A capacitated vehicle routing problem with order available time in e-commerce industry

Ling Liu, Kunpeng Li and Zhixue Liu

School of Management, Huazhong University of Science & Technology, Wuhan, PR China

ABSTRACT

In this article, a variant of the well-known capacitated vehicle routing problem (CVRP) called the capacitated vehicle routing problem with order available time (CVRPOAT) is considered, which is observed in the operations of the current e-commerce industry. In this problem, the orders are not available for delivery at the beginning of the planning period. CVRPOAT takes all the assumptions of CVRP, except the order available time, which is determined by the precedent order picking and packing stage in the warehouse of the online grocer. The objective is to minimize the sum of vehicle completion times. An efficient tabu search algorithm is presented to tackle the problem. Moreover, a Lagrangian relaxation algorithm is developed to obtain the lower bounds of reasonably sized problems. Based on the test instances derived from benchmark data, the proposed tabu search algorithm is compared with a published related genetic algorithm, as well as the derived lower bounds. Also, the tabu search algorithm is compared with the current operation strategy of the online grocer. Computational results indicate that the gap between the lower bounds and the results of the tabu search algorithm is small and the tabu search algorithm is superior to the genetic algorithm. Moreover, the CVRPOAT formulation together with the tabu search algorithm performs much better than the current operation strategy of the online grocer.

ARTICLE HISTORY

Received 28 May 2015
Accepted 26 April 2016

KEYWORDS

Order available time; vehicle routing; tabu search; Lagrangian relaxation

1. Introduction

This problem is motivated by the challenges faced by one of the biggest Chinese electronic commerce (e-commerce) companies. Owing to the fierce competition in this rapidly growing industry, short delivery time is especially important for the online grocer to gain market share. Today, the e-commerce giants invest in building warehouses, acquiring vehicles and recruiting crews in many cities to improve their service levels. The online grocer receives orders from the internet. When orders are placed online, they are dispatched to the warehouse immediately. Actually, as the orders are placed online, every 30 orders are put into one batch. Subsequently, the items in this batch are picked from the warehouse by an order picker. After that, obtained items in this batch are sorted, so that items in each order are gathered together. Then, each order is packed for delivery.

The online grocer has a powerful information system. The completion time of order picking and order packing is recorded accurately in the system. The operation in the warehouse runs steadily.

CONTACT Kunpeng Li likp@mail.hust.edu.cn

Supplemental data for this article can be accessed at http://dx.doi.org/10.1080/0305215X.2016.1188092 and https://figshare.com/s/759a84d53b7ebf375fc8

© 2016 Informa UK Limited, trading as Taylor & Francis Group
without interruptions. Thus, the completion time of order packing can be predicted, which is further proved or adjusted when the order picking is finished. Accurate prediction of order available time is expected to be achieved by the big data analysis tools. This will further improve the efficiency of the operations of the online grocer, which helps to schedule the transportation and delivery more efficiently by perceiving the finishing time of the previous stage. Another motivation of this research is that this work can be part of the application of big data analysis in today’s e-commerce industry. Currently, orders are batched for delivery subject to the first-come-first-serve rule, then the vehicles deliver nearby orders first and faraway orders last. It is desirable to consider the integration of order batching and vehicle routing by taking into account the completion time of order picking and packing, i.e. order available time, to improve the operating efficiency of the online grocers.

The problem considered in this article is called the capacitated vehicle routing problem with order available time (CVRPOAT), which is a variant of the well-known capacitated vehicle routing problem (CVRP). Toth and Vigo (2002) defined that in the CVRP, all the customers correspond to deliveries and the demands are deterministic, are known in advance and may not be split. The vehicles are identical and based at a single central depot, and only the capacity restrictions for the vehicles are imposed. The objective of the problem in this article is to minimize the total time for all vehicle routes to be completed, taking into account the order available time. In CVRP, it is assumed that the products are available at the depot before distribution. According to the reality of the operation of the online grocer, CVRPOAT assumes that the orders are not available for delivery at the beginning of the planning period. The order available time for the transportation stage is determined by the preceding processing stage, such as the production or order picking and packing process.

CVRPOAT is also observed in the operations of make-to-order (MTO) systems, where a finished product inventory is undesirable. The finished products are delivered to customers immediately or within a very short waiting time after production. Typical MTO operations can be observed in the newspaper industry (Van Buer, Woodruff, and Olson 1999), ready-mix concrete industry (Garcia and Lozano 2004), food catering industry (Chen and Vairaktarakis 2005) and consumer electronics industry (Li, Ganesan, and Sivakumar 2005; Li, Ganesan, and Sivakumar 2006), etc. It is worth mentioning that the order picking and packing process in the operations of the online grocer can also be treated as the production process before transportation. In these industries, finished products are commonly delivered by trucks. Thus, decisions about the loading and routing of the vehicles are constrained by the production scheduling. Chen (2010) reviews the existing work on the integrated production and outbound distribution scheduling problem. According to existing work on integrated production and outbound distribution, the transportation stage is simplified to direct shipping in most published articles. If routing is considered, the integrated scheduling of production and transportation is very complex. Recently, a few articles have considered the routing decisions in the integrated problem. Cakici, Mason, and Kurz (2012), Ullrich (2013) and Low, Li, and Chang (2013) consider delivery time windows and take the related time optimization as the objective. Adulyasak, Cordeau, and Jans (2012) and Jha and Shanker (2013) take the total cost as the objective and consider inventory cost in the integrated problem. Chang, Chang, and Chang (2013, 2014) consider both total delivery time and total distribution cost in the objective function. Viergutz and Knust (2014) consider a single vehicle to serve a selection of customers such that the total satisfied demand is maximized. Lee et al. (2014) consider a nuclear medicine production and delivery problem, where the objective is to minimize the total cost including production costs, fixed vehicle costs and travel costs. Chen, Hsueh, and Chang (2009) and Belo-Filho, Amorim, and Almada-Lobo (2015) consider the integrated production and distribution problem of perishable products with a lifespan. Chen, Hsueh, and Chang (2009) aim to maximize the expected total profit of the supplier. The objective in Belo-Filho, Amorim, and Almada-Lobo (2015) is to minimize the sum of production and distribution costs. Cheng, Yang, and Hu (2016) take the service span as the objective function of a supply chain scheduling problem, to minimize the period from the time when the first batch starts processing to the time when the last
product reaches its customer. A few articles mentioned the routing decisions in the e-commerce environment, whereas they neglected the order picking and packing time. The reader is referred to Du, Li, and Chou (2005), Agatz et al. (2011), Ehmke and Mattfeld (2012), Teo, Taniguchi, and Qureshi (2012), etc. Only Archetti, Feillet, and Speranza (2015) study the vehicle routing problem with release dates, where the goods are not all available at the depot at the start of the distribution. They consider two special cases, where the graph describing the locations of the depot and the customers is a star or a line. A single vehicle and an unlimited fleet of vehicles with infinite capacity are considered in these two cases. Their aim is to show that the problems are polynomial solvable on these two special graphs. In this article, the general graph is studied and a limited fleet of capacitated vehicles is considered. Since the problem is NP hard, heuristics are developed to solve it.

