

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/322409794>

Formulations and Branch-and-Cut Algorithms for Production Routing Problems with Time Windows

Article in *Transportmetrica A: Transport Science* · January 2018

DOI: 10.1080/23249935.2018.1427157

CITATIONS

5

READS

142

5 authors, including:



Yuzhuo Qiu

Nanjing University of Finance and Economics

19 PUBLICATIONS 233 CITATIONS

[SEE PROFILE](#)



P. M. Pardalos

University of Florida

373 PUBLICATIONS 8,703 CITATIONS

[SEE PROFILE](#)



Boris Goldengorin

University of Baltimore

146 PUBLICATIONS 1,109 CITATIONS

[SEE PROFILE](#)

Some of the authors of this publication are also working on these related projects:



Submodular optimization, binary search and truncation theorem will be able to change the models and algorithms in machine learning. [View project](#)



Tolerance Based Algorithms in Combinatorial Optimization [View project](#)



Formulations and branch-and-cut algorithms for production routing problems with time windows

Yuzhuo Qiu, Liang Wang, Xuanjing Fang, Panos M. Pardalos & Boris Goldengorin

To cite this article: Yuzhuo Qiu, Liang Wang, Xuanjing Fang, Panos M. Pardalos & Boris Goldengorin (2018) Formulations and branch-and-cut algorithms for production routing problems with time windows, *Transportmetrica A: Transport Science*, 14:8, 669-690, DOI: 10.1080/23249935.2018.1427157

To link to this article: <https://doi.org/10.1080/23249935.2018.1427157>



Accepted author version posted online: 11 Jan 2018.
Published online: 26 Jan 2018.



Submit your article to this journal [↗](#)




Article views: 77



View Crossmark data [↗](#)



Formulations and branch-and-cut algorithms for production routing problems with time windows

Yuzhuo Qiu ^{a,b}, Liang Wang^a, Xuanjing Fang^a, Panos M. Pardalos^b and Boris Goldengorin^c

^aJiangsu Key Laboratory of Modern Logistics, School of Marketing and Logistics Management, Nanjing University of Finance and Economics, Nanjing, People's Republic of China; ^bDepartment of Industrial and Systems Engineering, Faculty of Engineering, University of Florida, Gainesville, FL, USA; ^cDepartment of Industrial and Systems Engineering, The Russ College of Engineering and Technology, Ohio University, Athens, OH, USA

ABSTRACT

This paper presents a mixed integer optimization framework for incorporating time windows into production routing problems. This problem is a generalization of vehicle routing, inventory routing, and lot-sizing problems, and formulated as a mixed interlinear programming problem. An exact method within a branch-and-cut framework is developed to solve the model. Several families of valid cuts are adapted and a hybrid heuristic to obtain a good upper bound is also developed. The newly proposed (I, S) inequalities link production variables with inventory variables. From the computational results, the effectiveness of the valid inequalities is proved. The newly proposed (I, S) inequalities outperform previously related inequalities. The numerical results for the case study also show that the total cost results in an 11.6% decrease over a heuristic solution after applying the proposed model and algorithm.

ARTICLE HISTORY

Received 24 March 2017
Accepted 9 January 2018



KEYWORDS

Vehicle routing;
inventory/production;
deterministic; integer
programming; algorithms;
cutting plane

1. Introduction

Many real-world applications have widely shown that using computerized solution methods for the solution of vehicle routing problem (VRP) yields substantial savings in the global transportation costs (Toth and Daniele 2014; Chow and Nurumbetova 2015; You, Chow, and Ritchie 2016). Production routing problem (PRP) is a variant of VRP. This problem aims at optimizing decisions of production, inventory, distribution, and routing simultaneously (Adulyasak, Cordeau, and Jans 2015b). Solving such integrated optimization problems can contribute to finding better solutions in supply chain management, specifically in vendor managed inventory (VMI) and the practice of just-in-time (JIT) production and logistics.

In the past decade, the PRP was extended in various ways, such as multiple plants and a heterogeneous fleet of vehicles (Lei et al. 2006), uncapacitated production (Archetti et al. 2011), multiple homogeneous capacitated vehicles (Adulyasak, Cordeau, and Jans 2014b), and under demand uncertainty (Adulyasak, Cordeau, and Jans 2015a).

