A Bi-level Decomposition Algorithm for Multi-factory Scheduling With Batch Delivery Problem

Fateme Marandi
Amirkabir University of Technology
S.M.T. Fatemi Ghomi (fatemi@aut.ac.ir)
Amirkabir University of Technology https://orcid.org/0000-0003-4363-994X

Research Article

Keywords: Multi-factory, Bi-level decomposition, Scheduling, Batch delivery, Supply chain

Posted Date: January 4th, 2022

DOI: https://doi.org/10.21203/rs.3.rs-1151995/v1

A bi-level decomposition algorithm for multi-factory scheduling with batch delivery problem

Fateme Marandi and S.M.T. Fatemi Ghomi[1]

Department of Industrial Engineering, Amirkabir University of Technology, Tehran, Iran

Abstract

This paper introduces a multi-factory scheduling with batch delivery problem. A novel mixed-integer programming model is proposed to minimize the sum of total tardiness, holding and batching costs. A bi-level decomposition algorithm (BLDA) is developed involving two sub-problems: scheduling problem in the upper level and batching problem in the lower level. Four versions of the BLDA are created by combinations of CPLEX and simulated annealing in both levels, which interactively collaborate until the algorithm converges to a solution. The BLDAs are examined on several random and real-life test instances. A statistical analysis is performed by comparing the BLDAs' solutions with the exact minimum and lower bound values of the total cost. The results indicate that about all versions of the developed BLDA provide high quality solutions for real-world zinc industry problems as well as generated instances in a reasonably short time. Finally, some managerial insights are derived based on sensitivity analysis.

Keywords: Multi-factory, Bi-level decomposition, Scheduling, Batch delivery, Supply chain.

[1] Corresponding Author; E-mail address: fatemi@aut.ac.ir

1. Introduction

Recent growing competitiveness causes companies to consider the activities from the whole supply chain (SC) instead of the own manufacturers' activities. In the other hand, they prefer to optimize the SC from supplier to customer. SC consists of developed structure in suppliers, customers, manufacturers, flow of information and products. Then coordination in scheduling and planning of these components, sharing the information of goods and services lead to enhance the performance of SC, lower operating costs, less waiting of inventory, etc.

The significant reputation of scheduling in the context of multi-factory production has been one of the motivating topics in SC management because of the progress of globalization. Generally, there are three well-known configurations in the multi-factory environment, i.e. parallel, serial, and network configuration. In parallel configuration, each factory can produce the final products to supply them to customers while factories in serial configuration can produce final or intermediate products to send to another factory with the same processing route, e.g. see (Santos, Madureira et al. 2015, Karimi and Davoudpour 2016). In network configuration which is restricted to the studies of Kaminsky and Kaya (2008) and H'Mida and Lopez (2013), each factory can produce the final or intermediate products processed in different routes with possibility of reentrant processing, which has been paid less attention. Due to the number of factories through the SC, the scheduling activities are much more complicated in the latter configuration (Behnamian and Ghomi 2016). In addition, factories connection in such a complicated SC network heightens the complexity of this problem, which is studied by the current paper elaborately. The scheduling activities of network configuration are much more complicated than other configurations due to the structure of factories and non-identical processing routes consisting of reentrant processing on some factories. In fact, network configuration is the combination of parallel and serial configurations. There is no simple flow of material among factories that can be adapted in parallel or serial. It is varied based on rout of operations that may be different for each job. In order to schedule a number of factories located in various regions with different flow, multi-factory scheduling is applied to optimize the aims.

Interconnection among factories in the network configuration multi-factory SC leads to the great level of complexity. This means that the material shortage in one factory can influence on another one. In other words, it causes delay or production stoppage through the entire SC.

Therefore, production scheduling and transportation among factories should be synchronized to decline holding costs of inventory and in-process products. The products of a factory may be transmitted to the next factory as the input raw material then the total cost depends on the transportation cost of products. In such a network, the effective use of batches' capacities would reduce the batching costs, significantly. Thus, batch delivery is a reasonable approach which has severe effect on the overall performance of the SC (Govindan, Fattahi et al. 2017).

This paper introduces multi-factory scheduling with batch delivery problem (MFSBDP). It considers the SC including job scheduling among factories and allocation of in-process jobs with the same processing route to the same batches. In the assumed complex SC network, although batching of in-process jobs with the same processing routes may increase tardy and holding costs, it decreases the transportation costs, i.e. batching costs. Therefore, a trade-off has to be made between batching costs, tardy and holding costs to minimize total cost that is studied in this paper.

For the first time, as far as authors know, a novel mixed-integer programming (MIP) model is developed to provide optimal batching and scheduling solutions simultaneously using CPLEX. Since this problem is strongly NP-hard, a bi-level decomposition algorithm (BLDA) is presented which decomposes the main problem into two sub-problems: the scheduling problem at the upper level and the batching problem at the lower level. In the proposed approach each sub-problem is dealt separately in its own level and its solution is used as the input parameters to the other level. A simulated annealing (SA) as well as CPLEX is utilized to tackle each of the sub-problems in the BLDA, i.e. there are four versions of the BLDA based on the combinations of SA and CPLEX. According to a real-life zinc SC network, some different test problems are generated in the context of MFSBDP. Then, the BLDA is utilized to deal with the test problems. According to the numerical results, it is statistically shown that the proposed BLDA provides optimal and near-optimal solutions for even large-sized real-world problems. In summary, main contributions of this paper are listed as follows.

- A new network configuration multi-factory scheduling problem is introduced with batch delivery and reentrant process.
- A novel MIP model is formulated to solve the MFSBDP using CPLEX optimally.
- A new bi-level decomposition algorithm is devised to tackle the MFSBDP.

- A case study motivated by the real-world problem in a zinc SC network is applied to suggest some managerial insights.

The rest of the paper is structured as follows. In Section 2, the related papers to this study 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, Section 6 is devoted to conclusion.

2. Literature review

Since 1981, researchers have surveyed the issue of scheduling in multi-factory production networks to a great extent. Behnamian and Ghomi (2016) extensively review multi-factory scheduling problems with classification of shop environment, multi-factory configuration and solution approach. They also suggest some guidelines for future research. Peterson, Harjunkoski et al. (2014) study multi-factory scheduling with crane delivery. They present a new heuristic by presenting a decision tree of possible states of crane system. Sun, Chung et al. (2015) investigate integrated multi-factory scheduling with maritime transportation. They propose a novel genetic algorithm by fuzzy controller to minimize the total operating costs. Marandi and Fatemi Ghomi (2018) deal with integrated multi-factory scheduling with vehicle routing problem to satisfy customers' demand. They propose a novel improved imperialist competitive algorithm to minimize delivery and tardy costs. Gharaei and Jolai (2018) study parallel multi-factory scheduling with batch delivery to minimize two objectives of total tardy and distribution costs. They also propose a new multi-objective combined with bees algorithm in a decomposition approach.

Due to essence of transporting products among factories, transportation costs play an important role in optimization of multi-factory systems. Thus considering production and transportation in such systems is of special significance in the literature. Santos and Magazine (1985) firstly address batching problem for single machine scheduling. Afterward, batching problem increasingly has been considered by many researchers. Potts and Kovalyov (2000) comprehensively review batch delivery in machine scheduling problems with respect to models of different shop scheduling and design the efficient dynamic programming for solving these problems. Recently, Yin, Wang et al. (2020) consider parallel machine scheduling and batch delivery with due date assignment. They show significant reduction in total costs by batch delivery.

In contrast to the wide study in machine scheduling and batch delivery, few studies have investigated the multi-factory scheduling and batch delivery in the literature. Chung, Lau et al. (2010) study multi-factory scheduling with intermediate batch delivery to minimize makespan by presenting a modified genetic algorithm. Agnetis, Aloulou et al. (2014) survey semi-finished products scheduling with batch delivery within two transportation modes to minimize transportation costs. Karimi and Davoudpour (2015) present multi-factory scheduling in serial configuration with batch delivery by proposing a branch and bound method. Wang, Ma et al. (2016) study scheduling problem with batch delivery to customers to minimize the mean sum of arrival time by proposing two heuristics. Gharaei and Jolai (2018) study parallel multi-factory scheduling with batch delivery to minimize two objectives of total tardiness and distribution costs with bees algorithm in a decomposition approach. Ganji, Kazemipoor et al. (2020) study green multi-objective integrated production and distribution scheduling applying vehicle routing problem and batch delivery. These recently increasing attentions in the literature of this area motivate us to consider multi-factory scheduling and batch delivery in the current study.

In spite of considering exact methods to guarantee the optimal solution for small-sized problems (Zhou, Zhou et al. 2018), widespread studies have been increasingly developed heuristic and metaheuristic approaches in the literature, e.g. see (Chang, Li et al. 2014). As multi-factory scheduling is one of the most complex structures among scheduling problem, the decomposition-based approach can be applied to break the problem into smaller parts to tackle the intractable problem (Behnamian and Ghomi 2016). This innovative and superior approach is used in this study.

2.1. *Gap analysis*

In practice, batch delivery is considered as a key beneficial strategy to decrease operating costs in scheduling problems. Regardless of scheduling cost, most of the previous studies considered only single-factory instead of multi-factory scheduling problem without batching problem. Considering the operating costs, this paper introduces a new problem in the context of multi-factory scheduling with batch delivery to minimize batching, tardiness and holding costs. Additionally, it provides a new bi-level decomposition approach to deal with the problem. To the best of the authors' knowledge, this is the first study that discusses and deals with MFSBDP considering network configuration in the literature. Not only from a mere academic perspective but also from a practical view, this study bears potentials to reduce operational costs and coordinate the production and distribution scheduling in real life case study significantly, resulting in a more profitable SC.

