Systems Department, Faculty of Business Administration and Economics, University of Paderborn, Paderborn, Germany
[d] School of Management, Cranfield University, Bedfordshire, UK

ARTICLE INFO

*Keywords:*
Multi-factory scheduling
Batch delivery
Bi-levelstructured decomposition algorithm
Mixed-integer programming
Intelligent optimization
Supply chain decision support

ABSTRACT

Fluctuations in global customer demand increase the complexity of manufacturing systems and require adaptive production scheduling across multiple factories. Intelligent multi-factory scheduling is therefore essential for timely responses to demand variability and effective coordination between production and delivery decisions. This paper investigates a multi-factory scheduling problem with batch delivery and proposes a novel mixed-integer programming formulation that minimizes total system cost, including tardiness, holding, and batching costs. To efficiently solve the resulting large-scale problem, a bi-level structured decomposition algorithm (BLSDA) is developed, consisting of two interdependent subproblems: an upper-level scheduling problem and a lower-level batching problem. Four BLSDA variants are implemented by integrating CPLEX and simulated annealing at different levels of the hierarchy. This hybrid design combines deterministic optimization with stochastic search to enhance exploration, avoid local optima, and improve solution quality. The proposed approach is validated using a real-world zinc supply chain case. Computational experiments and statistical analyses benchmark the BLSDA variants against exact and lower-bound solutions, demonstrating their ability to generate high-quality solutions within reasonable computational time. Sensitivity analyses further yield managerial insights into delivery responsiveness and scheduling robustness, highlighting the practical value of the proposed intelligent optimization framework.

## 1. Introduction

Increasing global competition, decentralized production structures, and rapidly changing customer demands have compelled companies to optimize the performance of the entire supply chain (SC) rather than focusing solely on isolated manufacturing operations [1]. Modern SCs consist of interconnected networks of suppliers, manufacturers, distribution facilities, and customers, together with continuous flows of materials, products, and information across geographically distributed production environments [2]. In such dynamic and intelligent manufacturing systems, effective coordination of scheduling, production, inventory, and distribution decisions has become essential for improving operational responsiveness, reducing operating and transportation costs, minimizing

* Corresponding author at: Business Systems and Analytics Department, Distinguished Chair of Business Analytics, La Salle University, Philadelphia, PA 19141, United States.

*E-mail addresses:* Fateme.Marandi@bcu.ac.uk (F. Marandi), tavana@lasalle.edu (M. Tavana), Ying.Xie@cranfield.ac.uk (Y. Xie).

https://doi.org/10.1016/j.ins.2026.123602
Received 29 December 2025; Received in revised form 6 May 2026; Accepted 6 May 2026
Available online 10 May 2026
0020-0255/

inventory waiting times, and enhancing overall system efficiency and service performance [3].

The primary business context for multi-factory scheduling and batch distribution arises in manufacturing systems characterized by complex production processes, alternative routings, and coordinated order fulfillment requirements [4]. Furthermore, distributed production networks require integrated planning and scheduling approaches to improve reliability, resilience, and operational efficiency across multiple factories [5]. Multi-factory scheduling, therefore, involves coordinating production activities across geographically distributed factories while considering heterogeneous processing capabilities, local resource availability, and overall system objectives [6]. In addition, integrated multi-factory scheduling approaches increasingly incorporate distribution and transportation constraints to jointly optimize production and delivery decisions across interconnected supply chain networks [7].

For example, the automotive industry increasingly relies on advanced, intelligent manufacturing systems, including robotics and automated decision-support technologies, to improve production efficiency and operational performance in complex manufacturing environments [8]. Similarly, geographically distributed production systems can optimize manufacturing and delivery operations across multiple factory locations by integrating scheduling and routing decisions, thereby reducing completion times and tardiness [9]. Moreover, coordinated batch delivery and vehicle routing play an essential role in distributed manufacturing environments by improving service levels, reducing delivery delays, and enhancing the efficiency of integrated production–distribution operations [10].

Interconnection among factories in multi-factory SCs introduces considerable operational complexity, where disruptions or material shortages at one facility can propagate throughout the network and lead to production delays or stoppages. Consequently, production scheduling and transportation activities among factories must be effectively synchronized to reduce inventory accumulation, in-process holding costs, and delivery inefficiencies. In many distributed manufacturing environments, the output of one factory serves as the input material for subsequent production stages, making transportation coordination and delivery timing critical to overall system performance. In this context, batch delivery is an effective operational strategy for coordinating production and distribution, improving transportation efficiency, reducing delivery and fuel costs, and enhancing overall supply chain performance [11].

This study considers multi-factory scheduling with batch delivery problems (MFSBDP) in a multi-tier SC context. The core operational tasks in the SC under investigation include job scheduling across factories and allocating in-process jobs with identical processing routes to corresponding batches. In the SC network under consideration, while batching in-process jobs with identical processing routes may increase tardiness and holding costs, it concurrently decreases the transportation costs, namely, batching costs. Therefore, a trade-off must be made between batching, tardiness, and holding costs to minimize the total cost.

This study develops a rigorous mixed-integer programming (MIP) framework that jointly optimizes multi-factory scheduling and batch delivery decisions within a unified formulation, enabling simultaneous coordination of production and transportation activities using CPLEX. Given the strongly NP-hard nature of the MFSBDP, a bi-level structured decomposition algorithm (BLSDA) is proposed that systematically partitions the original problem into two tightly coupled subproblems: a scheduling problem at the upper (coordination) level and a batching problem at the lower (operational) level, while explicitly preserving their interdependence.

Within this structure, each subproblem is solved at its respective decision level, and its solution iteratively informs the other, forming a feedback-driven coordination mechanism that aligns production schedules with batching and transportation decisions and ensures consistency across the supply chain. The integration of CPLEX with simulated annealing (SA) establishes a complementary exact–metaheuristic hybrid solution approach, where CPLEX provides strong optimality guarantees and high-quality bounds, while SA introduces diversification and stochastic search capabilities that enable effective exploration of complex solution spaces and avoidance of local optima.

SA is embedded within the BLSDA to solve each subproblem, resulting in four algorithmic variants based on different combinations of SA and CPLEX. This design yields a flexible and scalable solution architecture that balances optimality, computational efficiency, and robustness across varying problem sizes and operational conditions.

To evaluate the effectiveness of the proposed approach, a real-world zinc supply chain network is used to generate multiple test scenarios within the MFSBDP context, reflecting different levels of scheduling complexity and batching requirements. The BLSDA is then applied to resolve scheduling and batching decisions across these scenarios jointly. The computational results demonstrate that the proposed framework consistently achieves optimal or near-optimal solutions with strong computational performance, even for large-scale, real-world instances characterized by significant combinatorial complexity and tightly coupled decision structures.

Previous studies, including Gharaei and Jolai [12], have examined integrated multi-factory scheduling and distribution problems or proposed decomposition-based approaches to coordinate interrelated operational decisions. However, these studies do not develop a rigorous bi-level MIP formulation that explicitly captures hierarchical decision-making by structuring the problem into distinct yet interdependent layers, where network-level configuration and coordination decisions are modeled at the upper level, and detailed factory-level scheduling with explicit batch-delivery interactions are modeled at the lower level within a unified optimization framework. In particular, prior work typically treats integration either in a single-level formulation or through loosely coupled decomposition schemes, without formally embedding the strategic–operational hierarchy and its feedback structure into the mathematical model itself.

The bi-level structure proposed in this study represents a realistic hierarchical decision architecture for multi-factory supply chains, in which higher-level coordination decisions guide system-wide resource allocation and policy setting, while lower-level operational decisions govern detailed scheduling and batching execution under these strategic directives. This structure enables the model to capture the bidirectional dependency between coordination and execution, where upper-level decisions shape feasible operational policies, and lower-level outcomes provide critical feedback that influences higher-level adjustments.

As a result, the proposed bi-level MIP is particularly well-suited for complex multi-factory environments in which decision-making authority is distributed across organizational layers. Yet, operational performance depends on tight synchronization between planning and execution activities. By explicitly modeling this hierarchy, the framework provides a more faithful representation of real-world supply chain decision processes, enhances coordination across factories, and improves the quality and practical applicability of the resulting solutions.

In summary, the main contributions of this paper are as follows.

- Introduce a novel network configuration–driven multi-factory scheduling problem that integrates batch delivery and reentrant process characteristics, capturing the operational complexity and inter-factory dependencies inherent in real-world supply chain networks.
- Develop a rigorous MIP formulation that jointly optimizes scheduling and batching decisions within a unified framework, enabling the derivation of high-quality optimal solutions using CPLEX.
- Propose a bi-level structured decomposition algorithm (BLSDA) that systematically exploits the problem's hierarchical structure, effectively coordinating upper-level scheduling and lower-level batching decisions to achieve computational efficiency and scalability for large-scale instances.
- Validate the proposed framework through a real-world zinc supply chain case study, generating practical managerial insights on cost trade-offs, coordination policies, and the impact of batching and scheduling integration on overall supply chain performance.

The remainder of the paper is structured as follows. In Section 2, the related studies are reviewed. Section 3 defines the problem and develops the mathematical model. The solution approach is presented in Section 4. Section 5 provides the case study and computational results. Finally, Sections 6 and 7 are devoted to research and managerial implications, respectively. Section 8 provides conclusions and future research directions.

## 2. Literature review

This section reviews the literature on multi-factory scheduling and batch delivery in supply chain systems. It begins by examining studies on multi-factory scheduling, emphasizing coordination among distributed production facilities. Next, it explores network configuration approaches that support efficient resource allocation and information flow across multi-factory supply chains. The review then focuses on research integrating scheduling with batch delivery decisions, highlighting the trade-offs between production efficiency and transportation performance. Finally, the identified research gaps are synthesized to position the present study within the broader context of multi-factory optimization and supply chain management.

### 2.1. Multi-factory scheduling

Since 1981, researchers have conducted extensive surveys of scheduling in multi-factory production networks. Bagheri Rad and Behnamian [1] review existing studies on multi-factory scheduling and summarize them to inform future research directions. Additionally, they discussed the virtual alliance and Industry 4.0 in multi-factory scheduling, as well as the importance of information sharing in a real-life data-driven approach in this field. Lohmer and Lasch [13] provide a systematic review of the literature on production planning and scheduling in a multi-factory environment. The authors illustrate the increasing trend in recent research in this area, driven by market uncertainty and new technological approaches, such as Industry 4.0 [14]. The research not only reviewed 128 studies but also proposed research opportunities, such as integrated planning and real-world objectives, given the dynamic nature of Industry 4.0. Behnamian and Ghomi [15] propose a multi-factory scheduling problem in which each factory has a parallel-machine scheduling problem. The authors aim to identify suitable factories for processing each job and schedule the allocated jobs at factories to minimize a multi-objective function that encompasses total earliness, tardiness, and completion duration.

Behnamian and Ghomi [16] provide an extensive review of multi-factory scheduling problems, categorizing shop environments, multi-factory configurations, and solution approaches. They also suggest some guidelines for future research. Peterson, Harjunkoski, et al. [17] studied multi-factory scheduling with crane delivery. They present a new heuristic based on a decision tree of possible crane system states. Sun et al. [18] investigate integrated multi-factory scheduling in the context of maritime transportation. They propose a novel genetic algorithm and a fuzzy controller to minimize the total operating costs. Marandi and Fatemi Ghomi [7] address integrated multi-factory scheduling with vehicle routing problems to meet customers' demands. They propose a novel improved imperialist competitive algorithm to minimize delivery and tardiness costs. Gharaei and Jolai [12] studied parallel multi-factory scheduling with batch delivery to minimize two objectives of total tardiness and distribution costs. They also propose a new multi-objective approach that combines the bees algorithm with a decomposition approach.

### 2.2. Network configuration in multi-factory supply chains

Generally, there are three well-known configurations in a multi-factory environment: parallel, serial, and network configurations. In a parallel configuration, each factory can produce the final products for customers. In contrast, factories in a serial configuration can produce final or intermediate products for shipment to another factory along the same processing route; see, e.g., Santos et al. [19], Karimi and Davoudpour [20], and Kazemi et al. [21].

In network configuration [7], each factory can produce final or intermediate products processed along different routes, with the possibility of reentrant processing, which has received less attention. Due to the number of factories along the SC, scheduling activities are significantly more complicated in the latter configuration [16]. In addition, factories' connections within such a complex SC network further complicate this problem, which is studied in detail in the current paper. The scheduling activities of network configuration are significantly more complex than those of other configurations due to the structure of factories and non-identical processing routes, which often involve reentrant processing in some factories. In fact, network configuration is the combination of parallel and serial configurations. There is no straightforward flow of material among factories that can be adapted in parallel or in series. It varies based on the route of operations, which can differ for each job. To schedule factories across various regions alongside other flows, multi-factory scheduling is applied to optimize the objectives.

### 2.3. Multi-factory scheduling and batch delivery

Given the importance of moving products among factories, transportation costs play a significant role in optimizing multi-factory systems. Thus, production and transportation in such systems are of special significance in the literature. Santos and Magazine [22] first addressed the batching problem for single-machine scheduling. Since then, batching problems have increasingly been studied by many researchers. Potts and Kovalyov [23] comprehensively review batch delivery in machine scheduling problems, with respect to models for different shop scheduling and design settings, as well as efficient dynamic programming for solving these problems. Yin et al. [24] consider scheduling and batch delivery with due date assignment. They show a significant reduction in total costs by batch delivery.