CONTACT Yuzhuo Qiu  jadeqyz@gmail.com  Jiangsu Key Laboratory of Modern Logistics, School of Marketing and Logistics Management, Nanjing University of Finance and Economics, Nanjing 210023, People's Republic of China

Incorporation of delivery or supply time windows within the PRP is one of the future research directions (Adulyasak, Cordeau, and Jans 2015b; Díaz-Madroñero, Peidro, and Mula 2015). To the best of our knowledge, little effort has been made in this direction. Our aim is to bridge this gap by incorporating the still-missing time windows in production routing problems.

There has been extensive research in the PRP with heuristics: approximation algorithms (Chandra and Fisher 1994), decoupled heuristics (Fumero and Vercellis 1999), greedy randomized adaptive search procedures (Boudia, Louly, and Prins 2007), memetic algorithms (Boudia and Prins 2009), tabu search (Bard and Nananukul 2009b; Armentano, Shiguemoto, and Lokketangen 2011), adaptive large neighborhood heuristics Adulyasak, Cordeau, and Jans (2014a), iterative mixed integer programming (Absi et al. 2015), and particle swarm optimization (Kumar et al. 2016). Research on the exact algorithms has only been addressed in a few publications. For example, branch-and-price (Bard and Nananukul 2009a, 2010; Qiu, Qiao, and Pardalos 2017) and branch-and-cut algorithms for single product PRPs were investigated recently (Archetti et al. 2011; Adulyasak, Cordeau, and Jans 2014b).

The PRP involves combinatorial optimization of both delivery and routing decisions. Current branch-and-cut algorithms for multi-vehicle PRPs can only solve the instances up to 50 customers, 3 periods, and 3 vehicles in 2 hours by using parallel computing (Adulyasak, Cordeau, and Jans 2014b). However, supply chain participants often face with problems with longer planning periods or more vehicles. An exact algorithm to solve larger instances is still necessary, especially when practical concerns such as time windows are further incorporated into the original PRP.

A limitation in previous exact algorithms for the PRP is the lack of lot sizing problems (LSP)-related valid inequalities. The LSP is a major component of the PRP (Adulyasak, Cordeau, and Jans 2015b). LSP-related valid inequalities are thus expected to enhance the performance of exact methods to solve the PRP. In this paper, we aim to strengthen the formulation of the PRP with time windows (PRPTW) by these valid inequalities.

The contributions of the paper are summarized as follows. First, we introduce a real-world variant of the PRP which addresses customer delivery windows. Second, we formulate the PRPTW as a mixed integer linear programming (MILP) problem. Third, we introduce four families of valid inequalities to tighten the MILP problem and design a branch-and-cut algorithm to find an optimal solution. Moreover, the proposed (I, S) inequalities outperform previous related valid cuts by linking inventory variables with production variables. Our computational results with the branch-and-cut algorithm show that the proposed valid inequalities can substantially improve the quality of lower bounds. The computational result from case study shows a more than 10% decrease in total costs. Our PRPTW model, algorithm, and computational results can serve as a stepping stone for further research of PRPs with startup times, change-over times, and other extensions (Adulyasak, Cordeau, and Jans 2015b).

The remainder of this paper is organized as follows. In Section 2, we provide a formal description of the problem. Several families of valid inequalities are then introduced in Section 3. Based on these valid inequalities and MILP formulation, we devise a branch-and-cut algorithm in Section 4. We present computational results of our algorithm in Sections 5 and 6. Conclusions and discussions are presented in Section 7.

2. Problem description and model formulations

We now introduce the notations used in our models. The MILP formulation with the Miller–Tucker–Zemlin (MTZ)-like constraints is then proposed, which is solvable for small- and medium-scale instances. For large-scale instances, another strengthened formulation with generalized capacity constraints and path inequalities is presented and used in the proposed branch-and-cut algorithm.