Table 1 presents a detailed summary of the papers related to this study. As illustrated in Table 1, although papers investigate the single-factory environment, there are many more papers considers multi-factory configurations such as network configuration that require more sophisticated approaches to study detailed aspects of the systems.

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 (2000)	Multi-factory, parallel configuration	✓		Single level, MIP	Fuzzy heuristic	Robust scheduling
Guinet (2001)	Multi-factory, parallel configuration	✓		Single level	Heuristic	Variable and fixed costs
Jia, Nee et al. (2003)	Multi-factory, parallel configuration	✓		Single level	GA	C_{max}
Jia, Fuh et al. (2007)	Multi-factory, parallel configuration	✓			GA	Operating costs
Kaminsky and Kaya (2008)	Multi-factory, network configuration	✓			Heuristic	C_{max}
Torabzadeh and Zandieh (2010)	Single-factory	✓		Single level, MIP	CSA	C_{max}
Nishi, Hiranaka et al. (2011)	Single-factory	✓		Bi-level, MIP	BLD & Lagrangian	Tardy costs
Chung, Chan et al. (2011)	Multi-factory, parallel configuration	✓		Single level, MIP	GA	
Ruifeng and Subramaniam (2011)	Multi-factory, serial configuration	✓		Single level, MIP		Operating costs
H'Mida and Lopez (2013)	Multi-factory, network configuration	✓		Single level, CSP		C_{max}
Behnamian and Ghomi (2013)	Multi-factory, parallel configuration	✓		Single level, MIP	GA-LS	C_{max}
(Peterson, Harjunkoski et al. (2014	Multi-factory, parallel configuration	✓		Single level, MIP	Heuristic	
Matusiak, de Koster et al. (2014)	Single-factory	✓	✓	Single level	A* algorithm & SA	Total delivery cost
(Santos, Madureira et al. (2015	Multi-factory, serial configuration	✓	✓	Single level, MIP		C_{max}
Nikzad, Rezaeian et al. (2015)	Single-factory	✓		Single level	ICA-SA	C_{max}
(Sun, Chung et al. (2015	Multi-factory, parallel configuration	✓		Single level, MIP	GA	Operating costs
(Cheng, Leung et al. (2015	Single-factory	✓			DP	C_{max}
Guo, Zhang et al. (2016)	Single-factory	✓	✓	Bi-level, MINLP	Bi-level evolutionary	Operating costs
Noroozi, Mazdeh et al. (2017)	Single-factory	✓	✓	Single level, MIP	PSO-GA	Maximize total benefit
Frazzon, Albrecht et al. (2018)	Single-factory	✓	✓	Single level, MIP	Simulation	Transport, production and storage costs
Gharaei and Jolai (2018)	Multi-factory, parallel configuration	✓	✓	Single-level, MIP	Bees algorithm	Total transport and tardy costs
Current study	Multi-factory, network configuration	✓	✓	Bi-level, MIP, LP	BLDA & SA	Batching, tardy and holding costs

SA: Simulated annealing	CSA: Cloud based simulated annealing	GA: Genetic algorithm
PSO: Particle swarm optimization	ICA: Imperialist competitive algorithm	LS: Local search
DP: Dynamic programming	BLD: Bi-level decomposition algorithm	CSP: Constraint satisfaction problem
LP: Linear programming	MIP: Mixed-integer linear programming	MINLP: Mixed-integer nonlinear programming

3. Problem description

This study investigates MFSBDP in a SC which contains a set of factories in a network configuration. A set of jobs are processed and scheduled via a set of operations with various reentrant processing routes performed in different factories that are transmitted to another factory as its input material. These unfinished products are referred to as work-in-process inventories (WIPs), which are transported between factories via a sufficient limited-capacity number of batches to reduce transportation cost of jobs, i.e. batching cost. An operation of a job cannot start in a factory until the required WIP is received from the departure factory. It is supposed that WIPs that pass the same route from one factory to another can be allocated to the same batch and delivered together. Also, it is assumed that a WIP cannot be divided and delivered in separate batches. Therefore, the completion time of each batch is calculated according to the production time of the last WIP in the batch at a factory. It is preferred to deliver jobs to customers before their due dates. Tardiness cost is induced when a job is delivered after its due date. As WIPs should wait to be assigned to the same batch, batching postpones starting time of some WIPs at the next factory. Hence, batching decisions is one of the critical aspects to be regarded in this paper that impacts on tardiness, holding and batching costs. Thus the goal of this paper is determination of production scheduling and assignment of WIPs to batches to minimize to the total tardiness, holding and batching costs.

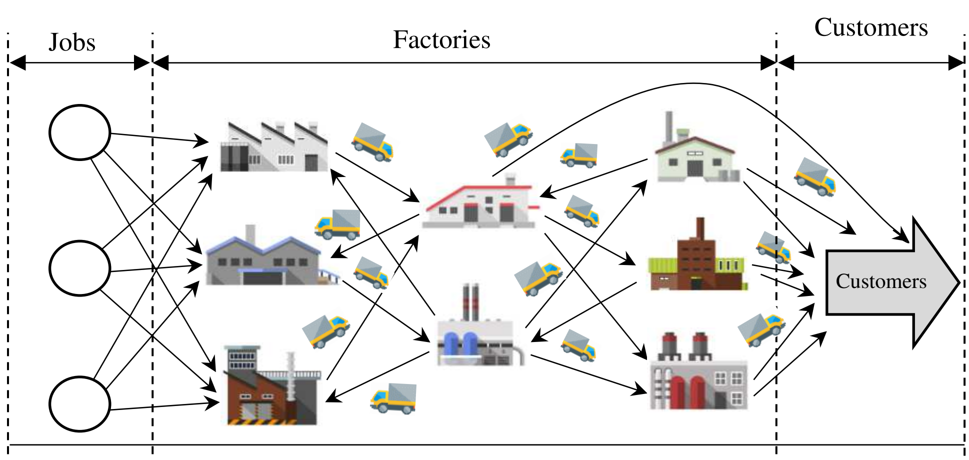

Figure 1. The supply chain network of multi-factory scheduling with batch delivery problem

Figure 1 illustrates the SC network of MFSBDP. It is conjectured that the batching assumption, as one of the key features of the current study, can reduce transportation costs of the company. However, it may increase the makespan because a batch can be released when the last WIP allocated to the batch is produced. Another problem raised by the batching approach is related to holding costs of WIPs, waiting for batch completion. Consequently, a compromise has to be made between batching costs, tardiness and holding costs, discussed with further explanations in this paper. The assumptions and notations of MFSBDP are listed as follows.

- All jobs are available at the start of planning era.
- 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 limited capacity and can be released when the last WIP of the batch is produced.
- A WIP consists of incomplete products which are produced by an operation that cannot be divided into two or more parts.

Indices

f, f'	index for factory,	$f = 1, 2, ..., F$
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 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^{th} operation of job i,
p_{ij}	processing time of operation o_{ij},
θ_{ij}	volume of each unit of job i produced at the j^{th} operation,
τ_{ij}	holding cost of operation o_{ij} while being transported between factories or waiting for batch completion as a work-in-process inventory,
r_{ijf}	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 job i,
α_i	tardiness cost of late delivery of job i per each unit time,
c^b	capacity of batch b,
λ^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 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 the model $\mathbf{M_1}$, the sum of holding and tardiness costs.
TC_2	objective function of the model $\mathbf{M_2}$, the batching costs.

3.1. *Mathematical model*

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

Model **M**:

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

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

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

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

$$\sum_{f=1}^{F} r_{iM_i f} (ST_{iM_i} + p_{iM_i} q_i) - d_i \le TD_i \qquad i = 1, 2, \ldots, N \quad (5)$$

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

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

$$\sum_{b=1}^{B} Y_{ij}^{\ b} = 1 \qquad i = 1, 2, \ldots, N;\ j = 1, 2, \ldots, M_i \quad (8)$$

$$\sum_{j=1}^{M_i} Y_{ij}^{\ b} \le 1 \qquad i = 1, 2, \ldots, N; b = 1, 2, \ldots, B \quad (9)$$

$$Y_{ij}^{\ b} + Y_{i'N}^{\ b} \le 1 \qquad i' \ne i = 1, 2, \ldots, N;\ j = 1, 2, \ldots, M_i;\ b = 1, 2, \ldots, B \quad (10)$$

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

$$\sum_{i=1}^{N} \sum_{j=1}^{M_i} \theta_{ij} q_i Y_{ij}^{\ b} \le c^b Z^b \qquad b = 1, 2, \ldots, B \quad (12)$$

$$X_{iji'j'}, Y_{ij}^b, Z^b \in \{0,1\} \qquad i' \le i = 1, 2, \ldots, N;\ j = 1, 2, \ldots, M_i - 1;\ j' = 1, 2, \ldots, M_i - 1; b = 1, 2, \ldots, B \quad (13)$$

$$ST_{ij}, TD_i \geq 0 \qquad i = 1,2,...,N;\ j = 1,2,...,M_i \qquad (14)$$

Equation (1) shows the objective function to minimize the sum of total tardiness, holding, and batching costs. Constraint set (2) ensures that jobs are scheduled based on their starting times, which means the starting time of each operation has to be greater than or equal to the time required for processing its preceding operation and transporting it to the current factory. Constraint sets (3) and (4) indicate the necessity that only one job can be processed in a factory at any time so that just one of the constraint sets will be active while the other set will be inactive. Constraint set (5) calculates the tardiness for each job. Constraint sets (6) and (7) determine starting time for each operation. When both operations o_{ij} and $o_{i'j'}$ are allocated to the same batch, the starting time of the next operation on the next factory must be greater than or equal to the time that it takes to finish all operations allocated to the batch. Constraint set (8) assures that each operation must be allocated to only one batch. Constraint set (9) shows that none of the operations of a job can be allocated to the same batch. Constraint (10) states that the last operation of each job has to be delivered separately leaving the multi-factory environment. Based on constraint set (11), just operations with the same route can be assigned to the same batch. Constraint set (12) enforces the capacity limitation of the acquired batches on the allocated operations. Constraint sets (13) and (14) define domain of the decision variables.