Chung et al. [25] studied multi-factory scheduling with intermediate batch delivery to minimize makespan by presenting a

**Table 1**
Detailed review of studies in production scheduling and delivery areas.

| Author(s) | Main features | Problem | | Mathematical model | Solution approach | Objective function |
|---|---|---|---|---|---|---|
| | | Scheduling | Batching | | | |
| Sauer and Appelrath [34] | Multi-factory, parallel configuration | ✓ | | Single level, MIP | Fuzzy heuristic | Robust scheduling |
| Guinet [35] | Multi-factory, parallel configuration | ✓ | | Single level | Heuristic | Variable and fixed costs |
| Jia, Nee et al. [36] | Multi-factory, parallel configuration | ✓ | | Single level | GA | $C_{max}$ |
| Jia, Fuh et al. [37] | Multi-factory, parallel configuration | ✓ | | | GA | Operating costs |
| Kaminsky and Kaya [38] | Multi-factory, network configuration | ✓ | | | Heuristic | $C_{max}$ |
| Torabzadeh and Zandieh [39] | Single-factory | ✓ | | Single level, MIP | CSA | $C_{max}$ |
| Nishi, Hiranaka et al. [40] | Single-factory | ✓ | | Bi-level, MIP | BLD & Lagrangian | Tardy costs |
| Chung et al. [25] | Multi-factory, parallel configuration | ✓ | | Single level, MIP | GA | |
| H'Mida and Lopez [41] | Multi-factory, network configuration | ✓ | | Single level, CSP | | $C_{max}$ |
| Behnamian and Ghomi [6] | Multi-factory, parallel configuration | ✓ | | Single level, MIP | GA-LS | $C_{max}$ |
| Peterson et al. [17] | Multi-factory, parallel configuration | ✓ | | Single level, MIP | Heuristic | |
| Matusiak et al. [42] | Single-factory | ✓ | ✓ | Single level | A* algorithm & SA | Total delivery cost |
| Santos et al. [19] | Multi-factory, serial configuration | ✓ | ✓ | Single level, MIP | | $C_{max}$ |
| Nikzad et al. [43] | Single-factory | ✓ | | Single level | ICA-SA | $C_{max}$ |
| Sun et al. [18] | Multi-factory, parallel configuration | ✓ | | Single level, MIP | GA | Operating costs |
| Cheng et al. [44] | Single-factory | ✓ | | | DP | $C_{max}$ |
| Guo et al. [45] | Single-factory | ✓ | ✓ | Bi-level, MINLP | Bi-level evolutionary | Operating costs |
| Noroozi et al. [46] | Single-factory | ✓ | ✓ | Single level, MIP | PSO-GA | Maximize total benefit |
| Frazzon et al. [47] | Single-factory | ✓ | ✓ | Single level, MIP | Simulation | Transport, production, and storage costs |
| Gharaei and Jolai [12] | Multi-factory, parallel configuration | ✓ | ✓ | Single-level, MIP | Bees algorithm | Total transport and tardy costs |
| Behnamian and Fatemi [18] | Multi-factory, parallel configuration | ✓ | | Single-level, MIP | Heuristic | Total tardiness and earliness of jobs, and total completion time |
| Kazemi et al. [21] | Multi-factory, parallel configuration | ✓ | | Single-level, MIP | Heuristic | $C_{max}$ |
| **Current study** | Multi-factory, network configuration | ✓ | ✓ | Bi-level, MIP, LP | BLSDA & SA | Batching, tardiness, and holding costs |

BLD (bi-level decomposition algorithm), CSA (cloud-based simulated annealing), CSP (constraint satisfaction problem), DP (dynamic programming), GA (genetic algorithm), ICA (imperialist competitive algorithm), LP (linear programming), LS (local search), MINLP (mixed-integer non-linear programming), MIP (mixed-integer linear programming), PSO (particle swarm optimization), and SA (simulated annealing).

modified genetic algorithm. Agnetis et al. [26] surveyed semi-finished products scheduling with batch delivery within two transportation modes to minimize transportation costs. Karimi and Davoudpour [20] present a multi-factory scheduling problem in a serial configuration with batch delivery and propose a branch-and-bound method. Wang et al. [27] studied the scheduling problem with batch delivery to customers to minimize the mean sum of arrival times, and proposed two heuristics. Gharaei and Jolai [12] studied parallel multi-factory scheduling with batch delivery to minimize two objectives: total tardiness and distribution costs, using the bees algorithm in a decomposition approach. Ganji et al. [28] studied green multi-objective integrated production and distribution scheduling applying the vehicle routing problem and batch delivery. The recent surge in attention to this area in the literature motivates us to consider multi-factory scheduling and batch delivery in the current study.

Although exact methods have been employed to obtain optimal solutions for small-sized problems [29], heuristic approaches have also been developed for integrated production and distribution scheduling problems [30]. Furthermore, metaheuristic methods have been proposed for collaborative supply chain scheduling environments [31]. As multi-factory scheduling is among the most complex scheduling problems, the decomposition-based approach can be applied to break the problem down into smaller parts, thereby addressing the intractability [16]. This innovative and superior approach is used in this study.

Beyond algorithmic improvement, this study contributes to the theoretical understanding of hierarchical decision coordination in distributed production systems. The proposed framework demonstrates how network-level configuration and factory-level scheduling decisions can be integrated through a decomposition-based coordination mechanism that relies on iterative information exchange. This structure provides insights into the design of hierarchical optimization architectures for complex production networks.

### *2.4. Gap analysis*

We have compiled a comparison of research problems, mathematical models, and solution approaches in Table 1 to provide a clear overview of the research landscape. Upon close examination of Table 1, it becomes apparent that while some papers focus on investigating scheduling and delivery within a single-factory setting, a substantial number of studies delve into the complexities of multi-factory configurations, including network configurations. These multi-factory scenarios demand more intricate and sophisticated approaches to explore the complicated facets of such systems. Although many mathematical models were developed to address multi-factory scenarios, several research gaps still exist:

1. As indicated in Table 1, most research focuses on single-level mathematical modeling, showing a significant gap in the exploration of bi-level mathematical models. However, bi-level mathematical modeling offers a more comprehensive and realistic approach by accounting for interrelated decision-makers, particularly within hierarchical structures where decisions at the upper level influence those at the lower level [32]. This structure better reflects the inherent complexity of optimizing a vast range of real-world problems. In the current study, the proposed bi-level model not only considers multi-factory scheduling in a network configuration that is practical for complex production processes, but also integrates batch deliveries between factories into the production routes. Our review of the published studies shows that no study has investigated these assumptions. The term "bi-level" refers to the hierarchical structure of the decision model, not to the specific solution technique used to solve it. In the proposed framework, batching decisions are optimized conditionally on scheduling decisions, and the evaluation of upper-level decisions depends on the optimal reaction of the lower level. While the solution procedure involves iterative and cooperative computation, the hierarchical nesting of decision problems remains intact.
2. Multi-factory supply chains face complicated logistical difficulties when scheduling and coordinating deliveries through a multi-factory environment. Existing studies primarily focus on broad supply chain optimization problems [33] while neglecting these detailed concerns, thereby leaving a research gap in this area. Moreover, the batching assumption, as a key feature of the current study, can reduce the company's transportation costs. However, it may increase the makespan because a batch can be released when the last unfinished product allocated to the batch is produced. Another problem raised by the batching approach concerns the holding costs of unfinished products awaiting batch completion. Thus, a compromise must be made between batching costs, tardiness, and holding costs.
3. The proposed network configuration differs fundamentally from classical flexible job shops and distributed flow shops. In flexible job shops, routing flexibility is limited to machine selection within a single facility. In distributed flow shops, each job is assigned to one factory and remains there throughout processing. In contrast, our model enables operation-level routing across multiple factories, allowing intermediate work-in-process (WIP) to be transferred in batches. These batch-delivery decisions create strong coupling between production scheduling and transportation timing, resulting in an integrated production–distribution scheduling problem.
4. While many studies focus on operating costs as their primary cost function, it is essential to consider costs caused by tardiness, holding costs, and batching costs. In our research, we place particular emphasis on minimizing tardiness costs. Tardiness costs are incurred when a job is delivered after its scheduled due date. Our objective is to ensure that products are delivered as swiftly as possible to mitigate these tardiness costs effectively. Furthermore, batching decisions significantly impact tardiness, holding, and batching costs. Specifically, when unfinished products are grouped into the same batch, it delays the start times of other processes in the following factory. Therefore, batching decisions are a critical aspect of our study, given their profound influence on the overall costs mentioned earlier.
5. SA and CPLEX have been used often to solve scheduling problems. However, in this study, they are particularly customized and developed by applying batch local search and three different local search scheduling methods to improve solutions. The proposed BLSDA consists of two subproblems decomposed into two levels. SA and CPLEX are used to tackle each of the interacting

subproblems. The BLSDA solution approach offers numerous benefits when investigating complicated optimization problems. This approach can provide a more efficient solution in terms of quality and computational time via separating the problem into two levels. This decomposition leads to a deeper understanding of the problem and its potential solutions. The combination of SA and CPLEX via BLSDA yields high-quality solutions, which are evaluated on several random and real-life test instances.

6. The contributions of this study are threefold. First, a multi-factory production–distribution scheduling model is developed that integrates operation-level routing decisions and batch delivery of WIP between factories. Second, a BLSDA is proposed to efficiently solve the resulting large-scale problem by separating network configuration decisions from factory-level scheduling. Finally, the proposed framework is validated through its application to a zinc supply chain involving multiple processing plants, demonstrating its effectiveness for coordinated scheduling in distributed metallurgical production networks.

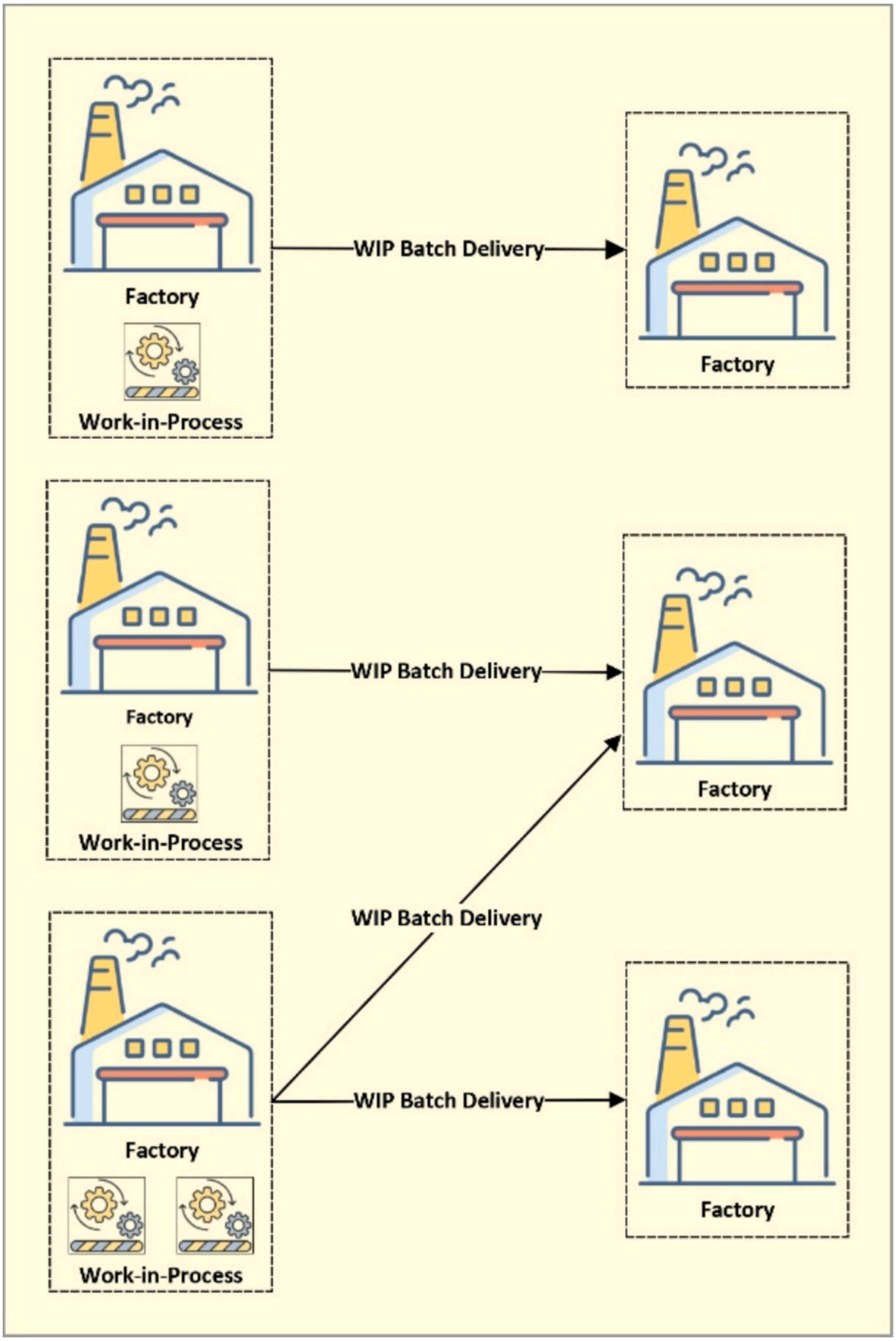


**Fig. 1.** Technical schematic of multi-factory scheduling with batch delivery problem.

## 3. Problem description

This study investigates the MFSBDP within an SC characterized by a network configuration of interconnected factories. A set of jobs is processed through multiple operations, potentially following reentrant processing routes, where jobs may revisit certain stages and be handled across different factories. After processing at one factory, intermediate products are transferred to downstream factories as input materials. These intermediate products are referred to as WIP inventories. To reduce transportation costs, WIPs are consolidated and transferred between factories using a limited number of capacity-constrained batches. The following assumptions govern the operational setting:

- **Precedence constraint:** An operation of a job at a given factory cannot begin until the required WIP has been received from the upstream (departure) factory.
- **Batch compatibility:** WIPs that share identical processing routes between two factories can be grouped into the same batch and transported jointly.
- **Indivisibility of WIPs:** Each WIP must be assigned to a single batch and cannot be split across multiple batches.

Under these conditions, the completion time of each batch is determined by the finishing time of the last WIP assigned to that batch at the originating factory. Timely delivery is critical, and jobs are expected to be completed and delivered before their due dates; otherwise, tardiness costs are incurred.

Batching decisions introduce an inherent trade-off. While consolidating WIPs into batches reduces transportation (batching) costs, it may delay the transfer of some WIPs, thereby postponing their start times at downstream factories. This delay can increase both tardiness and holding costs. Consequently, batching is not merely a logistical decision but a central coordination mechanism that directly influences system-wide performance.

Accordingly, the objective of this study is to jointly determine optimal production schedules and batch assignments for WIPs to minimize total cost, including tardiness, holding, and batching costs, within a tightly integrated multi-factory supply chain environment.

Fig. 1 illustrates the SC network of MFSBDP. It is conjectured that the batching assumption, as a key feature of the current study, can reduce the company's transportation costs. However, it may increase the makespan because a batch can be released when the last WIP allocated to the batch is produced. Another problem with the batching approach concerns the holding costs of WIPs awaiting batch completion. Consequently, a compromise must be reached among batching costs, tardiness, and holding costs, as discussed further in this paper. The assumptions and notations of MFSBDP are listed as follows.

- All jobs are available at the start of the planning era.
- The processing of each job in each factory cannot be interrupted.
- At most one job at a time can be processed in each factory.
- A job cannot be processed in more than one factory at the same time.
- Transportation times between factories are considered.
- All parameter data are known and deterministic.
- A batch has a limited capacity and can be released when the last WIP of the batch is produced.
- A WIP consists of incomplete products produced by an operation that cannot be divided into two or more parts.

