Hindawi Publishing Corporation
ISRN Industrial Engineering
Volume 2014, Article ID 163637, 12 pages
http://dx.doi.org/10.1155/2014/163637

Research Article

Production Scheduling and Customer Orders Assignment in a Three-Level Supply Chain

F. Sadeghi Naieni Fard,[1] B. Naderi,[2] and A. A. Akbari[1]

[1] *South Branch of Islamic Azad University, School of Industrial Engineering, Kafaee Amani Street, Gerami Street, Enghelab Avenue, Tehran 1151863411, Iran*
[2] *Kharazmi University, Daneshgah Square, Shahid Beheshti Avenue, Karaj 3197937551, Iran*

Correspondence should be addressed to B. Naderi; bahman_naderi62@yahoo.com

Received 27 September 2013; Accepted 23 November 2013; Published 28 January 2014

Academic Editors: T. F. Espino-Rodriguez and C.-C. Wu

In the classical production-distribution centers problem, only assignment of customers, distribution centers, and suppliers is determined. This paper extends the problem of production-distribution centers assignment by considering sequencing decisions in the supply network. Nowadays, meeting delivery time of products is a competitive benefit; therefore, the objective is to minimize total tardiness. This problem is mathematically formulated by a mixed integer programming model. Then, using the proposed model, small instances of the problem can be optimally solved by GAMS software. Moreover, two metaheuristics based on variable neighborhood search and simulated annealing are proposed to solve large instances of the problem. Finally, performance of the proposed metaheuristics is evaluated by two sets of balanced and unbalanced instances. The computational results show the superiority of the variable neighborhood search algorithm.

1. Introduction

In the supply chain, various organizations work together for converting raw materials to finished products and are connected with each other over far distances. The customer orders due date and the short life-cycle products need agile supply networks. To accelerate supply chain operations many activities such as keeping storage which are nonvalue added activities must be eliminated. It also should be stated that customer demands are generating the flow of supply network and they have to be specified or predicted in advance, but optimization of this flow through the network for minimizing the costs is an important purpose of the supply chain and, for optimizing, speed of the sources and capacity limitations are considered. Scheduling for order shipment to the customer destinations and using transportation means, in order to pick up the orders at the distribution centers (DCs) were done just by staff experiences. This will increase not only the waiting time of transportation means, but also accumulation of the orders and inventory costs. Ultimately delivery time of the orders will exceed the due date and cause penalty costs and dissatisfaction of the customers.

The model which is proposed in this paper presents a good solution for the above situation. It offers a good method for calculating the time of production or packing in the plants and distribution centers and the transportation between them to decrease the delivery time of the orders and define an optimal assignment and sequence for the customers according to the capacity of distribution centers and customer due dates. As regards the importance of scheduling and sequencing in the network of deteriorating jobs, it can be stated that the proposed model can be applied in different industries, such as food and drug supply chains and any products that little procrastination in delivery could cause large penalty and damage in their network.

Some researchers dealt with the integration of production and transportation in supply chain for scheduling purposes. Meanwhile we can mention Zegordi and Beheshti Nia [1] works who presented a model for the integration of production and transportation scheduling with the purpose of

minimizing total tardiness and total deviations of assigned workload of suppliers in which different sites were assumed for the sources. Later, they [2] proposed a model which is like the one that is stated above but with a single site for the suppliers and an integer programming model is formulated to minimize the makespan. Li et al. [3, 4] studied the similar subject by considering the air transportations and integrated with the assembly in the electronic supply chain. In the proposed model, optimal allocation of orders to different flights and release time for each order to complete the assembly are determined. The purpose of the two subproblems is minimizing the total costs including transportation cost of normal flight and special flight and earliness and tardiness penalty. Then, Zandieh and Molla-Alizadeh-Zavardehi [5] developed their research and studied the problem on two different situations, one of them with presence of commercial flights and the other one with absence of them. In the first situation the delivery tardiness has to be considered, whereas in the second one it is not necessary.

Prior to the papers mentioned above, Naso et al. [6] studied this integration by applying it for the distribution of ready-mixed concrete. Chen and Lee [7] considered the concept of scheduling in supply chain by assuming different importance for batching jobs and the goal was minimization of the weighted delivery time and total transportation costs.

In the online environment of supply chain scheduling, Averbakh and Xue [8] discussed a new class of problems that concerns with supply chain scheduling in the situation where the future is unknown and there is no information on the number of jobs. The purpose of the problem is minimizing sum of the total flow time of the jobs and the total delivery costs. Also, some researchers have studied the minimization of setup costs in the supply chain such as the model which is presented by Mansouri [9]. In this paper, the focus is on the sequencing of batches between two successive stages of a supply chain. Later, Liao et al. [10] followed his research and extended it by considering the problem in three stages and proposed a heuristic called LFFH (least flexibility first heuristic) for solving large sized problems.

According to the above review of research, it is obvious that many studies are done for scheduling and assignment problems in supply chain, but considering both of them in a model with the fulfillment of customer demands was paid less attention. Here in this paper we present a model which can bring all of these conditions together.

The remainder of this paper is organized as follows. In Section 2 the definition and formulation of the problem can be seen. The suggested algorithms and methods of generating new solutions are described in Section 3. Section 4 indicates the computational results and performance analysis and ultimately in Section 5 we conclude the study and give suggestions for future works.

2. Problem Definition and Formulation

The supply chain structure that this paper is concentrated on is shown in Figure 1. As it can be observed, it is a three-level structure including plants (resources), distribution centers, and customers (retailers). At the customer level, there are several batches of orders that part of the batches as a new set of orders or the total sets are processed in plants and transported to the distribution centers. After accumulation of the orders of customers, they are packed and delivered to the destinations (customers). The model attempts to minimize the completion time of each stage for the batch of orders by considering all of the supply chain restrictions and doing optimal assignment for satisfying the customer demands, in order to minimize the total tardiness.

There are some assumptions considered in this problem. First each of the plants and distribution centers has a specified order processing time. Plants, distribution centers, and customers are located in different sites. For the customers, their demands and due dates of their orders are available and their demands have to be fulfilled by only one of the distribution centers. Because we assumed that each of the customer orders as a batch has special attributes, so it must not synchronize with other customer orders for processing and packing. For presenting the model we use the following parameters and decision variables.

Parameters

j: Index of customer ($j = 1, \ldots, n$).

t: Index of plant ($t = 1, \ldots, m$).

l: Index of distribution center (DC) ($l = 1, \ldots, g$).

d_j: The demand of customer j.

p_t: The processing time of plant t.

p_l: The packing and loading time of distribution center l.

cap_l: The capacity of distribution center l for number of products.

