Li and Li *Journal of Inequalities and Applications* 2014, 2014:409
http://www.journalofinequalitiesandapplications.com/content/2014/1/409

Journal of Inequalities and Applications
a SpringerOpen Journal

RESEARCH Open Access

# Integrated production and distribution scheduling problems related with fixed delivery departure dates and number of late orders

Shanlin Li* and Maoqin Li

*Correspondence:
lishanlin56@hotmail.com
Department of Mathematics, Taizhou University, Taizhou, Zhejiang 317000, P.R. China

**Abstract**

We consider an integrated production and distribution scheduling problem faced by a typical make-to-order manufacturer which relies on a third-party logistics provider for finished product delivery to customers. In the beginning of a planning horizon, the manufacturer has received a set of orders to be processed on a single production line. Completed orders are delivered to customers by a finite number of vehicles (*e.g.* trucks, air freight containers on specific air flights) provided by the 3PL company which follows a fixed daily or weekly shipping schedule such that the vehicles have fixed departure dates which are not part of the decisions. The problem is to find a feasible schedule that minimizes one of the following objective functions: (1) the number of late orders, (2) the number of vehicles used subject to the condition that the number of late orders is minimum. We show that both problems are solvable in polynomial time.

**Keywords:** integrated production and distribution; 3PL; fixed departure dates; due dates; number of late orders

## 1 Introduction

Fierce competition in today's global market and heightened expectations of customers have forced companies to invest aggressively to reduce inventory levels across the supply chain on one hand and be more responsive to customers on the other. To reduce inventory, an increasing number of companies now adopt make-to-order (a.k.a. assemble-to-order, build-to-order) business models in which products are custom-made and delivered to customers within a very short lead time directly from the factory. Consequently, there is little or no finished product inventory in the supply chain such that production and outbound distribution are very intimately linked and must be scheduled jointly to achieve a desired on-time delivery performance at minimum total cost. To improve delivery timeliness without having to invest in logistics assets, a majority of the companies worldwide rely on third-party logistics (3PL) providers for their daily distribution and other logistics needs (Langley *et al.* [5]). 3PL providers often follow a fixed daily or weekly schedule for serving their customers. For example, many package delivery service providers such as UPS and FedEx have daily fixed package pickup times; and most 3PL rail, ocean, and

air freight service providers have a fixed weekly schedule for a specific origin-destination pair.

In this paper, we study integrated production and outbound distribution scheduling decisions commonly faced by many manufacturers that operate in a make-to-order mode and rely on a 3PL provider for finished product delivery to customers where the 3PL provider follows a fixed delivery schedule. Examples of such manufacturers include most high-end custom-made consumer electronics product manufacturers based in Asia that rely on air flights (which have fixed departure times) to deliver finished products to the US and European markets. The production and distribution scheduling problem faced by such a manufacturer can be described as follows. At the beginning of a planning horizon, the manufacturer has received a set $J = \{J_1, J_2, \ldots, J_n\}$ of $n$ independent orders from its customers to be processed on a single assembly line. Order $J_i$ has a processing time $p_i$ and a desired due date $d_i$ which is negotiated and agreed on by the manufacturer and the customer who placed the order. Finished orders are delivered by vehicles which have fixed departure times. In the planning horizon, there are $z$ possible vehicle departure time instants $T_1, T_2, \ldots, T_z$, whereby at time $T_j$, $1 \le j \le z$, there are $v_j$ vehicles available for delivery. In the air flight case, each vehicle represents an air freight container. Based on a contractual agreement between the manufacturer and the 3PL provider, the manufacturer can use a certain number (*e.g.* $v_j$) of containers available on a given flight with departure time $T_j$. Usually the 3PL provider charges the manufacturer a fixed transportation cost for each air freight container used. Thus, the total transportation cost is represented by the total number of vehicles (*i.e.* total number of containers) used. Each order is packaged into a standard-size pallet for delivery convenience regardless of the order size. Each vehicle can deliver at most $C$ orders (*e.g.* in the air flight case, each container can hold up to $C$ pallets). The $v_j$ vehicles can only deliver orders that are completed by time $T_j$. A feasible schedule is one in which each order has completed processing and delivered by one of the available vehicles. Without loss of generality, we may assume that $T_z \ge \sum_{i=1}^{n} p_i$, otherwise, there is at least one order that cannot be delivered and hence there is no feasible schedule.

In a given feasible schedule, if order $J_i$ is delivered at time $T_j$ and $T_j > d_i$, we define $U_i$ to be 1; if order $J_i$ is delivered at time $T_j$ and $T_j \le d_i$, we define $U_i$ to be 0. We say in a given feasible schedule an order $J_i$ is *early* if $U_i = 0$ and *late* if $U_i = 1$. The minimum number of late orders $\sum_{i=1}^{n} U_i$ measures the delivery timeliness relative to the customers' desired due dates and is one of the most commonly used measurements in practice. The problem is to find a feasible schedule that minimizes one of the following objective functions: (1) $\sum_{i=1}^{n} U_i$, (2) the number of vehicles used subject to the condition that $\sum_{i=1}^{n} U_i$ is minimum. We show that all two problems are solvable in polynomial time.

The remainder of this paper is organized as follows. We give a brief literature review in the rest of this section. In Section 2, we give a simple algorithm to check the feasibility of a given instance of the problem. In Sections 3 and 4, we give polynomial-time algorithms to solve problems (1) and (2), respectively. We conclude the paper in Section 5.

### 1.1 Related literature

Research on integrated production and outbound distribution scheduling problems is relatively recent, but it has attracted a rapidly growing interest in the last several years [1]. In most of the problems considered in the literature, vehicle departure times are not fixed and need to be determined along with other decisions. Only a handful of problems considered

in the literature involve fixed vehicle departure times. Such problems can be classified into two types based on vehicle availability. One type assumes that there are infinite number of vehicles available at each departure time, whereas the other type assumes that there are a limited number of vehicles available at each departure time. Stecke and Zhao [11], Melo and Wolsey [10] and Zhong *et al.* [14] all consider similar problems with an infinite number of vehicles where each order has a deadline which has to be satisfied and the objective is to minimize the total transportation cost. Their problems differ slightly in the structure of the transportation cost. Since the focus of this paper is on problems with a finite number of vehicles, we do not review these papers in detail.