For convenience in describing the proposed BLDA, two sub-models namely $\mathbf{M_1}$ and $\mathbf{M_2}$ are presented as follows. TC_1 is objective function of $\mathbf{M}_1$ that is sum of holding and tardiness costs. TC_2 is objective function of $\mathbf{M}_2$ that is sum of holding, tardiness and batching costs based on a given scheduling solution.

- Model $\mathbf{M_1}$:

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

Constraints (2)-(5), (13), (14)

- Model $\mathbf{M_2}$:

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

Constraints (8)-(12), (15)

3.2. *Complexity*

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

Proof. It is sufficient to show that the classical job shop problem, which is strongly NP-hard, is a special case of the MFSBDP, i.e. the job shop problem is not more difficult than the MFSBDP. Let $J_{M'} \| \sum_{j=1}^{N'} w'_j T'_j$ denote an instance of a job shop problem in which $\mathbf{M'}$ is the set of machines and $\mathbf{N'}$ denotes the set of jobs minimizing total weighted sum of tardiness, where w'_j and T'_j are the weight and tardiness of job j, respectively. Also, assume that the set of processing times is $\mathbf{P'}$ and the set of due dates is represented as $\mathbf{D'}$. Given this, an instance of a MFSBDP can be constructed as follows. The set of factories, jobs, processing times and due dates are represented as $\mathbf{M'}$, $\mathbf{N'}$, $\mathbf{P'}$, and $\mathbf{D'}$, respectively. Let parameters of transportation times, holding costs and batching costs all be equal to zero. Therefore, in the solution of MFSBDP, total holding and batching costs are equal to zero and total tardiness is in the minimum. This obtained solution is easily converted to the optimal solution of the job shop problem in a polynomial time. Therefore the MFSBDP is strongly NP-hard and the proof is complete. □

3.3. *Dominance condition*

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

Under some conditions, one may be able to batch WIPs without an increase in completion times. In other words, WIPs having the same routes can be batched and delivered in the same batch with no increase in completion times. Hence, the batching cost and also the total cost will decrease. These conditions are distinguished in a scheduling solution using the following lemma, called *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.

Lemma 2. 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 $TC([wip_{ij}–wip_{i'j'}]) < TC([wip_{ij}]–[wip_{i'j'}])$.

Proof. 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, $ST_{ij} \le ST_{i'j'}$ and at factory f', $ST_{i(j+1)} \ge ST_{i'(j'+1)}$. In addition, due to the precedence constraints in the scheduling problem, it holds that $ST_{i(j+1)} \ge ST_{ij} + p_{ij}q_i$ and $ST_{i'(j'+1)} \ge ST_{i'j'} + p_{i'j'}q_{i'}$. Hence, it can be shown that: $ST_{i'(j'+1)} \ge ST_{i'j'} + p_{i'j'}q_{i'} \ge ST_{ij} + p_{ij}q_i$ and $ST_{i(j+1)} \ge ST_{i'(j'+1)} + p_{i'(j'+1)}q_{i'} \ge ST_{i'j'} + p_{i'j'}q_{i'}$. This means that in the given scheduling solution, constraints $ST_{i'(j'+1)} \ge ST_{ij} + p_{ij}q_i$ and $ST_{i(j+1)} \ge ST_{i'j'} + p_{i'j'}q_{i'}$ are held for wip_{ij} and $wip_{i'j'}$. As a result, $wip_{i'j'}$ and wip_{ij} can be delivered in the same batch without tardiness and holding costs increase and hence $TC([wip_{ij} - wip_{i'j'}]) < TC([wip_{ij}] - [wip_{i'j'}])$.

□

3.4. *Lower bound*

TC is the objective function of **M** that is 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^* denote the minimum of 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 all equal to 0. It is clear that $TC^* \ge TC_1^* + TC_2^*$. Therefore, a lower bound (LB) can be found for the MFSBDP as $LB = TC_1^* + TC_2^*$.

4. Solution approach

The MFSBDP can be solved via BLDA consisting of two sub-problems decomposed into two levels, i.e. upper level and lower level. In the upper level, the scheduling sub-problem is to sequence a set of given jobs among factories with different reentering processing routes. In the lower level, the batching sub-problem is to assign WIPs to batches transported between factories. The BLDA's levels are interactively cooperating until the algorithm converges to a solution. A simulated annealing as well as CPLEX is utilized to tackle each of the sub-problems.

4.1. *The bi-level decomposition algorithm*

The BLDA starts by solving **M1** 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 **M2** including constraint sets (2), (15), and (16). In Equations (15)-(16), $\bar{X}_{iji'j'}$ 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 **M2** including the aforementioned constraints, using CPLEX or SA, the total cost, (*TC*) of the initial solution is obtained where the set of batched WIPs is represented as ψ. At the next iteration, the obtained batching solution is then converted into the input parameters of the upper level where **M1** is solved now with constraints (17)-(18). Given the scheduling solution, which might be different from the scheduling solution of the previous iteration, the constraint sets (2), (15)-(16) are added into **M2** and the procedure is continued as explained above. The algorithm terminates when a new *TC* could not be found or it reaches the maximum iteration limit.

The combination of SA and CPLEX will result into four versions of the BLDA: CPX-CPX, CPX-SA, SA-CPX, and SA-SA. For example, CPX-SA means that the scheduling problem is solved via CPLEX in the upper level, while SA is used to deal with the batching problem in the lower level. The BLDAs have been coded in Visual Studio C# 14 environment, and implemented on a computer with an Intel CORE i7 3.5GHz CPU and 16GB RAM memory. Figure 2 illustrates the process flow of the proposed BLDA.

$$M(2-r_{ijf}-r_{i'j'f})+M(1-\bar{X}_{iji'j'})+ST_{i'j'}-ST_{ij}\geq p_{ij}q_i \qquad 1\leq i<i'\leq N;\ j=1,2,...,M_i;\ j'=1,2,...,M_{i'};\ f=1,2,...,F \tag{15}$$

$$M(2-r_{ijf}-r_{i'j'f})+M\bar{X}_{iji'j'}+ST_{ij}-ST_{i'j'}\geq p_{i'j'}q_{i'} \qquad 1\leq i<i'\leq N;\ j=1,2,...,M_i;\ j'=1,2,...,M_{i'};\ f=1,2,...,F \tag{16}$$

$$ST_{i(j+1)}\geq ST_{i'j'}+p_{i'j'}q_{i'}+t_{ff'} \qquad 1\leq i<i'\leq N;\ j=1,2,...,M_i;\ j'=1,2,...,M_{i'};\ f=1,2,...,F;\ wip_{ij}\ and\ wip_{i'j'}\in\psi \tag{17}$$

$$ST_{i'(j'+1)} \geq ST_{ij} + p_{ij}q_i + t_{ff'} \qquad \begin{array}{l} 1 \leq i < i' \leq N;\ j = 1,2,\ldots,M_i; \\ j' = 1,2,\ldots,M_{i'};\ f = 1,2,\ldots,F; \\ wip_{ij} \text{ and } wip_{i'j'} \in \psi \end{array} \tag{18}$$