$\text{time}_{t,l}$: The transportation time between plant t and distribution center l.

$\text{time}_{l,j}$: The transportation time between distribution center l and customer j.

due_j: The due date for the orders of customer j.

Variables

tar_j: The continuous variable for tardiness of customer j orders.

$x_{t,l,j}$: The integer variable for number of orders of customer j that is manufactured by plant t and delivered by distribution center l.

$z_{t,j,b}$: Binary variable equal to 1, if customer j orders are processed before the orders of customer b in plant t, and zero otherwise.

$y_{l,j,b}$: Binary variable equal to 1, if customer j orders are packed and loaded before the orders of customer b in DC l, and zero otherwise.

$w_{t,l,j}$: Binary variable taking value 1 if any orders of customer j is manufactured by plant t and delivered by distribution center l, and zero otherwise.

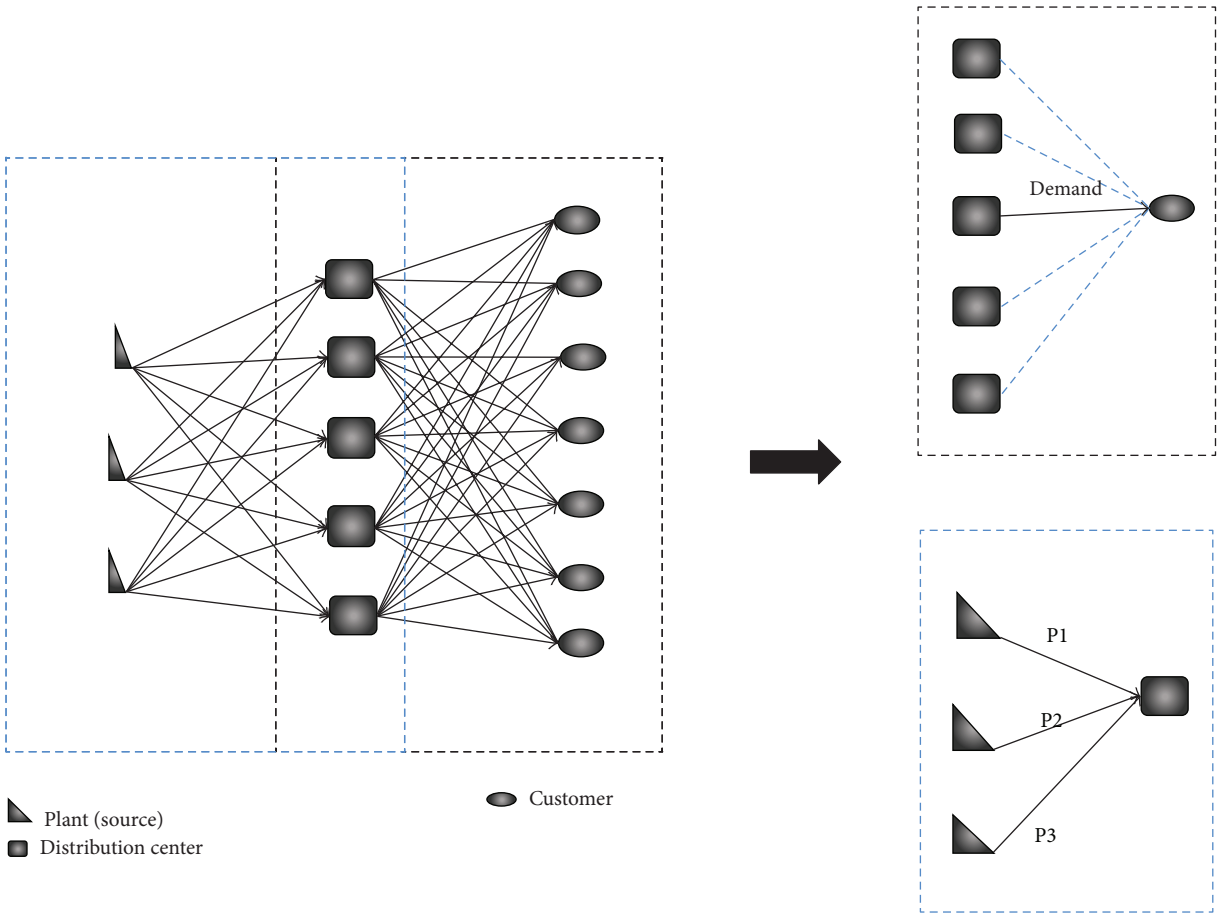

Figure 1: Supply chain structure which is studied in this paper.

$c_{t,j}^{1}$: Continuous variable for the completion time in the plants stage.

$c_{l,j}^{2}$: Continuous variable for the completion time in the second stage transported from plants to distribution centers.

$c_{l,j}^{3}$: Continuous variable for the third stage completion time in distribution centers.

$q_{l,j}$: A binary variable equal to 1 when orders of customer j are fulfilled by DC l.

c_{j}^{4}: Continuous variable for the last stage completion time when orders are delivered to the customers.

The MILP model is as follows.

$$\text{Minimize} \quad \sum_{j=1}^{n} \text{tard}_j \tag{1}$$

subject to

$$\sum_{t=1}^{m}\sum_{l=1}^{j} x_{t,l,j} \geq d_j \quad \forall_j \tag{2}$$

$$c_{t,j}^{1} \geq \sum_{l=1}^{g} x_{t,l,j} \times p_t \quad \forall_{j,t} \tag{3}$$

$$c_{t,j}^{1} \geq c_{t,b}^{1} + \sum_{l=1}^{g} x_{t,l,j} \times p_t - M\left(1 - Z_{t,j,b}\right) \quad \forall_{j,t,b|b<j} \tag{4}$$

$$c_{t,b}^{1} \geq c_{t,j}^{1} + \sum_{l=1}^{g} x_{t,l,b} \times p_t - M\left(Z_{t,j,b}\right) \quad \forall_{j,t,b|b<j} \tag{5}$$

$$Z_{t,j,b} = 0 \quad \forall_{j,t,b|b\geq j} \tag{6}$$

$$c_{l,j}^{2} \geq c_{t,j}^{1} + \text{time}_{t,l} - M\left(1 - w_{t,l,j}\right) \quad \forall_{j,l,t} \tag{7}$$

$$c_{l,j}^{3} \geq c_{l,j}^{2} + \sum_{t=1}^{m} x_{t,l,j} \times p_l \quad \forall_{j,l} \tag{8}$$