Li *et al.* [7–9] and Zandieh and Molla-Alizadeh-Zavardehi [13] study several similar problems with a finite number of vehicles at each departure time which are all motivated by applications involving synchronizing assembly operations of consumer electronics products such as PCs and air transportation schedules. Orders may have different sizes and the capacity of a vehicle is measured by the total size (weight or volume) of orders that it can carry. There is an earliness or tardiness penalty if an order is delivered earlier or later than the due date. The objective is to minimize the total transportation cost and total weighted earliness and tardiness penalty. Li *et al.* [9] consider the case where all the orders are processed on a number of parallel production lines, whereas the other papers consider the case with a single production line. The problems are strongly NP-hard as they contain the strongly NP-hard classical single-machine total weighted tardiness scheduling problem (Lenstra *et al.* [4]) as a special case when the delivery part is not considered. These papers propose various heuristics for solving their problems. Wang *et al.* [12] study a problem with a finite number of vehicles which involves coordinating mail processing and distribution schedules at a mail processing and distribution center. The objective is to minimize the total unused vehicle capacity. The authors show that this problem is strongly NP-hard and propose dispatching rules and heuristics.

Fu *et al.* [2] consider a problem where there is a limit on the total delivery capacity at each departure time. Each order has a delivery departure deadline, a production window, a size and a profit. The problem is to select a subset of orders to accept so as to maximize the total profit of the accepted orders under the constraint that each accepted order is processed within its production window, the delivery of this order is departed by its delivery departure deadline, and the total size of the orders delivered at each departure time does not exceed the available vehicle capacity limit. The problem is strongly NP-hard as it contains the bin packing problem as its special case when only the delivery part is considered. The authors propose a polynomial-time approximation scheme for the problem.

Leung and Chen [6] discuss an integrated production and distribution scheduling problem. In the beginning of a planning horizon, the manufacturer has received a set of orders to be processed on a single production line. Completed orders are delivered to customers by a finite number of vehicles provided by the 3PL company which follows a fixed daily or weekly shipping schedule such that the vehicles have fixed departure dates. The problem is to find a feasible schedule that minimizes one of the following objective functions: (1) the maximum lateness of orders, (2) the number of vehicles used subject to the condition that the maximum lateness is minimum, (3) the weighted sum of the maximum lateness and the number of vehicles used. They show that all three problems are solvable in polynomial time.

Finally we note that a remotely related class of problems - production scheduling problems that involve fixed delivery departure dates but do not involve delivery vehicles - have been extensively studied in the literature (*e.g.* Hall *et al.* [3]). These problems only consider production scheduling decisions without explicitly involving delivery vehicles. They do not consider any vehicle related objective function, and they can be viewed as special cases of our problems with an infinitely many delivery vehicles available at each departure time so that vehicle availability and vehicle capacity are never a constraint.

## 2 Feasibility

Given an instance of the problem, we first need to determine whether there is any feasible schedule. The following, Algorithm VA, comes from Leung and Chen [6] and is now stated. The idea is to schedule the orders in Smallest-Processing-Time first (SPT) order. Let $S$ be a SPT schedule. Let $S(T_i, T_j)$ denote the set of orders completed in the interval $(T_i, T_j]$ in $S$. Let $T_{z+1}$ be any integer greater than $T_z$. We then assign the orders to the vehicles by the following algorithm.

**Algorithm VA**

**Input:** An SPT schedule $S$. For each departure time $T_j$, for $1 \le j \le z$, there are $v_j$ vehicles available for delivery at $T_j$.

**Output:** 'Yes' if it is possible to deliver all the orders in the schedule $S$; 'No' otherwise.

**Method:**

1. $S(T_z, T_{z+1}) := \emptyset$; $T_0 := 0$.
2. For $j = 1$ to $z$ do
   (a) Assign the orders in $S(T_{j-1}, T_j)$ to one of the $v_j$ vehicles available at time $T_j$. After an order is assigned to a vehicle, it is removed from $S(T_{j-1}, T_j)$.
   (b) If all of the $v_j$ vehicles are full and there is still at least one unassigned order in $S(T_{j-1}, T_j)$, then put all the unassigned orders into $S(T_j, T_{j+1})$.
3. If $S(T_z, T_{z+1}) = \emptyset$ then return 'Yes', else return 'No'.

If the algorithm returns 'Yes', then there is a feasible schedule; otherwise, there is no feasible schedule.

Let $(S_1, S_2)$ be a SPT schedule of the instance of the problem. Clearly, there is no feasible schedule for the instance of the problem if $|S_2| = \sum_{j=k}^{z} v_j C$ and $\sum_{J_i \in S_1} p_i > T_{k-1}$, where $1 \le k \le z$. In the remainder of this paper, we will assume that there is a feasible schedule for the given instance.

## 3 Number of late orders

In this section we give a polynomial-time algorithm to solve the number of late orders problem. Similar to the idea in Leung and Chen [6], we first classify the set of orders based on certain criteria, and then do iteration for the types to obtain an optimal schedule. For each $1 \le i \le n$, we compute the maximal departure time $\bar{d}_i = \max\{T_m | T_m \le d_i, 1 \le m \le z\}$ such that order $J_i$ must be a late order if it is delivered after time $\bar{d}_i$. For each $1 \le j \le z$, we classify the set of orders based on $\bar{d}_i : N_{0,j} = \{J_i | \bar{d}_i = T_j, 1 \le i \le n\}$. Clearly, in a given schedule an order $J_i \in N_{0,j}$ is early if and only if it is completed and delivered by time $T_j$. The following algorithm decides whether there is a feasible schedule that minimizes the number of late orders. To break ties when sequencing the orders in increasing order of

their processing times, we employ the last-in first rule, *i.e.*, we arrange order $J_j$ before order $J_i$ if $J_j$ is merged into a set of orders $S$ and $p_j = p_i$, where $J_i \in S$.