2.1. Notations and problem description

We define the PRPTW on a complete directed graph $G = (N_0, A)$ with sets of nodes $N_0 = N \cup \{0\}$ and arcs $A = \{(i, j) : i, j \in N_0, i \neq j\}$, respectively. The node set $N_0 = N \cup \{0\}$ contains a set $N = \{1, 2, \dots, n\}$ of customers (retailers) and the plant (depot) represented by node 0. Under limited production capacity C , a commodity is produced at the plant with a fixed cost of f and a unit cost of u , and shipped to the retailers by a fleet of identical vehicles over a finite set T of time periods.

In every period t , the service for each customer i should start within a given time window $[a_{it}, b_{it}]$. The vehicle must arrive at customer i no later than b_{it} . If the vehicle arrives earlier than a_{it} , the vehicle must wait until the earliest starting time for service. We assume that the vehicle traverses each arc with an average speed of \mathcal{V} , hence it takes time c_{ij}/\mathcal{V} to transport over an arc (i, j) of length c_{ij} . Assume it takes time γ_i to serve a customer i , it thus takes at least time $\tau_{ij} = c_{ij}/\mathcal{V} + \gamma_i$ to serve customer i and reach the next customer j . When we formulate the MILP model, we assume the transport cost over an arc (i, j) of length c_{ij} is also c_{ij} to keep the formulation simple. Incorporation of a unit transport cost per unit length is straightforward and will not change the formulation much. The average speed of \mathcal{V} is also left out in the following formulation for the same simplicity reasons.

The inventory capacities L_i are imposed at the end of a period, after a delivery is made from the plant to a customer and the customer's demand is satisfied. Given the initial inventory levels and the unit inventory holding costs h_i of the product at node i , the PRPTW is to determine the production, shipment, and inventory quantities as well as the set of routes in each period to minimize the total production, inventory, and routing costs and meet the customers' demand within time windows requirements.

Since the Vehicle Routing Problem with Time Windows (VRPTW) (Savelsbergh 1985) is \mathcal{NP} -hard, and the PRPTW reduces to multi-period VRP when production variables are constants, and delivery quantities are known beforehand, one may conclude that the PRPTW is also \mathcal{NP} -hard. Note that finding a feasible solution to the VRPTW for a fixed number of vehicles is an \mathcal{NP} -complete problem. Thus finding a feasible solution to the PRPTW is also an \mathcal{NP} -complete problem.

Notations for production, inventory, and routing parameters and decision variables are as follows.

Notations and Parameters

f	fixed production setup cost
u	unit production cost
h_i	unit inventory holding cost at node i
c_{ij}	transportation cost from node i to node j

γ_i	service time at node i
d_{it}	demand of customer (retailer) i in period t
a_{it}	opening time, i.e. the earliest possible starting time for servicing customer i
b_{it}	closing time, i.e. the latest possible time for servicing customer i
M_{ijt}	$\max\{b_{it} + \gamma_i + \tau_{ij} - a_{jt}, 0\}$
B_t	$= \min\{C, \sum_{k=t}^{ T } \sum_{i \in N} d_{ik}\}$
E_{it}	$= \min\{L_i, \sum_{k=t}^{ T } d_{ik}\}$
$ T $	number of time periods
Q	vehicle capacity
K	number of available vehicles
L_i	maximum or target inventory level at node i
C	production capacity
$\delta_t^+(S)$	set of arcs leaving from the node set S in period t , $\delta_t^+(S) = \{(i, j) \in A : i \in S, j \notin S\}$, and $\delta_t^+(i) := \delta_t^+(\{i\})$
$\delta_t^-(S)$	set of arcs entering the node set S in period t , $\delta_t^-(S) = \{(i, j) \in A : i \notin S, j \in S\}$, and $\delta_t^-(i) := \delta_t^-(\{i\})$
\mathcal{Z}^+	set of positive integers
P	set of paths
P_G	set of all minimal infeasible paths in G
p	path $p = (v_1, v_2, \dots, v_l), p \in P$
$N(p)$	node set of path $p = (v_1, v_2, \dots, v_l)$
$ N(p) = l$	number of nodes on path $p = (v_1, v_2, \dots, v_l)$
$A(p)$	arc set of path $p = (v_1, v_2, \dots, v_l)$
$ A(p) = l - 1$	number of arcs on path $p = (v_1, v_2, \dots, v_l)$.