$$c_{l,j}^{3} \geq c_{l,b}^{3} + \sum_{t=1}^{m} x_{t,l,j} \times p_l - M\left(1 - y_{l,j,b}\right) \quad \forall_{j,l,b|b<j} \tag{9}$$

$$c_{l,b}^{3} \geq c_{l,j}^{3} + \sum_{t=1}^{m} x_{t,l,b} \times p_l - M\left(y_{l,j,b}\right) \quad \forall_{j,l,b|b<j} \tag{10}$$

$$y_{l,j,b} = 0 \quad \forall_{j,l,b|b\geq j} \tag{11}$$

$$\sum_{t=1}^{m}\sum_{j=1}^{n} x_{t,l,j} \leq \text{cap}_l \quad \forall_l \tag{12}$$

$$c_j^{4} \geq c_{l,j}^{3} + \text{time}_{l,j} - M\left(1 - w_{t,l,j}\right) \quad \forall_{j,l,t} \tag{13}$$

$$x_{t,l,j} \leq M w_{t,l,j} \quad \forall_{j,l,t} \tag{14}$$

$$w_{t,l,j} \leq x_{t,l,j} \quad \forall_{j,l,t} \tag{15}$$

$$w_{t,l,j} \leq q_{l,j} \quad \forall_{j,l,t} \tag{16}$$

$$\sum_{l=1}^{g} q_{l,j} = 1 \quad \forall_j \tag{17}$$

$$\text{tar}_j \geq c_j^{4} - \text{due}_j \quad \forall_j \tag{18}$$

$$c_{t,j}^{1}, c_{l,j}^{2}, c_{l,j}^{3}, c_j^{4}, \text{tar}_j, x_{t,l,j} > 0 \quad \forall_{j,l,t} \tag{19}$$

$$z_{t,j,b}, y_{l,j,b}, w_{t,l,j}, q_{l,j} \in \{0,1\} \quad \forall_{j,l,t,b}. \tag{20}$$

Constraint set (2) ensures the fulfillment of the customer demands with processed orders. The first stage completion time in plants is determined by constraint set (3). Constraint set (4) and constraint set (5) guarantee that orders of two customers could not be processed simultaneously as a batch and constraint set (6) excludes redundant variables from the model. The second stage completion time is presented by constraint set (7), in which the transportation time between plants and DCs is added to the first stage makespan. The completion time of the third stage consists of packing and loading the orders is denoted by constraint set (8). Similar to the assumption in the plant stage, orders of different customers could not be packed or loaded at the same time, and therefore constraints sets (9) and (10) present this assumption. Redundant variables are omitted by constraint set (11).

Constraint set (12) assures that number of orders which are delivered by a special distribution center must not exceed the capacity limitation. Constraint set (13) determines each customer orders makespan as a batch, which is the completion time of the last stage. Constraint sets (14) and (15) indicate that if variable $X_{t,l,j}$ takes value except zero, the assignment variable $W_{t,l,j}$ would take value 1 and if $W_{t,l,j}$ is equal to zero the assignment orders variable $X_{t,l,j}$ should be zero too. It means that if the orders of customer j are processed by plant t and delivered by DC l, the assignment of the orders of customer j to plant t and DC l has taken place. In the same way constraint set (16) shows that if any assignment from customer j to DC l has taken place then at least with one plant, $W_{t,l,j}$ has to take value 1. Constraint set (17) shows that each of the customers could be served only by one distribution center.

Table 1: Example parameters related to the plants.

Plants	Time distance to DCs			Processing time
	1	2	3	
1	3	4	2	0.13
2	7	1	5	0.14

Constraint set (18) determines tardiness value according to the customer orders makespan. Constraint set (19) or constraint set (20) indicates that each of the variables should take the specified range of values. Finally (1) is the objective function that shows that the model attempts to minimize the accumulation of customer orders tardiness.

For making the problem clear, it is illustrated with an example. Here an example of a supply chain with 2 manufacturers, 3 distribution centers, and 5 retailers is considered. Numbers of product units that each of the customers requires are 16, 47, 55, 26, and 20 for the first customer to the last one, respectively. Each customer has a specified due date for delivery of their demands that follows 10, 15, 9, 6, and 11 sequentially. As shown in Figure 2 each of the customers is served by a predetermined distribution center.

Some other parameters that are needed to be given for this example, like supplier and DCs processing time, time distances between the elements of the supply network, and capacity of the distribution centers for packing and loading of product units, are inserted in Tables 1 and 2. This example was solved by the GAMS software. Tables 3 and 4 show the assigned number of units and tardiness values. As it can be observed that the optimal sequence in the first plant is 5-3-4-2 and in the second one follows the order 3-4-1-2, and in the same way the optimal sequence of customers in the distribution centers can be determined. The Gant chart of the completion time of different stages for customer batches of order is shown in Figures 3, 4, and 5. The optimum objective value is 5.939 which was gained by the software.

3. Proposed Algorithms

Metaheuristic algorithms can provide a good solution for an optimization problem by searching over a large set of feasible solutions and they are applied for the problems with limited computation capacity and they can give a near optimal solution in a reasonable amount of time.

The model which is developed in this paper is an extended format of Zegordi's model which has two levels in the supply chain and he proved that his model belongs to NP-hard class problems; therefore the model proposed in this paper which consists of three levels in supply chain is NP-hard and here we suggested two metaheuristic algorithms for solving the large sized problems. Here in this section two metaheuristic algorithms known as variable neighborhood search and simulated annealing are selected according to the problem characteristics and we developed their features to generate the near optimal solutions.

Table 2: Example parameters related to the DCs.

DC	Time distance to customer					Packing and loading time	DCs capacity
	1	2	3	4	5		
1	4	3	2	3	7	0.038	100
2	1	2	5	6	7	0.037	90
3	6	1	3	3	5	0.014	80

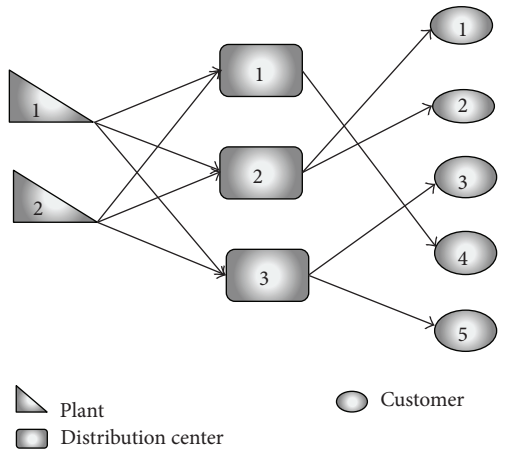

Figure 2: Supply chain structure of the example.