The mathematical notations are presented below:

| **Indices** | |
|---|---|
| $f, f'$ | Index for factory $f = 1, 2, \ldots, F$ |
| $i, i'$ | Index for job $i = 1, 2, \ldots, N$ |
| $j, j'$ | Index for operation $j = 1, 2, \ldots, M_i$ |
| $B$ | Index for batch $b = 1, 2, \ldots, B$ |
| **Parameters** | |
| $F$ | Number of factories working in the SC network |
| $N$ | Number of jobs, ready for processing at the beginning of the planning period |
| $M_i$ | Number of operations belonging to job $i$ |
| $B$ | maximum number of available batches, which is equal to $\sum_{i=1}^{N} M_i$ |
| $o_{ij}$ | $j^{\text{th}}$ operation of job $i$ |
| $p_{ij}$ | Processing time of operation $o_{ij}$ |
| $\theta_{ij}$ | Volume of each unit of job $i$ produced at the $j^{\text{th}}$ operation |
| $\tau_{ij}$ | Holding cost of operation $o_{ij}$ while being transported between factories or waiting for batch completion as a WIP inventory |
| $r_{ij}$ | Route indicator which is equal to 1 if operation $o_{ij}$ is processed in factory $f$ or 0 otherwise |
| $t_{ff'}$ | Transportation time between factories $f$ and $f'$ |
| $q_i$ | Demand quantity of job $i$ |
| $d_i$ | Due date of delivering the job $i$ |
| $\alpha_i$ | Tardiness cost of late delivery of the job $i$ per unit time |
| $c^b$ | Capacity of batch $b$ |

(continued on next page)

(*continued*)

| **Indices** | |
|---|---|
| $f, f'$ | Index for factory $f = 1, 2, \ldots, F$ |
| $\lambda^b$ | Delivery cost of batch $b$ |
| $M$ | A very large positive number |
| **Variables** | |
| $ST_{ij}$ | Starting time of operation $o_{ij}$ |
| $TD_i$ | Tardiness of the job $i$ |
| $Y_{ij}^b$ | Binary variable which is equal to 1 if operation $o_{ij}$ is assigned to batch $b$ or 0 otherwise |
| $X_{iji'j'}$ | Binary variable which is equal to 1 if operation $o_{ij}$ precedes operation $o_{i'j'}$ or 0 otherwise |
| $Z^b$ | Binary variable which is equal to 1 if at least one product is assigned to batch $b$ or 0 otherwise |
| TC | Objective function of the model **M**, the sum of holding, tardiness, and batching costs |
| $TC_1$ | Objective function of model $\mathbf{M_1}$, the sum of holding and tardiness costs |
| $TC_2$ | Objective function of model $\mathbf{M_2}$, the batching costs |
| **Constraints** | |
| (2) | Precedence constraint ensuring technological order |
| (3)–(4) | Machine capacity constraints |
| (5) | Computes the tardiness of each job |
| (6)–(7) | Determine the starting time of operations, considering batching decisions |
| (8) | Assignment constraint for batch allocation |
| (9) | Prevent multiple operations from allocating the same job to the same batch. |
| (10) | Ensures the last operation of each job is delivered individually when leaving |
| (11) | Routing compatibility constraint |
| (12) | Batch capacity constraint |

### 3.1. Mathematical model

In the following, an integrated mathematical model, called model **M**, is presented to describe the MFSBDP.
Model **M**:

$$\min TC = \sum_{i=1}^{N} \alpha_i TD_i + \sum_{i=1}^{N} \sum_{j=1}^{M_i-1} \tau_{ij}\left(ST_{i(j+1)} - \left(ST_{ij} + p_{ij}q_i\right)\right) + \sum_{b=1}^{B} \lambda^b \tag{1}$$

$$\sum_{f=1}^{F} r_{ijf}\left(ST_{ij} + p_{ij}q_i\right) + \sum_{f=1}^{F} \sum_{f'=1}^{F} r_{ijf} r_{i(j+1)f'} t_{ff'} \le \sum_{f'=1}^{F} r_{i(j+1)f'} ST_{i(j+1)}, i = 1, 2, \cdots, N; j = 1, 2, \cdots, M_i - 1 \tag{2}$$

$$M\left(2 - r_{ijf} - r_{i'j'f}\right) + M\left(1 - X_{iji'j'}\right) + ST_{i'j'} - ST_{ij} \ge p_{ij}q_i, 1 < i < i' \le N; j = 1, 2, \cdots, M_i; j' = 1, 2, \cdots, M_{i'}; f = 1, 2, \cdots, F \tag{3}$$

$$M\left(2 - r_{ijf} - r_{i'j'f}\right) + MX_{iji'j'} + ST_{ij} - ST_{i'j'} \ge p_{i'j'}q_{i'}, 1 < i < i' \le N; j = 1, 2, \cdots, M_i; j' = 1, 2, \cdots, M_{i'}; f = 1, 2, \cdots, F \tag{4}$$

$$\sum_{f=1}^{F} r_{iM_if}\left(ST_{iM_i} + p_{iM_i}q_i\right) - d_i \le TD_i, i = 1, 2, \cdots, N \tag{5}$$

$$ST_{i(j+1)} \ge ST_{i'j'} + p_{i'j'}q_{i'} + t_{ff'} + M\left(2 - Y_{ij}^b - Y_{i'j'}^b\right), i' \ne i = 1, 2, \cdots, N; j = 1, 2, \cdots, M_i; j' = 1, 2, \cdots, M_{i'}; b = 1, 2, \cdots, B \tag{6}$$

$$ST_{i'(j'+1)} \ge ST_{ij} + p_{ij}q_i + t_{ff'} + M\left(2 - Y_{ij}^b - Y_{i'j'}^b\right), i' \ne i = 1, 2, \cdots, N; j = 1, 2, \cdots, M_i; j' = 1, 2, \cdots, M_{i'}; b = 1, 2, \cdots, B \tag{7}$$

$$\sum_{b=1}^{B} Y_{ij}^b = 1, i = 1, 2, \cdots, N; j = 1, 2, \cdots, M_i \tag{8}$$

$$\sum_{j=1}^{M_i} Y_{ij}^b \le 1, i = 1, 2, \cdots, N; b = 1, 2, \cdots, B \tag{9}$$

$$Y_{ij}^b + Y_{iMi}^b \le 1, i, j = 1, 2, \cdots, M_i - 1; b = 1, 2, \cdots, B \tag{10}$$

$$Y_{ij}^b + Y_{i'j'}^b \le 1 + \sum_{f=1}^{F} \sum_{f'=1}^{F} r_{ijf} r_{i'j'f} r_{i(j+1)f'} r_{i'(j'+1)f'} i' \ne i = 1, 2, \ldots, N; j = 1, 2, \ldots, M_i; b = 1, 2, \ldots, B \tag{11}$$

$$\sum_{i=1}^{N} \sum_{j=1}^{M_i} \theta_{ij} q_i Y_{ij}^b \le c^b Z^b, b = 1, 2, \cdots, B \tag{12}$$

$$X_{iji'j'}, Y_{ij}^b, Z^b \in \{0,1\}, i' \leq i = 1,2,\cdots,N; j = 1,2,\cdots,M_i - 1; j' = 1,2,\cdots,M_{i'-1}; b = 1,2,\cdots,B \tag{13}$$

$$ST_{ij}, TD_i \geq 0, i = 1,2,\cdots,N; j = 1,2,\cdots,M_i \tag{14}$$

The objective function and constraint groups of Model M are summarized as follows:

- **Objective function (1):** Minimizes the total cost, including tardiness, holding, and batching costs.
- **Precedence and flow constraints (2):** Ensure that each operation starts only after the completion and transfer of its preceding operation to the current factory.
- **Machine capacity and sequencing constraints (3)–(4):** Guarantee that at most one job is processed at a factory at any time by determining the processing order of operations.
- **Tardiness constraints (5):** Compute the tardiness of each job based on its completion time relative to its due date.
- **Batch-dependent timing constraints (6)–(7):** Define the starting times of operations considering batch formation and inter-factory transportation. When operations are assigned to the same batch, the start time of the subsequent operation must account for the completion of all operations within that batch.
- **Batch assignment constraints (8):** Ensure that each operation is assigned to exactly one batch.
- **Job-level batching restriction (9):** Prevent multiple operations of the same job from being assigned to the same batch.
- **Final operation constraint (10):** Ensure that the last operation of each job is not batched with other operations and must be delivered separately.
- **Route compatibility constraints (11):** Allow only operations with identical processing routes to be grouped into the same batch.
- **Batch capacity constraints (12):** Enforce capacity limits on batches based on the total assigned workload.
- **Variable domain constraints (13)–(14):** Define the binary and continuous decision variables and their feasible ranges.

All cost components in the objective function are expressed in monetary units to ensure consistency. To avoid numerical instability caused by excessively large constants, the Big-M parameter is defined as an upper bound based on the problem data. Let $p_{ij}$ denote the processing time of the operation $j$ of job $i$, $q_i$ the job quantity, and $M_i$ the number of operations. Jobs are indexed by $i = 1,\cdots,N$. Each job $i$ consists of a sequence of operations indexed by $j = 1,\cdots,M_i$, where $M_i$ denotes the number of operations required for the job $i$. Operation $j$ of job $i$ is denoted by $o_{ij}$. All parameters and decision variables associated with operations are indexed using this job-based notation.

For convenience in describing the proposed BLSDA, two submodels, $\mathbf{M_1}$ and $\mathbf{M_2}$, are presented as follows. $TC_1$ is the objective function of $\mathbf{M_1}$, which is the sum of holding and tardiness costs. $TC_2$ is the objective function of $\mathbf{M_2}$, which is the sum of holding, tardiness, and batching costs based on a given scheduling solution.

- Model $\mathbf{M_1}$:

$$\min TC_1 = \sum_{i=1}^{N} \alpha_i TD_i + \sum_{i=1}^{N} \sum_{j=1}^{M_i - 1} \tau_{ij}(ST_{i(j+1)} - (ST_{ij} + p_{ij}q_i))$$

Subject to constraints (2)-(5), (13), (14)

- Model $\mathbf{M_2}$:

$$\min TC_2 = \sum_{i=1}^{N} \alpha_i TD_i + \sum_{i=1}^{N} \sum_{j=1}^{M_i - 1} \tau_{ij}(ST_{i(j+1)} - (ST_{ij} + p_{ij}q_i)) + \sum_{b=1}^{B} \lambda^b Z^b$$

Subject to constraints (8)-(12), (15)

### 3.2. Complexity

**Lemma 1.** MFSBDP is NP-hard in the strong sense.

### 3.3. 2.3. Dominance condition

Given a scheduling problem, the decision maker may seek a batching solution to minimize transportation costs. This objective can be reached by delivering those WIPs with the exact departure and destination simultaneously. However, production of other WIPs may be delayed. Therefore, tardiness costs and holding costs will increase.

Under certain conditions, one may be able to batch WIPs without increasing completion times. In other words, WIPs with the same routes can be batched and delivered in the same batch with no increase in completion times. Hence, the batching and total cost will decrease. These conditions are distinguished in a scheduling solution by the following lemma, known as the *dominance condition*. For convenience, let [$wip_{ij}$]-[$wip_{i'j'}$] and [$wip_{ij}$-$wip_{i'j'}$] denote WIPs being delivered separately and in the same batch, respectively.

**Dominance Rule**: In a given scheduling solution, for $wip_{ij}$ and $wip_{i'j'}$ that have the same route, if the production of $wip_{ij}$ proceeds $wip_{i'j'}$ in the departure factory, and production of $wip_{i'j'}$ precedes $wip_{ij}$ in the destination factory, then $TC$ ([$wip_{ij}$-$wip_{i'j'}$]) $<$ $TC$ ([$wip_{ij}$]-[$wip_{i'j'}$]). The proof of this dominance rule is provided in Appendix B.

### 3.4. Lower bound

$TC$ is the objective function of **M**, which is the sum of tardiness, holding, and batching costs. For each of the scheduling and batching problems, the optimal solution can be found by solving **M**, separately. For this purpose, let $TC^*$ is the minimum of tardiness, holding, and batching costs. In addition, $TC_1^*$ be the minimum of tardiness and holding costs obtained by solving $\mathbf{M_1}$, and $TC_2^*$ denotes the minimum of the batching cost obtained by solving $\mathbf{M_2}$. It should be noted that, by solving $\mathbf{M_2}$, the optimal values of $ST_{ij}$ and $TD_i$ are both 0. It is clear that $TC^* \geq TC_1^* + TC_2^*$. Therefore, a lower bound (LB) for MFSBDP can be found as $LB \geq TC_1^* + TC_2^*$.

## 4. Proposed solution approach

The MFSBDP can be solved via BLSDA, which consists of two subproblems, decomposed into two levels: the upper and lower levels. At the upper level, the scheduling subproblem is to sequence a set of given jobs across factories with different reentering process routes. In the lower level, the batching subproblem is to assign WIPs to batches transported between factories. The BLSDA's levels are interactively cooperating until the algorithm converges to a solution. Simulated annealing and CPLEX are used to tackle each subproblem.

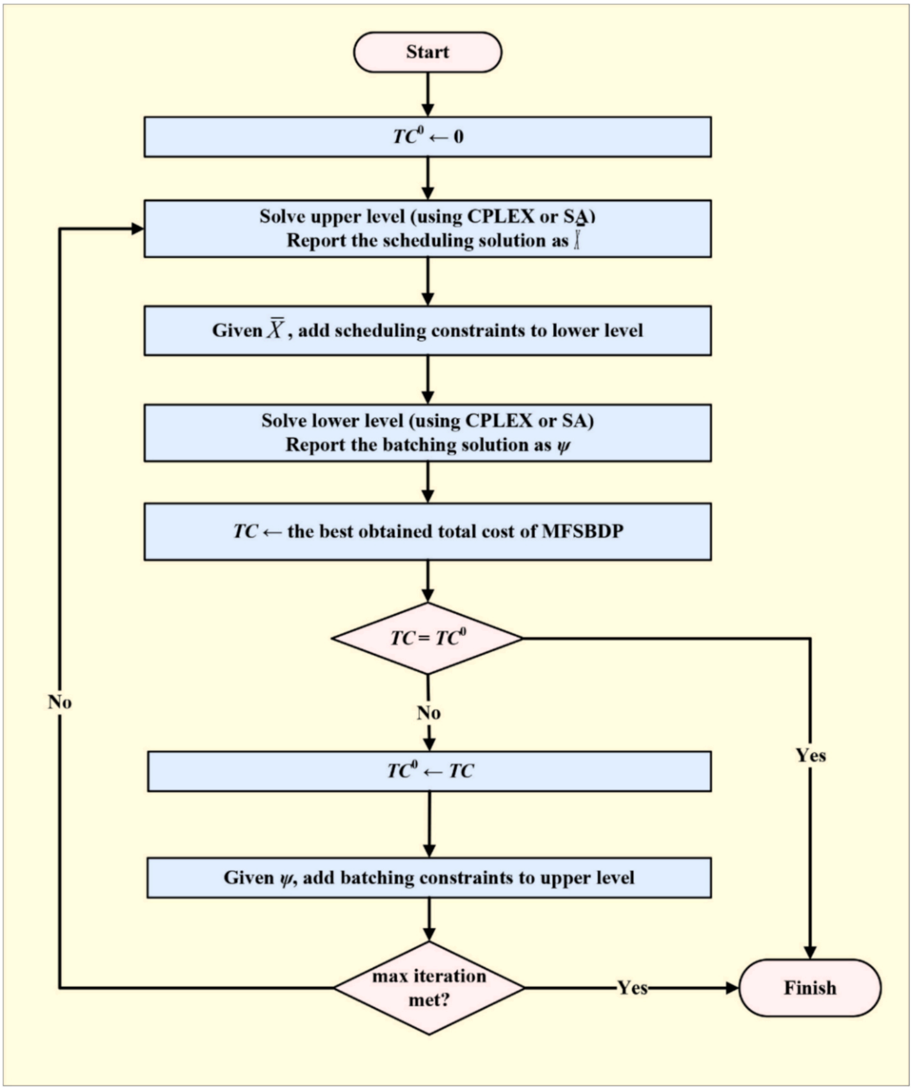


**Fig. 2.** Process flow of the proposed BLSDA.

Although the two levels are solved iteratively and cooperatively, the model preserves a strict decision hierarchy: batching decisions are conditionally optimized given scheduling outcomes, and the upper-level evaluation depends on the optimal lower-level response. Therefore, the problem satisfies the defining characteristics of bilevel programming, namely a nested decision structure and reaction-based optimization [48], even though the computational approach relies on iterative coordination rather than an analytical single-level reformulation.