Decision Variables

q_t	production quantity in period t
l_{it}	inventory at node i at the end of period t
r_{it}	shipment quantity to customer (retailer) i in period t
x_{ijt}	binary variable if arc (i, j) is traversed in period t , 0 otherwise
y_t	binary variable, equal to 1 if the product is set up for production in period t , 0 otherwise
z_{it}	integer variable if vertex i is visited in period t , 0 otherwise
v_{it}	total quantity carried on the vehicle at retailer i in period t on a feasible route
W_{it}	start of service time at retailer i in period t on a feasible route.

2.2. A MILP formulation for the PRPTW: formulation I

Let $x_t(S) = \sum_{i \in S, j \in S} x_{ijt}$, $r_t(S) = \sum_{i \in S} r_{it}$ and $z_t(S) = \sum_{i \in S} z_{it}$, using the notations given in Section 2.1, we formulate the PRPTW as follows:

$$\min \sum_{t \in T} \left(uq_t + fy_t + \sum_{i \in N_0} h_i l_{it} \right) + \sum_{t \in T} \sum_{(i, j) \in A} c_{ij} x_{ijt} \quad (1)$$

subject to

$$l_{0,t-1} + q_t = l_{0t} + \sum_{i \in N} r_{it}, \quad \forall t \in T, \quad (2)$$

$$l_{i,t-1} + r_{it} = l_{it} + d_{it}, \quad \forall i \in N, t \in T, \quad (3)$$

$$q_t \leq B_t y_t, \quad \forall t \in T, \quad (4)$$

$$l_{it} \leq L_i, \quad \forall i \in N_0, t \in T, \quad (5)$$

$$r_{it} \leq E_{it} z_{it}, \quad \forall i \in N, t \in T, \quad (6)$$

$$\sum_{j \in N_0} x_{ijt} = z_{it}, \quad \forall i \in N_0, t \in T, \quad (7)$$

$$\sum_{j \in N_0} x_{ijt} = \sum_{j \in N_0} x_{jit}, \quad \forall i \in N_0, t \in T, \quad (8)$$

$$v_{it} - v_{jt} + Qx_{ijt} \leq Q - r_{jt}, \quad \forall i, j \in N, t \in T, \quad (9)$$

$$W_{it} - W_{jt} + M_{ijt}x_{ijt} \leq M_{ijt} - \tau_{ij}, \quad \forall i, j \in N, t \in T, \quad (10)$$

$$q_t \geq 0, \quad y_t \in \{0, 1\}, \quad \forall t \in T, \quad (11)$$

$$l_{it} \geq 0, \quad r_{it} \geq 0, \quad \forall i \in N_0, t \in T, \quad (12)$$

$$z_{it} \in \{0, 1\}, \quad \forall i \in N, t \in T, \quad (13)$$

$$z_{0t} \in \mathcal{Z}^+, \quad z_{0t} \leq K, \quad \forall t \in T, \quad (14)$$

$$x_{ijt} \in \{0, 1\}, \quad \forall i, j \in N_0, t \in T. \quad (15)$$

$$r_{it} \leq v_{it} \leq Q, \quad \forall i \in N, t \in T, \quad (16)$$

$$0 \leq W_{it} \leq b_{it}, \quad \forall i \in N, t \in T \quad (17)$$

The objective function (1) measures the operational cost of production, inventory and routing. Constraints (2) and (3) enforce inventory flow balance at the plant and retailer for two sequential time periods $t-1$ and t , respectively. Constraints (4) ensure that the binary

variables of setup $y_t = 1$ if production takes place. These constraints limit the production quantity to the minimum of production capacity and the total demand in the remaining time periods. Constraints (5) impose inventory limits both at the plant and the retailers. Constraints (6) permit a positive delivery quantity to node i in period t if this node is visited in period t . Constraints (7) ensure $z_{it} > 0$ if a truck leaves from retailer or depot i . Constraints (8) enforce vehicle flow conservation. Constraints (9) are MTZ constraints of vehicle capacity for the PRPTW. The MTZ formulation for the TSP reduces the number of constraints at the cost of weaker lower bounds than subtour elimination constraints (Miller, Tucker, and Zemel 1960). Constraints (10) are MTZ-like constraints of time windows for the PRPTW. Note that in 1-hour CPU time, the largest PRPTW instance based on the Formulation I and solved by the proposed branch-and-cut algorithm is 25 customers and 3 time periods. Hence, it is necessary to strengthen the formulation and develop a corresponding branch-and-cut algorithm.