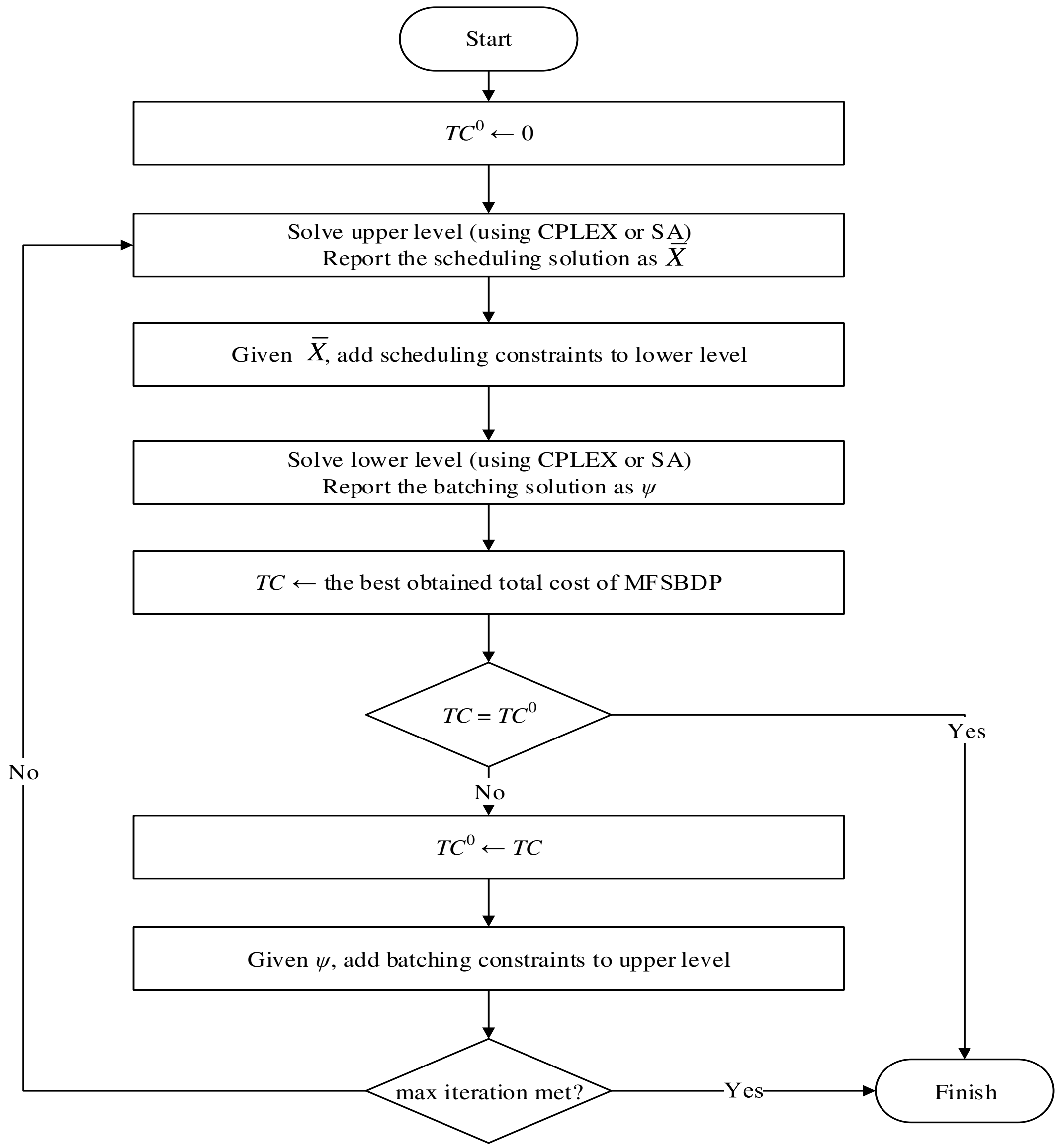

Figure 2. Process flow of the proposed BLDA

4.2. *Simulated annealing*

The SA algorithm starts with a randomly generated solution upgraded gradually 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 interchange the current solution with the worsening neighbor depending on the Metropolis acceptance measurement using

Equation (19), where *rnd* denotes a random uniform number and *temp_co* is coefficient of the temperature. The procedure is repeated for P_{max} times at every temperature's level, which is reduced based on Equation (20), where *cooling_co* is the cooling schedule coefficient. The general framework of SA used dealing with either scheduling or batching sub-problems is presented via a flow process chart, in Figure 3, and a pseudocode, in Figure 4.

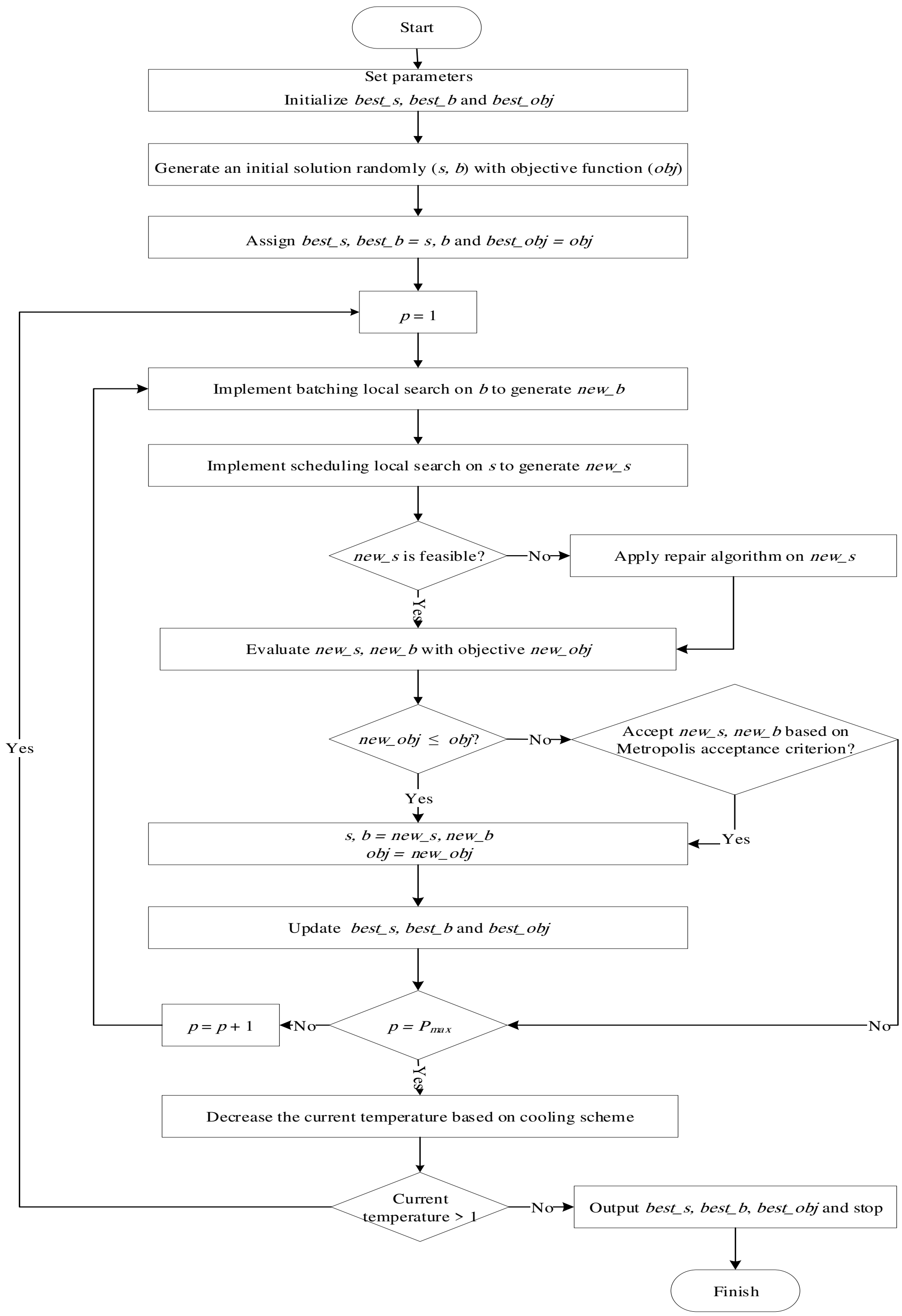

Figure 3. Flow process chart of the SA

Algorithm 1: simulated annealing algorithm

```
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
```

Figure 4. Pseudocode of the SA

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

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

Since the SA is used to solve each of the scheduling or the batching problems different solution representations are designed for each of these problems.

4.3. *Application of the SA for scheduling problem*

The scheduling solution s is created randomly based on the uniform distribution U (0, 1). Then, by scheduling local search, its neighbors are created through three different algorithms called *local search I*, *local search II*, and *local search III*, explained in Figure 5 for an example with three jobs each with three operations processed in three factories. The probability of selecting each local search algorithm is equal to 1/3. In order to make the scheduling solution feasible, a repair algorithm proposed by Qing-dao-er-ji and Wang (2012) is utilized.

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		0.873							**0.212**
f_3			0.569		0.835		0.678		

random uniform number ∈ (0,1)

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		

Figure 5. Local search methods of *s* implemented with equal chance

Pseudocodes of scheduling local search methods are provided in Figures 6-8.

Algorithm 2: schedule_local_search_I

INPUT: *s*
Step 1. *new_s* ← *s*
Step 2. Select an operations 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*

Figure 6. 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*

Figure 7. 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*

Figure 8. Pseudocode of scheduling local search III

4.4. *Application of the SA for batching problem*

To initialize a feasible batching solution b, it would be sufficient to assign all WIPs to all existing batches randomly. Then, by batching local search process some WIPs may get into the same batch; then, some other batches become vacant. Figure 9 depicts an example of b which is altered via two local search strategies, i.e. grouping and ungrouping strategies. More detail of the procedure is provided in Figure 10, where C_R denotes the set of operations that have common processing routes.