### 4.1. The bi-level structured decomposition algorithm

The BLSDA starts by solving $\mathbf{M_1}$ using CPLEX or SA in the upper level. Then, the obtained scheduling solution is converted into the input parameters of the lower level to solve $\mathbf{M_2}$, including constraint sets (2), (15), and (16). In Eqs. (15)-(16), $\overline{X}_{iji'j'}$ is the scheduling solution represented as a binary parameter which is equal to one if $o_{ij}$ has been scheduled before $o_{i'j'}$, or zero otherwise. By solving $\mathbf{M_2}$, including the constraints above, using CPLEX or SA, the total cost ($TC$) of the initial solution is obtained, where the set of batched WIPs is represented as $\psi$. At the next iteration, the obtained batching solution is converted into the input parameters for the upper level, and $\mathbf{M_1}$ is now solved with constraints (17)-(18). Given the scheduling solution, which might differ from that of the previous iteration, the constraint sets (2), (15)-(16) are added to $\mathbf{M_2}$, and the procedure is continued as explained above. The algorithm ends when a new $TC$ cannot be found or when the maximum iteration limit is reached.

The combination of SA and CPLEX yields four versions of the BLSDA: CPX-CPX, CPX-SA, SA-CPX, and SA-SA. For example, CPX-SA means that the scheduling problem is solved via CPLEX at the upper level, while SA is used to handle the batching problem at the lower level. The BLSDAs have been coded in the Visual Studio C# 14 environment and implemented on a computer with an Intel Core i7 3.5 GHz CPU and 16 GB RAM. Fig. 2 illustrates the process flow of the proposed BLSDA.

$$M\left(2-r_{ijf}-r_{i'j'f}\right)+M\left(1-\overline{X}_{iji'j'}\right)+ST_{i'j'}-ST_{ij}\geq p_{ij}q_i 1\leq i\leq i'\leq N;$$

$$j=1,2,\cdots,M_i;$$

$$j'=1,2,\cdots,M_{i'};$$

$$f=1,2,\cdots,F \tag{15}$$

$$M\left(2-r_{ijf}-r_{i'j'f}\right)+M\overline{X}_{iji'j'}+ST_{ij}-ST_{i'j'}\geq p_{i'j'}q_{i'} 1\leq i\leq i'\leq N;$$

$$j=1,2,\cdots,M_i;$$

$$j'=1,2,\cdots,M_{i'};$$

$$f=1,2,\cdots,F \tag{16}$$

$$ST_{i(j+1)}\geq ST_{i'j'}+p_{i'j'}q_{i'}+t_{ff'} 1\leq i\leq i'\leq N;$$

$$j=1,2,\cdots,M_i;$$

$$j'=1,2,\cdots,M_{i'};$$

$$f=1,2,\cdots,F;$$

$$wip_{ij} and wip_{i'j'}\in\psi \tag{17}$$

$$ST_{i'(j'+1)}\geq ST_{ij}+p_{ij}q_i+t_{ff'} 1\leq i\leq i'\leq N;$$

$$j=1,2,\cdots,M_i;$$

$$j'=1,2,\cdots,M_{i'};$$

$$f=1,2,\cdots,F;$$

$$wip_{ij}\text{and}wip_{i'j'}\in\psi \tag{18}$$

### 4.2. Simulated annealing

The SA algorithm starts with a randomly generated solution and gradually improves it through neighborhood search. If the objective function of a generated neighbor (*new_obj*) is less than the current solution's (*obj*), then SA accepts it. Otherwise, it may replace the current solution with a worse neighbor according to the Metropolis acceptance criterion in Eq. (19), where rnd denotes a

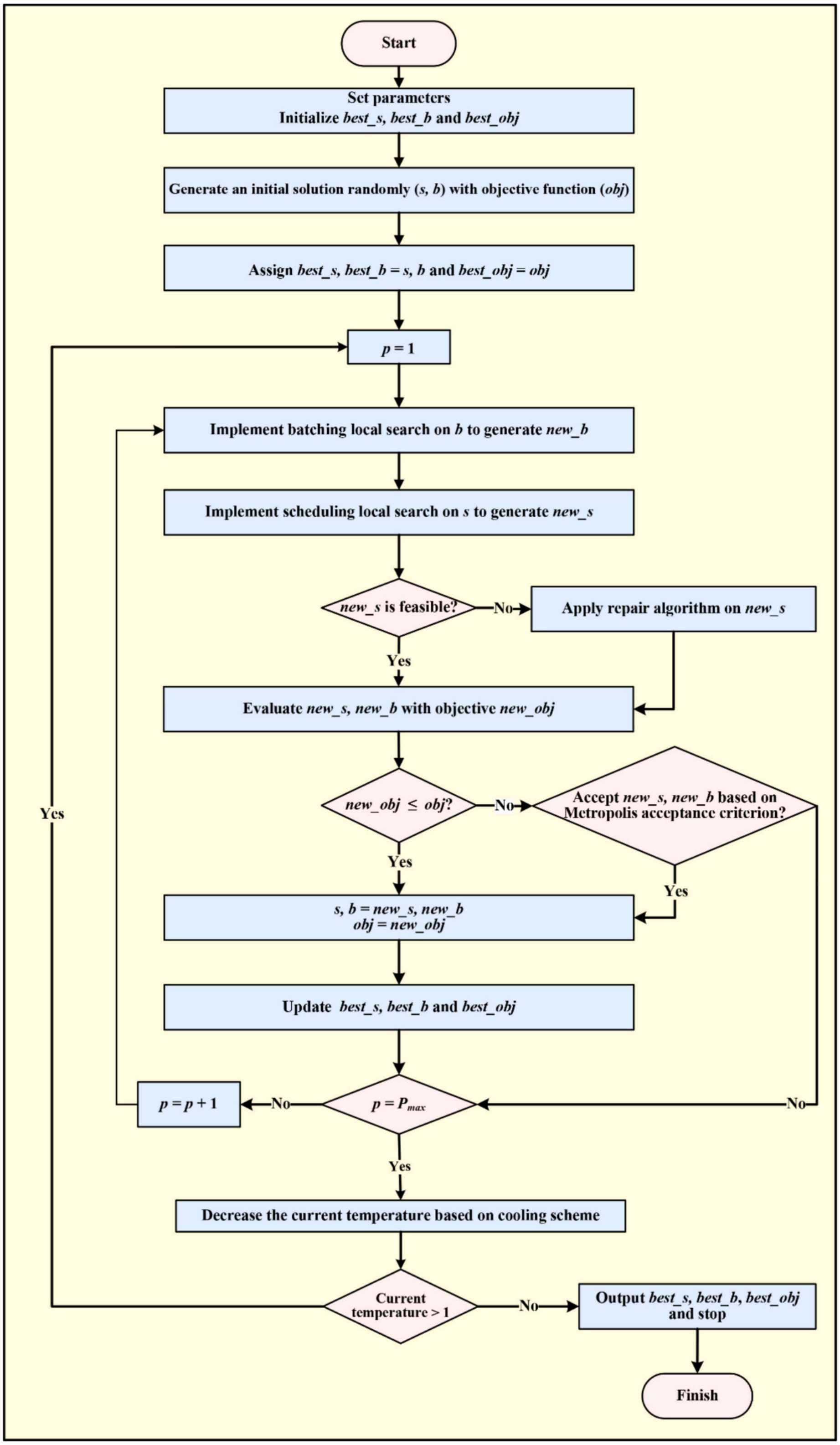


**Fig. 3.** Flow process chart of the SA.

uniformly distributed random number, and temp_co is the temperature coefficient. The procedure is repeated $P_{max}$ times at each temperature level, with the temperature level reduced according to Eq. (20), where *cooling_co* is the cooling schedule coefficient. The general SA framework for handling scheduling or batching subproblems is presented as a flowchart in Fig. 3 and as pseudocode in Fig. 4. Based on Algorithm 1 in Fig. 4, the parameter-tuning strategy is implemented as a controlled balance between exploration and exploitation, tailored to the combined scheduling-batching problem.

Here are the detailed steps in tuning parameters in the SA algorithm, which explain the main parameters as follows.

**1 Temperature** governs the acceptance of a worse solution through Eq. (19)

$$exp\left\{\frac{obj - new\_obj}{temp\_co \times temp}\right\} \geq rnd \tag{19}$$

The initial temperature is set sufficiently high to allow frequent acceptance of worse solutions. This is important because we are exploring two coupled decision specs: scheduling and batching. In practice, the initial temperature is chosen so that the acceptance probability for worse solutions is relatively high. For example, if (obj −new_obj) $= -500$, $temp = 1000$, and $temp\_co = 1$, then the acceptance probability is $e^{-0.5} \approx 0.61$, which supports sufficient exploration in the early stage.

**2 Cooling coefficient (cooling_co)** is applied to update temperature using Equation (20)

$$temp = cooling\_co \times temp \tag{20}$$

A gradual cooling schedule is used to ensure enough iterations at higher temperatures for exploration, followed by a smooth transition toward intensification. Typically, the value of *cooling_co* is chosen close to 1. For example, if $temp = 1000$ and $cooling\_co = 0.95$, then the temperature decreases gradually as 1000→950→902.5, which maintains sufficient exploration before convergence. The temperature scaling parameter *temp_co* is applied in the acceptance function in (19) to control sensitivity to objective differences and to effectively rescale the acceptance probability.

```
Algorithm 1: SA
INPUT: initial_temp; cooling_co; temp_co; P_max
Step 1. temp ← initial_temp
Step 2. (s, b) ← initial_solution_generator ( )
Step 3. obj ← evaluate (s, b)
Step 4. best_s, best_b > ← s, b
        best_obj ← obj
Step 5. WHILE (temp > 1)
            FOR (p = 1 to P_max)
                new_b ← batching local search (b)
                rnd ← a random number ∈ (0, 1)
                IF (rnd < 0.33)
                    new_s← scheduling local search_I (s)
                ELSE IF (rnd < 0.66)
                    new_s← scheduling local search_II (s)
                ELSE
                    new_s← scheduling local search_III (s)
                END
                new_obj ← evaluate (new_s, new_b)
                rnd ← a random number ∈ (0, 1)
                IF (new_obj ≤ obj)
                    s, b ← < new_s, new_b
                    obj ← new_obj
                ELSE IF (exp{ (obj – new_obj) / (temp × temp_co) } ≥ rnd)
                    s, b ← < new_s, new_b
                    obj ← new_obj
                END
                Update best_s, best_b and best_obj
            END
            temp ← temp × cooling_co
        END
OUTPUT: best_s, best_b, best_obj
```

**Fig. 4.** Pseudocode of the SA.

**3 Iterations per temperature (*Pmax*)** is used in the inner loop to determine how thoroughly the solution space is explored at each temperature level. A sufficiently large value of *Pmax* is set to allow multiple neighborhood explorations and to stabilize improvements before colliding. A sufficiently large value of $P_{max}$ is selected to allow multiple neighborhood explorations and to stabilize improvements before cooling. For example, $P_{max} = 50$ allows limited exploration, while $P_{max} = 200$ enables a deeper search but increases computational time

**4 Implicit tuning via neighborhood design** is also implemented, including 1 batching local search and 3 scheduling local searches (randomly selected with equal probability), which ensures balanced exploration across multiple neighborhoods and avoids the dominance of a single search pattern

Since the SA is used to solve scheduling and batching problems, different solution representations are designed for each.

### 4.3. Application of the SA for scheduling problems

The scheduling solution s is created randomly using uniform distribution U (0, 1). Next, through local search scheduling, three distinct algorithms, called local search I (Swap), local search II (Shift), and local search III (Cross-Factory Swap), are employed to generate neighbors, as depicted in Fig. A1 in Appendix A. This illustrates three jobs, each with three operations processed across three factories. The probability of selecting each local search algorithm is equal to 1/3. To make the scheduling solution feasible, a repair algorithm proposed by Wang [49] is utilized. The pseudocode for the scheduling local search methods is provided in Figs. A2, A3, and A4 of Appendix A.

### 4.4. Application of the SA for the batching problem

To initialize a feasible batching solution *b*, it suffices to assign all WIPs randomly to all existing batches. Then, by batching the local search process, some WIPs may be included in the same batch; next, other batches become vacant. Details of the procedure are provided in Fig. A5 in Appendix A, where *C_R* denotes the set of operations that share common processing routes.

## 5. Case study

In this section, the BDLA developed in Section 4 was applied to a Zinc supply chain for multi-factory scheduling and batch delivery. Zinc is one of the most widely consumed non-metallic minerals, and its development is driven by its wide range of applications across different industries. Iran has many zinc production plants that process zinc ore into final ingots. Fig. 5 illustrates a zinc SC network in

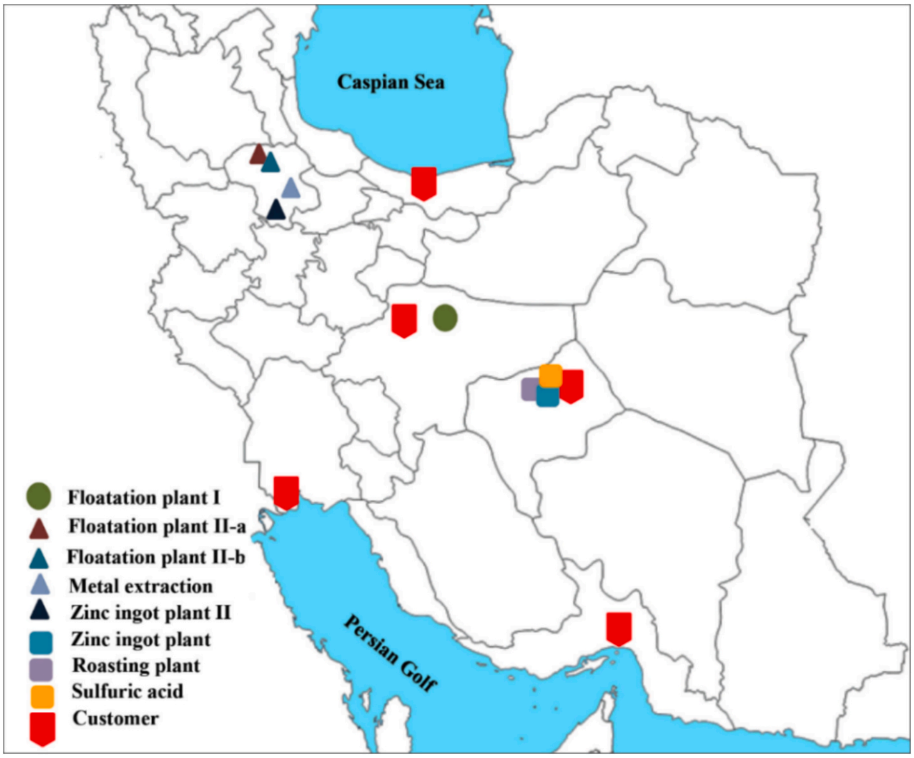


**Fig. 5.** The SC network of the zinc case study on the map of Iran.

Iran, which is the motivating real-world case problem studied in this paper. The main products produced in the zinc SC network are concentrate, zinc ingot, zinc powder, and sulfuric acid. Also, some by-products are yielded from the wastes produced in the zinc SC. The by-products include heavy metals such as silver, cadmium, cobalt, nickel, and lead. Table 2 presents detailed information on zinc SC's products and production plants of the SC network. In this regard, there are two main kinds of concentrate products: sulfate concentrate and oxide concentrate. Sulfate concentrate is delivered to the roasting plant for further processing or can be sold as a final product. Oxide concentrate is divided into high-purity (HP) and low-purity (LP) concentrates. In addition, the zinc ingot group comprises low-grade and high-grade ingots, designated type I and type II, respectively.