2.3. A strengthened MILP formulation for the PRPTW: formulation II

As indicated in Section 2.2, the lower bound obtained by MTZ formulation for the PRPTW is rather weak. To further strengthen the formulation, we replace the MTZ-like constraints (9) and (10) with the following constraints.

$$Qx_t(S) \leq Qz_t(S) - r_t(S), \quad \forall S \subseteq N, |S| \geq 2, t \in T, \quad (18)$$

$$\sum_{(i,j) \in A(p)} x_{ijt} \leq |A(p)| - 1, \quad \forall p \in P_G, t \in T. \quad (19)$$

Constraints (18) serve as the subtour eliminations and vehicle capacity constraints. Constraints (19) are known as the incompatible path inequalities or path inequalities to forbid the generation of paths that violate time windows and vehicle capacity requirements. In the following branch-and-cut algorithm, these constraints are first relaxed from the linear programming (LP) relaxation and then added back as cutting planes.

The strengthened PRPTW formulation (1)–(8), (11)–(15), (18) and (19) (Formulation II) now serves as the basis for solving large size instances with the proposed branch-and-cut algorithm.

3. Valid inequalities

In this section, we introduce several families of valid inequalities to strengthen the LP relaxation of the PRPTW formulation (1)–(15). The proof of these valid inequalities is left in the online supplement. For the sake of brevity, we use similar notations to describe the inequalities related to the VRPTW by dropping the subscript t of the PRPTW notations and describe the inequalities related to the LSP by dropping the subscript i of the PRPTW notations.

3.1. Strengthened reachability inequalities

The Reachability Inequalities (RIs) are rather efficient for the VRPTW (Lysgaard 2006). For any node $i \in N$, let R_i^- and R_i^+ denote the reaching and reachable set of arcs for node i , respectively.

Definition 3.1 (Lysgaard 2006): For any customer $i \in N$, the reaching arc set $R_i^- \subset A$ is defined as the minimum arc set such that any feasible path $(0, \dots, i)$ lies inside R_i^- , i.e. $(j, k) \in A$ for each arc (j, k) on the feasible path.

Definition 3.2 (Lysgaard 2006): For any customer $i \in N$, the reachable arc set $R_i^+ \subset A$ is defined as the minimum arc set such that any feasible path $(i, \dots, 0)$ lies inside R_i^+ , i.e. $(j, k) \in A$ for each arc (j, k) on the feasible path.

For any $S \subseteq N$ and $i \in S$, the following RIs

$$x(\delta^-(S) \cap R_i^-) \geq 1 \quad (20)$$

and

$$x(\delta^+(S) \cap R_i^+) \geq 1 \quad (21)$$

are valid for the VRPTW (Lysgaard 2006). We extend the above definitions and valid inequalities to a multi-period version as follows.

Definition 3.3: For any customer $i \in N$, the reaching arc set $R_{it}^- \subset A_t$ is defined as the minimum arc set such that any feasible path $(0, \dots, i)$ lies inside R_{it}^- , i.e. $(j, k) \in A_t$ for each arc (j, k) on the feasible path.

Definition 3.4 (Lysgaard 2006): For any customer $i \in N$, the reachable arc set $R_{it}^+ \subset A_t$ is defined as the minimum arc set such that any feasible path $(i, \dots, 0)$ lies inside R_{it}^+ , i.e. $(j, k) \in A_t$ for each arc (j, k) on the feasible path.

Moreover, we tighten the generalized RIs for the PRPTW as follows.