**Algorithm NF**

**Input:** A set of $n$ orders $J_1, \ldots, J_n$.

**Output:** A feasible schedule that minimizes the number of late orders.

**Method:**

1. For each $1 \le i \le n$, let $\bar{d}_i = \max\{T_m | T_m \le d_i, 1 \le m \le z\}$.
2. For each $1 \le j \le z$, let $N_{0,j} = \{J_i | \bar{d}_i = T_j, 1 \le i \le n\}$.
3. $P = \sum_{i=1}^{n} p_i$; $P' := P$; $t := 0$. $T_0 := 0$. $N_{0,0} := \emptyset$.
4. For $j = z$ down to 1 do
   (a) Let $R_{t,j}$ be all the orders in $N_{t,j}$, arranged in non-decreasing order of their processing times.
   (b) If $v_j C < |R_{t,j}|$, then update $N_{t,j-1} := N_{t,j-1} \cup F_{t,j}$ and $R_{t,j} := R_{t,j} \backslash F_{t,j}$, where $F_{t,j}$ are the first $|R_{t,j}| - v_j C$ orders from $R_{t,j}$.
   (c) $p := \sum_{J_i \in R_{t,j}} p_i$.
   (d) Schedule the orders in $R_{t,j}$ from time $P' - p$ to $P'$. These orders will be delivered by the vehicles available at time $T_j$.
   (e) $P' := P' - p$.
   (f) If $P' > T_{j-1}$, find $J_{l_t} \in N_{t,j_l}$ such that $0 \le j_l \le j - 1$ and $p_{l_t} = \max\{p_i | J_i \in \bigcup_{i=0}^{j-1} N_{t,i}\}$. Update $N_{t,0} := N_{t,0}$, $N_{t,1} := N_{t,1}, \ldots, N_{t,j_l-1} := N_{t,j_l-1}$, $N_{t,j_l} := N_{t,j_l} \backslash \{J_{l_t}\}$, $N_{t,j_l+1} := N_{t,j_l+1}, \ldots, N_{t,j-1} := N_{t,j-1}$, $N_{t,j} := R_{t,j}, \ldots, N_{t,z-1} := R_{t,z-1}$, $N_{t,z} := R_{t,z} \cup \{J_{l_t}\}$ and $t := t + 1$, return to 4.
5. Stop. The schedule $(R_{t,1}, \ldots, R_{t,z})$ is an optimal feasible schedule, where the orders in $R_{t,j}$ will be delivered by the vehicles available at time $T_j$ and $t$ denotes the number of late orders in the optimal feasible schedule.

Algorithm NF consists of $t+1$ iterations. $t$ in Algorithm NF decides not only the number of iterations but also the number of late orders. The iteration starts from $t := 0$. The $t+1$th iteration checks whether there is a feasible schedule such that it contains $t$ late orders exactly. If not, select an order as a late order, update $t := t + 1$ and other data, proceed to the next iteration. For ease of presentation, we list detailed output data generated by the $t + 1$ iterations as follows:

$$(R_{0,j_0}, \ldots, R_{0,z}), J_{l_0}; \quad \ldots; \quad (R_{t-1,j_{t-1}}, \ldots, R_{t-1,z}), J_{l_{t-1}}; \quad (R_{t,1}, \ldots, R_{t,z}), \tag{1}$$

where for $k = 0, 1, \ldots, t-1$, $1 \le j_k \le z$, $P - \sum_{J_i \in \bigcup_{j=j_k}^{z} R_{k,j}} p_i > T_{j_k - 1}$, $J_{l_k} \in J \backslash \bigcup_{j=j_k}^{z} R_{k,j}$, and $p_{l_k} = \max\{p_i | J_i \in J \backslash \bigcup_{j=j_k}^{z} R_{k,j}\}$.

The following lemma describes some properties of the data obtained by Algorithm NF.

**Lemma 1** *Let data in* (1) *be obtained by Algorithm* NF. *All of the following hold.*

(i) $j_0 \ge j_1 \ge \cdots \ge j_t = 1$.

(ii) $p_{l_0} \ge p_{l_1} \ge \cdots \ge p_{l_{t-1}}$.

(iii) *For each* $0 \le k \le t-1$ *and* $k+1 \le i \le t$, $J_{l_k} \in \bigcup_{j=j_k}^{z} R_{i,j}$ *and is a late order if the orders in* $R_{i,j}$ *are delivered at time* $T_j$.

*Proof* Let $(R_{0,j_0},\dots,R_{0,z})$, $J_{l_0}$ be the output data obtained by the first iteration of Algorithm NF. Then we have $P-\sum_{J_i\in\bigcup_{j=m}^{z}R_{0,j}}p_i\le T_{m-1}$, $|R_{0,m}|\le v_mC$ for $m=z,z-1,\dots,j_0+1$ and $P-\sum_{J_i\in\bigcup_{j=j_0}^{z}R_{0,j}}p_i>T_{j_0-1}$, $|R_{0,j_0}|\le v_{j_0}C$. When running the second iteration on the data $(N_{1,0},N_{1,1},\dots,N_{1,z})$, $R_{1,z}=R_{0,z}\cup\{J_{l_0}\}$ if $|R_{0,z}|<v_zC$ and $R_{1,z}=R_{0,z}\cup\{J_{l_0}\}\backslash\{J_l\}$ otherwise, where $p_l=\min\{p_i|J_i\in R_{0,z}\cup\{J_{l_0}\}\}$. In either case, $\sum_{J_i\in R_{1,z}}p_i\ge\sum_{J_i\in R_{0,z}}p_i$. This implies that $P-\sum_{J_i\in R_{1,z}}p_i\le P-\sum_{J_i\in R_{0,z}}p_i\le T_{z-1}$. Further, we have for $m=z-1,\dots,j_0$,

$$\bigcup_{j=m}^{z}R_{1,j}=\bigcup_{j=m}^{z}R_{0,j}\cup\{J_{l_0}\}\quad\text{if }\sum_{j=m}^{z}|R_{0,j}|<\sum_{j=m}^{z}v_jC, \tag{2}$$

$$\bigcup_{j=m}^{z}R_{1,j}=\bigcup_{j=m}^{z}R_{0,j}\cup\{J_{l_0}\}\backslash\{J_l\}\quad\text{if }\sum_{j=m}^{z}|R_{0,j}|=\sum_{j=m}^{z}v_jC, \tag{3}$$

where $p_l=\min\{p_i|J_i\in\bigcup_{j=m}^{z}R_{0,j}\cup\{J_{l_0}\}\}$. This implies that for $m=z-1,\dots,j_0+1$, $P-\sum_{J_i\in\bigcup_{j=m}^{z}R_{1,j}}p_i\le P-\sum_{J_i\in\bigcup_{j=m}^{z}R_{0,j}}p_i\le T_{m-1}$. Thus, $j_0\ge j_1$ holds. The proof of the following inequalities in (i) are similar to that of the first inequality. We conclude that (i) holds.