Some products may have only one route, e.g., oxide concentrate LP, while other products can be produced via more than one route, e.g., zinc ingot I. Each production route is specialized by a unique color to better represent the SC network, as shown in Fig. 6. It should be noted that, to the best of the authors' knowledge, at this time, the plant of producing heavy metals, i.e., $f_8$, does not exist in Iran. However, to complement our motivating real-world case study, we temporarily assume that such a central plant has been established in the network. This assumption can also motivate decision makers who read this article and compare the results to set up such a plant.

Details of the SC network are depicted in Fig. 6, where color routes are obtained from Table 2. For example, oxide concentrate LP starts its production route (shown in light blue) from flotation plant II-a ($f_2$). Then, the production continues by flotation plant II-b ($f_3$), and finally, the product is delivered to the end customer. In addition, all possible routes of those products with multiple routes are illustrated. The case instances, explained in the next section, are based on real data and multiple routes. Data on heavy metals are based on interviews with experts.

### 5.1. Data collection for model input of real-life zinc case study

The problem data for factory scheduling and batching capacity in the zinc supply chain are available. Moreover, the parameters for transportation costs and the distance between factories are provided by Iran's Road Maintenance & Transportation Organization and the relevant companies. The parameters for zinc mines are derived from the National Geoscience Database of Iran and experts' knowledge.

### 5.2. Results

From the experimental results, the solution algorithms are validated on a diverse set of case instances drawn from the zinc SC network, i.e., case-I: 2 operations across 2 active factories, case-II: 3 operations across 4 active factories, case III: 3 operations across 6 active factories, caseIV: 4 operations across 6 active factories, case −V: 4 operations across 8 active factories. The units for WIPs are tons. Finally, for each case, two sets of due dates are considered: loose due dates (*LD*), which are about 25% wider than tight due dates (*TD*).

To evaluate the performance of the BLSDAs, detailed cost values of the solutions with minimum total cost are reported in Table 3. Also, the exact optimal total cost values obtained by solving **M** via CPLEX, as well as the lower bounds (LBs), are provided. Due to memory limitations, CPLEX could not find a feasible solution for Case-V, as shown by (−).

According to Table 3, the minimum total costs obtained by BLSDAs are close to one another and also close to the optimal total costs found by CPLEX. Regarding tardiness, holding, and batching costs, Table 3 indicates that the BLSDA algorithms converge to the optimal solution, as their cost values are close to those obtained by CPLEX. LB values are used to evaluate the performance of BLSDAs on large problems. In this regard, the BLSDAs' performances are close to the LB values. The CPU times elapsed by the algorithms, reported in Table 3, show that CPX-SA is the fastest, i.e., the maximum time amount of CPX-SA is about 6 h, while for SA-SA, SA-CPX, and CPX-CPX it is about 10, 11, and 15 h, respectively. The aforementioned comparison of *STA*s approves this result. In Table 3, the best values, including the lower bounds, are shown in bold.

**Table 2**
Information about the product of the zinc supply chain network.

| Product type | Product group | Product variant | Product index | Product full name | Production routes | Route color |
|---|---|---|---|---|---|---|
| Main product | Concentrate | Sulfate | $j_1$ | Sulfate concentrate | $f_1$ | Pink |
| | | Oxide LP | $j_2$ | Oxide concentrate LP | $f_2 - f_3$ | Light blue |
| | | Oxide HP | $j_3$ | Oxide concentrate HP | $f_2 - f_3 - f_2$ | Dark green |
| | Zinc ingot | Type I | $j_4$ | Zinc ingot I | $f_1 - f_4 - f_6$ | Yellow |
| | | | | | $f_1 - f_4 - f_7$ | |
| | | | | | $f_2 - f_3 - f_2 - f_6$ | |
| | | | | | $f_2 - f_3 - f_2 - f_7$ | |
| | | Type II | $j_5$ | Zinc ingot II | $f_1 - f_4 - f_6$ | Dark blue |
| | | | | | $f_1 - f_4 - f_7$ | |
| | | | | | $f_2 - f_3 - f_2 - f_6$ | |
| | | | | | $f_2 - f_3 - f_2 - f_7$ | |
| By-product | Zinc powder | – | $j_6$ | Zinc powder | $f_2 - f_3 - f_6$ | Light green |
| | | | | | $f_2 - f_3 - f_7$ | |
| | Sulfuric acid | – | $j_7$ | Sulfuric acid | $f_1 - f_4 - f_5$ | Red |
| | Heavy metals | – | $j_8$ | Heavy metals | $f_2 - f_3 - f_7 - f_8$ | Purple |

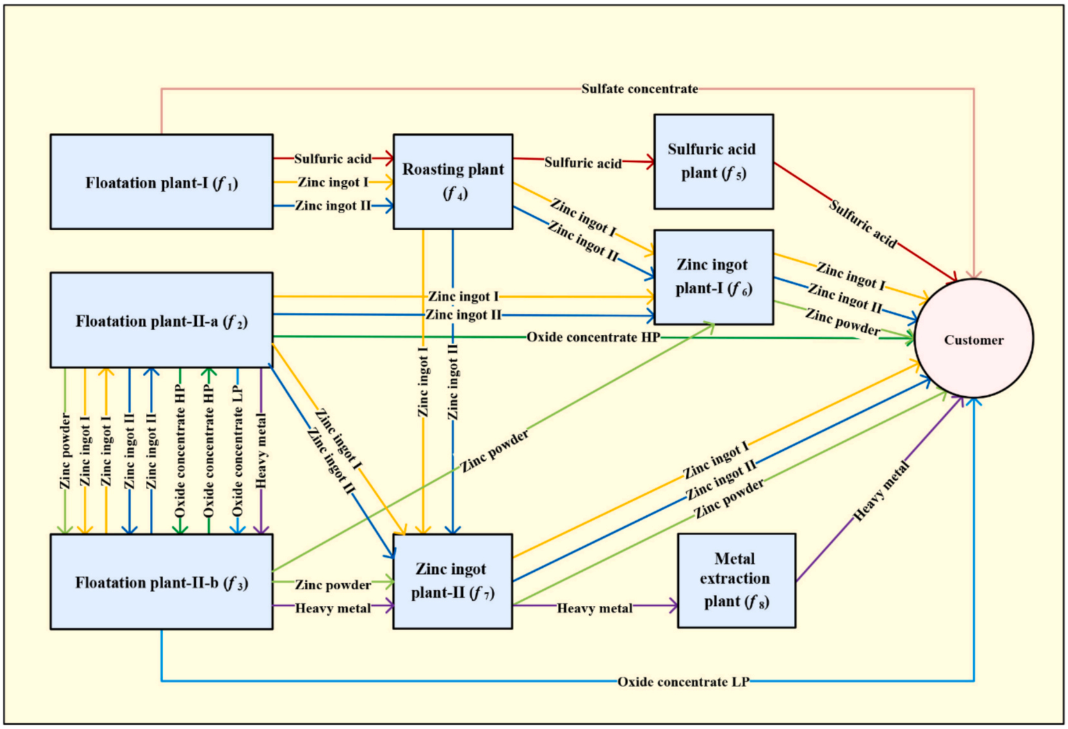


**Fig. 6.** Processing routes of different jobs in the zinc SC.

**Table 3**
Computational results of costs and elapsed time based on loose and tight due dates (case study).

| Measure | Algorithm | Tight due date (TD) | | | | | Loose due date (LD) | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Case | | | | | Case | | | | |
| | | I | II | III | IV | V | I | II | III | IV | V |
| **Tardiness and Holding Cost ($)** | SA-SA | 1,750 | **10,998** | 56,319 | 96,976 | 109,527 | 275 | **1,718** | 11,042 | 22,061 | 78,827 |
| | SA-CPX | 1,603 | 11,056 | 51,971 | 95,714 | 104,590 | 308 | 1,729 | 10,750 | 21,324 | 73,712 |
| | CPX-SA | 1,338 | **10,998** | 50,054 | 93,867 | **100,552** | 258 | 1,780 | **9,388** | 20,099 | **72,107** |
| | CPX-CPX | 1,338 | 11,003 | 50,046 | 93,867 | 102,996 | 264 | 1,730 | **9,388** | 20,099 | **72,107** |
| | CPLEX | 1,338 | 11,003 | **50,031** | 93,850 | – | 303 | **1,718** | 9,545 | 20,265 | – |
| | LB | **1,321** | **10,998** | **50,031** | **93,831** | **100,552** | **251** | **1,718** | **9,388** | **20,051** | **72,107** |
| **Batching Cost ($)** | SA-SA | 6,990 | 8,970 | 15,170 | 15,020 | 21,320 | 6,990 | 8,840 | 14,890 | 14,890 | 21,320 |
| | SA-CPX | 6,840 | 8,670 | 14,910 | 14,740 | 21,320 | 6,840 | 8,540 | 14,760 | 14,610 | 21,320 |
| | CPX-SA | 6,970 | 8,970 | 15,170 | 14,890 | 21,170 | 6,970 | 8,540 | 15,320 | 15,040 | 21,320 |
| | CPX-CPX | 6,970 | 8,670 | 15,170 | 14,890 | 21,320 | 6,970 | 8,540 | 15,320 | 15,040 | 21,320 |
| | CPLEX | 6,970 | 8,670 | 15,170 | 14,890 | – | 6,840 | 8,540 | 15,020 | 14,760 | – |
| | LB | **6,690** | **8,390** | **14,460** | **14,610** | **20,460** | **6,690** | **8,390** | **14,460** | **14,610** | **20,460** |
| **Total Cost ($)** | SA-SA | 8,740 | 19,968 | 69,489 | 111,996 | 130,847 | 7,265 | 10,558 | 25,932 | 36,951 | 100,147 |
| | SA-CPX | 8,443 | 19,726 | 66,881 | 110,454 | 125,910 | 7,148 | 10,269 | 25,038 | 35,706 | 95,032 |
| | CPX-SA | 8,308 | 19,968 | 65,224 | 108,757 | 121,722 | 7,228 | 10,320 | 24,708 | 35,139 | 93,427 |
| | CPX-CPX | 8,308 | 19,673 | 65,216 | 108,757 | 124,316 | 7,234 | 10,270 | 24,708 | 35,139 | 93,427 |
| | CPLEX | 8,308 | 19,673 | 65,201 | 108,740 | – | 7,143 | 10,258 | 24,565 | 35,025 | – |
| | LB | **8,011** | **19,388** | **64,491** | **108,441** | **121,012** | **6,941** | **10,108** | **23,848** | **34,661** | **92,567** |
| **Running Time (s)** | SA-SA | 5,464 | 7,850 | 23,258 | 35,774 | 4,037 | 1,751 | 6,230 | 18,687 | 35,774 | 4,037 |
| | SA-CPX | 3,825 | 7,415 | 21,823 | 22,426 | 9,327 | 5,340 | 10,560 | 27,782 | 40,537 | 15,960 |
| | CPX-SA | 53 | 48 | 692 | 11,841 | 20,854 | 55 | 128 | 313 | 20,669 | 10,964 |
| | CPX-CPX | 81 | 390 | 2,215 | 26,113 | 53,266 | 48 | 77 | 445 | 20,346 | 11,835 |
| | CPLEX | 31 | 2,729 | 3,606 | 3,805 | – | 21 | 43 | 3,608 | 3,605 | – |
| | LB | 7 | 10 | 30 | 27 | 9 | 7 | 10 | 30 | 27 | 9 |

– Out of memory in CPLEX

$$GAP_1 = \frac{TotalCost_{BLDA} - TC^*}{TC^*} \tag{21}$$

$$GAP_2 = \frac{TotalCost_{BLDA} - LB}{LB} \tag{22}$$

Additional insight is provided in Table 4 through the relative gap measures, $GAP_1$ and $GAP_2$. In Eq. (21), $GAP_1$ is computed using exact solutions for cases I-IV obtained with CPLEX. In contrast, Eq. (22) evaluates Case V using a lower bound instead of the optimal cost, since the optimal solution is not available. In these equations, $TotalCost_{BLSDA}$, $TC^*$, and LB represent the total cost of the BLSDA variants, the optimal total cost, and the lower bound, respectively. As shown in Table 4, all GAP1 and GAP2 values are below 10%, indicating that the proposed approach produces high-quality solutions even for large-scale instances. In particular, the low gap values achieved by CPX-SA and CPX-CPX support the effectiveness of the BLSDAs. In Table 4, the minimum GAP values in each row are shown in bold.

Although five problems are extracted from a real zinc supply chain in Table 3, a total of 32 additional instances are randomly generated, as reported in Table 5, to enable more robust statistical comparisons among the solution approaches. The number of instances, 32, is chosen to balance computational effort and statistical reliability, consistent with prior studies in scheduling and metaheuristic optimization [50]. Table 4 presents the optimality gaps for both TD and LD under different solution strategies. The results show that the CPX-SA and CPX-CPX approaches consistently achieve the lowest gap values across most cases, highlighting the effectiveness of incorporating exact optimization within the bi-level framework. In Table 5, the minimum total cost for each instance is indicated in bold.

The statistical analyses, conducted using MINITAB 17 and reported in Table 6, are based on algorithm outputs across the 32 generated instances. The mean total costs shown in Table 6, therefore, represent averages across these instances, rather than averages obtained from multiple repeated runs on the same instance. Accordingly, the statistical results should be interpreted as cross-instance comparisons among the generated test problems. Table 6 presents the Tukey pairwise comparison results for total cost, where the SA-SA approach forms a distinct group with a higher average cost, while all other methods fall into a lower-cost grouping. The corresponding mean total costs further confirm that CPX-SA achieves the lowest average cost among the evaluated approaches.

Except for SA-SA, the results of all other versions of the BLSDA, as well as CPLEX's results, and the lower bounds are placed in one group, which means the applied SA is robust in finding optimal or near-optimal solutions in each level of the BLSDA. Also, it shows that SA-CPX, CPX-SA, and CPX-CPX find optimal solutions in different instances. Moreover, as shown in Table 6, the mean total costs for CPX-SA and CPX-CPX are 7,551,746 and 7,603,818, respectively. This shows that the mean total cost of CPX-SA is lower than that of CPX-CPX, and that applying SA at a lower level yields solutions of very high quality, compared with using CPLEX. That is why $SRM_{CPX\text{-}CPX}$ and $STA_{CPX\text{-}CPX}$ are the largest.

The strong performance of CPX-SA can be attributed to its hybrid search mechanism. The crossover component promotes information exchange between high-quality solutions, enabling promising routing and batch-delivery configurations to be combined. Meanwhile, the simulated annealing component provides controlled stochastic exploration by occasionally accepting non-improving solutions, which helps the algorithm escape local optima. This balance between intensification and diversification is particularly important for large-scale multi-factory scheduling problems, where the solution space grows combinatorially with the number of jobs, operations, and factories. In addition, within the BLSDA framework, the algorithm operates on a reduced decision space associated with network configuration decisions, while detailed scheduling is handled separately. This decomposition further improves search efficiency and contributes to the superior computational performance observed in the experiments.

To provide further explanation of the superior performance of CPX-SA, additional analytical insights are provided. From a computational complexity point of view, solving both levels using CPLEX (CPX-CPX) incurs a significantly higher computational burden due to the problem's NP-hardness, as the branch-and-bound procedure may grow exponentially with problem size. In contrast, CPX-SA cuts this burden by applying simulated annealing (SA) at the lower level with polynomial per-iteration complexity. This causes the algorithm to search a larger number of candidate solutions while maintaining the same computational time.