Grouping strategy

	o_{11}	o_{12}	o_{13}	o_{21}	o_{22}	o_{23}	o_{31}	o_{32}	o_{33}
b_1			1						
b_2									1
b_3					1				
b_4		1							
b_5				1					
b_6								1	
b_7									
b_8	1					1			
b_9							1		

Ungrouping strategy

	o_{11}	o_{12}	o_{13}	o_{21}	o_{22}	o_{23}	o_{31}	o_{32}	o_{33}
b_1				1					
b_2								1	
b_3					1		1		
b_4			1						
b_5	1								
b_6									
b_7						1			
b_8		1							
b_9									1

Figure 9. Batching local search: grouping and ungrouping strategies for b

Algorithm 5: batch_local_search

```
INPUT: b; C_R
Step 1. new_b ← b
Step 2. Select two WIPs (wip_ij , wip_i'j') ∈ C_R, randomly
Step 3. IF (wip_ij AND wip_i'j' belong to the same batch β)
            Remove wip_i'j' from β
            Assign wip_i'j' to the first available batch β′ with the lowest batching cost
            Update b
        ELSE
            β ← index of the batch containing wip_ij
            β′ ← index of the batch containing wip_i'j'
            IF (remaining capacity of β ≥ required capacity of batching wip_i'j')
                Remove wip_i'j' from β′
                Insert wip_i'j' into β
                Find operation wip_i"j" ∉ C_R assigned to batch β″ with highest batching cost
                IF (batching cost of β″ > batching  cost of β′)
                    Remove wip_i"j" from β″
                    Insert wip_i"j" into β′
                END
            END
            Update b
        END
OUTPUT: new_b
```

Figure 10. Pseudocode of batching local search

5. Case study

Zinc is one of the most consuming non-metallic minerals which have been developed due to its variety of applications in different industries. Iran has many zinc production plants processing zinc metal from ores to final ingots. Figure 11 illustrates a zinc SC network in 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, there are some by-products, which are yielded from the wastes produced in the zinc SC. The by-products, say heavy metals, are silver, cadmium, cobalt, nickel, and lead. Table 2 presents detailed information of zinc SC's products and production plants of the SC network. In this regard, basically, there are two kinds of concentrate products: sulfate concentrate and oxide concentrate. Sulfate concentrate is delivered to 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, zinc ingot group consists of low grade and high grade ingots, represented as type I and type II, respectively.

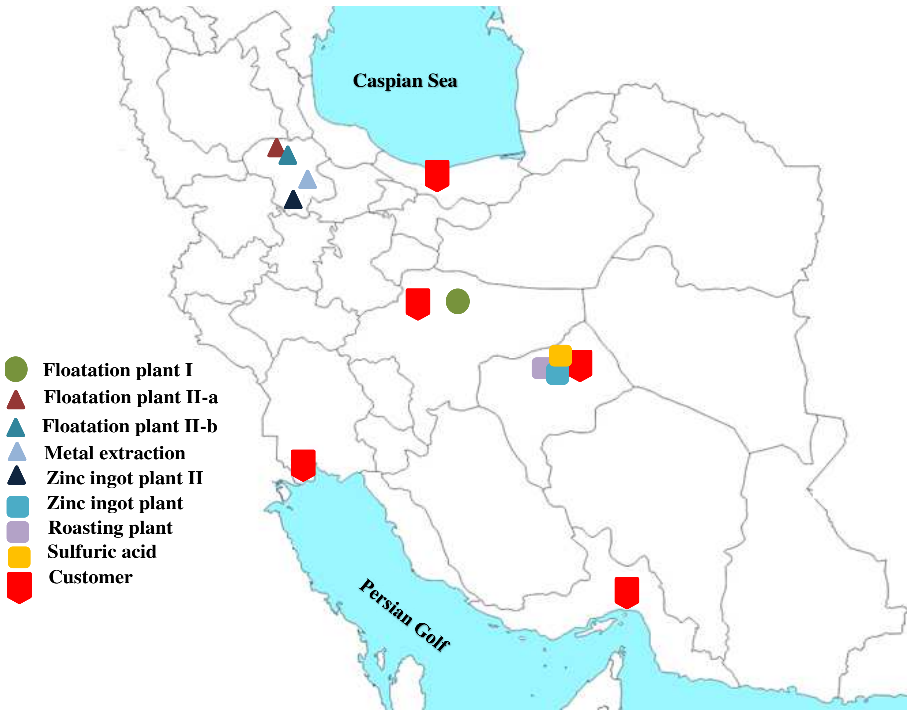

Figure 11. The SC network of the zinc case study on the map of Iran

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

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

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 via a unique color to better represent the SC network via Figure 12. It should be noted that, to the best of 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, it is temporarily assumed that such a central plant have been established in the network. This assumption can also motivate decision makers who read this article and compare the results to establish such a plant.

Details of the SC network are depicted in Figure 12, 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 next section, are based on real data and multiple routes. Data of heavy metals are assumed according to some interviews with experts.

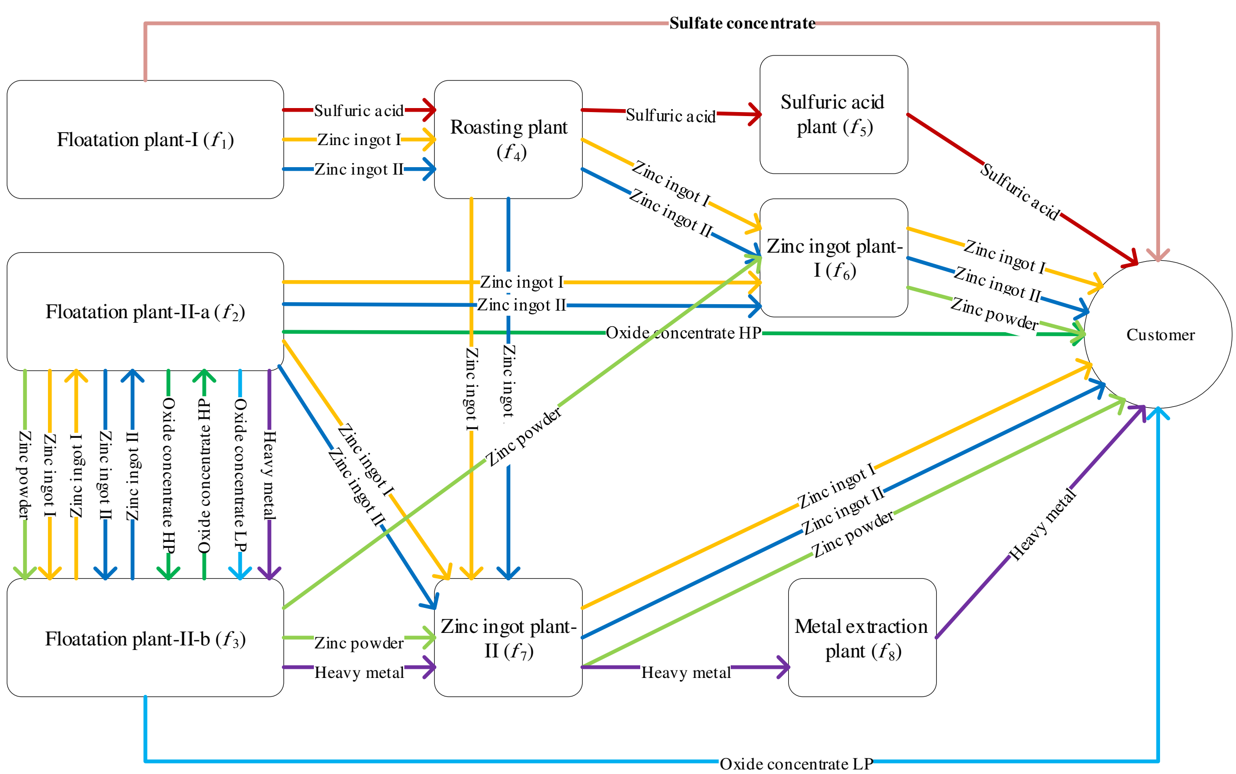

Figure 12. Process route of different jobs in the zinc SC

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

The problem data related to factory scheduling and batching capacity based on zinc supply chain is available. Moreover, the parameter of transportation costs and distance between factories are provided by Iran's Road Maintenance & Transportation Organization and the related company. The parameters related to zinc mines are applied based on National Geoscience Database of Iran and experts' knowledge.

5.2. *Results*

From the experimental results, the solution algorithms are validated by a diverse set of case instances adopted from the zinc SC network, i.e. case-I to case-V with at most 4 operations over 8 active factories. The units of WIPs are based on ton. 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*).

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

According to Table 3, minimum total costs obtained by BLDAs are close to each other and also they are close to the optimal total costs found by CPLEX. Regarding tardiness, holding and batching costs, Table 3 indicates that the BLDA algorithms converge to the optimal solution since their cost values are close to the values obtained by CPLEX. To evaluate performance of the BLDAs for dealing with large-sized problems, LB values are used. In this regard, performances of the BLDAs are close to the LB values as well. 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 hours while for SA-SA, SA-CPX, and CPX-CPX it is about 10, 11 and 15 hours, respectively. The aforementioned comparison of *STA*s approves this result.

Further useful information is provided by Table 4 in terms of relative gaps GAP_1 and GAP_2. In Equation (21), GAP_1 is calculated based on exact solutions of case-I to case-IV obtained by CPLEX. However, in Equation (22) lower bound is used for evaluating Case-V results instead of optimal cost, because optimal cost cannot be calculated. In these equations, $TotalCost_{BLDA}$, TC^*

and LB denote the total cost of all versions of the BLDA, optimal total cost and the lower bound, respectively. According to Table 4, the GAP_1 and GAP_2 for all BLDAs are less than 10%, which shows that the proposed approach can reach high quality solutions even for large-sized problems. Specifically, the gap values of CPX-SA and CPX-CPX which approves the appropriateness of BLDAs.

Table 3. Computational results of costs and elapsed time based on loose and tight due dates in real case study problems

Measure	Algorithm	Tight due date (TD)					Loose due date (LD)				
		Case-I	Case-II	Case-III	Case-IV	Case-V	Case-I	Case-II	Case-III	Case-IV	Case-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

$$GAP_1 = \frac{TotalCost_{BLDA} - TC^*}{TC^*} \quad (21)$$

$$GAP_2 = \frac{TotalCost_{BLDA} - LB}{LB} \quad (22)$$

Table 4. Gaps of 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%

In spite of five problems extracted from real zinc SC in Table 3 in order to make strong statistical inferences thirty two instances are generated randomly shown in Table 5.