By Algorithm NF, we have $J_{l_0}\in J\backslash\bigcup_{j=j_0}^{z}R_{0,j}$ and $J_{l_1}\in J\backslash\bigcup_{j=j_1}^{z}R_{1,j}$, where $p_{l_0}=\max\{p_i|J_i\in J\backslash\bigcup_{j=j_0}^{z}R_{0,j}\}$ and $p_{l_1}=\max\{p_i|J_i\in J\backslash\bigcup_{j=j_1}^{z}R_{1,j}\}$. Due to (i), $j_0\ge j_1$. It follows from the argument of (i) that $\bigcup_{j=j_0}^{z}R_{1,j}=\bigcup_{j=j_0}^{z}R_{0,j}\cup\{J_{l_0}\}$ if $\sum_{j=j_0}^{z}|R_{0,j}|<\sum_{j=j_0}^{z}v_jC$. In the case, since $J\backslash\bigcup_{j=j_0}^{z}R_{0,j}\supseteq J\backslash\bigcup_{j=j_1}^{z}R_{1,j}$, we have $p_{l_0}\ge p_{l_1}$. $\bigcup_{j=j_0}^{z}R_{1,j}=\bigcup_{j=j_0}^{z}R_{0,j}\cup\{J_{l_0}\}\backslash\{J_l\}$ if $\sum_{j=j_0}^{z}|R_{0,j}|=\sum_{j=j_0}^{z}v_jC$, where $p_l=\min\{p_i|J_i\in\bigcup_{j=j_0}^{z}R_{0,j}\cup\{J_{l_0}\}\}$. In the case, since $J\backslash\bigcup_{j=j_0}^{z}R_{0,j}\supseteq\{J_l\}\cup J\backslash\bigcup_{j=j_1}^{z}R_{1,j}$, along with $p_l\le p_{l_0}$, we have $p_{l_0}\ge p_{l_1}$. Thus, the first inequality $p_{l_0}\ge p_{l_1}$ in (ii) holds. The proofs of the following inequalities in (ii) are similar to that of the first inequality. We conclude that (ii) holds.

For any $k\in\{0,1,\dots,t-1\}$, Suppose $J_{l_k}\in N_{0,k_0}$. Clearly, $J_{l_k}$ is a late order if it is delivered at time $T_j$, where $k_0<j\le z$. To show (iii), there are two cases to consider: (a) $k_0<j_k$ and (b) $k_0\ge j_k$.

(a) $k_0<j_k$. Assume that $J_{l_k}\in\bigcup_{j=j_k}^{z}R_{k+1,j}$ does not hold. Due to the argument of (i), we have $\sum_{j=j_k}^{z}|R_{k,j}|=\sum_{j=j_k}^{z}v_jC$, $\bigcup_{j=j_k}^{z}R_{k+1,j}=\bigcup_{j=j_k}^{z}R_{k,j}$, and $p_{l_k}=\min\{p_i|J_i\in\bigcup_{j=j_k}^{z}R_{k,j}\cup\{J_{l_k}\}\}$. This, along with $p_{l_k}=\max\{p_i|J_i\in J\backslash\bigcup_{j=j_k}^{z}R_{k,j}\}$, implies that for any $J_{i_1}\in J\backslash\bigcup_{j=j_k}^{z}R_{k,j}$ and $J_{i_2}\in\bigcup_{j=j_k}^{z}R_{k,j}$, $p_{i_1}\le p_{i_2}$. This, along with $\sum_{j=j_k}^{z}|R_{k,j}|=\sum_{j=j_k}^{z}v_jC$ and $P-\sum_{J_i\in\bigcup_{j=j_k}^{z}R_{k,j}}p_i>T_{j_k-1}$, implies that there is no feasible schedule for the instance of the problem, which contradicts the assumption that there is a feasible schedule for the given instance. Thus, we have $J_{l_k}\in\bigcup_{j=j_k}^{z}R_{k+1,j}$ and is a late order if the orders in $R_{k+1,j}$ are delivered at time $T_j$. Further, due to the argument of (i), we have $\bigcup_{j=j_k}^{z}R_{k+2,j}=\bigcup_{j=j_k}^{z}R_{k+1,j}\cup\{J_{l_{k+1}}\}$ if $\sum_{j=j_k}^{z}|R_{k+1,j}|<\sum_{j=j_k}^{z}v_jC$, $\bigcup_{j=j_k}^{z}R_{k+2,j}=\bigcup_{j=j_k}^{z}R_{k+1,j}\cup\{J_{l_{k+1}}\}\backslash\{J_l\}$ if $\sum_{j=j_k}^{z}|R_{k+1,j}|=\sum_{j=j_k}^{z}v_jC$, where $p_l=\min\{p_i|J_i\in\bigcup_{j=j_k}^{z}R_{k+1,j}\cup\{J_{l_{k+1}}\}\}$. This, along with $p_{l_k}\ge p_{l_{k+1}}$ and $J_{l_k}$ is merged into $\bigcup_{j=j_k}^{z}R_{k+1,j}$ before $J_{l_{k+1}}$ merged, implies that $J_{l_k}\in\bigcup_{j=j_k}^{z}R_{k+2,j}$ and is a late order if the orders in $R_{k+2,j}$ are delivered at time $T_j$. Similarly, we can show $J_{l_k}\in\bigcup_{j=j_k}^{z}R_{i,j}$ and it is a late order if the orders in $R_{i,j}$ are delivered at time $T_j$ for $i=k+3,\dots,t$.

(b) $k_0\ge j_k$. Note the fact that order $J_{l_k}$ is pushed by the iterations from $N_{0,k_0}$ into $J\backslash\bigcup_{j=j_k}^{z}R_{k,j}$. By Step 4(a) and Step 4(b), we have $|R_{k,j}|=v_jC$ for $j=j_k,\dots,k_0$ and $p_{l_k}=\min\{p_i|J_i\in\bigcup_{j=j_k}^{k_0}R_{k,j}\}$. This implies $J_{l_k}\in\bigcup_{j=k_0+1}^{z}R_{k+1,j}$ $(\subseteq\bigcup_{j=j_k}^{z}R_{k+1,j})$. Otherwise, due to the argument of (i), we have $\sum_{j=k_0+1}^{z}|R_{k,j}|=\sum_{j=k_0+1}^{z}v_jC$, $\bigcup_{j=k_0+1}^{z}R_{k+1,j}=\bigcup_{j=k_0+1}^{z}R_{k,j}$, and $p_{l_k}=\min\{p_i|J_i\in\bigcup_{j=k_0+1}^{z}R_{k,j}\cup\{J_{l_k}\}\}$, and then we have $\sum_{j=j_k}^{z}|R_{k,j}|=\sum_{j=j_k}^{z}v_jC$, $\bigcup_{j=j_k}^{z}R_{k+1,j}=\bigcup_{j=j_k}^{z}R_{k,j}$, and $p_{l_k}=\min\{p_i|J_i\in\bigcup_{j=j_k}^{z}R_{k,j}\cup\{J_{l_k}\}\}$. Similar to the argument of case (a), we

can derive a contradiction. Similarly, we can show $J_{l_k} \in \bigcup_{j=j_k}^{z} R_{i,j}$ and it is a late order if the orders in $R_{i,j}$ are delivered at time $T_j$ for $i = k+2, \ldots, t$. This ends the proof for (iii). □

**Theorem 1** *Algorithm* NF *correctly finds a feasible schedule that minimizes the number of late orders in* $O(zn^2 \log n)$ *time*.