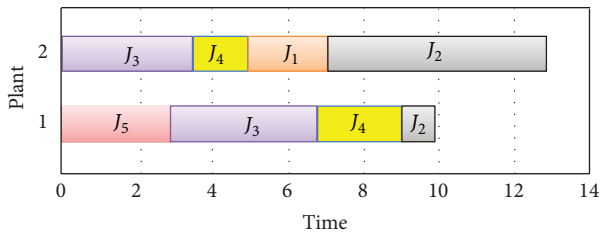

Figure 3: Completion time of the batches in the first stage.

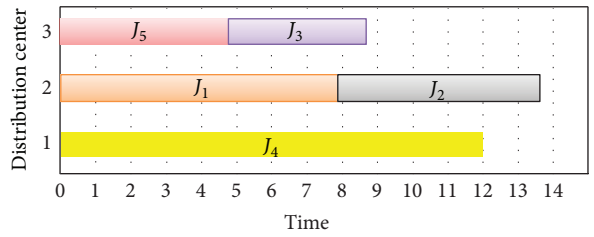

Figure 4: Completion time of the batches in the second stage.

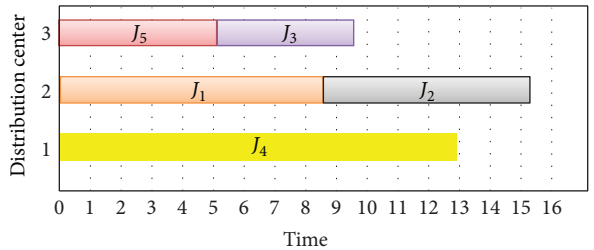

Figure 5: Completion time of the batches in the third stage.

Table 3: Assigned number of units.

Plants	DCs	Customers				
		1	2	3	4	5
1	1	0	0	0	17	0
	2	0	4	0	0	0
	3	0	0	27	0	20
2	1	0	0	0	9	0
	2	16	43	0	0	0
	3	0	0	28	0	0

Table 4: Last stage completion time and tardiness values.

Customer	1	2	3	4	5
Completion time of last stage	10	17.26	12.41	16	11
Tardiness	0	2.259	3.41	0	0

3.1. Variable Neighborhood Search Algorithm. The variable neighborhood search algorithm is a metaheuristic algorithm for solving optimization problems whose basic idea is systematic change of neighborhood solutions in two phases. Firstly, descent to find a local optimum, finally a perturbation phase to escape from the corresponding valley.

For the description of the process, first we should define a set of solution structures, $N_k\,(k = 1, 2, \ldots, k_{\max})$, and an initial solution should be generated randomly. In the kth neighborhood of initial solution, some feasible solutions are generated. The algorithm explores the solution space and tries to find the best solution. This exploration is done by two functions called shaking and local search. Shaking function verifies the solution structures and the local search explores the space of the solution. Then the best solution of the local search is compared with the initial solution and, if it is better, replaces it and searching in the kth neighborhood structure is continued until it becomes exhausted. Finally the whole process is terminated when the stopping condition is met. It should be stated that the stopping condition could be the maximum CPU time allowed or the maximum number of iterations. Also the order and choice of the structures are important for ultimate near optimal solutions. The pseudo code of the algorithm is shown in Algorithm 1 [11].

3.2. Simulated Annealing Approach. The basic idea of simulate annealing (SA) algorithm is originated from the process of cooling and freezing metals into minimum energy which is called the annealing process. This algorithm was first developed to highly nonlinear problems. The procedure of this algorithm can be described as follows. First an initial

solution has to be generated randomly, and then some new test solutions are produced through applying various methods. If the objective function of the test solution is better than the current solution, it is replaced by the new one. But if it is worse than the current solution it will be replaced in a probabilistic situation and the acceptance probability is calculated.

3.2.1. The Acceptance Probability. In this paper the Metropolis criterion is used for the acceptance probability. It is calculated as follows:

$$P(A) = \exp\left(\frac{-\Delta E}{K_b T}\right)$$
$$\Delta E = \frac{d}{f} \tag{21}$$
$$k_b = \frac{-\Delta E_1}{T_1 \ln P(A)_1}.$$

ΔE shows the distance between the worse solution and the current solution. d is the distance between the objective function of the test solution and the current solution and f represents the objective function of the current solution. k_b is introduced as the Boltzmann constant which reflects the probability of accepting the test solution at the initial temperature or T_1. After each of the iterations the temperature is decreased and the above process will be carried out. The process will be terminated after a number of iterations or the maximum CPU time allowed is met or when the minimum temperature is reached. The pseudo code, shown in Algorithm 2 is an overview of the algorithm [9].

3.3. Generating New Neighborhood Solutions. Since we assumed three levels for the supply chain concluding plants, distribution centers, and customers, the representation scheme is a matrix that has been designed in which rows are assigned to the plants and columns are assigned to the customers and distribution centers. In order to make it obvious, consider that there are two plants, two distribution centers, and three customers, and then the resulted matrix is like the one presented in Table 5.

As shown in Table 6, rows and columns are dedicated to plants and customers, respectively. The triple group of columns (customers) forms a distribution center. In this scheme each cell contains a number of orders processed by a special plant and assigned to a distinct distribution center that belonged to a specified customer. The last row presents sequence of customers in each distribution center and is specified by the system of ranking. It means that, for example, the customer who is first has a rank of one, and the second rank shows that it is the second customer whose orders have to be processed. For the orders of customers in each plant, a matrix structure is designed in which rows and columns are dedicated to the plants and customers, respectively, and the sequence of orders batch is determined by customer numbers.

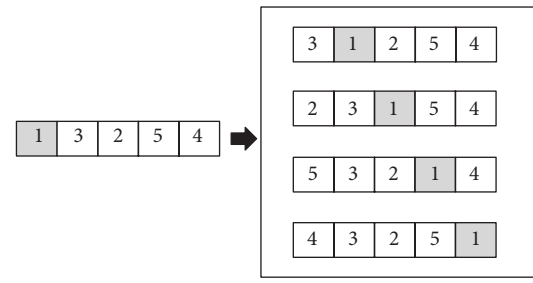

Figure 6: Example for 2-OPT heuristic.

Table 5: Matrix structure excluding customer sequences in plants.

	DC1 J_1	DC1 J_2	DC1 J_3	DC2 J_1	DC2 J_2	DC2 J_3
P1	d1 (J_1)	0	d1 (J_3)	0	d1 (J_2)	0
P2	d2 (J_1)	0	d2 (J_3)	0	d2 (J_2)	0
P3	d3 (J_1)	0	d3 (J_3)	0	d3 (J_2)	0
	1	2	3	1	3	2

Table 6: Customer sequences in plants.