Besides, SA has a well-established convergence specification through proper cooling schedules that converges in probability to an overall optimum. While such guarantees are asymptotic, SA offers deeper diversification by probabilistically accepting non-improving solutions, enabling the search to escape local optima. When combined with CPLEX, which provides strong intensification at the upper level, CPX-SA gains an effective balance between exploration and exploitation.

Further, the lower-level scheduling problem illustrates a greatly rugged solution landscape with many near-optimal solutions. Similarly, exact solvers may spend an excessive amount of time demonstrating optimality without making substantial progress in

**Table 4**
Gaps in computational results of *TD* and *LD*.

| | | Solution approaches for TD | | | | Solution approaches for LD | | | |
|---|---|---|---|---|---|---|---|---|---|
| GAP | Problem | SA-SA | SA-CPX | CPX-SA | CPX-CPX | SA-SA | SA-CPX | CPX-SA | CPX-CPX |
| $GAP_1$ | Case-I | 1.7% | **0.1%** | 1.2% | 1.3% | 5% | 2% | **0%** | **0%** |
| | Case-II | 2.9% | **0.1%** | 0.6% | **0.1%** | 1% | **0%** | 1% | **0%** |
| | Case-III | 5.6% | 1.9% | **0.6%** | **0.6%** | 7% | 3% | **0%** | **0%** |
| | Case-IV | 5.5% | 1.9% | **0.3%** | **0.3%** | 3% | 2% | **0%** | **0%** |
| $GAP_2$ | Case-V | 8.2% | 2.7% | **0.9%** | **0.9%** | 8% | 4% | **1%** | 3% |

**Table 5**
Computational results of total costs and time based on loose and tight due dates (test instances).

| Loose due date (LD) | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Instance | CPX-CPX | Time(s) | CPX-SA | Time(s) | SA-SA | Time(s) | SA-CPX | Time(s) | CPLEX | Time(s) |
| 1 | **6,023** | 10 | **6,023** | 1 | **6,023** | 59 | **6,023** | 0 | **6,023** | 8 |
| 2 | **16,863** | 8 | **16,863** | 2 | **16,863** | 56 | **16,863** | 1 | **16,863** | 12 |
| 3 | 94,916 | 25 | 96,125 | 6 | 102,040 | 150 | 94,008 | 2 | **94,000** | 15 |
| 4 | 129,948 | 29 | 127,532 | 3 | 128,250 | 188 | 118,484 | 2 | **115,450** | 34 |
| 5 | **612,004** | 202 | 626,077 | 32 | 629,697 | 1,255 | 663,571 | 14 | | |
| 6 | **562,817** | 197 | 636,857 | 30 | 576,740 | 1,558 | 587,145 | 31 | | |
| 7 | **2,623,153** | 3,286 | 2,764,195 | 647 | 3,795,315 | 15,927 | 2,982,814 | 378 | | |
| 8 | **2,884,416** | 2,564 | 3,075,385 | 425 | 4,809,336 | 14,849 | 3,375,011 | 336 | | |
| 9 | **3,097,307** | 2,596 | 3,528,420 | 362 | 4,548,802 | 13,674 | 3,718,767 | 239 | | |
| 10 | **2,573,096** | 3,232 | 2,775,237 | 430 | 3,445,111 | 13,280 | 3,137,443 | 280 | | |
| 11 | **5,079,125** | 2,551 | 5,397,253 | 180 | 8,316,533 | 12,538 | 6,309,352 | 236 | | |
| 12 | **3,675,953** | 3,601 | 4,069,380 | 898 | 6,768,200 | 17,861 | 4,034,813 | 700 | | |
| 13 | **6,166,399** | 3,604 | 6,205,671 | 2,878 | 11,391,059 | 22,660 | 6,940,479 | 1,297 | | |
| 14 | **9,003,460** | 3,602 | 9,060,684 | 1,693 | 15,055,941 | 19,884 | 9,607,693 | 786 | | |
| 15 | 36,124,577 | 3,625 | 36,324,185 | 3,613 | 48,164,257 | 26,439 | **34,685,664** | 3,617 | | |
| 16 | 40,738,050 | 3,614 | 38,131,750 | 2,525 | 51,890,765 | 28,991 | **37,443,986** | 3,625 | | |
| **Tight due date (TD)** | | | | | | | | | | |
| **Instance** | **CPX-CPX** | **Time(s)** | **CPX-SA** | **Time(s)** | **SA-SA** | **Time(s)** | **SA-CPX** | **Time(s)** | **CPLEX** | **Time(s)** |
| 1 | **11,131** | 11 | **11,131** | 2 | **11,131** | 61 | **11,131** | 0 | **11,131** | 16 |
| 2 | **18,050** | 9 | **18,050** | 2 | **18,050** | 64 | **18,050** | 1 | **18,050** | 17 |
| 3 | 142,238 | 28 | 143,311 | 28 | 148,718 | 222 | 139,775 | 3 | **138,050** | 11 |
| 4 | 142,465 | 55 | 138,990 | 4 | 143,680 | 226 | 139,185 | 4 | **130,870** | 12 |
| 5 | **612,577** | 212 | 647,171 | 40 | 664,351 | 1,335 | 749,704 | 16 | | |
| 6 | **638,091** | 214 | 639,767 | 286 | 662,113 | 1,783 | 724,627 | 34 | | |
| 7 | **3,667,005** | 3,521 | 3,830,623 | 787 | 6,364,667 | 16,081 | 4,455,598 | 452 | | |
| 8 | **3,091,864** | 3,488 | 3,400,059 | 579 | 5,191,279 | 14,946 | 3,889,471 | 415 | | |
| 9 | **3,426,437** | 2,826 | 3,745,922 | 633 | 4,580,541 | 14,077 | 3,910,404 | 454 | | |
| 10 | **2,848,995** | 3,601 | 3,049,967 | 1,066 | 4,038,017 | 16,830 | 3,414,988 | 506 | | |
| 11 | **5,440,334** | 3,601 | 5,521,703 | 1,856 | 8,712,138 | 17,818 | 6,666,147 | 338 | | |
| 12 | **4,972,723** | 3,601 | 5,112,570 | 988 | 7,478,488 | 18,590 | 5,360,350 | 755 | | |
| 13 | **7,898,561** | 3,604 | 8,253,928 | 3,298 | 12,909,019 | 22,790 | 9,228,509 | 1,477 | | |
| 14 | **9,527,496** | 3,603 | 10,175,773 | 2,308 | 15,904,666 | 22,943 | 11,024,405 | 1,599 | | |
| 15 | 41,431,140 | 3,639 | 41,060,211 | 3,615 | 52,801,129 | 28,997 | **38,201,291** | 3,635 | | |
| 16 | 46,064,946 | 3,630 | 43,065,065 | 3,611 | 58,346,631 | 31,208 | **42,606,528** | 3,626 | | |

**Table 6**
Tukey pairwise comparison with respect to total cost.

| | Algorithm | | Grouping | |
|---|---|---|---|---|
| **Total cost** | SA-SA | | A | |
| | SA-CPX | | B | |
| | CPX-CPX | | B | |
| | CPX-SA | | B | |
| | CPLEX | | B | |
| | LB | | B | |
| **Algorithm** | SA-SA | SA-CPX | CPX-CPX | CPX-SA |
| **Mean total cost** | 10,550,611 | 7,633,196 | 7,603,818 | 7,551,746 |

solution quality within practical time limits. In contrast, SA efficiently identifies high-quality feasible solutions and explores diverse regions of the solution space. This explains why CPX-SA exhibits better average performance than CPX-CPX and SA-CPX on large-scale instances.

Although SA-SA has been categorized separately, one can achieve lower total costs by applying a tuning scheme, such as erruchi, to SA's parameters at both levels. Also, the application of an integrated SA, rather than the bi-level SA (SA-SA), would be more beneficial and is suggested for future research.

Computational results for implementing all versions of the BLSDA for the *TD* and *LD* case instances are illustrated in Figs. 7 and 8, respectively. According to these figures, the BLSDA finds a solution with the minimum total cost before reaching the maximum-iteration limit. However, in Fig. 7(a), 7(f), 7(i), 7(j), 7(m), and 7(n), where BLSDA uses SA at the upper level, the algorithm does not converge and must be terminated at the maximum iteration limit. This is the same for Fig. 8(f), 8(i), 8(j), 8(m), and 8(n). The main reason is that the SA's schedules are mostly local optima, which leads to significant diversity in BLSDA's outputs. In this regard, the average coefficient of variation (CV) of SA-SA and SA-CPX is 85% and 31%, respectively, while it is 1% for each of CPX-SA and CPX-CPX. Fig. 7(p) and 7(t) also indicate that CPX-SA and CPX-CPX may have high diversity in their outputs. However, in other cases, the BLSDA generally reaches the minimum within half the maximum number of iterations when solving the upper level via CPLEX; see


**Fig. 7.** Total costs in terms of iteration on tight due dates. The vertical axes show the objective value, and the horizontal axes show the number of algorithm iterations, ranging from 1 to 10 (the maximum number of iterations), where 0 denotes the initial iteration. The red markers show the minimum total costs.

Fig. 7(k), 7(l), 8(g), and 8(h).

In Fig. 7(g), 7(t), 8(k), 8(l), 8(s), and 8(t), the BLSDA using CPLEX in its upper level cannot improve its initial solution. This may be due to two reasons: first, the algorithm may get stuck in a local optimum; for example, in Fig. 7 (g), CPX-SA's output is even worse than SA-CPX's. Second, the algorithm may reach the optimum solution in its initial starting iteration. This is because the total tardiness and holding costs are much higher than the batching costs. Therefore, no reduction will be made to the total batching cost. For example, in Fig. 8(s) and 8(t), it seems that the optimum solution is found in the initial iteration, since the total cost is remarkably close to the lower bound (a gap of less than 1%).

In this analysis, the paper defines two performance measures for the BLSDAs: the speed to reach the minimum total cost (SRM) and the speed to terminate the algorithm (*STA*). Fig. 9 depicts cumulative frequencies of the algorithms that reached their minimums versus the iteration index. For example, at half of the maximum iteration, see the area surrounded by a dashed line on the 5th iteration. CPX-CPX has found the minimum across all TD and LD cases, with a cumulative frequency of 100%. It is, however, 90% for CPX-SA, 80% for SA-SA, and 60% for SA-CPX, which, equivalently, means the SRM of CPX-CPX is higher than that of the other algorithms. The algorithms are compared in terms of *SRM* as follows:

$SRM_{\text{ CPX-CPX}} > SRM_{\text{ CPX-SA}} > SRM_{\text{ SA-SA}} = SRM_{\text{ SA-CPX}}$.

Fig. 10 illustrates the cumulative STA frequencies of the algorithm. At the 5th iteration, CPX-CPX and CPX-SA terminated in 80% of the cases, and this percentage is much lower for SA-SA (30%) and SA-CPX (20%). In this regard, CPX-SA is the fastest algorithm:

$STA_{\text{ CPX-CPX}} = STA_{\text{ SA-SA}} = STA_{\text{ SA-CPX}} > STA_{\text{ CPX-SA}}$.

The *SRM* and *STA* measures indicate that a trade-off has to be made between the size of the search space and the search time. In this regard, CPX-SA is the algorithm whose search space is not too limited like CPX-CPX nor too wide like SA-CPX and SA-SA. Therefore, it reaches high-quality solutions in a shorter time than other algorithms.

A sensitivity analysis is performed to assess the effect of due-date parameters on the network's costs. So, the optimal cost values of CPLEX are studied by comparing *TDs* vs. *LDs*. Since CPLEX could not solve case-V, the results of CPX-CPX are reported for case-V. Details of total costs are depicted in Fig. 11, showing that *LDs*' total costs are 43% lower than *TDs*', on average. Due to the

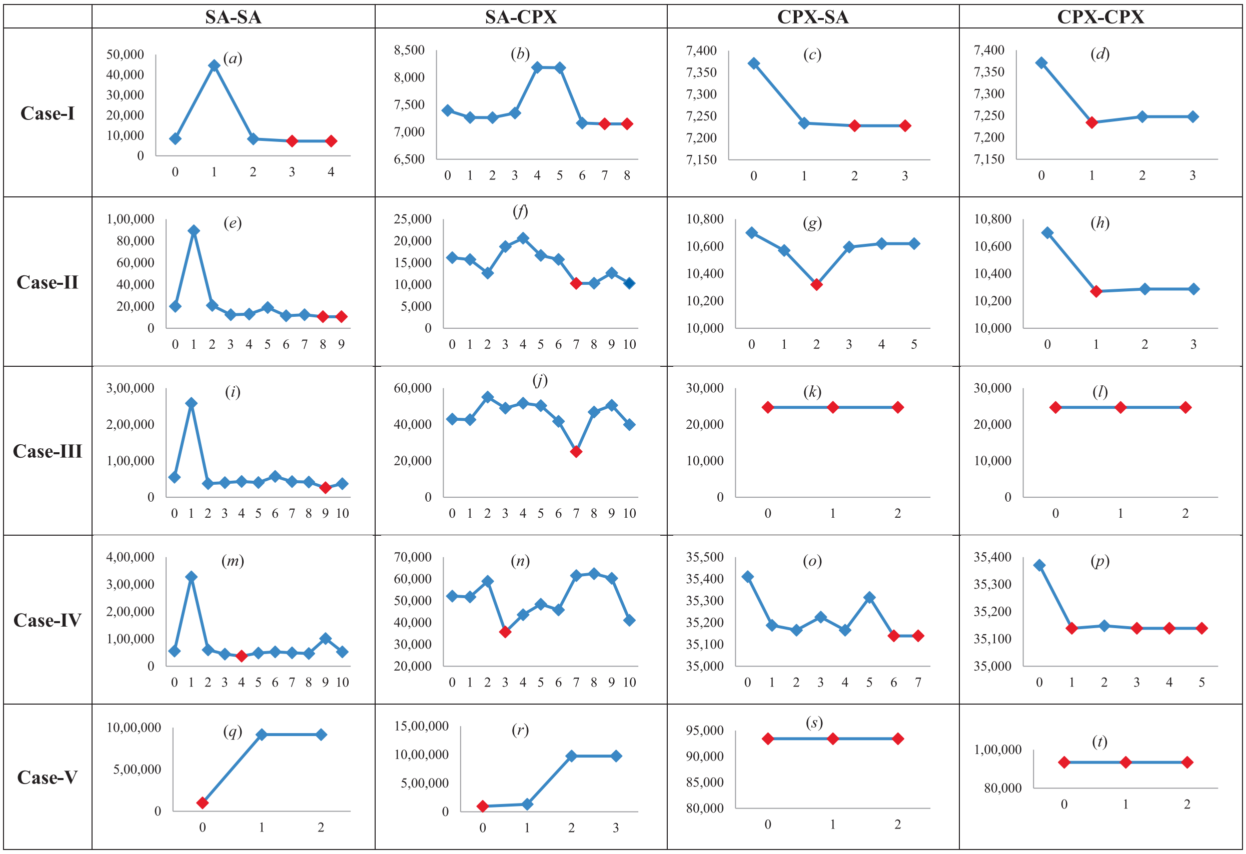


**Fig. 8.** Total costs in terms of iterations on loose due dates. The vertical axes show the objective value, and the horizontal axes show the number of algorithm iterations, ranging from 1 to 10 (the maximum number of iterations), where 0 denotes the initial iteration. The red markers show the minimum total costs.