Proposition 3.5: Let R_{it}^- and R_{it}^+ be the reaching and reachable set of arcs to node i at period t , respectively. The following strengthened reachability inequalities

$$Qx_t(\delta_t^-(i) \cap R_{it}^-) \geq r_{it}, \quad \forall i \in S, S \subseteq N, t \in T \quad (22)$$

and

$$Qx_t(\delta_t^+(i) \cap R_{it}^+) \geq r_{it}, \quad \forall i \in S, S \subseteq N, t \in T \quad (23)$$

are valid for the PRPTW.

3.2. Strengthened (I, S) inequalities

LSPs are also essential to the PRPTW. When there is only one supplier but no retailers, and when costs are Wagner–Whitin costs, i.e. $u_t + h_t > u_{t+1}$, the following (I, S) inequalities are valid for the LSP (Pochet and Wolsey 2006)

$$I_{t-1} \geq \sum_{j=t}^{\eta} d_j(1 - y_t - y_{t+1} - \dots - y_j), \quad \forall \eta : t < \eta \leq |T|, t \in T \quad (24)$$

where d_j is the demand for period j at the supplier, y_j is the binary set-up variable for period j . Note that in our model, $u_t = u_{t+1} = u$. Therefore, our model features with Wagner–Whitin

costs. In the PRPTW, we must also deal with the demand and inventories at the retailers. Thus, inspired by the aggregated subtour elimination constraints (Adulyasak, Cordeau, and Jans 2014a), we have the following proposition.

Proposition 3.6: *The following strengthened (I, S) inequalities*

$$\sum_{i \in N_0} l_{i,t-1} + \sum_{j=t+1}^{\eta} \sum_{k=j}^{\eta} \sum_{i \in N} d_{ik} y_j \geq \sum_{j=t}^{\eta} \sum_{i \in N} d_{ij} (1 - y_t), \quad \forall \eta : t < \eta \leq |T|, t \in T \quad (25)$$

are valid for the PRPTW.

Note that the proposed (I, S) inequalities are different from the following inventory-related valid inequalities (Archetti et al. 2011)

$$\left(\sum_{j=0}^s \delta_{i,t-j} \right) \left(1 - \sum_{k=0}^s z_{i,t-k} \right) \leq l_{i,t-s-1}^d, \quad \forall i \in N, t \in T, 0 \leq s \leq t-1, \quad (26)$$

where constraints (26) guarantee that in each time instant, product inventory is enough to cover accumulated delivery demand. The difference lies in that the proposed (I, S) inequalities (25) link inventory and production variables, whereas constraints (26) only deal with inventory variables.

3.3. Logical inequalities

The formulations (1)–(8), (11)–(15), (18) and (19) can be strengthened by adding the following valid inequalities, known as logical inequalities (LIs):

$$z_{it} \leq z_{0t}, \quad \forall i \in N, t \in T, \quad (27a)$$

$$z_{it} \leq r_{it}, \quad \forall i \in N, t \in T, \quad (27b)$$

$$l_{i|T|} = 0, \quad \forall i \in N_0. \quad (27c)$$

The inequalities (27a) and (27c) have appeared in the PRP literature (Archetti et al. 2007; Adulyasak, Cordeau, and Jans 2014a).

The inequalities (27b) we propose for the PRPTW ensure that the number of visits to node i equals zero if no delivery occurs at node i . This is not guaranteed by the original formulation but implied by an optimal solution if the triangular inequality $c_{ik} \leq c_{ij} + c_{jk}$, $\forall i, j, k \in N_0$ holds strictly. When the triangular inequality does not hold strictly, we can not assert (27b) hold because visiting customer j without demand does not incur extra transportation cost, if $c_{ik} = c_{ij} + c_{jk}$. Visiting customer j does not form a detour in this case. Our computational study shows that adding (27b) makes the lower bounds tighter in the instances for which triangular inequality holds strictly.