*Proof* We first prove that $\bigcup_{j=j_k}^{z} R_{k,j}$ has not only exactly $k$ late orders but also the maximum total processing time among all schedules for $k = 0, 1, \ldots, t-1$. By the definition of $N_{0,z}$ and Step 4(a) and Step 4(b) of Algorithm NF, we see that $R_{0,z}$ is a set of early orders delivered at time $T_z$ with the maximum total processing time among all schedules. Similarly, we see that $\bigcup_{j=m}^{z} R_{0,j}$ is a set of early orders with the maximum total processing time among all schedules for $m = z-1, \ldots, j_0$, where the orders in $R_{0,j}$ are delivered at time $T_j$.

By (2) and (3), we have $\bigcup_{j=j_0}^{z} R_{1,j} = \bigcup_{j=j_0}^{z} R_{0,j} \cup \{J_{l_0}\}$ if $\sum_{j=j_0}^{z} |R_{0,j}| < \sum_{j=j_0}^{z} \nu_j C$, $\bigcup_{j=j_0}^{z} R_{1,j} = \bigcup_{j=j_0}^{z} R_{0,j} \cup \{J_{l_0}\} \setminus \{J_l\}$ if $\sum_{j=j_0}^{z} |R_{0,j}| = \sum_{j=j_0}^{z} \nu_j C$ where $p_l = \min\{p_i | J_i \in \bigcup_{j=j_0}^{z} R_{0,j} \cup \{J_{l_0}\}\}$. This, along with (iii) in Lemma 1, $p_{l_0} = \max\{p_i | J_i \in J \setminus \bigcup_{j=j_0}^{z} R_{0,j}\}$, and $\bigcup_{j=j_0}^{z} R_{0,j}$ being a set of early orders with the maximum total processing time among all schedules, implies that $\bigcup_{j=j_0}^{z} R_{1,j}$ does not only have exactly a late order but also the maximum total processing time among all schedules. Further, by Step 4(a) and Step 4(b) of Algorithm NF, we see that $\bigcup_{j=m}^{z} R_{1,j}$ does not only exactly have a late order but also the maximum total processing time among all schedules for $m = j_0 - 1, \ldots, j_1$. Similarly, we can show the result for $k = 2, \ldots, t-1$.

We below prove that $(R_{t,1}, \ldots, R_{t,z})$ is a feasible schedule minimizing the number of late orders. By Algorithm NF, the feasibility is obvious. For any feasible schedule $S = (R_1, \ldots, R_z)$, where the orders in $R_j$ are delivered at time $T_j$ for $j = 1, \ldots, z$, we see that $\bigcup_{j=j_0}^{z} R_j$ contains at least a late order. Otherwise, by $\sum_{J_i \in \bigcup_{j=j_0}^{z} R_j} p_i \le \sum_{J_i \in \bigcup_{j=j_0}^{z} R_{0,j}} p_i$, we see that $\sum_{J_i \in S \setminus \bigcup_{j=j_0}^{z} R_j} p_i \ge P - \sum_{J_i \in \bigcup_{j=j_0}^{z} R_{0,j}} p_i > T_{j_0-1}$. This contradicts the feasibility of $S$. Given that $\bigcup_{j=j_0}^{z} R_j$ contains at least a late order, $\bigcup_{j=j_1}^{z} R_j$ contains at least two late orders. Otherwise, $\bigcup_{j=j_1}^{z} R_j$ contains a late order. By $\sum_{J_i \in \bigcup_{j=j_1}^{z} R_j} p_i \le \sum_{J_i \in \bigcup_{j=j_1}^{z} R_{1,j}} p_i$, we have $\sum_{J_i \in S \setminus \bigcup_{j=j_1}^{z} R_j} p_i \ge P - \sum_{J_i \in \bigcup_{j=j_1}^{z} R_{1,j}} p_i > T_{j_1-1}$. This contradicts the feasibility of $S$. Similarly, we can show that $\bigcup_{j=j_k}^{z} R_j$ contains $k+1$ late orders at least for $k = 2, \ldots, t-1$. Thus, $(R_{t,1}, \ldots, R_{t,z})$ is a feasible schedule minimizing the number of late orders.

We now show that the algorithm can be implemented to run in $O(zn^2 \log n)$ time. The algorithm consists of $n+1$ iterations at most since there are $n$ late orders at most. Step 1 of the algorithm takes $O(n \log z)$ time. Step 2 takes $O(n \log n)$ time since we can sort the jobs in ascending order of $\bar{d}_i$ and then divide the jobs into various $N_{0,j}$. Step 3 takes $O(n)$ time. Step 4 is iterated $z$ times. Within each iteration, the most time-consuming step is Step 4(a), sorting the orders in ascending order of their processing times, which takes $O(n \log n)$ time. Hence Step 4 takes $O(zn \log n)$ time. Thus, the overall running time of the algorithm is $O(zn^2 \log n)$ time. □

## 4 Minimum number of vehicles used

In this section we show that the problem of minimizing the number of vehicles used subject to the constraint that the number of late orders is minimum can be solved in polynomial time. We assume that we have found the minimum number of late orders using the algorithm given in the previous section. The sets $(R_{t,1}, \ldots, R_{t,z})$ were obtained by Algorithm NF, where $t$ is the minimum number of late orders. By the fact that if $|R_{t,j}| < \nu_j C$, then each order in $\bigcup_{i=1}^{j-1} R_{t,i}$ is an early order and will be a late order if we push the order to be delivered by vehicles at $T_j$. Thus, either there is no feasible schedule or the number of

late orders must will be increased if we push some order in $R_{t,i}$ to be delivered by vehicles at $T_{i+1}, \ldots, T_z$. By the optimality of $t$, we see that the number of late orders must will not be decreased if we push some order in $R_{t,i}$ to be delivered by vehicles at $T_1, \ldots, T_{i-1}$. The following algorithm finds a solution with a minimum number of vehicles used under the constraint that the number of late orders is minimum by pushing some orders to an earlier departure time for delivery.

**Algorithm MV**

**Input:** $z$ sets of orders $R_{t,1}, \ldots, R_{t,z}$ given by Algorithm NF, where $t$ is the minimum number of late orders.