Based on the above descriptions, CVRPOAT can be formulated as follows. \( n \) orders are placed by \( m \) customers. Each order has the order size and the available time, which can also be taken as the production completion time of the order. A fleet of homogeneous vehicles is available in the single depot or the production plant. A vehicle can only depart from the depot when the orders belonging to the vehicle are all loaded. Each vehicle can be used only once and must go back to the depot. The vehicle departure time is the maximum production completion time of the orders allocated to the vehicle. This time is defined as the vehicle release time. The objective is to minimize the total vehicle route time, which is the sum of the vehicle travel time and the vehicle release time. Recently, time minimization has been widely considered as the objective function in routing problems. Malandraki and Daskin (1992) and Dabia et al. (2013) aim to minimize total route time in the time-dependent vehicle routing problem. Avella, Boccia, and Vasilyev (2013) and Zachariadis, Tarantilis, and Kiranoudis (2013) also minimize the total vehicle travel time in vehicle routing problems with time windows. Hong and Park (1999) take the total vehicle travel time as the first objective and the customer waiting time as the second objective for solving bi-objective vehicle routing with time window constraints. Balseiro, Loiseau, and Ramonet (2011), Gong et al. (2012) and Ehmke et al. (2015) have the primary objective of minimizing the number of vehicles and the secondary objective of minimizing the total route travel time or total route duration time in vehicle routing problems with time windows. The objective of total route time minimization helps to evaluate the overall efficiency of the transportation facilities.

In this study, a granular tabu search (GTS) algorithm is proposed to solve CVRPOAT. To evaluate the performance of the proposed GTS algorithm, a Lagrangian relaxation (LR) algorithm is proposed to find a lower bound of CVRPOAT. Another lower bound based on the optimal solution of the benchmark CVRP instances is also derived. In computational experiments, a published genetic algorithm (GA) for a related problem is adapted to compare with the GTS algorithm. Test instances are derived from benchmark CVRP data.

The rest of this article is organized as follows. Section 2 defines CVRPOAT formally and introduces the mathematical formulation. A GTS algorithm is proposed in Section 3. Section 4 employs two algorithms to obtain the lower bound for the problem. Section 5 presents computational results, followed by the concluding Section 6.

### 2. Mathematical formulation

Let \( G = (V, E) \) be a complete and undirected graph, where \( V = \{0, 1, 2, \ldots, n\} \) is the set of vertices and \( E = \{(i, j), i, j \in V; i \neq j\} \) is the set of arcs. Vertex 0 represents the depot at which a fleet of homogeneous vehicles of capacity \( Q \) is located with the vehicle set \( H = 1, 2, \ldots, K \). The number of available vehicles is \( K \). Vertices set \( N = \{1, 2, \ldots, n\} \) corresponds to the customers. A non-negative travel time \( t_{ij} \) is associated with every arc \( e = (i, j) \). The real value corresponds to the travel time between vertices \( i \) and \( j \). Each order is associated with each customer \( i \) with the corresponding order size \( q_i \) and an available time \( r_i \), which represents the finishing time of the order picking and packing process. It is obvious that CVRPOAT with each customer having multiple orders can also be converted into CVRPOAT with each customer having one order. When a set of orders is loaded on to a vehicle,
the earliest time that the vehicle can depart is the largest order available time among the loaded orders. Thus, this earliest vehicle departure time is regarded as the route (vehicle) release time. CVRPOAT aims to assign the orders to each vehicle and find a set of routes to minimize the sum of the route times of all vehicles. The route time of each vehicle includes the vehicle release time and the vehicle travel time. Constraint conditions are as follows:

1. Each order is assigned to one route and each order cannot be split.
2. The available time of each customer’s order cannot be violated, i.e. the vehicle cannot depart before the available time of each onboard order.
3. Each route must begin and end at the depot, and visit at least one customer.
4. The total demand served by any vehicle does not exceed its capacity.

Before presenting the mathematical model, all variables, parameters and notations are introduced below.

| Variables | Description |
|-----------|-------------|
| $x_{ijk}$ | equals 1 if vehicle $k$ travels from vertex $i$ to vertex $j$ on arc $(i, j)$ and equals 0 otherwise, $\forall (i, j) \in E$. |
| $y_{ik}$ | equals 1 if vehicle $k$ loads the order belonging to customer $i$ and equals 0 otherwise, $\forall i \in N$. |
| $A^k_i$ | arrival time of vehicle $k$ at customer $i$. |

| Parameters and notations | Description |
|--------------------------|-------------|
| $i, j$ | index of the customer belonging to the customer set $N = \{1, 2, \ldots, n\}$. |
| $k$ | index of the vehicle belonging to the vehicle set $H = \{1, 2, \ldots, K\}$. |
| $t_{ij}$ | travel time between customers $i$ and $j$. |
| $q_i$ | size of the order belonging to customers $i$. |
| $r_i$ | available time of the order belonging to customers $i$. |

The mathematical model is presented as follows:

$$\text{Min} \sum_{k=0}^{K} \left( \sum_{i=0}^{n} \sum_{j=0}^{n} t_{ij}x_{ijk} + \max_{j \in N}(r_{j}y_{jk}) \right)$$

(1)

$$\sum_{k=0}^{K} \sum_{j=0}^{n} x_{ijk} = 1, \quad i = 1, \ldots, n$$

(2)

$$\sum_{i=0}^{n} x_{ihk} - \sum_{j=0}^{n} x_{hjk} = 0, \quad h = 1, 2, \ldots, n, \quad k = 1, 2, \ldots, K$$

(3)

$$\sum_{i=1}^{n} q_i \sum_{j=0}^{n} x_{ijk} \leq Q, \quad k = 1, \ldots, K$$

(4)

$$A^k_i + t_{ij} - A^k_j \leq (1 - x_{ijk})M, \quad i = 0, \ldots, n, \quad j = 0, \ldots, n, \quad k = 1, \ldots, K$$

(5)

$$A^k_0 \geq \max_{j \in N}(r_{j}y_{jk}), \quad k = 1, \ldots, K$$

(6)

$$y_{jk} = \sum_{i=0}^{n} x_{ijk}, \quad j = 1, \ldots, n, \quad k = 1, \ldots, K$$

(7)
\[ x_{ijk} \in \{0, 1\}, \quad i = 0, \ldots, n, \quad j = 0, \ldots, n, \quad k = 1, \ldots, K \]  
(8)

\[ y_{jk} \in \{0, 1\}, \quad j = 1, \ldots, n, \quad k = 1, \ldots, K \]  
(9)

The objective is to minimize the sum of the route time of the vehicles, which includes the total vehicle travel time and the total vehicle release time. Constraints (2) ensure that each order should be transported and delivered to its customer. Constraints (3) are the flow conservation constraints. Constraints (4) guarantee that the vehicle capacity will not be exceeded for each route. Constraints (5) are the sub-tour elimination constraints, where \( M \) is a large non-negative number. Constraints (6) ensure that each vehicle cannot depart before the largest order available time among the loaded orders. Constraints (7) denote that if a vehicle visits a customer, the order belonging to the customer must be loaded on the vehicle. Constraints (8) and (9) are integer conditions.