In addition, valid inequalities preventing stockout are usually added prior to solving the problem. Denote by t' the earliest time period that a production must be made, we know that $t' = \arg \min_{t \in T} \{ \sum_{i \in N} \max\{0, \sum_{j=1}^t d_{ij} - l_{i0}\} - l_{00} > 0 \}$. Then, denote by t'' the earliest

time period such that at least one customer must be replenished to avoid stockout, we know that $t'' = \arg \min_{i \in N} t'_i$, with $t'_i = \arg \min_{t \in T} \{\sum_{j=1}^t d_{ij} - l_{i0} > 0\}$. Finally, denote by $\kappa = \sum_{i \in N} \max\{0, \sum_{j=1}^{t''} d_{ij} - l_{i0}\}$ the minimum shipping quantity to prevent stockout. We have the following valid inequalities (Archetti et al. 2007, 2011; Adulyasak, Cordeau, and Jans 2014b)

$$\sum_{t=1}^{t'} y_t \geq 1, \quad (28a)$$

$$\sum_{t=1}^{t''} z_{0t} \geq \left\lceil \frac{\kappa}{Q} \right\rceil, \quad (28b)$$

for the PRPTW.

We also propose the following logical valid inequalities

$$\sum_{t=1}^{t'_i} z_{it} \geq 1, \quad \forall i \in N \quad (29a)$$

$$\sum_{t=1}^{t'_i} r_{it} \geq \sum_{j=1}^{t'_i} d_{ij} - l_{i0}, \quad \forall i \in N \quad (29b)$$

The above constraints are largely ignored in the PRP literature. Our computational study indicates that these inequalities can tighten the lower bounds.

3.4. VRPTW-related inequalities

We propose the following generalized Asymmetric-Traveling-Salesman-Problem-with-Time-Windows (ATSPTW) inequalities for the PRPTW

$$\begin{aligned} Q \left(x_{v_k v_1 t} + \sum_{h=2}^k x_{v_{h-1} v_h t} + 2 \sum_{h=2}^{k-1} x_{v_h v_1 t} + \sum_{h=3}^{k-1} \sum_{j=2}^{h-1} x_{v_j v_h t} \right) \\ \leq Qz_t(N(p)) - r_t(N(p)), \quad \forall t \in T \end{aligned} \quad (30)$$

and

$$\begin{aligned} Q \left(x_{v_1 v_k t} + \sum_{h=2}^k x_{v_h v_{h-1} t} + 2 \sum_{h=2}^{k-1} x_{v_1 v_h t} + \sum_{h=3}^{k-1} \sum_{j=2}^{h-1} x_{v_j v_h t} \right) \\ \leq Qz_t(N(p)) - r_t(N(p)), \quad \forall t \in T \end{aligned} \quad (31)$$

where $z_t(N(p)) = \sum_{v_i \in p} z_{v_i t}$, $r_t(N(p)) = \sum_{v_i \in p} r_{v_i t}$ and $k \geq 3$.

We also introduce generalized tournament inequalities (GTIs) for the PRPTW as follows:

$$\sum_{i=1}^{k-1} \sum_{j=i+1}^k x_{v_i v_j t} \leq k - 2 (= |A(p)| - 1), \quad \forall t \in T \quad (32)$$

4. Branch-and-cut algorithm

Since the VRP and the LSP within the PRPTW usually comprise of exponential numbers of subtour elimination constraints, it is natural to resort to dynamic column or row generation. In this section by using the MILP formulation and valid inequalities described in previous sections, we devise a branch-and-cut algorithm to solve the PRPTW exactly. A description of our branch-and-cut algorithm is as follows.

4.1. Linear programming relaxation and a priori cutting planes

The linear programming relaxation of the MILP formulation without the generalized capacity and path constraints is solved at each node of the branch-and-cut tree. Except for experiments to test the effectiveness of the valid inequalities, logical valid inequalities are added a priori to the linear programming relaxation of the MILP formulation. Note that the number of LSP-related valid inequalities usually increases exponentially with time periods. Thus adding LSP-related inequalities a priori will increase computational complexity when the number of time periods is relatively large.

4.2. Separation procedures

We adapt the well-known connected component-based separation procedures (Naddef and Rinaldi 2001; Bard, Kontoravdis, and Yu 2002; Kallehauge, Boland, and Madsen 2007) for GSECs (18). The auxiliary graph for each period is defined by arcs with non-zero values. Besides capacity inequalities, we also adapt the separation heuristic (Pochet and Wolsey 2006) for generalized (I, S) inequalities, heuristics (Fischetti and Toth 1997) for ATSPTW inequalities, and separation techniques (Bard, Kontoravdis, and Yu 2002; Lysgaard 2006) for other VRPTW inequalities.