Table 5. Computational results of total costs and time based on loose and tight due dates in generated test instances

Loose due date (LD)											
Instance	CPX-CPX	Time	CPX-SA	Time	SA-SA	Time	SA-CPX	Time	CPLEX	Time
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	CPX-SA	Time	SA-SA	Time	SA-CPX	Time	CPLEX	Time
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		

The statistical analyses, performed using MINITAB 17, are shown in Table 6. Except for SA-SA, the results of all other versions of the BLDA 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 BLDA. Also, it shows that SA-CPX, CPX-SA, and CPX-CPX find optimal solution in different instances. Moreover, as shown in Table 6 the mean total cost of CPX-SA and CPX-CPX are 7,551,746 and 7,603,818 respectively. This shows the mean total cost of CPX-SA is lower than CPX-CPX that the application of SA in the lower level provides very high quality solutions rather than using CPLEX. That is why $SRM_{CPX\text{-}CPX}$ and $STA_{CPX\text{-}CPX}$ are the largest.

Although SA-SA has been categorized separately, one can find lower total costs by applying a tuning scheme such as Taguchi on SA's parameters in both levels. Also, the application of an integrated SA instead of the bi-level SA (SA-SA) would be more beneficial suggested for future research activities.

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

Computational results of implementing all versions of the BLDA dealing with the *TD* and *LD* case instances are illustrated in Figures 13 and 14, respectively. According to these figures, the BLDA finds solution with minimum total cost before the max-iteration limit. However, in Figures 13-(*a*), 13-(*f*), 13-(*i*), 13-(*j*), 13-(*m*), 13-(*n*), where BLDA uses SA in its upper level, the algorithm does not converge normally and must be terminated at the maximum iteration limit. This is the same for Figures 14-(*f*), 14-(*i*), 14-(*j*), 14-(*m*), and 14-(*n*). The main reason is that the SA's schedules are mostly local optimums which cause a significant diversity in BLDA's outputs. In this regard, the average coefficient of variation (CV) of SA-SA and SA-CPX are 85% and 31%, respectively while it is 1% for each of CPX-SA and CPX-CPX. Figures 13-(*p*) and 13-(*t*) also indicate that CPX-SA and CPX-CPX may have high diversity in their outputs. However, according to other cases, the BLDA generally reaches the minimum within half of the maximum iteration when solving the upper level via CPLEX, e.g. see Figures 13-(*k*), 13-(*l*), 14-(*g*), and 14-(*h*).

In Figures 13-(*g*), 13-(*t*), 14-(*k*), 14-(*l*), 14-(*s*), and 14-(*t*) the BLDA using CPLEX in its upper level cannot improve its initial solution. This may have two reasons: first, the algorithm may get stock in a local optimum, e.g. in Figure 13-(*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. It is because the total of tardiness and holding costs becomes much higher than batching costs. Therefore, no reduction will occur in the total cost via batching. For example, in Figures 14-(*s*) and 14-(t), it seems that the optimum solution is found in the initial iteration since the total cost is very close to the lower bound (less than 1% gap).

In a specific analysis, this paper defines two performance measures on the BLDAs: the speed of reaching the minimum total cost (*SRM*) and the speed of terminating the algorithm (*STA*). Figure 15 depicts cumulative frequencies of the algorithms reached to their minimums versus iteration's index. For example, at half of the maximum iteration, see the area surrounded by dashed line on the 5th iteration, CPX-CPX has found the minimum of all of the *TD* and *LD* cases, where the cumulative frequency is 100%. It is, however, 90% for CPX-SA, 80% for SA-SA and 60% for SA-CPX, which equivalently means *SRM* of CPX-CPX is higher than other algorithms. The algorithms are compared in terms of *SRM* as follows.

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

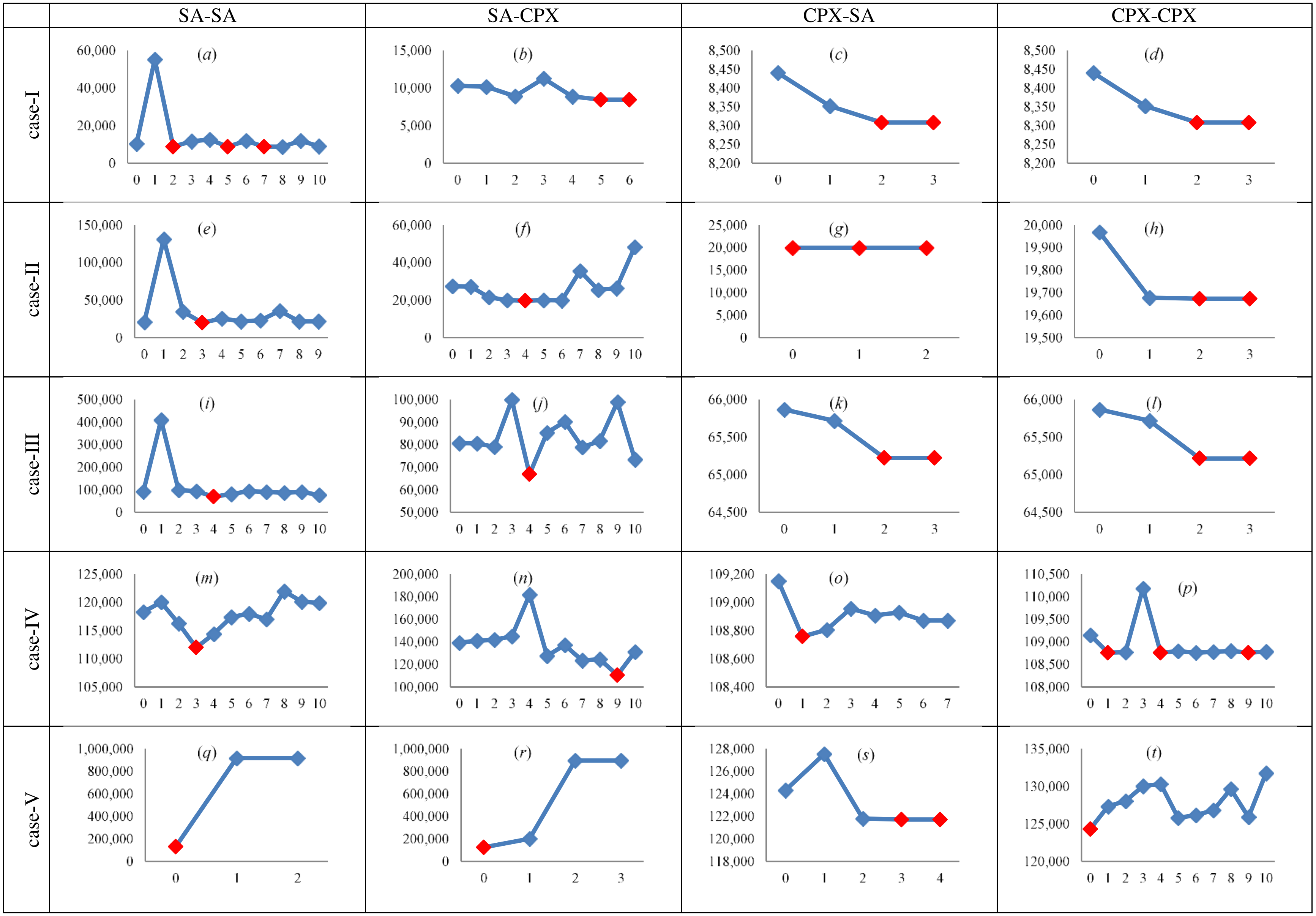

Figure 13. Total costs results in terms of iteration on tight due dates. The vertical axes show the objective value and the horizontal axes illustrate the number of algorithm's iterations ranging from 1 to 10 (the max-iteration), where 0 denotes the initial starting iteration. The red markers show minimum total costs.