If the order available time is ignored, the problem is simplified to a CVRP. Since CVRP is generally an NP-hard problem, this problem is also an NP-hard problem.

3. A granular tabu search heuristic

A GTS heuristic is presented for CVRPOAT, which is based on the well-known tabu search algorithm. The GTS heuristic is introduced by Toth and Vigo (2003), and considers drastically restricted neighbourhoods (granular neighbourhoods) that just contain the moves being likely to generate good feasible solutions. Compared with traditional tabu search, the GTS searches a smaller neighbourhood to reduce the computing time of neighbourhood exploration within a local search. The GTS heuristic in this article is composed of the following stages and the major procedure is described in Figure 1:

- **Stage 1:** Construction of an initial solution. A greedy algorithm and a random algorithm are proposed.
- **Stage 2:** Granular neighbourhoods. The granular neighbourhoods are designed and a simple strategy to calculate the neighbours is proposed.
- **Stage 3:** The individual routes in the best neighbour solution are improved using a local optimization algorithm applied to each route.
- **Stage 4:** Tabu search phase. Tabu list, intensification and diversification are designed. In each iteration, the best neighbour solution is selected subject to the constraint of tabu list. Then, the selected neighbour solution is taken as the current solution. Subsequently, new neighbour solutions are generated based on the current solution. In addition, if the best solution is not updated for a predefined number of iterations, the current solution is replaced by a randomly generated solution.

3.1. Construction of an initial solution

The initial solution consists of several routes. Each route can be treated as the result of the travelling salesman problem (TSP). Two algorithms are adopted to generate a set of solutions. Then, the best one is selected as the initial solution of the GTS. For the generated initial solutions, one is obtained by a greedy algorithm and the remaining ones are generated by a random algorithm, with the details presented below.

3.1.1. Greedy algorithm

For the greedy algorithm, the customers are sequenced according to non-increasing order of order available time. Then, the customers are inserted into the route of each vehicle by a greedy method.
A pseudo-code for the greedy algorithm is given below:

**Procedure 1: Greedy algorithm**

Set \( C = \{1, 2, \ldots, n\} \), \( R = \{1, 2, \ldots, K\} \)

Sequence the customers in \( C \) according to non-increasing order available time

While \( C \) is not empty

Select the first customer \( f \) in \( C \)

Calculate the travel time \( t_{f, \text{last}} \) between customer \( f \) and the last customer in route \( r \in R \)

If (the total quantity of orders in a route \( r > \) the capacity \( Q \) of a vehicle)

The travel time \( t_{f, \text{last}} \) in the route is equal to \( M \) (\( M \) is a huge number)

Select a route which results in the minimum value of \( t_{f, \text{last}} \)

Append customer \( f \) to the end of the selected route

Delete customer \( f \) from the set \( C \)

Return each route

### 3.1.2. Random algorithm

For the random algorithm, the customers are randomly sequenced. Then, each customer is inserted into the best position among the routes of the partial solution, such that the resulting increase of the objective value is minimized. The customers are inserted one by one until all of them are assigned. When a customer is inserted into a route, if the capacity of the vehicle is exceeded, a large number is assigned to the objective value so that this insertion is not chosen.

### 3.2. Granular neighbourhoods

To reduce the computing time of neighbourhood exploration, the search of a granular neighbourhood considers only the moves that are generated by edges with a cost up to a certain threshold, as ‘long’ (i.e., high-cost) edges have a small probability of being part of high-quality solutions (Toth and Vigo 2003). Thus, a reduced graph \( G' \) is considered, in which only arcs \((i, j)\) linking customers having a cost \((t_{ij})\) smaller than the granularity threshold \( v \) are allowed, where \( v = \beta * (z/(n + K)) \). \( \beta \) is a suitable positive parameter which is set between 1.0 to 2.0. According to preliminary tests, the initial value of \( \beta \) is set to 1.0, which could be increased when no improved solution can be found for a certain time during execution. \( z \) is the value of the initial solution. \( n \) and \( K \) indicate the number of customers and vehicles, respectively.

The neighbourhoods are generally based on edge exchanges and customer movements, such as 1-0-exchange, 1-1-exchange, 1-1-arc-exchange and piece-exchange, which can be seen in Figure 2. Each move results in new arcs being generated. The dashed lines denote the new generated edges based on different moves in Figure 2. For CVRPOAT, when a move is applied to the current solution, if there is at least one new edge belonging to \( G' \), the move can be accepted and the neighbour solution can be generated. For a solution of CVRPOAT, a long edge (not belonging to \( G' \)) may reduce the time of a route, although it results in a long travel time. Then, the total objective value may be decreased. Therefore, when generating a neighbour, long edges are allowed to be introduced in a move, but moves involving only long edges are not considered.

The four neighbourhoods are briefly introduced as follows:

- **1-0-exchange**: removing one customer in a route and inserting it into the remaining routes
- **1-1-exchange**: exchanging two customers in any two different routes in the current solution
- **1-1-arc-exchange**: exchanging two edges (except for the arcs connected with the depot) in any two different routes in the current solution
- **piece-exchange**: swapping pieces of any two routes in the current solution. For these two routes, once a cutting point is specified in each route, the two pieces from the cutting point to the end of each route can be exchanged to generate one neighbour solution.
When each neighbour solution is generated, the feasibility is checked. If any route in a neighbour exceeds the vehicle capacity constraint, the neighbour is infeasible and a large penalty value is assigned to the objective value of the neighbour. Otherwise, a simple strategy is applied to calculate the objective value of the feasible neighbour, which just calculates the cost of the affected part of the current solution. When a neighbour solution is generated, only a limited part of the current solution structure is modified. This modified part consists of a set of the new generated arcs, which causes the
change in the objective function. For CVRPOAT, the cost of the affected part includes two aspects, which are route travel time and route release time. Thus, both the travel time of the affected arcs and the release time of the routes involved in these affected arcs need to be calculated. This new strategy for calculating the objective value of the generated neighbours can save computing time compared to recalculating the travel time of each arc and each route release time.

### 3.3. A local optimization algorithm for improving the best neighbour solution

As a neighbour solution consists of several routes, these routes can be improved by local search algorithms. A local optimization algorithm named the unstringing and stringing (US) algorithm is used to improve the quality of the generated neighbours. To save computing time, the local search algorithm is only applied to the best neighbour in each iteration of the GTS. The local search algorithm is proposed by Li et al. (2014) and its features are briefly described below.

First, only type 1 unstring step and type 1 string step in Gendreau, Hertz, and Laporte (1992) are employed. Secondly, the two nearest nodes within the $p$-neighbourhood of the insertion point are selected as the two adjacent nodes. Thirdly, all the possible sequences are enumerated and the best one is selected if the number of nodes in the route does not exceed 7. Then, if the customer number in the route is greater than or equal to 7, the adapted US algorithm will be used. Readers can refer to Gendreau, Hertz, and Laporte (1992) for details of the standard US algorithm.