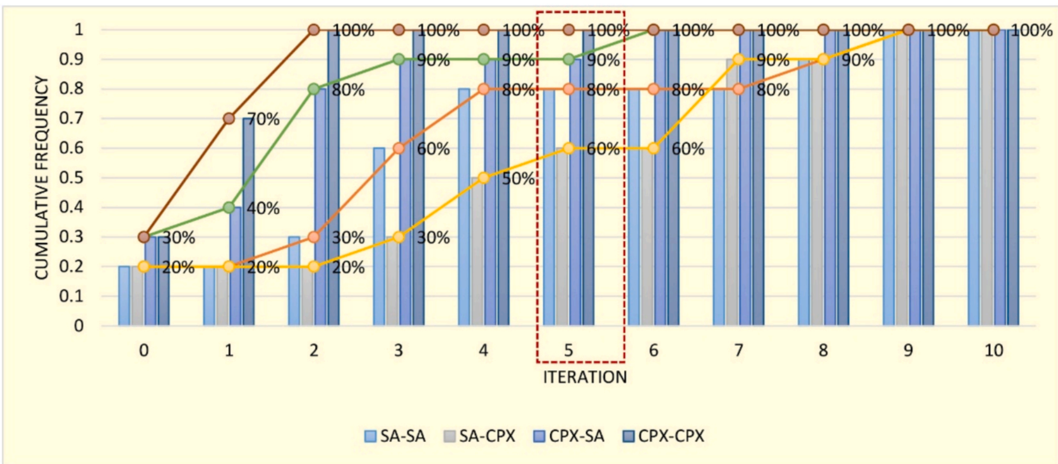


**Fig. 9.** SRM cumulative frequencies.

increasing number of due dates, job completion can be postponed for batching purposes without increasing tardiness or holding costs. Also, tardiness costs will decrease as the due date parameters are extended. Fig. 12 illustrates the scheduling costs, i.e., tardiness and holding costs, with TDs costing 30% more than *LDs*. Unlike scheduling costs, batching costs are not significantly affected by changes to due date parameters, as shown in Fig. 13. The average batching costs of *LDs* are 1% lower than those of *TDs*. This means the batching solution is less sensitive to due-date parameters than the scheduling solution.

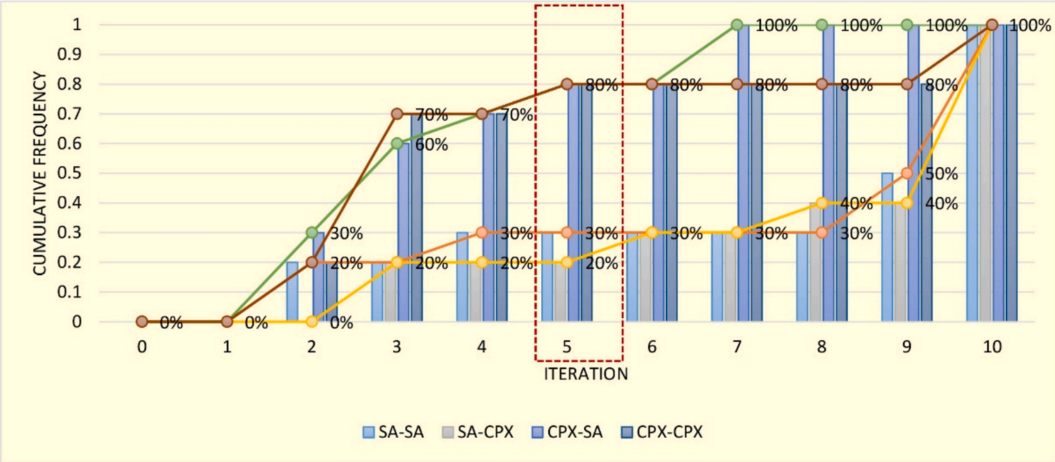


**Fig. 10.** STA cumulative frequencies.

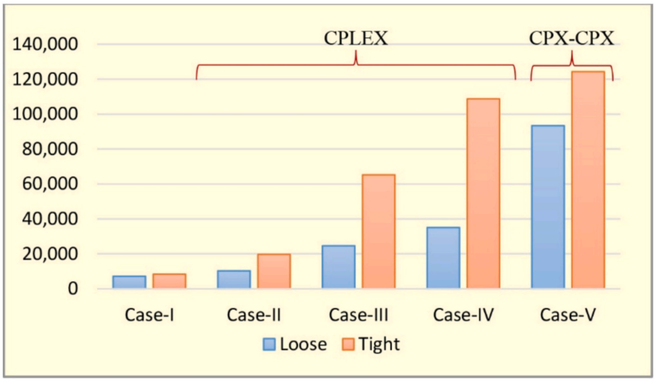


**Fig. 11.** Total costs analysis (*TD* vs *LD*).

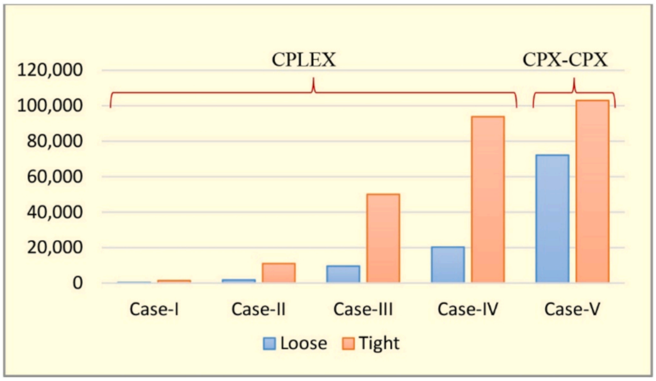


**Fig. 12.** Scheduling costs (tardiness costs and holding costs) analysis (*TD* vs *LD*).

The proposed bi-level decomposition framework effectively coordinates network-level batching and factory-level scheduling through an iterative feedback mechanism. In each cycle, batching decisions provide feasible lower bounds for operation start times, while CPLEX determines detailed schedules at the factory level; updated completion times and tardiness values are fed back to refine batching and routing decisions, preventing cycling and ensuring convergence. Numerical results from the zinc supply chain case illustrate the practical implications of this interaction: looser due dates reduce total costs by 43% on average by allowing jobs to be delayed strategically for batching without increasing tardiness or holding costs. While scheduling costs decrease significantly under

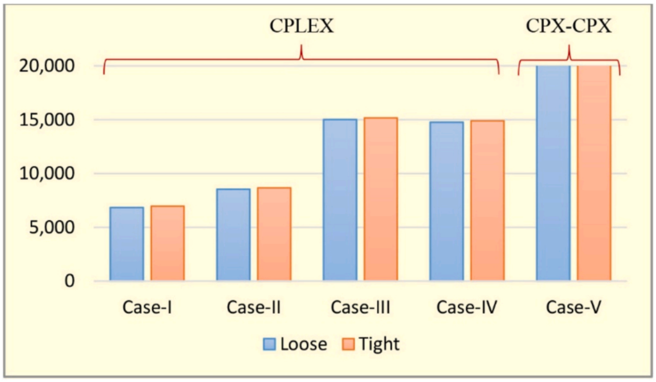


**Fig. 13.** Batching costs analysis (*TD* vs *LD*).

looser due dates (approximately 30%), batching costs remain largely unaffected, indicating that the framework enables managers to leverage batching strategically while prioritizing scheduling efficiency. This demonstrates how the decomposition approach not only improves computational tractability but also provides actionable, data-driven insights for balancing delivery flexibility, production synchronization, and cost efficiency in multi-factory, reentrant production systems.

## 6. Research implications

This study contributes to the information sciences literature by demonstrating how a BLSDA can effectively address the complexity of reentrant flows and multi-factory routing in large-scale manufacturing systems. The framework highlights the necessity of decomposition: solving the full mixed-integer programming (MIP) model for multi-factory scheduling with batch delivery is computationally intractable due to the combinatorial explosion of routing and sequencing decisions. By separating network-level routing and batch-delivery decisions from factory-level operation scheduling, the proposed approach provides a tractable, interpretable, and data-driven decision-support system that preserves coordination across hierarchical decision levels. This decomposition enables the integration of intelligent search mechanisms (simulated annealing) at the strategic level with exact optimization (CPLEX) at the operational level, thereby improving solution quality and computational efficiency. The study thus advances the methodological understanding of intelligent, hierarchical decision-support systems for data-intensive, reentrant production networks, offering practical guidance for managing multi-factory operations where routing flexibility and batch synchronization are critical. The computational results confirm the effectiveness of hybrid deterministic–stochastic algorithms in solving real-world, high-dimensional optimization problems, reinforcing their relevance for future research in intelligent systems, algorithm design, and applied optimization.

## 7. Managerial implications

The managerial implications of this study are particularly relevant for industries operating complex, distributed manufacturing networks, such as the zinc industry, where production involves reentrant flows and multiple facilities, including flotation and roasting plants. In such systems, bottlenecks at critical stages and the timing of batch deliveries can significantly affect operational performance. The proposed BLSDA serves as an intelligent decision-support tool, enabling managers to decompose complex scheduling and routing problems into tractable components and to coordinate production schedules with batch deliveries across facilities. Grounded in the numerical results, the sensitivity analyses reveal that loosening due-date policies can reduce total operational costs by up to 43%, highlighting a major actionable insight: managers can achieve substantial cost savings by strategically relaxing delivery constraints in coordination with batch scheduling. The analyses further demonstrate how adjustments to batch sizes, routing, and resource allocation affect WIP accumulation and tardiness, guiding the balancing of service levels against logistics and production costs. By enhancing synchronization between production and batch delivery, enabling iterative feedback across decision levels, and incorporating real operational data, the framework equips managers with a data-driven, adaptive tool for proactive decision-making, improving cost efficiency, delivery reliability, and overall system performance in multi-factory, reentrant production environments.

## 8. Conclusions and future research directions

This study demonstrates that the proposed BLSDA provides an effective and scalable framework for coordinating multi-factory scheduling and batch delivery in complex, distributed production networks. Numerical experiments based on a real-world zinc supply chain show that the approach can reduce total operational costs by up to 43% under looser due-date policies, while simultaneously improving synchronization between production and delivery, reducing WIP accumulation, and enhancing delivery reliability. The iterative feedback mechanism linking network-level routing and batching decisions with factory-level scheduling enables more informed, adaptive, and data-driven decision-making in large-scale supply chain environments.

Despite these contributions, the study has several limitations. All model parameters, including processing times, demand, and transportation conditions, are assumed to be deterministic, which does not fully capture the inherent variability and uncertainty present in real-world operations. In addition, the current framework primarily focuses on operational and logistical coordination and does not explicitly account for factors such as energy consumption, maintenance planning, or stochastic disruptions in production and transportation systems.

In practice, uncertainty in demand, processing times, and transportation conditions can significantly influence system performance and decision quality. Variations in processing times may lead to schedule deviations, increased idle times, and higher tardiness costs. Demand uncertainty can disrupt batch formation, resulting in underutilized or delayed shipments and reduced transportation efficiency. Similarly, unexpected disruptions in production or logistics may affect routing feasibility and coordination across factories, potentially amplifying delays and increasing holding costs. As a result, solutions derived under deterministic assumptions may underestimate operational risks and overestimate system performance, highlighting the importance of explicitly incorporating uncertainty into the modeling framework.

Future research can extend the proposed framework in several meaningful directions. Incorporating uncertainty in key parameters, such as processing times, demand, and transportation conditions, would enhance the model's robustness and realism. These extensions can be pursued through stochastic programming, robust optimization, or simulation-based approaches to better support decision-making under uncertainty.

Furthermore, integrating sustainability considerations, such as energy-aware scheduling and emission-sensitive transportation decisions, would broaden the model's applicability to environmentally conscious supply chain management. The inclusion of real-time data streams, learning-based mechanisms, and predictive maintenance information would further strengthen the framework's decision-support capabilities, enabling adaptive, intelligent, and resilient coordination across multi-factory systems operating in dynamic and uncertain environments.

Ethics approval and consent to participate: Not Applicable.

Consent for publication: All authors have agreed to this submission.

Data availability statement: The authors will make the anonymized data supporting this study's findings available upon reasonable request.

Funding

Not Applicable.

Declaration of Generative Artificial Intelligence Use: ChatGPT (OpenAI) was used for language editing and readability. The authors reviewed and revised all content and take full responsibility for the final manuscript.

## CRediT authorship contribution statement

**Fateme Marandi:** Writing – review & editing, Writing – original draft, Visualization, Validation, Methodology, Formal analysis, Conceptualization. **Madjid Tavana:** Writing – review & editing, Writing – original draft, Visualization, Validation, Methodology, Formal analysis. **Ying Xie:** Writing – review & editing, Writing – original draft, Visualization, Validation, Methodology.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgments

The authors gratefully acknowledge the support of industrial partners and domain experts who contributed to the data collection and validation processes for this study. Production and supply chain data, including processing times, routing structures, and capacity information, were obtained through collaboration with firms operating in the zinc supply chain. These data reflect typical operational conditions over a representative planning horizon and were provided in aggregated and anonymized form due to confidentiality agreements.

Logistics and transportation parameters, such as transportation times, batch capacities, and cost estimates, were compiled from company inputs, industry reports, and publicly available sources. When direct data access was limited, parameter values were calibrated using industry benchmarks and further validated through discussions with practitioners to ensure consistency with real-world operations.

Information on zinc processing stages and raw material flows was gathered from publicly available technical reports and industry publications, which helped ensure that the modeled supply chain reflects realistic material flows and processing characteristics.

In addition, the model assumptions and parameter settings were reviewed with industry professionals and subject-matter experts in supply chain management and metal processing. Their input was valuable in confirming the case study's practical relevance and realism.