**Output:** A vehicle assignment: $V_1, \ldots, V_z$, where the orders in $V_j$ will be delivered by the vehicles available at time $T_j$, so that the number of vehicles used is minimum.

**Method:**

1. $T_0 := 0$. $v_0 := 0$.
2. For $j = z$ down to 1 do
   (a) Let $V'_j$ be all the orders in $R_{t,j}$, arranged in non-decreasing order of their processing times, and $|V'_j| = kC + r$, where $k$ and $r$ are nonnegative integers, and $0 \le r < C$, and $F$ is the set of the first $r$ orders from $V'_j$.
   (b) If $\sum_{J_i \in \bigcup_{m=1}^{j-1} R_{t,m} \cup F} p_i > T_{j-1}$, then update $A := \emptyset$ and $V_j := V'_j$, and the orders in $V_j$ will be delivered by the vehicles at time $T_j$ and proceed to the next $j$.
   (c) Let $A'$ are the first $k_0 C + r$ orders from $V'_j$, where $k_0$ is the maximal nonnegative integer such that $\sum_{J_i \in \bigcup_{m=1}^{j-1} R_{t,m} \cup A'} p_i \le T_{j-1}$.
   (d) If $\lceil |R_{t,j-1} \cup A'|/C \rceil \le v_{j-1} C$, then update $A := A'$, $R_{t,j-1} := R_{t,j-1} \cup A$ and $V_j := V'_j \backslash A$ and the orders in $V_j$ will be delivered by the vehicles available at time $T_j$ and proceed to the next $j$.
   (e) Call Subalgorithm CA to computer $A$, then update $R_{t,j-1} := R_{t,j-1} \cup A$ and $V_j := V'_j \backslash A$, and the orders in $V_j$ will be delivered by the vehicles at time $T_j$ and proceed to the next $j$.

The algorithm schedules the order delivery backwards, starting from $T_z$ and going down to $T_1$. Now, suppose we are considering the orders in $R_{t,j}$, $1 \le j \le z$. Step 2(a) sorts these orders in $R_{t,j}$ in ascending order of their processing times, and assigns these sorted orders to the set $V'_j$, and expresses the number of orders in $V'_j$ as $kC + r$ and assigns the first $r$ orders from $V'_j$ to the set $F$. As we will see later, some orders from a later departure time, *e.g.*, some orders from $R_{t,h}$ for some $h > j$, may be pushed to an earlier departure time for delivery and join the set $R_{t,j}$. Therefore, there could be more orders in $R_{t,j}$ than are in the initially defined set $R_{t,j}$. We will try to use the minimum number of vehicles to deliver all or part of the orders in $V'_j$. Step 2(b) checks whether it is possible to push the orders in $F$ to be delivered by vehicles at $T_{j-1}, \ldots, T_1$. If it is not possible, Step 2(b) stops the algorithm and outputs $A = \emptyset$ and $V_j = V'_j$. Otherwise, Step 2(c) computes the set of orders $A'$ which made $\lceil |V'_j|/C \rceil - \lceil |A'|/C \rceil$ the minimal number possibly of vehicles used at $T_j$. Step 2(d) and 2(e) exactly decide the set of orders $A$ which made $\lceil |V'_j|/C \rceil - \lceil |A|/C \rceil$ minimal number of vehicles used at $T_j$.

**Subalgorithm CA**

1. $A'$ := the first $k_0 C + r$ orders from $V'_j$, arranged in non-decreasing order of their processing times.

2. Sort the orders in $R_{t,j-1} \cup A'$ in ascending order of their processing times.
3. If $|R_{t,j-1} \cup A'| \le v_{j-1}C$, then stop and output $A = A'$.
4. For $h = 1$ to $j-2$, let $R'_{t,h} := R_{t,h}$.
5. $B :=$ the first $|R_{t,j-1} \cup A'| - v_{j-1}C$ orders from $R_{t,j-1} \cup A'$.
6. For $h = j-2$ down to 1 do
   (a) Sort the orders in $R'_{t,h} \cup B$ in ascending order of their processing times.
   (b) If $\sum_{J_i \in \bigcup_{m=1}^{h} R'_{t,m} \cup B} p_i > T_h$,
      (b1) if $|A'| < C$, stop and output $A = \emptyset$;
      (b2) update $A' :=$ the first $|A'| - C$ orders from $A'$ and return 2.
   (c) If $|R'_{t,h} \cup B| \le v_h C$, then stop and output $A = A'$.
   (d) If $h = 1$, stop and output $A = \emptyset$ if $|A'| < C$ and update $A' :=$ the first $|A'| - C$ orders from $A'$ and return 2 else.
   (e) Update $B :=$ the first $|R'_{t,h} \cup B| - v_h C$ orders from $R'_{t,h} \cup B$ and proceed to the next $h$.

Subalgorithm CA operates as follows. The algorithm consists of $k_0 + 1$ main iterations for the first $k_0 C + r$ orders, the first $(k_0 - 1)C + r$ orders, ..., the first $r$ orders from $V'_j$, respectively, and exactly decides the set of orders $A$ which made $\lceil |V'_j|/C \rceil - \lceil |A|/C \rceil$ minimal number of vehicles used at $T_j$. In Step 1, it sorts the orders in the initial $A'$ in ascending order of their processing times. Now, suppose we are considering the $s$th main iteration, where $1 \le s \le k_0 + 1$. In Step 2, it sorts the orders in $R_{t,j-1} \cup A'$ in ascending order of their processing times. If $|R_{t,j-1} \cup A'| \le v_{j-1}C$, all orders in $A'$ can be delivered by vehicles at $T_{j-1}$. Step 3 stops the algorithm and outputs $A = A'$. Otherwise, Step 4 assigns the orders in $R_{t,h}$ to a temporary set $R'_{t,h}$ for each $h$ from 1 to $j-2$. This is necessary since Subalgorithm CA operates on $R'_{t,h}$ without changing the content of $R_{t,h}$. Step 5 assigns the first $|R_{t,j-1} \cup A'| - v_{j-1}C$ orders from $R_{t,j-1} \cup A'$ to a set $B$. Step 6 consists of $j-2$ secondary iterations, starting from $j-2$ and going down to 1. Step 6 checks whether it is possible to push the orders in $A'$ to be delivered by vehicles at $T_{j-2}, \ldots, T_1$. Suppose we are considering the $h$th secondary iteration for the $s$th main iteration. In Step 6(a), it sorts these orders in $R'_{t,h} \cup B$ in ascending order of their processing times. There are two cases to consider. In case 1, $\sum_{J_i \in \bigcup_{m=1}^{h} R'_{t,m} \cup B} p_i > T_h$. It is impossible to deliver all orders in $A'$ by the vehicles at $T_1, \ldots, T_h$. If $|A'| < C$, Step 6(b1) stops the algorithm and outputs $A = \emptyset$. Otherwise, Step 6(b2) assigns the first $|A'| - C$ orders from $A'$ to $A'$ and proceed to the next main iteration. In case 2, $\sum_{J_i \in \bigcup_{m=1}^{h} R'_{t,m} \cup B} p_i \le T_h$. If $|R'_{t,h} \cup B| \le v_h C$, all orders in $A'$ can be delivered by vehicles at $T_{j-1}, \ldots, T_1$. Step 6(c) stops the algorithm and outputs $A = A'$. Otherwise, when we reach $h = 1$, it is impossible to deliver all orders in $A'$ by the vehicles at $T_1, \ldots, T_h$. Step 6(d) stops the algorithm and outputs $A = \emptyset$ if $|A'| < C$, and assigns the first $|A'| - C$ orders from $A'$ to $A'$ else, and proceed to the next main iteration. Step 6(e) assigns the first $|R'_{t,h} \cup B| - v_h C$ orders from $R'_{t,h} \cup B$ to $B$ and proceed to the next secondary iteration.