### 3.4. Tabu search phase

Tabu list is an essential part of the GTS algorithm, storing two kinds of information. The first one is the two routes that are chosen to generate neighbourhood solutions. The other one is the cutting points in these two routes, where the pieces of these two routes are cut and exchanged. When a new neighbourhood solution is generated, it is checked according to the tabu list. First, the two routes are obtained based on which new neighbour is generated. If these two routes are recorded in the tabu list, the cutting points of these two routes are checked. If the cutting points of these two routes are also recorded in the tabu list, this new neighbourhood solution is not accepted.

Intensification and diversification are also two important rules. By intensification, the search focuses on promising portions of the solution space, while diversification moves the algorithm to other unexplored regions. If the current best solution is not improved for Non-Improvement iterations,
Procedure 2: Granular tabu search algorithm

Generate initial solution $s$
Initialize the tabu list
Set $Cur = s$, $Best = s$, $Div = 0$, Non-Improvement = 0, Is-Useful = false
While $Div < 50$
do
Generate neighbours of $Cur$ based on granular neighbourhoods
Select the best neighbour
Apply the US algorithm to improve the best neighbour
While Is-Useful = false do
If the selected neighbour is non-tabu then
If the selected neighbour is better than $Obj_Best$ then
Non-Improvement = 0, $Cur =$ the selected neighbour
$Best =$ the selected neighbour
Is-Useful = true, clear tabu list (intensification)
If the selected neighbour is not better than $Obj_Best$ then
$Cur =$ the selected neighbour, Is-Useful = true
Update the tabu list
Non-Improvement = Non-Improvement + 1
If Non-Improvement $> 25$ then
$Div = Div + 1$, Non-Improvement = 0
Clear tabu list
Generate another initial solution $s_1$ (diversification)
$Cur = s_1$
If the selected neighbour is tabu then
If the selected neighbour is better than $Obj_Best$ then
Non-Improvement = 0, $Cur =$ the selected neighbour
$Best =$ the selected neighbour
Is-Useful = true, clear tabu list (aspiration)
If the selected neighbour is no better than $Obj_Best$ then
The selected neighbour = the next neighbour, Is-Useful = false
Return the best solution $Obj_Best$.

the search process may have reached a local minimum. Thus, diversification is executed by starting the search with a randomly generated initial solution. Meanwhile, the tabu list is cleared.

Before presenting the detailed steps of the GTS algorithm, the notations are defined. $Cur$ indicates the current solution, $Best$ indicates the best solution, $Obj_Best$ denotes the value of the best solution, and $Div$ denotes the number of times that diversification is executed. Non-Improvement records the number of times the solution has not been improved. Is-Useful is a conditional judgement.

4. Lower bounds for CVRPOAT

To evaluate the performance of the proposed GTS algorithm, an LR algorithm is proposed to find the lower bound of the CVRPOAT. Meanwhile, another lower bound is derived based on the optimal solution of the benchmark CVRP instances. This lower bound can be used to evaluate the quality of the LR algorithm, as well as the proposed GTS.

4.1. Lagrangian relaxation algorithm to obtain the lower bound for CVRPOAT

To assess the performance of the proposed tabu search algorithm, a lower bound for CVRPOAT can be found by the LR algorithm. By relaxing some constraints, the original complex problem can be decomposed into several simple subproblems, which then can yield a lower bound of the optimal solution to the original problem (Jörnsten, Näsberg, and Smeds 1985; Guignard and Kim 1987). The foundations of the LR algorithm and related applications can be found in Fisher (1981, 1985).

The LR of CVRPOAT is considered with respect to the assignment constraints (2), by introducing a vector of Lagrangian multipliers $\alpha = (\alpha_1, \ldots, \alpha_n)$. $\alpha_i$ is associated with the $i$th constraint in (2).
Thus, the model of the relaxed Lagrangian problem can be presented as:

$$LR = \text{Min} \left\{ \sum_k \left( \sum_{i \in V} \sum_{j \in V} x_{ijk} t_{ij} + \max_{j \in N} (r_{jy}) \right) + \sum_{i \in V} \alpha_i \left( \sum_k \sum_{j \in V} x_{ijk} - 1 \right) \right\} \quad (10)$$

subject to Constraints (3)–(9).

### 4.1.1. Lagrangian relaxation subproblems

The LR algorithm decomposes the original problem into a number of subproblems with known usable structures. Normally, the resulting model decomposes the original problem into subproblems for each vehicle (Kohl and Madsen 1997). The following property of the relaxed model provides a powerful guarantee for decomposing the relaxed model.

**Property 1.** For the optimal solution of the relaxed problem, each vehicle’s route can be identical.

**Proof:** Assume that the routes are not identical in the optimal solution $S$ of the relaxed problem. Each customer $i$ corresponds to a Lagrangian multiplier $\alpha_i$, with $i = (1, 2, \ldots, n)$. The objective function can be written as:

$$LR = \text{Min} \left\{ \sum_{i \in V} \sum_{j \in V} x_{ijk} t_{ij} + \max_{j \in N} (r_{jy}) \right\} + \sum_{i \in V} \alpha_i \left( \sum_k \sum_{j \in V} x_{ijk} - \sum_{i \in V} \alpha_i \right) \quad (11)$$

$\sum_{i \in V} \sum_{j \in V} x_{ijk} t_{ij} + \max_{j \in N} (r_{jy})$ denotes the total route time of the $k$th route (the sum of travel time and the maximum onboard order available time); let $T_k = \sum_{i \in V} \sum_{j \in V} x_{ijk} t_{ij} + \max_{j \in N} (r_{jy})$; $\sum_{i \in V} \alpha_i \sum_{j \in V} x_{ijk}$ denotes the sum of Lagrangian multipliers, which belongs to customers in route $k$; let $S_k = \sum_{i \in V} \alpha_i \sum_{j \in V} x_{ijk}$. Therefore, Formula (11) could be written as:

$$LR = \sum_k T_k + \sum_k S_k - \sum_{i \in V} \alpha_i = \sum_k (T_k + S_k) - \sum_{i \in V} \alpha_i \quad (12)$$

Let $M_k = T_k + S_k$, then Formula (12) could be written as:

$$LR = \sum_k M_k - \sum_{i \in V} \alpha_i \quad (13)$$

The value of $\sum_{i \in V} \alpha_i$ is constant, since each Lagrangian multiplier can be calculated by Formula (17) in Section 4.1.3. According to Formula (13), the LR algorithm decomposes the original problem into $K$ identical subproblems for each vehicle. Thus, if all vehicles’ $M_k$ are equal and minimum, $LR$ in (13) is the minimum value. That is to say, when each vehicle’s route is identical and minimum, the solution of the relaxed problem is optimal. Thus, to obtain the optimal solution of the relaxed problem, each vehicle’s route should be identical. ■

According to Property 1, the route of each vehicle should have identical route time. Thus, the original problem can be decomposed into subproblems. Also, the variables $x_{ijk}$ and $y_{ik}$ can be replaced
The DP algorithm consists of the following steps.

subject to Constraints (3)–(9). Then, (14) can be rewritten as:

Each time a customer \( j \) is inserted into the current route, both the route travel time and the route release time need to be updated. Generally, the route release time would be increased or unchanged when adding customer \( j \). Let \( R_{\text{current}} \) represent the release time of the current route. When \( x_{ij} \) is equal to 1, and if the available time of the order belonging to customer \( j \) is larger than \( R_{\text{current}} \), the release time of the route is increased; otherwise, the release time is unchanged. Thus, (15) can be rewritten as:

Let \( c_{ij} = t_{ij} + \alpha_i + \max(0, r_j - R_{\text{current}}) \), which now denotes the travel time from node \( i \) to node \( j \) in the relaxed problem. Since \( K \) and \( \alpha_i \) are constants, each subproblem can be treated as a shortest path problem with capacity constraints (Handler and Zang 1980; Desrosiers et al. 1995).