P1	1	3	2
P2	2	3	1
P3	1	2	3

The ultimate structure is built by joining the matrix of customer sequences to the first structure which is shown in Table 7.

In this section various methods of generating new solutions are introduced and illustrated with some examples.

3.3.1. Verifying Assigned Distribution Centers. By applying this method, a customer is randomly selected and entered a loop including the range of distribution centers and assigned to each of them sequentially. In this loop the assignment is accepted after checking two conditions. First the capacity limitation of the distribution center and the second condition is improvement of the objective function. As a simple example for this verification, assume that there are six customers and four distribution centers. The assignment of distribution centers to the customers follows the sequence 2,1,2,3,1,4. Then if we want to verify the first customer assignment we can use this procedure and new assignments would be 1,1,2,3,1,4 or 3,1,2,3,1,4 or 4,1,2,3,1,4. Therefore we can have 4^6 new assignments for all of the customers.

3.3.2. Verifying Number of Orders Assigned to a Plant. The method that is presented in this section reduces the gap between optimal solution and the one which is generated by the algorithm in a large extent. The procedure is demonstrated and for more clarification, a set of steps for this method is presented in Algorithm 3.

For this method initially a distribution center is selected randomly. Then a random number in the range between 0 and 1 is generated and named as $a1$. Here, there are two loops:

Initial sequence of assignment

2	3	1	2	4

Customer	Supplier	Selected position	New sequence of assignment
1	2	3	3 1 **2** 2 4
2	3	1 4	**3** 2 1 2 4 2 1 2 **3** 4
3	1	2 4	2 **1** 3 2 4 2 3 2 **1** 4
4	2	1 5	**2** 2 3 1 4 2 3 1 4 **2**
5	4	2	2 **4** 3 1 2

Figure 7: Customer assignment perturbation examples.

Table 7: The whole representation scheme.

	DC1 J_1	DC1 J_2	DC1 J_3	DC2 J_1	DC2 J_2	DC2 J_3			
P1	d1 (J_1)	0	d1 (J_3)	0	d1 (J_2)	0	1	3	2
P2	d2 (J_1)	0	d2 (J_3)	0	d2 (J_2)	0	2	3	1
P3	d3 (J_1)	0	d3 (J_3)	0	d3 (J_2)	0	1	2	3
	1	2	3	1	3	2	0	0	0

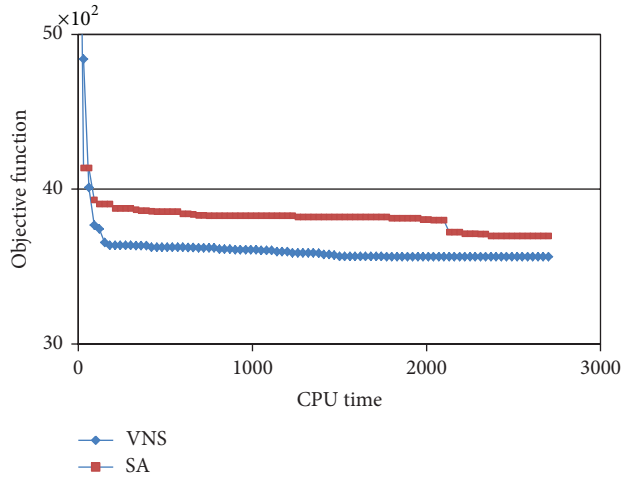

Figure 8: CPU time usage and convergence diagram.

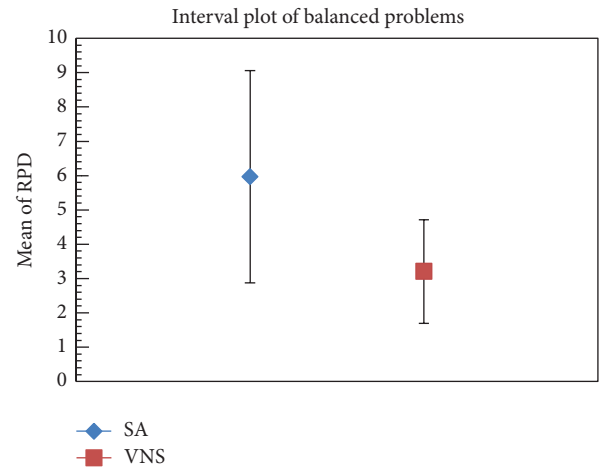

Figure 9: The average RPD intervals for balanced problems.

the outer loop includes customers and the inner loop consists of plants. In the inner loop the number of orders which is assigned to a plant multiplying to $a1$ and the result subtracted from the mentioned plant and added to the dedicated orders of other plants for the customer that is already mentioned. If the objective function is improved the process continues for other plants, but if it does not reflect any improvement, the half of $a1$ is calculated and it is replaced with the resulting

Table 8: Range of selection for test problems.

Parameter	Type of the selection set	Assumed interval
Number of plants	Discrete set	{2, 4}
Number of distribution centers	Discrete set	{4, 6}
Time distances: plants to DCs – DCs to customers	Balanced uniform distribution	$u(1–8) – u(1–8)$
	Unbalanced uniform distribution	$u(1–8) – u(8–16)$
Customer demands	Uniform distribution	$u(10–50)$
Processing time of the plants	Uniform distribution	$u(1\text{-}2)$
Packing and loading time of the distribution centers	Uniform distribution	$u(1–1.5)$

Table 9: Design of experiment results and the efficient set.

Parameter	Description	Proposed levels	The most efficient value
ΔE_1	Maximum distance, new solution can get in the temperature t_1	{0.3, 0.2, 0.1}	0.1
$P(A)_1$	The acceptance probability in temperature t_1	{0.6, 0.3, 0.1}	0.1
CR	Changing rate of the temperature in each iteration	{0.99, 0.78, 0.6}	0.99
T_1	Primary temperature	{40, 30, 10}	40

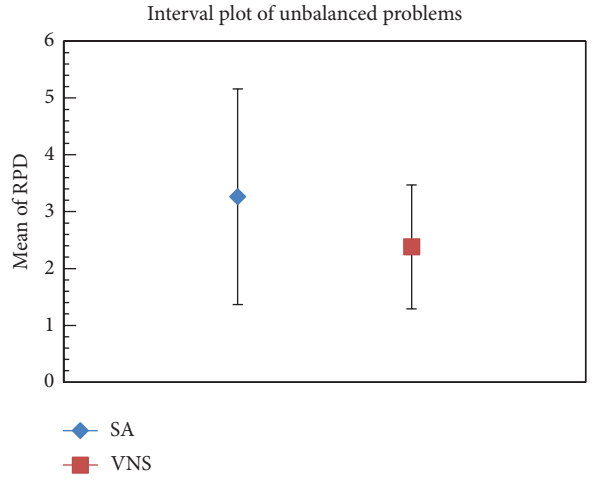

Figure 10: The average RPD intervals for unbalanced problems.