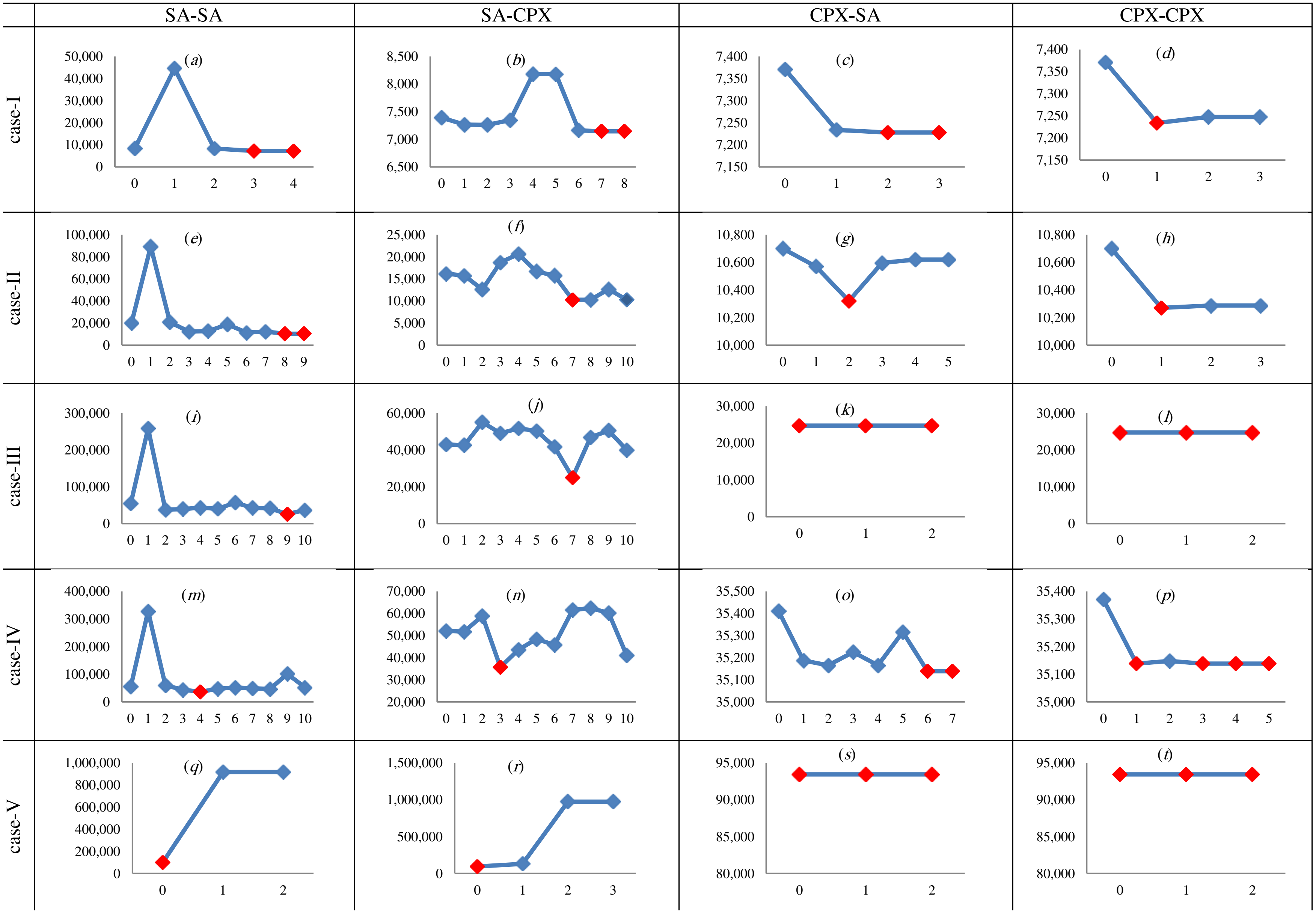

Figure 14. Total costs results in terms of iteration on loose due dates. The vertical axes show the objective value and the horizontal axes illustrate the number of algorithm's iterations ranging from 1 to 10 (the max-iteration), where 0 denotes the initial starting iteration. The red markers show minimum total costs.

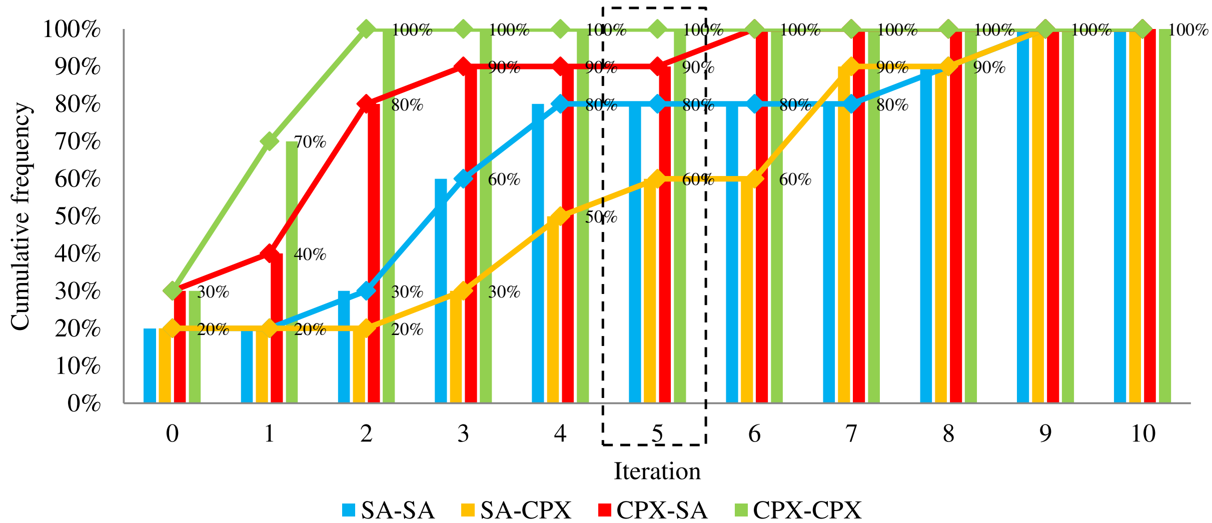

Figure 15. *SRM* cumulative frequencies

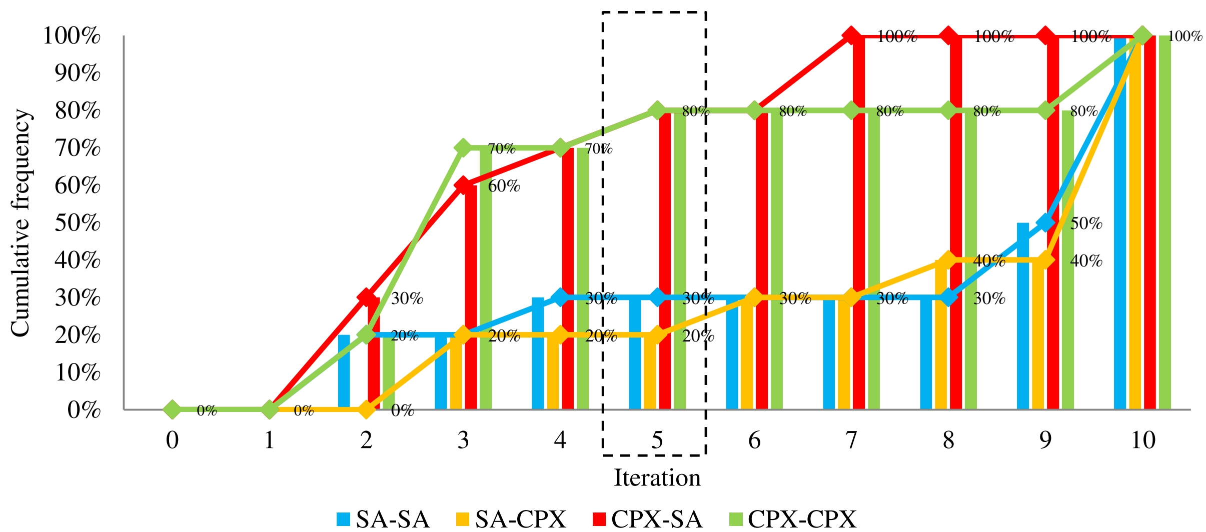

Figure 16. *STA* cumulative frequencies

Figure 16 illustrates *STA* cumulative frequencies of the algorithms. At the 5th iteration, CPX-CPX and CPX-SA have terminated in 80% of the cases, where this percent 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 tradeoff has to be made between the size of 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 time shorter than other algorithm.

A sensitivity analysis is performed to see the effect of due date parameters on costs of the network. So, the optimal cost values of CPLEX are studied comparing *TD*s vs. *LD*s. Since CPLEX could not solve case-V, the results of CPX-CPX are reported for case-V. Details of total costs are depicted in Figure 17 showing that *LD*s' total costs are 43% lower than *TD*s', on average. Because of increasing due dates, the completion of jobs can be postponed for batching purposes without increase in tardiness and holding costs. Also, tardiness costs will decrease by extending due date parameters. Figure 18 illustrates the scheduling costs, i.e. tardiness and holding costs, where costs of *TD*s are 30% higher than costs of *LD*s. Unlike scheduling costs, the batching costs are not significantly affected by altering due date parameters, see in Figure 19. The average batching costs of *LD*s are 1% lower than *TD*s. This means that the batching solution is not very sensitive to due date parameters compared to the scheduling solution.

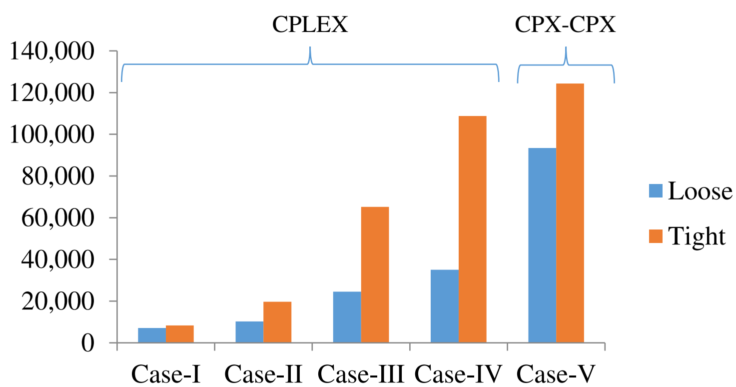

Figure 17. Total costs analysis: *TD* vs *LD*

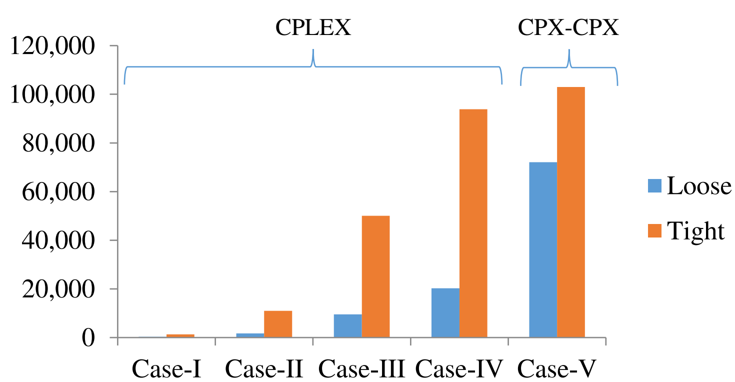

Figure 18. Scheduling costs (tardiness costs and holding costs) analysis: *TD* vs *LD*