4.1.2. Solving the subproblem to obtain the lower bound

A labelling algorithm based on the dynamic programming (DP) algorithm is proposed to solve the subproblem. The labelling algorithm extends the dynamic algorithm in Desrosiers et al. (1995) to avoid cyclic routes in the network. Similarly to Desrosiers et al. (1995), two items are added to a label in the labelling algorithm. One is the onboard quantity of a vehicle when it arrives at a customer site. The other is the immediate predecessor of the customer and the route index from the depot to the immediate predecessor.

For each node \( i \), there is a set of labels. Each label can be denoted by \( \text{EFF}(T^L_i, \text{current Quantity}(i, L), \text{Front}(\hat{i}, L^\prime)) \). \( T^L_i \), and current Quantity\((i, L)\) indicates the travel time and onboard quantity of the \( L \)th path from \( o \) to \( i \), respectively. On the \( L \)th path, \( \hat{i} \) denotes the immediate predecessor of \( i \) and \( L^\prime \) is the corresponding route index from \( o \) to \( \hat{i} \). \( Q_i \) is the set of labels for node \( i \). \( P_i \) refers to the set of labels exploited in the DP procedure. \( N \) denotes the set of nodes. \( o \) and \( d \) denote the depot that the vehicle departs from and returns to, respectively. \( V \) denotes nodes including \( N \) and \( d \). \( \Gamma(i) \) denotes successors of node \( i \). To avoid cyclic routes, a node included in the route from \( o \) to \( i \) cannot be included in \( \Gamma(i) \).

Similarly to Desrosiers et al. (1995), among the set of labels of node \( i \), the condition is defined that one label dominates another label. Let \( \text{label1} = (T^{L1}_i, \text{current Quantity}(i, L1), \text{Front}(\hat{i}^1, L1^\prime)) \) and \( \text{label2} = (T^{L2}_i, \text{current Quantity}(i, L2), \text{Front}(\hat{i}^2, L2^\prime)) \).

**Definition:** if \( T^{L1}_i < T^{L2}_i \) and \( \text{current Quantity}(i, L1) < \text{current Quantity}(i, L2) \), \( (T^{L1}_i, \text{current Quantity}(i, L1), \text{Front}(\hat{i}^1, L1^\prime)) \) dominates \( (T^{L2}_i, \text{current Quantity}(i, L2), \text{Front}(\hat{i}^2, L2^\prime)) \)

That is to say, only both the travel time and onboard quantity of label1 are less than those of label2, therefore label1 is superior to label2.

\( \text{EFF}(T^L_i, \text{current Quantity}(i, L), \text{Front}(\hat{i}, L^\prime)) \) denotes the set of non-dominated labels of node \( i \). The DP algorithm consists of the following steps.

\[ LR = \min \left( K \left( \sum_{i \in V} \sum_{j \in V} x_{ij} t_{ij} + \max(r_{jy_j}) \right) + K \sum_{i \in V} \alpha_i \sum_{j \in V} x_{ij} - \sum_{i \in V} \alpha_i \right) \]
1. Initialization
   \[ Q_0 = \{(T_0^1 = 0, \text{current Quantity}(0, 1), \text{Front}(0, 1) = (0, 0))\}; \]
   \[ Q_i = \emptyset, \forall i \in N \cup \{d\}; P_i = \emptyset, \forall i \in V. \]

2. Selection of node \( i^* \)
   From the set \( U_i \in (Q_i \setminus P_i), \forall i \in N \cup \{o\}, \) select the label
   \( (T_{i^*}^L, \text{current Quantity}(i^*, L^*), \text{Front}(i^*, L^*)) \)
   with the minimum value of \( \text{current Quantity}(i^*, L^*). \)

   If \( U_i = \emptyset, \forall i \in N \cup \{o\}, \) stop and go to step 4; Otherwise, go to the next step.

3. Treatment of label \( (T_{i^*}^L, \text{current Quantity}(i^*, L^*), \text{Front}(i^*, L^*)) \) for all \( j \in \Gamma(i^*); \)
   \[ Q_j = \text{EFF}(f_{r^*j}(T_{i^*}^L), \text{current Quantity}(i^*, L^*), \text{Front}(i^*, L^*)) \cup Q_j; \]
   \[ P_i = P_i \cup \{(T_{i^*}^L, \text{current Quantity}(i^*, L^*), \text{Front}(i^*, L^*))\}; \]

   note that \( f_{r^*j}(T_{i^*}^L, \text{current Quantity}(i^*, L^*), \text{Front}(i^*, L^*)) \) is the treatment of the labels of node \( j, \) which is the successor of node \( i^*; \)

   \( f_{r^*j}(T_{i^*}^L, \text{current Quantity}(i^*, L^*), \text{Front}(i^*, L^*)) = (T_j^L, \text{current Quantity}(j, L), \)
   \[ \text{Front}(j^*, L^*) \) with current Quantity \( (j, L) \leq \text{Vehicle Capacity}, T_j^L = T_{i^*}^L + c_{r^*j}, \] and
   \[ \text{Front}(j^*, L^*) = (i^*, L^*), \text{current Quantity}(j, L) = \text{current Quantity}(i^*, L^*) + D_j. \] \( D_j \) denotes the demand quantity of customer \( j; \)

   If \( \text{current Quantity}(j, L) > \text{Vehicle Capacity}, f_{r^*j}(T_{i^*}^L, \text{current Quantity}(i^*, L^*), \text{Front}(i^*, L^*)) \) is empty.

   When the treatment is executed, return to step 2.

4. Find the shortest path
   From all the labels of \( \{d\}, \) select the label \( T_d^L \) with the minimum value of \( T_d^L. \)

   Then, find the shortest path based on this label.