Table 10: Performance analysis of large sized balanced problems.

Balanced problems				
m	g	n	Mean of RPD	
			SA	VNS
2	4	40	2.7	0
2	4	60	1.73	6.2
2	4	80	9.02	0.93
2	6	40	0.47	4.07
2	6	60	1.52	5.53
2	6	80	2.9	6
4	4	40	4.78	5.72
4	4	60	16.96	0
4	4	80	2.29	0
4	6	40	21.2	0
4	6	60	0	9.78
4	6	80	8.051	0.188
Total mean of RPD			5.97	3.202

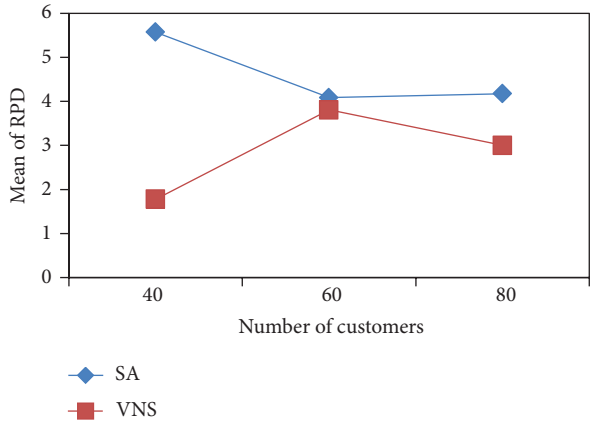

Figure 11: The average RPD of the tested algorithms versus the number of customers.

amount. The procedure is carried out with the same plant, and again if any improvement has not been observed the operation should be continued since $a1$ takes value zero.

3.3.3. 2-OPT Heuristic to Achieve Optimal Sequence. In this section, we use a heuristic method called 2-OPT for creating optimal sequence of customer orders (customer batches) in plants or distribution centers; some researchers like Tavakkoli-Moghaddam et al. [12] applied this method, when using simulated annealing approach for optimizing a vehicle routing problem. 2-OPT algorithm was first used to solve the traveling salesman problem. It is an improvement algorithm which systematically modifies the starting solution and evaluates the resulting modified solution. If it is better the modification is made permanent; if not, the systematic modification is continued until it is no longer possible to

Procedure: Variable Neighborhood Search Algorithm
Initialization: select a set of neighborhood structures $(N_k, k = 1, 2, \ldots, k_{\max})$
Find an initial solution (x)
While stopping criterion is not met (maximum allowed CPU time)
  set $k = 1$
  If $(k \leq k_{\max})$
    y = generate a new solution from the kth neighborhood of x
    If tardiness$(y) \leq$ tardiness(x)
      $x = y$
    End if
    continue the search with $N_1(k = 1)$
    Else
    $k = k + 1$
  End if
End while
Report x as the best solution

Algorithm 1: The sod code of variable neighborhood search algorithm.

Procedure: Simulated annealing Algorithm
Generate a random initial solution (x)
While the stopping criterion is not met (the maximum allowed CPU time)
  Generate a test solution from a current solution via mutation (y)
  If Tardiness$(y) \leq$ Tardiness(x)
    $x = y$ (replace current solution with test solution)
  Else check metropolis Criterion
    Rand = Generate a random number between 0 and 1
    If Rand < P(A)
      $x = y$ (replace current solution with test solution)
    End if
  End if
End while
Report x as the best solution

Algorithm 2: The sod code of simulated annealing algorithm.

Step 1. set x as the current solution and go to *Step 2*.
Step 2. generate a random number between 1 and number of DCs and put it in rand(1) and go to *Step 3*.
Step 3. set j as the customer number and put $j = 1$ and go to *Step 4*.
Step 4. if ($j <$ number of customers), go to *Step 5*; else go to *Step 2*.
Step 5. set i as plant number and put $i = 1$ and set $p(i)$ as the number of product of plant i that is part of customer j demands which is assigned to $\mathrm{DC}_{\mathrm{rand}(1)}$.
Step 6. set k as plant number and put $k = 1$ and set $p(k)$ as the number of product of plant k that is part of customer j demands which is assigned to $\mathrm{DC}_{\mathrm{rand}(1)}$.
Step 7. if $(i = k)$, put $k = k + 1$ and go to *Step 8*; else go to *Step 8*.
Step 8. generate a random number between 0 and 1 and put it in rand(2) and go to *Step 9*.
Step 9. set $a_1 = p(i) \times \mathrm{rand}(2)$ and go to *Step 10*.
Step 10. replace $p(i)$ with $p(i) - a_1$ and $p(k)$ with $p(k) + a_1$ and set the test solution as a new solution y and go to *Step 11*.
Step 11. set $f(x)$ and $f(y)$ as the objective value of solution x and solution y respectively and go to *Step 12*.
Step 12. if $(f(x) \geq f(y))$ set $x = y$ and put $j = j + 1$ and go to *Step 4*; else go to *Step 13*.
Step 13. if ($k <$ number of plants), put $k = k + 1$ and go to *Step 7*; else go to *Step 14*.
Step 14. if ($i <$ number of plants), put $i = i + 1$ and $k = 1$ and go to *Step 9*; else go to *Step 15*.
Step 15. if (rand(2) > 0), put rand(2) = rand(2)/2 and go to *Step 8*; else set $j = j + 1$ and go to *Step 4*.

Algorithm 3: The steps of verifying number of orders assigned to a plant.

Table 11: Performance analysis of large sized unbalanced problems.

Unbalanced problems				
m	g	n	Mean of RPD	
			SA	VNS
2	4	40	7.96	0
2	4	60	0.4	4.2
2	4	80	1.2	1.16
2	6	40	3.36	1.63
2	6	60	1.27	4.17
2	6	80	2.16	3.13
4	4	40	0	1.29
4	4	60	7.74	0.57
4	4	80	7.78	4.02
4	6	40	4.11	1.5
4	6	60	3.08	0
4	6	80	0	6.89
Total mean of RPD			3.255	2.38

produce better solutions (Heragu). 2-OPT algorithm considers pairwise exchanges of the customer positions. As an application of this method one distribution center (plant) is randomly selected and the positions of two customers in a sequence mutually exchange. This operation continues since all of the customer positions are tested or the OFV improved. For realization of the concept an example; is illustrated in Figure 6.