**Theorem 2** *Algorithm* MV *finds an optimal solution with the minimum number of vehicles used under the constraint that the number of late orders is minimum in* $O(z^2 n^2 \log n)$ *time*.

*Proof* We first point out that the solution by Algorithm MV does not change the optimality of the number of late orders, since the solution is found by pushing some orders in $R_{t,2}, \ldots, R_{t,z}$ to an earlier departure time for delivery.

Let $V_z$ and $A$ be the output data obtained by the first iteration of Algorithm MV, where $V_z = V'_z \backslash A$ and $A$ is the set of the first $|A|$ orders from $V'_z$ (the orders of $V'_z = R_{t,z}$ have been arranged in non-decreasing order of their processing times). Step 2(b) corresponds to the case where $\sum_{J_i \in \bigcup_{m=1}^{z-1} R_{t,m} \cup F} p_i > T_{z-1}$, where $F$ the set of the first $|F|$ orders from $V'_z$ and $0 < |F| < C$. In the case, it is impossible to deliver all orders in $F$ by the vehicles at $T_1, \ldots, T_{z-1}$. This, along with $F$ consists of the first $|F|$ orders from $V'_z$, means that the number of vehicles used at $T_z$ cannot decrease by one. Thus, $\lceil |V'_z|/C \rceil$ is the minimal number of vehicles used at $T_z$. However, we cannot push part of orders in $F$ to be delivered by vehicles at $T_{z-1}, \ldots, T_1$, since if we do that, it not only does not decrease the number of vehicles used at $T_z$, but it also increases the amount of orders delivered by vehicles at $T_{z-1}, \ldots, T_1$. In the case, the data $V_z = V'_z$ and $A = \emptyset$ are optimal. Under the case of $\sum_{J_i \in \bigcup_{m=1}^{z-1} R_{t,m} \cup F} p_i \le T_{z-1}$, Step 2(c) computes the set of orders $A'$ which made $\lceil |V'_z|/C \rceil - \lceil |A'|/C \rceil$ the minimal number of vehicles possibly used at $T_z$, where $A'$ are the first $k_0 C + r$ orders from $V'_z$. This is because $k_0$ is the maximal nonnegative integer such that $\sum_{J_i \in \bigcup_{m=1}^{z-1} R_{t,m} \cup A'} p_i \le T_{z-1}$ and $A'$ consists of some small orders in $V'_z$. Step 2(d) corresponds to the case where $\lceil |R_{t,z-1} \cup A'|/C \rceil \le \nu_{z-1} C$. This means that we can push all of orders in $A'$ to be delivered by vehicles at $T_{z-1}$. Thus, in the case, the output data $A = A'$ and $V_z = V'_z \backslash A$ make $\lceil |V_z|/C \rceil$ the minimal number of vehicles used at $T_z$. Under the case of $\lceil |R_{t,z-1} \cup A'|/C \rceil > \nu_{z-1} C$, Step 2(e) calls Subalgorithm CA to decide a set of orders $A$ such that $\lceil |V_z|/C \rceil$ is the minimal number of vehicles used at $T_z$, where $V_z = V'_z \backslash A$. We show below that Subalgorithm CA can really do that.

Subalgorithm CA consists of $k_0 + 1$ main iterations for the first $k_0 C + r$ orders, the first $(k_0 - 1)C + r$ orders, $\ldots$, the first $r$ orders from $V'_z$, respectively. We now run the first main iteration for the first $k_0 C + r$ orders $A'$ from $V'_z$. Due to the corresponding case by Step 2(e), we have $A' \neq \emptyset$ and $|R_{t,z-1} \cup A'| > \nu_{z-1} C$. We need to proceed through Step 4. Step 4 assigns the orders in $R_{t,h}$ to a temporary set $R'_{t,h}$ for each $h$ from 1 to $j - 2$. This is necessary since Subalgorithm CA operates on $R'_{t,h}$ without changing the content of $R_{t,h}$. Step 5 assigns the first $|R_{t,z-1} \cup A'| - \nu_{z-1} C$ orders from $R_{t,z-1} \cup A'$ to a set $B$. If we want to push the orders in $A'$ to be delivered by vehicles at $T_{z-1}, \ldots, T_1$, all orders of $B$ are the minimal increment undertaken by vehicles at $T_{z-2}, \ldots, T_1$ to deliver. Now we proceed through Step 6.