4.1.3. Subgradient method

The subgradient method is used to find the appropriate multipliers for the LR technique (Held, Wolfe, and Crowder 1974; Kohl and Madsen 1997). A sequence of Lagrangian multipliers is generated by the formula:

\[
\alpha_{i+1} = \alpha_i + \frac{\mu_n \ast (UB - LB)}{2 \left( K \sum_{j=0}^{n} x_{ij} - 1 \right)} \]

(17)

The notations in (17) are presented as follows. The initial multipliers can be set to 0. \( \mu_n \) is a scalar satisfying \( 0 < \mu_n \leq 2. \) UB is the upper bound obtained by the tabu search algorithm to the original problem. LB is the lower bound obtained by the DP algorithm. The initial value of \( \mu_n \) is set to 2. Each time when the lower bound is not improved, \( \mu_n = \mu_n / 2. \) The iteration of the LR algorithm is set to 1000. Through preliminary experiments, it is observed that it is not always necessary for the iteration to be 1000. In some cases, LR cannot be increased when \( \mu_n \) is reduced to certain values. Thus, if LR stops increasing for both a value of \( \mu_n \) and the following \( \mu_n / 2, \) the algorithm is terminated to save computing time.

According to the experiments, the convergence of the subgradient method is fast at the beginning, whereas the convergence is slow afterwards. The convergence progress is consistent with that of Kohl and Madsen (1997).

4.2. Another lower bound derived based on the optimal solution of the CVRP benchmark

To evaluate the quality of the LR algorithm, another lower bound of CVRPOAT is derived based on the optimal solution of the original CVRP benchmark instances, which are introduced in Section
5. The objective of the original CVRP is to minimize the total vehicle travel time. It is obvious that the optimal solution of each CVRP instance can be treated as the lower bound of the corresponding derived CVRPOAT instance. If the route release time based on the CVRP optimal solution is taken into account, the lower bound should be closer to the optimal solution of CVRPOAT.

To reduce the value of the sum of vehicle release times, first, all orders are sequenced in non-increasing order of available time of each order. Then, the orders are loaded on to vehicles from the first order of the sequence, until the remaining capacity of the current vehicle is not sufficient to load the next order in the sequence. To ensure that the lower bound can be obtained, it is assumed that current vehicle still can load the next order even if the capacity of the vehicle is exceeded. Otherwise, if the current vehicle has remaining capacity, it cannot ensure that the lower bound of the sum of vehicle release times is obtained. The reason for sequencing the orders in the non-increasing order of available time of each order is that all the orders with larger available time are loaded on to one vehicle, so that the remaining vehicles can load orders with small available time.

Thus, the lower bound of each CVRPOAT instance can be obtained by the summation of this minimum total vehicle release time and the optimal solution of the corresponding CVRP instance. This lower bound method is denoted by LB_CVRP. Note that the Lagrangian relaxation algorithm is denoted by LB_LR in the following sections.

5. Computational results

In this section, CPLEX is applied to validate the mathematical model. A published GA (Vallada and Ruiz 2011) is adapted to evaluate the performance of the proposed GTS algorithm. Two lower bounds LB_LR and LB_CVRP are also derived to evaluate the performance of the proposed tabu search algorithm. All algorithms are coded in C++. The computational experiments are conducted on a personal computer with an Intel Pentium dual-core 2.8 GHz CPU and 2 GB RAM.

5.1. Instance generation

To assess the performance of the proposed algorithms, some test instances are generated. Since there are no benchmark data available for the problem in this article, two sets of test instances are constructed. The first set of data is generated based on the well-known CVRP benchmarks with series A, B and P, including 72 instances. CVRP benchmark instances are available on the website www.branchandcut.org. Customer locations, customer demands and the number of vehicles are given by the benchmark data. There are 18–100 customers and two to 15 vehicles in each instance. The second set of data consists of small-sized instances, which are generated based on five to 10 customers (nodes) and two vehicles. The capacity of each vehicle is 20. The demands of each customer/order are generated randomly in [1, 10] and the total demands of all customers in each instance do not exceed 40. The coordinates of each customer are generated randomly in [0, 50].

The travel distance between the two nodes in each instance is taken as the travel time between the corresponding nodes in CVRPOAT. For each customer in each test instance, the order available time is generated randomly from the range [1/8, 1/4]*T/K, where T is the travel time of the giant route which visits all customers from index 1 to n in each instance, and K is the number of vehicles.

5.2. Computational results for small-sized instances

Table 1 provides the solutions obtained by CPLEX and the GTS algorithm for small-sized instances. The column ‘Case’ represents the instance name, where n is the number of customers and k is the number of vehicles. CPLEX is used to validate the mathematical model. The results show that the optimal solutions for small-sized test instances could be obtained both by CPLEX and GTS, and the average running time of the GTS is less than that of CPLEX for all instances.
Note that Constraints (6) are nonlinear constraints in the original mathematical model, which is presented in Section 3. Thus, to solve the problem conveniently by CPLEX, a new variable $F_{jk}$ is introduced, where $F_{jk} = r_j y_{jk}$. Then, Constraints (10) are replaced by two linear constraints of $r_j \leq F_{jk} + M(1 - y_{jk})$ and $0 \leq F_{jk} + My_{jk}$, where $M$ is a huge number, for all $j$ and $k$.

### 5.3. Computational results for larger sized instances

To assess the quality of the proposed GTS algorithm, a GA proposed by Vallada and Ruiz (2011) is introduced. The GA is presented for an unrelated parallel machine scheduling problem, which is quite similar to the present problem. Each machine can be treated as a vehicle. Jobs on each machine correspond to the orders loaded on to each vehicle. In Vallada and Ruiz (2011), the crossover operation only allows the job insertion in the same array of the selected chromosome and the corresponding offspring. For modification in this article, jobs can be interchanged or inserted between different arrays of the selected chromosome and the corresponding offspring. This modification expands the search space, increasing the possibility of obtaining better solutions. The population size of the GA is set to 80. The Pressure used in the selection operator is set to 10. The probabilities of crossover and mutation are each set to 0.5. The probability of local search is set to 1.

Table 2 provides the average results of the instance groups A, B and P, which are obtained by the algorithms GTS, GA, LB_LR and LB_CVRP for CVRPOAT. The column ‘Group’ gives the instance group name. Columns ‘Z1’ and ‘Z2’ give the average results of CVRPOAT by GTS and GA. Columns ‘T1’ and ‘T2’ are the average running times of GTS and GA. Columns ‘LB1’ and ‘LB2’ are the average lower bounds of CVRPOAT by LB_LR and LB_CVRP. Column ‘Gap1’ indicates the average percentage gap between the GA results and GTS results, which is calculated by $\text{Gap1} = 100\% \times \left(\frac{\text{GA} - \text{GTS}}{\text{GA}}\right)$. ‘Gap2’ indicates the average percentage gap between the GTS results and the lower bounds obtained by LB_LR, which is calculated by $\text{Gap2} = 100\% \times \left(\frac{\text{GTS} - \text{LB1}}{\text{GTS}}\right)$. ‘Gap3’ indicates the average percentage gap between the LB_LR results and LB_CVRP results, which is calculated by $\text{Gap3} = 100\% \times \left(\frac{\text{LB1} - \text{LB2}}{\text{LB1}}\right)$.