3.3.4. Customer Assignment Perturbation. This heuristic has high quality and effectiveness for reducing the gap and it is illustrated completely with an example, this method was firstly used by some researchers such as Parthasarathy and Rajendran [13] who applied it for optimizing jobs sequence. Suppose that there are four distribution centers and five customers that their related assignment to the distribution centers is {2-3-1-2-4} for customer 1 up to customer 5, respectively. Hence the distribution center in the first position can be inserted at any position between 2 and n (here $n = 5$). Then a random number generated between 2 and n is used to select the job position. Assume that the selected position is 3. The assignment found in this position is inserted to the third position and a new assignment for the customers is presented as {3-1-2-2-4}. Consider the assignment in the second position; it can be inserted to the positions between three and five on its left position and inserted to the first position on its right. The same process should be continued for all of the positions. After doing this procedure, any sequence of assignment is accepted that does not exceed the capacity of the distribution centers. Some examples for this method are shown in Figure 7.

3.3.5. Generating Initially Random Solutions. A procedure is considered in this paper for generating initial random solutions. According to this process first we should decide about the assignment of customers to distribution centers. Hence a random sequence of customers is produced, and then by considering the time distances between the plants and DCs, the one which is the nearest distribution center to the first customer in the sequence is selected for assignment; after that for the second up to the end customers, this process is continued but the assignment occurs when the capacity limitation is not exceeded. For the second step after determining the customer assignment, the number of orders which is dedicated to a special plant for each customer should be determined. So to reach this goal, generate random numbers in the range between (0, 100) and this generation is done according to the number of plants. Then the produced numbers are normalized and multiplying to the specific customer demands and they are inserted in the right positions in the related matrix structure. Then the accumulation of the dedicated orders is subtracted from the customer demands and the result is randomly added to one of the plants orders for that specific customer. Finally sequences of customers in plants and distribution centers are absolutely generated randomly. This procedure repeats since the stopping number of iterations or the CPU time allowed is met and in each iteration the objective function improves. Then final solution in this section is considered as an initial solution for the metaheuristic approach.

4. Experimental Evaluation

In this section, we evaluate the performance of the proposed metaheuristic algorithms. For this purpose, two sets of experiments are conducted; first we evaluate the performance of each metaheuristic by comparing it with the optimal solution for small instances, and then the evaluation is done by comparing two metaheuristics in large sized problems.

These algorithms are implemented in Visual C#. Net and run on a PC with 2.4 GHz Intel Core i3-370 M processor CPU and 4 GB of RAM memory. The performance measure used in this study is relative percentage deviation (RPD). It can be calculated as follows:

$$\mathrm{RPD} = \frac{\mathrm{Alg}_{\mathrm{sol}} - \mathrm{Min}_{\mathrm{sol}}}{\mathrm{Min}_{\mathrm{sol}}} \times 100, \tag{22}$$

where $\mathrm{Alg}_{\mathrm{sol}}$ is the objective function obtained by any of the algorithm. $\mathrm{Min}_{\mathrm{sol}}$ is the lowest objective function obtained for a given instance.

4.1. Test Problems. For generating test problems a special feature of supply chain which is called balances is considered. A balanced problem is the one that similar parameters of different stages have equal scales [9]. Here, one of these parameters is the distance between the supply chain elements. Therefore for generating this parameter we consider balanced and unbalanced cases. Some of the parameters such as processing time, distances between supply chain elements, customer orders and due dates, and capacity of the distribution centers are randomly generated from uniform distributions. Other parameters such as the number of plants and distribution centers and also number of customers are selected from discrete sets. Table 8 presents the interval value of the test problems.

Table 12: Performance analysis of small sized balanced problems.

Problem specifications					Balanced problems			
					SA		VNS	
m	g	n	Example number	Optimal solution	Objective function	PRD	Objective function	PRD
2	4	10	1	191.73	210.5	9.79	209.21	9.12
2	4	10	2	99.61	109.29	10.22	111.63	12.07
2	6	10	1	267.45	305.43	14.2	300.87	12.5
2	6	10	2	317.75	349.89	10.11	336.72	5.97
2	4	20	1	480.65	529.19	10.1	518.54	7.88
2	4	20	2	462.09	492.29	6.54	503.31	8.92
2	6	20	1	703.36	772.87	9.88	729.7	3.74
2	6	20	2	488.92	532.66	8.95	502.27	2.73

Table 13: Performance analysis of small sized unbalanced problems.

Problem specifications					Unbalanced problems			
					SA		VNS	
m	g	n	Example number	Optimal solution	Objective function	PRD	Objective function	PRD
2	4	10	1	136.64	149.6	9.48	151.09	10.58
2	4	10	2	128.22	143.14	11.64	132.48	3.32
2	6	10	1	354.76	369.02	9.86	345.77	2.94
2	6	10	2	171.87	186.98	8.79	181.32	5.5
2	4	20	1	649.66	731.2	12.55	724.54	11.53
2	4	20	2	522.95	571.56	9.3	542	3.64
2	6	20	1	833.32	903.25	8.39	893.01	7.16
2	6	20	2	649.20	688.58	6.07	653.41	0.65

For generating customer due dates we follow the uniform distribution $[p(1 - \mathrm{TF} - \mathrm{RDD}/2), p(1 - \mathrm{TF} + \mathrm{RDD}/2)]$ that is first proposed by Potss and Van Wassenhove [8] for single machine scheduling problems. According to his research the due dates of jobs followed uniform distribution $[\sum_{j=1}^{n} P_j(1 - \mathrm{TF} - \mathrm{RDD}/2), \sum_{j=1}^{n} P_j(1 - \mathrm{TF} + \mathrm{RDD}/2)]$ in which TF is the due date tightness and RDD is the due date range. $\sum_{j=1}^{n} P_j$ is the accumulation of processing time of all jobs considered as the delivery time of the jobs. The estimation of delivery time of each customer in the proposed model could be as follows:

$$\left(\frac{\sum_{t=1}^{m} p_t}{m} \times \frac{\sum_{j=1}^{n} d_j}{n}\right) \times \left(\frac{n}{m}\right) + \frac{\sum_{t=1}^{m}\sum_{l=1}^{g} \mathrm{time}_{t,l}}{m \times g} + \left(\frac{\sum_{l=1}^{g} p_l}{g} \times \frac{\sum_{j=1}^{n} d_j}{n}\right) \times \left(\frac{n}{g}\right) + \frac{\sum_{l=1}^{g}\sum_{j=1}^{n} \mathrm{time}_{l,j}}{n \times g}. \tag{23}$$

For generating distribution capacities, according to the customer demands the following uniform distribution is used:

$$\left(\frac{3}{2}\frac{\sum_{j=1}^{n} d_j}{g}, \frac{5}{2}\frac{\sum_{j=1}^{n} d_j}{g}\right). \tag{24}$$

Focusing on the above descriptions for generating the test problems, 16 problems were generated and solved by the proposed algorithms and comparisons with the optimal solutions were done and 72 test problems in large size were produced in order to study and investigate the superior metaheuristic algorithm by doing comparisons on the performance parameter. For the CPU time usage, different examples for both VNS and SA approaches have been executed and the related equation which is dependent on the CPU time was obtained. Consider

$$0.40 \times \left[\frac{(g+m)}{4} \times \frac{n}{2}\right] \times 60. \tag{25}$$

Figure 8 shows the CPU time usage by any of the algorithms and we can come to a conclusion that VNS algorithm solutions converge earlier than the SA algorithm solutions and the objective function value is nearer to the optimal solution.