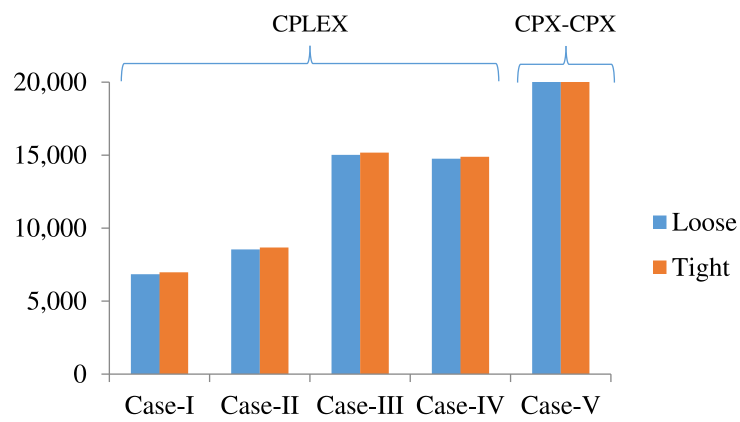

Figure 19. Batching costs analysis: *TD* vs *LD*

6. Conclusion

This paper dealt with the multi-factory scheduling with batch delivery problem (MFSBDP). A new mixed-integer programming model was proposed to provide optimal solution of the MFSBDP. The problem was divided into two sub-problems: scheduling problem and batching problem. Then, a novel bi-level decomposition algorithm (BLDA) was developed to deal with the MFSBDP. In the developed BLDA, the scheduling solution from the upper level was utilized to find an appropriate batching solution in the lower level. Both the upper and lower levels interactively collaborated leading the algorithm to converge to a solution or reach the maximum iteration limit. Based on the solution approach utilized in each level, i.e. CPLEX or simulated annealing, four versions of the BLDA were developed. According to the comparison made between the results of BLDAs, exact minimum, and lower bound values, it was statistically shown that the solution approach was able to deal with large-sized real-world case instances providing high quality solutions in a reasonably short time. Finally, a sensitivity analysis was performed by altering due date parameters on the case instances, which led to some useful managerial insights.

There are some considerable achievements obtained in this study in the view point of SCs' managerial boards, particularly for zinc SC networks. The supply chain manager(s) may employ the proposed algorithm to obtain applicable high quality solutions rather than traditional methods. They would be more interested in new aspects of transportation costs especially the bathing costs throughout the SC network. This paper reveals that how batching decisions may impose extra costs onto the network. More in detail, the simultaneous consideration of batching and scheduling decisions, proposed by this paper, would be worth to study. For future research, this study can be followed under uncertain parameters in other real-life SC networks.

Compliance with Ethical Standards

Funding: There was no funding for this study.

Conflict of Interest: The authors declare that they have no conflict of interest.

Ethical approval: This article does not contain any studies with human participants or animals performed by any of the authors.

Informed consent: With respect to the explanations given above, response to this issue is not applicable.

References

Agnetis, A., et al. (2014). "Coordination of production and interstage batch delivery with outsourced distribution." European journal of operational research **238**(1): 130-142.

Behnamian, J. and S. F. Ghomi (2013). "The heterogeneous multi-factory production network scheduling with adaptive communication policy and parallel machine." Information Sciences **219**: 181-196.

Behnamian, J. and S. F. Ghomi (2016). "A survey of multi-factory scheduling." Journal of Intelligent Manufacturing **27**(1): 231-249.

Chang, Y.-C., et al. (2014). "An ant colony optimization heuristic for an integrated production and distribution scheduling problem." Engineering Optimization **46**(4): 503-520.

Cheng, B.-Y., et al. (2015). "Integrated scheduling of production and distribution to minimize total cost using an improved ant colony optimization method." Computers & Industrial Engineering **83**: 217-225.

Chung, S., et al. (2010). "Application of genetic approach for advanced planning in multi-factory environment." International Journal of Production Economics **127**(2): 300-308.

Chung, S. H., et al. (2011). "Minimization of order tardiness through collaboration strategy in multifactory production system." IEEE Systems Journal **5**(1): 40-49.

Frazzon, E. M., et al. (2018). "Hybrid approach for the integrated scheduling of production and transport processes along supply chains." **56**(5): 2019-2035.

Ganji, M., et al. (2020). "A green multi-objective integrated scheduling of production and distribution with heterogeneous fleet vehicle routing and time windows." Journal of Cleaner Production: 120824.

Gharaei, A. and F. Jolai (2018). "A multi-agent approach to the integrated production scheduling and distribution problem in multi-factory supply chain." Applied Soft Computing **65**: 577-589.

Govindan, K., et al. (2017). "Supply chain network design under uncertainty: A comprehensive review and future research directions." European journal of operational research **263**(1): 108-141.

Guinet, A. (2001). "Multi-site planning: A transshipment problem." International Journal of Production Economics **74**(1): 21-32.

Guo, Z., et al. (2016). "A bi-level evolutionary optimization approach for integrated production and transportation scheduling." Applied Soft Computing **42**: 215-228.

H'Mida, F. and P. Lopez (2013). "Multi-site scheduling under production and transportation constraints." International Journal of Computer Integrated Manufacturing **26**(3): 252-266.

Jia, H., et al. (2007). "Integration of genetic algorithm and Gantt chart for job shop scheduling in distributed manufacturing systems." Computers & Industrial Engineering **53**(2): 313-320.

Jia, H., et al. (2003). "A modified genetic algorithm for distributed scheduling problems." Journal of Intelligent Manufacturing **14**(3): 351-362.

Kaminsky, P. and O. Kaya (2008). "Inventory positioning, scheduling and lead-time quotation in supply chains." International Journal of Production Economics **114**(1): 276-293.

Karimi, N. and H. Davoudpour (2015). "A branch and bound method for solving multi-factory supply chain scheduling with batch delivery." Expert Systems with Applications **42**(1): 238-245.

Karimi, N. and H. Davoudpour (2016). "Integrated production and delivery scheduling for multi-factory supply chain with stage-dependent inventory holding cost." Computational and Applied Mathematics **36**(4): 1-16.

Marandi, F. and S. Fatemi Ghomi (2018). "Integrated multi-factory production and distribution scheduling applying vehicle routing approach." International Journal of Production Research: 1-27.

Matusiak, M., et al. (2014). "A fast simulated annealing method for batching precedence-constrained customer orders in a warehouse." European journal of operational research **236**(3): 968-977.

Nikzad, F., et al. (2015). "Scheduling of multi-component products in a two-stage flexible flow shop." Applied Soft Computing **32**: 132-143.

Nishi, T., et al. (2011). "A bilevel decomposition algorithm for simultaneous production scheduling and conflict-free routing for automated guided vehicles." Computers & Operations Research **38**(5): 876-888.

Noroozi, A., et al. (2017). "Evolutionary computation algorithms to coordinating order acceptance and batch delivery for an integrated supply chain scheduling." Computational and Applied Mathematics: 1-51.

Peterson, B., et al. (2014). "Scheduling multiple factory cranes on a common track." Computers & Operations Research **48**: 102-112.

Potts, C. N. and M. Y. Kovalyov (2000). "Scheduling with batching: A review." European journal of operational research **120**(2): 228-249.

Qing-dao-er-ji, R. and Y. Wang (2012). "A new hybrid genetic algorithm for job shop scheduling problem." Computers & Operations Research **39**(10): 2291-2299.

Ruifeng, C. and V. Subramaniam (2011). "Performance evaluation for tandem multi-factory supply chains: an approximate solution." International Journal of Production Research **49**(11): 3285-3305.

Santos, A., et al. (2015). Scheduling and batching in multi-site flexible flow shop environments. Information Systems and Technologies (CISTI), 2015 10th Iberian Conference on, IEEE.

Santos, C. and M. Magazine (1985). "Batching in single operation manufacturing systems." Operations Research Letters **4**(3): 99-103.

Sauer, J. and H.-J. Appelrath (2000). Integrating transportation in a multi-site scheduling environment. System Sciences, 2000. Proceedings of the 33rd Annual Hawaii International Conference on, IEEE.

Sun, X., et al. (2015). "Integrated scheduling of a multi-product multi-factory manufacturing system with maritime transport limits." Transportation Research Part E: Logistics and Transportation Review **79**: 110-127.

Torabzadeh, E. and M. Zandieh (2010). "Cloud theory-based simulated annealing approach for scheduling in the two-stage assembly flowshop." Advances in Engineering Software **41**(10): 1238-1243.

Wang, K., et al. (2016). "Coordinated scheduling of production and transportation in a two-stage assembly flowshop." International Journal of Production Research **54**(22): 6891-6911.

Yin, Y., et al. (2020). Integrated Production, Inventory, and Batch Delivery Scheduling with Due Date Assignment. Due Date-Related Scheduling with Two Agents, Springer**:** 93-141.

Zhou, H., et al. (2018). "A kriging metamodel-assisted robust optimization method based on a reverse model." Engineering Optimization **50**(2): 253-272.