It can be seen from Table 2 that for the two lower bound algorithms LB_LR and LB_CVRP, the average gaps between LB_LR and the LB_CVRP are 2.20%, 0.21% and 1.14% for instance groups A, B and P, respectively. It is clear that LB_LR is better than LB_CVRP based on the instances derived from the benchmark CVRP instances. This indicates that LB_LR is able to generate tight lower bounds of

### Table 1. Computational results of CPLEX and granular tabu search (GTS) algorithm.

| Case      | Objective | Time (s) | Objective | Time (s) | Optimal? |
|-----------|-----------|----------|-----------|----------|----------|
| Small-n5-k2 | 126       | 3        | 126       | 2        | Yes      |
| Small-n6-k2 | 184       | 3        | 184       | 2        | Yes      |
| Small-n7-k2 | 196       | 6        | 196       | 2        | Yes      |
| Small-n8-k2 | 248       | 6        | 248       | 2        | Yes      |
| Small-n9-k2 | 288       | 10       | 288       | 2        | Yes      |
| Small-n10-k2 | 277      | 25       | 277       | 2        | Yes      |

Note: CPLEX = Integer Linear Programming; GTS = Granular Tabu Search; Optimal? = Whether the solution is optimal.

### Table 2. Computational results of four algorithms for the capacitated vehicle routing problem with order available time (CVRPOAT) (A, B, P).

| Group | GTS | GA | LB_LR | LB_CVRP | Gap1 | Gap2 | Gap3 |
|-------|-----|----|-------|---------|------|------|------|
| Z1    |    |    |       |         |      |      |      |
| Z2    |    |    |       |         |      |      |      |
| T1    |    |    |       |         |      |      |      |
| T2    |    |    |       |         |      |      |      |
| LB1   |    |    |       |         |      |      |      |
| LB2   |    |    |       |         |      |      |      |
| Gap1  |    |    |       |         |      |      |      |
| Gap2  |    |    |       |         |      |      |      |
| Gap3  |    |    |       |         |      |      |      |

Note: GA = genetic algorithm; LB = lower bound; LR = Lagrangian relaxation; CVRP = capacitated vehicle routing problem.
**Table 3.** Computational results of the operation method for the capacitated vehicle routing problem with order available time (CVRPOAT) (A, B, P).

| Group | GTS Z1 | T1 (s) | OM Z3 | T3 (s) | Gap (Z3 – Z1)/Z3 × 100% |
|-------|-------|-------|-------|-------|------------------------|
| A     | 1659  | 7     | 2325  | 2     | 39.86                  |
| B     | 1561  | 8     | 2279  | 2     | 46.53                  |
| P     | 853   | 10    | 1235  | 2     | 39.52                  |

Note: GTS = granular tabu search; OM = operating method.

CVRPOAT for the general instances. The average gaps between GTS and LB_LR are 5.85%, 7.93% and 5.59% for instance groups A, B and P, respectively. It is clear that GTS can provide near-optimal solutions for CVRPOAT.

According to the results in Table 2, GTS is better than GA in terms of the results of CVRPOAT and the running time. The average percentage gaps between the GA and the GTS are 2.13%, 1.10% and 1.83% for instance groups A, B and P, respectively. For test instances in set A and B, the average running time of the GTS is almost half that of the GA. For test instances in set P, the average running time of the GTS is similar to that of the GA.

Detailed computational results for CVRPOAT (A,B,P) are available in four tables in an online supplement to this article (see supplemental data).

### 5.4. Comparing granular tabu search with the current operating method in the industry

Currently, the online grocer runs the warehouse and the delivery vehicles based on simple rules. The received orders are batched randomly for order picking. When the order picking and final packing are finished, the orders are loaded on to the waiting vehicle for delivery according to the first-come-first-serve rule. As it is hard to obtain real data, the generated CVRPOAT instances are used to simulate the operation of the real practice by applying the operating method of the online grocer. Orders with close picking finishing times, i.e., available time, are loaded on to the same vehicle. The local search method in Section 3.3 is applied to generate the routing of each vehicle, which should be better than the manually determined routing in practice.

Table 3 gives the results of the instance groups A, B and P, which are obtained by applying the operating method in practice. The average gap between the current operating method and GTS is approximately 40% for all instances. This indicates that the proposed GTS approach has the potential to make a significant improvement to the efficiency of the operation strategy in the real world.

### 6. Conclusion

This article conducted investigations on a new variant of the CVRP, taking the order available time into account. This problem is observed in the operations of the current e-commerce industry and the context of integrated scheduling of production and transportation. Considering the special constraints of order available time in this problem, the objective is to minimize the sum of the route time of each vehicle. For this NP-hard problem, a GTS algorithm is proposed, which uses drastically restricted neighbourhoods and employs a simple strategy to evaluate the neighbour solutions. Then, an adapted US algorithm is developed to improve the quality of the best neighbour solution in each iteration of the tabu search. To evaluate the performance of the proposed tabu search, the LR algorithm is employed to obtain the lower bound of the problem, which decomposes the relaxed problem by exploiting its properties. To evaluate the quality of the LR algorithm, another lower bound is derived based on the optimal solutions of benchmark instances. A published GA is employed for comparison with the GTS algorithm. Finally, the GTS algorithm is compared with the current operation strategy of the online grocer. Computational results show that the LR algorithm can provide tight
lower bounds for the problem. The GTS algorithm can provide solutions close to the lower bounds. Also, the GTS algorithm is better than the GA and the current operation strategy in practice.

In the future, an exact algorithm such as the branch-and-cut algorithm should be developed for this problem. In addition, this problem should be integrated into the production stage preceding distribution, where an integrated scheduling method should be addressed.

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

This work was supported by National Natural Science Foundation of China [grant number 71372133], [grant number 71131004].