## Appendix A

**Local search I**

| | $o_{11}$ | $o_{12}$ | $o_{13}$ | $o_{21}$ | $o_{22}$ | $o_{23}$ | $o_{31}$ | $o_{32}$ | $o_{33}$ |
|---|---|---|---|---|---|---|---|---|---|
| $f_1$ | 0.213 | | | 0.738 | | 0.477 | | 0.312 | |
| $f_2$ | | | | | | | | | |
| $f_3$ | | | 0.569 | | 0.835 | | 0.678 | | |

*Exchange with a random uniform number*

**Local search II**

| | $o_{11}$ | $o_{12}$ | $o_{13}$ | $o_{21}$ | $o_{22}$ | $o_{23}$ | $o_{31}$ | $o_{32}$ | $o_{33}$ |
|---|---|---|---|---|---|---|---|---|---|
| $f_1$ | 0.213 | | | 0.738 | | 0.477 | | 0.312 | |
| $f_2$ | | 0.873 | | | | | | | 0.212 |
| $f_3$ | | | **0.569** | | **0.835** | | 0.678 | | |

**Local search III**

| | $o_{11}$ | $o_{12}$ | $o_{13}$ | $o_{21}$ | $o_{22}$ | $o_{23}$ | $o_{31}$ | $o_{32}$ | $o_{33}$ |
|---|---|---|---|---|---|---|---|---|---|
| $f_1$ | 0.213 | | | **0.738** | | 0.477 | | 0.312 | |
| $f_2$ | | **0.873** | | | | | | | 0.212 |
| $f_3$ | | | 0.569 | | 0.835 | | 0.678 | | |

**Fig. A1.** Local search methods of *s* are implemented with equal chance

**Algorithm 2: schedule_local_search_I**

**INPUT:** *s*
*Step* 1. *new_s* ← *s*
*Step* 2. Select an operation $o_{ij}$, randomly
*Step* 3. Generate a random number ∈ (0, 1)
*Step* 4. Replace *new_s* ($o_{ij}$) with the generated random number in *Step* 3
*Step* 5. *status* ← **schedule_feasible_checker** (*new_s*)
    **IF** (*status* == *infeasible*)
        *new_s* ← **schedule_repair** (*new_s*)
    **END**
**OUTPUT:** *new_s*

**Fig. A2.** Pseudocode of scheduling local search I

**Algorithm 3: schedule_local_search_II**

**INPUT:** *s*
*Step* 1. *new_s*← *s*
*Step* 2. Select a factory *f*, randomly
*Step* 3. Select two random operations ($o_{ij}$ , $o_{i'j'}$) from *f*
*Step* 4. Swap *new_s* ($o_{ij}$) and *new_s*($o_{i'j'}$)
*Step* 5. *status* ← **schedule_feasible_checker** (*new_s*)
    **IF** (*status* == *infeasible*)
        *new_s* ← **schedule_repair** (*new_s*)
    **END**
**OUTPUT:** *new_s*

**Fig. A3.** Pseudocode of scheduling local search II

**Algorithm 4: schedule_local_search_III**

**INPUT:** $s$
*Step* 1. $new\_s \leftarrow s$
*Step* 2. Select two factories $f$ and $f'$, randomly
*Step* 3. Select random operations $o_{ij}$ from $f$ and $o_{i'j'}$ from $f'$
*Step* 4. Swap $new\_s(o_{ij})$ and $new\_s(o_{i'j'})$
*Step* 5. $status \leftarrow$ **schedule_feasible_checker** ($new\_s$)
    **IF** ($status == infeasible$)
        $new\_s \leftarrow$ **schedule_repair** ($new\_s$)
    **END**
**OUTPUT:** $new\_s$

**Fig. A4.** Pseudocode of scheduling local search III

**Algorithm 5: batch_local_search**

**INPUT:** $b$; $C\_R$
*Step* 1. $new\_b \leftarrow b$
*Step* 2. Select two WIPs ($wip_{ij}$, $wip_{i'j'}$) $\in C\_R$, randomly
*Step* 3. **IF** ($wip_{ij}$ **AND** $wip_{i'j'}$ belong to the same batch $\beta$)
        Remove $wip_{i'j'}$ from $\beta$
        Assign $wip_{i'j'}$ to the first available batch $\beta'$ with the lowest batching cost
        Update $b$
    **ELSE**
        $\beta \leftarrow$ index of the batch containing $wip_{ij}$
        $\beta' \leftarrow$ index of the batch containing $wip_{i'j'}$
        **IF** (remaining capacity of $\beta \geq$ required capacity of batching $wip_{i'j'}$)
            Remove $wip_{i'j'}$ from $\beta'$
            Insert $wip_{i'j'}$ into $\beta$
            Find operation $wip_{i''j''} \notin C\_R$ assigned to batch $\beta''$ with highest batching cost
            **IF** (batching cost of $\beta'' >$ batching cost of $\beta'$)
                Remove $wip_{i''j''}$ from $\beta''$
                Insert $wip_{i''j''}$ into $\beta'$
            **END**
        **END**
        Update $b$
    **END**
**OUTPUT:** $new\_b$

**Fig. A5.** Pseudocode of batching local search

## Appendix B

**Proof of Dominance Rule.**

Let factories $f$ and $f'$ be the departure and destination of $wip_{ij}$ and $wip_{i'j'}$, respectively. For convenience, since the transportation time from $f$ to $f'$ is the same for both WIPs, it is not considered in this proof. It is obvious that at factory $f$, $ST_{ij} \leq ST_{i'j'}$ and at factory $f'$, $ST_{i(j+1)} \geq ST_{i'(j'+1)}$. In addition, due to the precedence constraints in the scheduling problem, it holds that $ST_{i(j+1)} \geq ST_{ij} + p_{ij}q_i$ and $ST_{i'(j'+1)} \geq ST_{i'j'} + p_{i'j'}q_{i'}$. Hence, it can be shown that: $ST_{i'(j'+1)} \geq ST_{i'j'} + p_{i'j'}q_{i'} \geq ST_{ij} + p_{ij}q_i$ and $ST_{i(j+1)} \geq ST_{i'(j'+1)} + p_{i'(j'+1)}q_{i'} \geq ST_{i'j'} + p_{i'j'}q_{i'}$. This means that in the given scheduling solution, constraints $ST_{i'(j'+1)} \geq ST_{ij} + p_{ij}q_i$ and $ST_{i(j+1)} \geq ST_{i'j'} + p_{i'j'}q_{i'}$ are held for $wip_{ij}$ and $wip_{i'j'}$. As a result, $wip_{ij}$ and $wip_{i'j'}$ can be delivered in the same batch without tardiness and holding costs increase, and hence $TC(\left[wip_{ij} - wip_{i'j'}\right]) < TC(\left[wip_{ij}\right] - \left[wip_{i'j'}\right])$.

## Data availability

Data will be made available on request.

## References


[1] N. Bagheri Rad, J. Behnamian, Recent trends in distributed production network scheduling problem, Artif. Intell. Rev. 55 (4) (2022) 2945–2995.
[2] U.M. Modibbo, S. Gupta, A. Ahmed, I. Ali, An integrated multi-objective multi-product inventory managed production planning problem under uncertain environment, Ann. Oper. Res. 339 (3) (2024) 1679–1723.
[3] S. Chorghe, R. Kumar, M.S. Kulkarni, V. Pandhare, B.K. Lad, Smart scheduling for next generation manufacturing systems: a systematic literature review, J. Intell. Manuf. 36 (7) (2025) 4447–4476.
[4] H. Hilali, V. Hovelaque, V. Giard, Integrated scheduling of a multi-site mining supply chain with blending, alternative routings and co-production, Int. J. Prod. Res. 61 (6) (2023) 1829–1848.
[5] S.A. Razavi Al-e-hashem, A. Papi, M.S. Pishvaee, M. Rasouli, Robust maintenance planning and scheduling for multi-factory production networks considering disruption cost: a bi-objective optimization model and a metaheuristic solution method, Oper. Res. 22 (5) (2022) 4999–5034.
[6] J. Behnamian, S.F. Ghomi, The heterogeneous multi-factory production network scheduling with adaptive communication policy and parallel machine, Inf. Sci. 219 (2013) 181–196.
[7] F. Marandi, S.M.T. Fatemi Ghomi, Integrated multi-factory production and distribution scheduling applying vehicle routing approach, Int. J. Prod. Res. 57 (3) (2019) 722–748.
[8] C.P. Garg, Ö.F. Görçün, P. Kundu, H. Küçükönder, An integrated fuzzy MCDM approach based on Bonferroni functions for selection and evaluation of industrial robots for the automobile manufacturing industry, Expert Syst. Appl. 213 (2023) 118863.
[9] Y. Fu, Z. Zhang, K. Gao, Q. Pan, H.F. Rahman, Integrated distributed flexible job shop scheduling and vehicle routing problem via Q-learning-based evolutionary algorithms, Inf. Sci. 713 (2025) 122169.
[10] K. Zhang, B. Zhou, A self-adaptive Gaussian mutation-based arithmetic optimiser algorithm for integrated production scheduling and vehicle routing problem in the distributed manufacturing environment, Int. J. Prod. Res. (2024) 1–29.
[11] M. Ganji, R. Rabet, S.M. Sajadi, A new coordinating model for green supply chain and batch delivery scheduling with satisfaction customers, Environ. Dev. Sustain. (2022) 1–36.
[12] A. Gharaei, F. Jolai, A multi-agent approach to the integrated production scheduling and distribution problem in multi-factory supply chain, Appl. Soft Comput. 65 (2018) 577–589.
[13] J. Lohmer, R. Lasch, Production planning and scheduling in multi-factory production networks: a systematic literature review, Int. J. Prod. Res. 59 (7) (2021) 2028–2054.
[14] Z. Xu, A. Elomri, R. Baldacci, L. Kerbache, Z. Wu, Frontiers and trends of supply chain optimization in the age of industry 4.0: an operations research perspective, Ann. Oper. Res. (2024) 1–43.
[15] J. Behnamian, S.M.T.F. Ghomi, Multi-Objective Multi-Factory Scheduling. RAIRO-Operations Research 55 (2021) S1447–S1467.
[16] J. Behnamian, S.M.T. Fatemi Ghomi, A survey of multi-factory scheduling, J. Intell. Manuf. 27 (2016) 231–249.
[17] B. Peterson, I. Harjunkoski, S. Hoda, J.N. Hooker, Scheduling multiple factory cranes on a common track, Comput. Oper. Res. 48 (2014) 102–112.
[18] X.T. Sun, S.H. Chung, F.T. Chan, Integrated scheduling of a multi-product multi-factory manufacturing system with maritime transport limits, Transportation Research Part E: Logistics and Transportation Review 79 (2015) 110–127.
[19] A.S. Santos, A.M. Madureira, M.L.R. Varela, G.D. Putnik, H.E. Kays, A.N.M. Karim, in: June). Scheduling and Batching in Multi-Site Flexible Flow Shop Environments, IEEE, 2015, pp. 1–6.
[20] N. Karimi, H. Davoudpour, A branch and bound method for solving multi-factory supply chain scheduling with batch delivery, Expert Syst. Appl. 42 (1) (2015) 238–245.
[21] H. Kazemi, M. Nourelfath, M. Gendreau, The multi-factory two-stage assembly scheduling problem, J. Ind. Inf. Integr. 38 (2024) 100574.
[22] C. Santos, M. Magazine, Batching in single operation manufacturing systems, Oper. Res. Lett. 4 (3) (1985) 99–103.
[23] C.N. Potts, M.Y. Kovalyov, Scheduling with batching: a review, Eur. J. Oper. Res. 120 (2) (2000) 228–249.
[24] Y. Yin, D. Wang, T.C.E. Cheng, Y. Yin, D. Wang, T.C.E. Cheng, Integrated production, inventory, and batch delivery scheduling with due date assignment, Models and Algorithms, Due Date-Related Scheduling with Two Agents, 2020, pp. 93–141.
[25] S.H. Chung, H.C. Lau, K.L. Choy, G.T. Ho, Y.K. Tse, Application of genetic approach for advanced planning in multi-factory environment, Int. J. Prod. Econ. 127 (2) (2010) 300–308.
[26] A. Agnetis, M.A. Aloulou, L.L. Fu, Coordination of production and interstage batch delivery with outsourced distribution, Eur. J. Oper. Res. 238 (1) (2014) 130–142.
[27] K. Wang, W.Q. Ma, H. Luo, H. Qin, Coordinated scheduling of production and transportation in a two-stage assembly flowshop, Int. J. Prod. Res. 54 (22) (2016) 6891–6911.
[28] M. Ganji, H. Kazemipoor, S.M.H. Molana, S.M. Sajadi, A green multi-objective integrated scheduling of production and distribution with heterogeneous fleet vehicle routing and time windows, J. Clean. Prod. 259 (2020) 120824.
[29] H. Zhou, Q. Zhou, C. Liu, T. Zhou, A kriging metamodel-assisted robust optimization method based on a reverse model, Eng. Optim. 50 (2) (2018) 253–272.
[30] Y.C. Chang, V.C. Li, C.J. Chiang, An ant colony optimization heuristic for an integrated production and distribution scheduling problem, Eng. Optim. 46 (4) (2014) 503–520.
[31] L. Tang, Z. Jin, X. Qin, K. Jing, Supply chain scheduling in a collaborative manufacturing mode: model construction and algorithm design, Ann. Oper. Res. 275 (2019) 685–714.
[32] I. Gunawan, D. Trihastuti, A. Kumar, K.H. Tan, Integrated DANP and binary goal programming model in generating joint-decision making for packaging postponement and supplier selection, Ann. Oper. Res. (2023) 1–30.
[33] Ç. Cergibozan, A.S. Tasan, Order batching operations: an overview of classification, solution techniques, and future research, J. Intell. Manuf. 30 (2019) 335–349.
[34] J. Sauer, H.J. Appelrath, January). Integrating Transportation in a Multi-Site Scheduling Environment, IEEE, 2000.
[35] A. Guinet, Multi-site planning: a transshipment problem, Int. J. Prod. Econ. 74 (1–3) (2001) 21–32.
[36] H.Z. Jia, A.Y. Nee, J.Y. Fuh, Y.F. Zhang, A modified genetic algorithm for distributed scheduling problems, J. Intell. Manuf. 14 (2003) 351–362.
[37] H.Z. Jia, J.Y. Fuh, A.Y. Nee, Y.F. Zhang, Integration of genetic algorithm and Gantt chart for job shop scheduling in distributed manufacturing systems, Comput. Ind. Eng. 53 (2) (2007) 313–320.
[38] P. Kaminsky, O. Kaya, Inventory positioning, scheduling and lead-time quotation in supply chains, Int. J. Prod. Econ. 114 (1) (2008) 276–293.
[39] E. Torabzadeh, M. Zandieh, Cloud theory-based simulated annealing approach for scheduling in the two-stage assembly flowshop, Adv. Eng. Softw. 41 (10–11) (2010) 1238–1243.
[40] T. Nishi, Y. Hiranaka, I.E. Grossmann, A bilevel decomposition algorithm for simultaneous production scheduling and conflict-free routing for automated guided vehicles, Comput. Oper. Res. 38 (5) (2011) 876–888.
[41] F. H'Mida, P. Lopez, Multi-site scheduling under production and transportation constraints, Int. J. Comput. Integr. Manuf. 26 (3) (2013) 252–266.
[42] M. Matusiak, R. de Koster, L. Kroon, J. Saarinen, A fast simulated annealing method for batching precedence-constrained customer orders in a warehouse, Eur. J. Oper. Res. 236 (3) (2014) 968–977.
[43] F. Nikzad, J. Rezaeian, I. Mahdavi, I. Rastgar, Scheduling of multi-component products in a two-stage flexible flow shop, Appl. Soft Comput. 32 (2015) 132–143.
[44] B.Y. Cheng, J.Y.T. Leung, K. Li, Integrated scheduling of production and distribution to minimize total cost using an improved ant colony optimization method, Comput. Ind. Eng. 83 (2015) 217–225.
[45] Z. Guo, D. Zhang, S.Y.S. Leung, L. Shi, A bi-level evolutionary optimization approach for integrated production and transportation scheduling, Appl. Soft Comput. 42 (2016) 215–228.

[46] A. Noroozi, M. Mahdavi Mazdeh, M. Rasti-Barzoki, Coordinating order acceptance and batch delivery for an integrated supply chain scheduling, Int. J. Eng. 30 (5) (2017) 700–709.
[47] E.M. Frazzon, A. Albrecht, M. Pires, E. Israel, M. Kück, M. Freitag, Hybrid approach for the integrated scheduling of production and transport processes along supply chains, Int. J. Prod. Res. 56 (5) (2018) 2019–2035.
[48] S. Dempe, Foundations of bilevel programming, Springer, US, Boston, MA, 2002.
[49] Y. Wang, A new hybrid genetic algorithm for job shop scheduling problem, Comput. Oper. Res. 39 (10) (2012) 2291–2299.
[50] F. Glover, K. Sörensen, Metaheuristics. Scholarpedia 10 (4) (2015) 6532.