4.2. Design of Experiments. There are some parameters in SA algorithm whose values should be determined through some experiments. These parameters are ΔE_1, $P(A)_1$, CR, T_1. For finding an efficient set for these parameters, three levels were specified for each of them and by executing 81 experiments with 3 iterations on some test problems, the efficient set is determined and presented in Table 9. The test problems were generated in balanced case and for 10, 20, 50 number of customers. Number of plants and distribution centers were selected from the sets {2} and {4, 6}, respectively.

4.3. Performance Analysis. Here the results of our computations and the comparisons between the performances of algorithms are presented. For generating large problems, number of customers selected from the set {40, 60, 80} and for each of the specified problem size three disparate examples have been generated. Each problem was solved 3 times and the result for each of them is the average of these three executions. Tables 10, 11, 12, and 13 show the results of balanced and unbalanced large problems and the interval plots of the results, shown in Figures 9 and 10, indicate that variable neighborhood search algorithm has better performance than the proposed simulated annealing.

Various examples were solved as small sized problems and the performance parameter confirmed the prominence of VNS. The basis for the size selection of the problems for comparison with optimal solutions was the execution time of the GAMS software which does not require time greater than 2 hours.

The results show that the relative parameter for the VNS algorithm in balanced and unbalanced problems has the mean of 7.866 and 5.6, respectively, and for the SA algorithm has the mean of 9.97 for balanced problems and 9.51 for unbalanced ones. According to the computational results, VNS shows better performance than SA in both balanced and unbalanced small instances. Also as shown in Figure 11 according to different number of jobs VNS algorithm has better performance than SA algorithm, but it should be stated that for middle sized problems two algorithms nearly have the same performance.

5. Conclusion and Future Research

Because of increasing the competitiveness, satisfaction of customers plays an important role for the survival of the markets. Different factors can have an effect on their satisfaction, and one of them is the delivery time of the products. In supply networks many operations should be done to convert the raw materials to finished products and deliver them to the customers; therefore scheduling these jobs and deciding about their sequence in the network help us to perform more effectively and do the best in this field. In this study we try to integrate different parts of the supply network and proposed a model in order to make proper decisions to accelerate the production and delivery process of the orders. Then two algorithms were suggested for solving the model and after presenting the computational results we introduced VNS approach as the superior algorithm for solving this model in large sized problems.

In this model, transportation means has not been considered, so for future research, using transportation means both in the first and the third stage could be a suggestion to extend the problem. Also in metaheuristic approach using hybrid approach might be more effective for solving the large instances.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

[1] S. H. Zegordi and M. A. Beheshti Nia, "A multi-population genetic algorithm for transportation scheduling," *Transportation Research E*, vol. 45, no. 6, pp. 946–959, 2009.

[2] S. H. Zegordi, I. N. K. Abadi, and M. A. B. Nia, "A novel genetic algorithm for solving production and transportation scheduling in a two-stage supply chain," *Computers and Industrial Engineering*, vol. 58, no. 3, pp. 373–381, 2010.

[3] K. Li, A. I. Sivakumar, and V. K. Ganesan, "Complexities and algorithms for synchronized scheduling of parallel machine assembly and air transportation in consumer electronics supply chain," *European Journal of Operational Research*, vol. 187, no. 2, pp. 442–455, 2008.

[4] K. Li, V. K. Ganesan, and A. I. Sivakumar, "Scheduling of single stage assembly with air transportation in a consumer electronic supply chain," *Computers and Industrial Engineering*, vol. 51, no. 2, pp. 264–278, 2006.

[5] M. Zandieh and S. Molla-Alizadeh-Zavardehi, "Synchronizing production and air transportation scheduling using mathematical programming models," *Journal of Computational and Applied Mathematics*, vol. 230, no. 2, pp. 546–558, 2009.

[6] D. Naso, M. Surico, B. Turchiano, and U. Kaymak, "Genetic algorithms for supply-chain scheduling: a case study in the distribution of ready-mixed concrete," *European Journal of Operational Research*, vol. 177, no. 3, pp. 2069–2099, 2007.

[7] B. Chen and C.-Y. Lee, "Logistics scheduling with batching and transportation," *European Journal of Operational Research*, vol. 189, no. 3, pp. 871–876, 2008.

[8] I. Averbakh and Z. Xue, "On-line supply chain scheduling problems with preemption," *European Journal of Operational Research*, vol. 181, no. 1, pp. 500–504, 2007.

[9] S. A. Mansouri, "A simulated annealing approach to a bi-criteria sequencing problem in a two-stage supply chain," *Computers and Industrial Engineering*, vol. 50, no. 1-2, pp. 105–119, 2006.

[10] C.-J. Liao, C.-C. Shyu, and C.-T. Tseng, "A least flexibility first heuristic to coordinate setups in a two- or three-stage supply chain," *International Journal of Production Economics*, vol. 117, no. 1, pp. 127–135, 2009.

[11] Z. Sevkli and F. E. Sevilgen, "variable neighborhood search for the orienting problem," in *Lecture Notes in Computer Science*, vol. 4263, pp. 134–143, Springer, 2006.

[12] R. Tavakkoli-Moghaddam, N. Safaei, and Y. Gholipour, "A hybrid simulated annealing for capacitated vehicle routing problems with the independent route length," *Applied Mathematics and Computation*, vol. 176, no. 2, pp. 445–454, 2006.

[13] S. Parthasarathy and C. Rajendran, "A simulated annealing heuristic for scheduling to minimize mean weighted tardiness in a flowshop with sequence-dependent setup times of jobs—a case study," *Production Planning and Control*, vol. 8, no. 5, pp. 475–483, 1997.