References

Adulyasak, Y., J. F. Cordeau, and R. Jans. 2012. “Optimization-based adaptive large neighbourhood search for the production routing problem.” Transportation Science 48 (1): 20–45.
Agatz, N., A. Campbell, M. Fleischmann, and M. Savelsbergh. 2011. “Time slot management in attended home delivery.” Transportation Science 45 (3): 435–449.
Archetti, C., D. Feillet, and M. G. Speranza. 2015. “Complexity of routing problems with release dates.” European Journal of Operational Research 247 (3): 797–803.
Avella, P., M. Boccia, and I. Vasilyev. 2013. “Lifted and local reachability cuts for the vehicle routing problem with time windows.” Computers & Operations Research 40 (8): 2004–2010.
Balseiro, S. R., I. Loiseau, and J. Ramonet. 2011. “An ant colony algorithm hybridized with insertion heuristics for the time dependent vehicle routing problem with time windows.” Computers & Operations Research 38 (6): 954–966.
Belo-Filho, M. A. F., P. Amorim, and B. Almada-Lobo. 2015. “An adaptive large neighbourhood search for the operational integrated production problem and distribution problem of perishable products.” International Journal of Production Research 53 (20): 6040–6058.
Cakici, E., S. J. Mason, and M. E. Kurz. 2012. “Multi-objective analysis of an integrated supply chain scheduling problem.” International Journal of Production Research 50 (10): 2624–2638.
Chang, Y. C., K. H. Chang, and T. K. Chang. 2013. “Applied column generation-based approach to solve supply chain scheduling problems.” International Journal of Production Research 51 (13): 4070–4086.
Chang, Y. C., V. C. Li, and C. J. Chiang. 2014. “An ant colony optimization heuristic for an integrated production and distribution scheduling problem.” Engineering Optimization 46 (4): 503–520.
Chen, Z. L. 2010. “Integrated production and outbound distribution scheduling: review and extensions.” Operations Research 58 (1): 130–148.
Chen, H. K., C. F. Hsueh, and M. S. Chang. 2009. “Production scheduling and vehicle routing with time windows for perishable food products.” Computers & Operations Research 36 (7): 2311–2319.
Chen, Z. L., and G. L. Vairaktarakis. 2005. “Integrated scheduling of production and distribution operations.” Management Science 51 (4): 614–628.
Cheng, B., Y. Yang, and X. Hu. 2016. “Supply chain scheduling with batching, production, and distribution.” International Journal of Computer Integrated Manufacturing 29 (3): 251–262.
Du, T. C., E. Y. Li, and D. Chou. 2005. “Dynamic vehicle routing for online B2C delivery.” Omega 33 (1): 33–45.
Desrosiers, J., Y. Dumas, M. M. Solomon, and F. Soumis. 1995. “Time constraints routing and scheduling.” In Network Routing, Handbooks in Operations Research and Management Science 8, edited by M. O. Ball, T. L. Magnanti, C. L. Monma, and G. L. Nemhauser, 35–139. Amsterdam: North Holland.
Dabia, S., S. Ropke, T. Van Woensel, and T. De Kok. 2013. “Branch and price for the time-dependent vehicle routing problem with time windows.” Transportation Science 47 (3): 380–396.
Ehmke, J. F., A. M. Campbell, and T. L. Urban. 2015. “Ensuring service levels in routing problems with time windows and stochastic travel times.” European Journal of Operational Research 240: 539–550.
Ehmke, J. F., and D. C. Mattfeld. 2012. “Vehicle routing for attended home delivery in city logistics.” Procedia—Social and Behavioral Sciences 39: 622–632.
Fisher, M. L. 1981. “The lagrangian relaxation method for solving integer programming problems.” Management Science 27 (1): 1–18.
Fisher, M. L. 1985. “An applications oriented guide to lagrangian relaxation.” Interfaces 15 (2): 10–21.
Garcia, J. M., and S. Lozano. 2004. “Production and vehicle scheduling for ready-mix operations.” Computers & Industrial Engineering 46 (4): 803–816.
Gendreau, M., A. Hertz, and G. Laporte. 1992. “New insertion and post-optimization procedures for the traveling salesman problem.” Operations Research 40 (6): 1086–1094.

Gong, Y. J., J. Zhang, O. Liu, R. Z. Huang, S. H. Chung, and Y. H. Shi. 2012. “Optimizing the Vehicle Routing Problem with Time Windows: A Discrete Particle Swarm Optimization Approach.” IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 42 (2): 254–267.

Guignard, M., and S. Kim. 1987. “Lagrangean Decomposition: A model yielding stronger lagrangean bounds.” Mathematical Programming 39: 215–228.

Handler, G. Y., and I. Zang. 1980. “A dual algorithm for the constrained shortest path problem.” Networks 10 (4): 293–309.

Held, M., P. Wolfe, and H. P. Crowder. 1974. “Validation of subgradient optimization.” Mathematical Programming 6 (1): 62–88.

Hong, S. C., and Y. B. Park. 1999. “A heuristic for bi-objective vehicle routing with time window constraints.” International Journal of Production Economics 62 (3): 249–258.

Jörnsten, K., M. Näsberg, and P. Smeds. 1985. “Variable splitting—A new lagrangean approach to some mathematical programming models.” Report LITHMAT-85-04. Department of Mathematics, Linköping Institute of Technology, Linköping, Sweden.

Jha, J. K., and K. Shanker. 2013. “A coordinated two-phase approach for operational decisions with vehicle routing in a single-vendor multi-buyer system.” International Journal of Production Research 51 (5): 1426–1450.

Kohl, N., and O. B. G. Madsen. 1997. “An optimization algorithm for the vehicle routing problem with time windows based on lagrangian relaxation.” Operations Research 45 (3): 395–406.

Lee, J., B. I. Kim, A. L. Johnson, and K. Lee. 2014. “The nuclear medicine production and delivery problem.” European Journal of Operational Research 236 (2): 461–472.

Li, K., B. Chen, A. I. Sivakumar, and Y. Wu. 2014. “An inventory-routing problem with the objective of travel time minimization.” European Journal of Operational Research 236 (3): 936–945.

Li, K. P., V. K. Ganesan, and A. I. Sivakumar. 2005. “Synchronized scheduling of assembly and multi-destination air-transportation in a consumer electronics supply chain.” International Journal of Production Research 43 (13): 2671–2685.

Li, K. P., V. K. Ganesan, and A. I. Sivakumar. 2006. “Scheduling of single stage assembly with air transportation in a consumer electronic supply chain.” Computers & Industrial Engineering 51 (2): 264–278.

Low, C., R. K. Li, and C. M. Chang. 2013. “Integrated scheduling of production and delivery with time windows.” International Journal of Production Research 51 (3): 897–909.

Malandraki, C., and M. S. Daskin. 1992. “Time dependent vehicle routing problems: formulations, properties and heuristic algorithms.” Transportation Science 26 (3): 185–200.

Teo, J. S., E. Taniguchi, and A. G. Qureshi. 2012. “Evaluating city logistics measure in e-commerce with multiagent systems.” Procedia—Social and Behavioral Sciences 39: 349–359.

Toth, P., and D. M. Vigo. 2002. The Vehicle Routing Problem. Philadelphia, PA: Society for Industrial and Applied Mathematics.

Toth, P., and D. Vigo. 2003. “The Granular Tabu Search and Its Application to the Vehicle-Routing Problem.” INFORMS Journal on Computing 15 (4): 333–346.

Ullrich, C. A. 2013. “Integrated machine scheduling and vehicle routing with time windows.” European Journal of Operational Research 227 (1): 152–165.

Vallada, E., and R. Ruiz. 2011. “A genetic algorithm for the unrelated parallel machine scheduling problem with sequence dependent setup times.” European Journal of Operational Research 211 (3): 612–622.

Van Buer, M. G., D. L. Woodruff, and R. T. Olson. 1999. “Solving the medium newspaper production/distribution problem.” European Journal of Operational Research 115 (2): 237–253.

Viergutz, C., and S Knust. 2014. “Integrated production and distribution scheduling with lifespan constraints.” Annals of Operations Research 213 (1): 293–318.

Zachariadis, E. E., C. D. Tarantilis, and C. T. Kiranoudis. 2013. “Designing vehicle routes for a mix of different request types, under time windows and loading constraints.” European Journal of Operational Research 229 (2): 303–317.