Step 6 consists of $z - 2$ secondary iterations. We now run the first secondary iteration. If $\sum_{J_i \in \bigcup_{m=1}^{z-2} R'_{t,m} \cup B} p_i > T_{z-2}$, it is impossible to deliver all orders in $B$ by the vehicles at $T_1, \ldots, T_{z-2}$. This means that the number of vehicles used at $T_z$ should to be at least $\lceil |V'_z|/C \rceil - \lceil |A'|/C \rceil + 1$. Step 6(b1) corresponds to the case where $|A'| < C$ and outputs $A = \emptyset$. This, along with $A' \neq \emptyset$ and $V_z = V'_z$, implies that $\lceil |V_z|/C \rceil$ is the minimal number of vehicles used at $T_z$. Step 6(b2) corresponds to the case where $|A'| \ge C$ and starts the second main iteration for the first $(k_0 - 1)C + r$ orders $A'$ from $V'_z$ to check whether it is possible to push the orders in $A'$ to be delivered by vehicles at $T_{z-1}, \ldots, T_1$. On the other hand, $\sum_{J_i \in \bigcup_{m=1}^{z-2} R'_{t,m} \cup B} p_i \le T_{z-2}$. Step 6(c) corresponds to the case where $|R'_{t,z-2} \cup B| \le \nu_{z-2} C$. This means that we can push all of orders in $A'$ to be delivered by vehicles at $T_{z-1}$ and $T_{z-2}$. Thus, the output data $A = A'$ by Step 6(c) and $V_z = V'_z \backslash A$ make $\lceil |V_z|/C \rceil$ the minimal number of vehicles used at $T_z$. Otherwise, Step 6(e) start the second secondary iteration to check whether it is possible to push the orders in a set of the first $|R'_{t,z-2} \cup B| - \nu_{z-2} C$ orders from $R'_{t,z-2} \cup B$ to be delivered by vehicles at $T_{z-3}, \ldots, T_1$. Similarly, we can show the result for the following secondary iterations. If need may be, we proceed through the last secondary iteration. Suppose that $B$ is a set of orders output by the last time secondary iteration. In

the case of $\sum_{J_i \in R'_{t,1} \cup B} p_i > T_1$, Step 6(b1) outputs $A = \emptyset$ if $|A'| < C$ and Step 6(b2) starts the second main iteration for the first $(k_0 - 1)C + r$ orders $A'$ from $V'_z$ else. In the case of $\sum_{J_i \in R'_{t,1} \cup B} p_i \le T_1$, Step 6(c) outputs data $A = A'$ if $|R'_{t,1} \cup B| \le \nu_1 C$. Otherwise, Step 6(d) outputs $A = \emptyset$ if $|A'| < C$ and starts the second main iteration for the first $(k_0 - 1)C + r$ orders $A'$ from $V'_z$ else, since it is impossible to deliver all orders in the set of the first $|R'_{t,1} \cup B| - \nu_1 C$ orders from $R'_{t,1} \cup B$ by the vehicles at $T_0 = 0$. Similarly, we can show the result for the following main iterations in Subalgorithm CA, which can really decide a set orders $A$ such that $\lceil |V_z|/C \rceil$ is the minimal number of vehicles used at $T_z$, where $V_z = V'_z \backslash A$. For the following iterations in Algorithm MV, we can show the result. At last, the algorithm must be able to find an optimal solution with the minimum number of vehicles used.

We now look at the time complexity of Algorithm MV. In the algorithm, Step 1 takes constant time. Step 2 is iterated $z$ times. Inside the iteration loop, the most time-consuming steps are 2(a) and 2(e). Step 2(a) calls for sorting the jobs which takes $O(n \log n)$ time. Step 2(e) calls Subalgorithm CA which takes $O(zn^2 \log n)$ time. Thus, the overall time complexity of Algorithm MV is $O(z^2 n^2 \log n)$. □

## 5 Conclusion

In this paper, we have given polynomial-time algorithms for minimizing: (1) the number of late orders, (2) the number of vehicles used subject to the condition that the number of late orders is minimum. An interesting open question is whether the problem related with release dates is NP-hard or not.

**Competing interests**
The authors declare that they have no competing interests.

**Authors' contributions**
All authors contributed equally to the manuscript and typed, read, and approved the final manuscript.

**Acknowledgements**
This work was supported in part by the Zhejiang Natural Science Foundation of China grant Y6110054. The authors wish to thank Weiya Zhong and Zhi-Long Chen for their technical advice, and C Cascante for his editorial work.

Received: 23 May 2014 Accepted: 16 September 2014 Published: 16 Oct 2014

**References**

1. Chen, Z-L: Integrated production and outbound distribution scheduling: review and extensions. Oper. Res. **58**, 130-148 (2010)
2. Fu, B, Hou, Y, Zhao, H: Coordinated scheduling of production and delivery with production window and delivery capacity constraints. Theor. Comput. Sci. **422**, 39-51 (2012)
3. Hall, NG, Lesaoana, M, Potts, CN: Scheduling with fixed delivery dates. Oper. Res. **49**, 134-144 (2001)
4. Lenstra, JK, Rinnooy Kan, AHG, Brucker, P: Complexity of machine scheduling problems. Ann. Discrete Math. **1**, 343-362 (1977)
5. Langley, CJ, van Dort, E, Sykes, SR: 2005 Third-Party Logistics: Results and Findings of the 10th Annual Study (2006)
6. Leung, JY-T, Chen, Z-L: Integrated production and distribution with fixed delivery departure dates. Oper. Res. Lett. **41**, 290-293 (2013)
7. Li, KP, Ganesan, VK, Sivakumar, AI: Synchronized scheduling of assembly and multi-destination air-transportation in a consumer electronics supply chain. Int. J. Prod. Res. **43**, 2671-2685 (2005)
8. Li, KP, Ganesan, VK, Sivakumar, AI: Scheduling of single stage assembly with air transportation in a consumer electronic supply chain. Comput. Ind. Eng. **51**, 264-278 (2006)
9. Li, KP, Sivakumar, AI, Ganesan, VK: Complexities and algorithms for synchronized scheduling of parallel machine assembly and air transportation in consumer electronic supply chain. Eur. J. Oper. Res. **187**, 442-455 (2008)
10. Melo, RA, Wolsey, LA: Optimizing production and transportation in a commit-to-delivery business mode. Eur. J. Oper. Res. **203**, 614-618 (2010)
11. Stecke, KE, Zhao, X: Production and transportation integration for a make-to-order manufacturing company with a commit-to-delivery business mode. Manuf. Serv. Oper. Manag. **9**, 206-224 (2007)
12. Wang, Q, Batta, R, Szczerba, RJ: Sequencing the processing of incoming mail to match an outbound truck delivery schedule. Comput. Oper. Res. **32**, 1777-1791 (2005)
13. Zandieh, M, Molla-Alizadeh-Zavardehi, S: Synchronizing production and air transportation scheduling using mathematical programming models. J. Comput. Appl. Math. **230**, 546-558 (2009)
14. Zhong, W, Chen, Z-L, Chen, M: Integrated production and distribution scheduling with committed delivery dates. Oper. Res. Lett. **38**, 133-138 (2010)

10.1186/1029-242X-2014-409
**Cite this article as:** Li and Li: Integrated production and distribution scheduling problems related with fixed delivery departure dates and number of late orders. *Journal of Inequalities and Applications* 2014, 2014:409