In all the adaptation, customers with positive values of variable z_{it} are included in and those with zero values of variable z_{it} are excluded from the resulting networks in the corresponding period t .

4.3. Separation strategy

We treat the root node of the branching tree differently from the other nodes (Lysgaard, Letchford, and Eglese 2004). If the optimal LP solution at the root node is fractional, we call the above-mentioned separation procedures for generalized capacity inequalities (18), the generalized (I, S) inequalities (25), ATSPTW inequalities, and other VRPTW inequalities to search for violated cuts by the LP optimal solution. If at least one such cut is found, we add the violated constraints and the LP is reoptimized. The optimization and separation cycle of valid constraints iterates until no more violated inequalities are found. At other nodes, rather than the root node, if the solution is fractional, we branch. If the solution at the root node and other nodes is integer, we use the lazy constraints call-back (defined in C++ API of IBM ILOG CPLEX) containing the generalized capacity inequalities (12), the generalized tournament inequalities (32), and strengthened reachability inequalities (22) and (23) to check for violations (capacity and/or time windows).

4.4. Branching strategy

Three sets of binary or integer variables can be prioritized to branch in the PRPTW. We test different orders of variables to branch on. Computational experiments show that priority

should be given first to z_{it} , then y_t , and finally x_{ijt} variables. An effective strategy of branching on the number of vehicles $z_{0t} = \sum_{j \in N_0} x_{0jt}$ in the VRP literature (Naddef and Rinaldi 2001; Lysgaard, Letchford, and Eglese 2004) corresponds to branching on z_{0t} variables in the PRPTW, and means branching on cut-set. Hence, we always branch on z_{0t} first before we branch on other z_{it} , $i > 0$. Our computational study shows that this special branching order works much better than the branching order of y_t , z_{it} , and x_{ijt} variables (Adulyasak, Cordeau, and Jans 2014a).

Our branch-and-cut algorithm uses a mixed strategy of most infeasibility and best-bound-first as the variable-and-node-selection rule (Lysgaard, Letchford, and Eglese 2004). The variable closest to 0.5 and the node with the largest lower bound are always processed first. This strategy combines the heuristic consideration and the fact that best-bound-first search leads to the smallest possible branch-and-cut tree (Naddef and Rinaldi 2001; Lysgaard, Letchford, and Eglese 2004). Note that default strong branching strategy (Achterberg, Koch, and Martin 2005) in CPLEX is also switched on.

4.5. Heuristics

To speed up the solution process, we implement a hybrid heuristic method. The heuristic combines the ideas of large neighborhood search, local search, tabu search, and simulated annealing. We outline the heuristic in Algorithm 1.

The heuristic contains two phases, namely, construction and improvement. In the first phase, we construct initial solutions for the PRPTW by solving two decomposed problems of production–distribution and the VRPTW (Adulyasak, Cordeau, and Jans 2014b). The formulation is similar. However, the calculation of visiting cost is modified. In the previous study (Adulyasak, Cordeau, and Jans 2014b),

$$\sigma_i = \min \left(2c_{0i}, \min_{j, k \in N_0, j \neq k} (c_{ji} + c_{ik}) \right).$$

We modify the approximate cost of visiting customer i as

$$\sigma_i = \min \left(2c_{0i}, \min_{j, k \in N, j \neq k} 0.5(c_{ji} + c_{ik}) \right).$$

The total visiting cost is probably closer to the actual routing cost with our modification. The reason is that, what we need in the formulation of the production–distribution subproblem is an estimate of the routing cost. We divide the estimated routing cost and attribute the cost to each customer to get the ‘visiting cost’. Without the multiplication by 0.5, we will approximately double the routing cost because each possible arc would be counted twice (once for each of its endpoints) in the final possible routes.

We use the Solomon insertion heuristic (Solomon 1987) to quickly obtain a solution of the VRPTW subproblem. We fix the shipment quantity variables r_{it} and site-visiting indicator variables z_{it} according to the solution of the solved production–distribution subproblem which could be formulated as follows:

$$\min \sum_{t \in T} \left(uq_t + fy_t + \sum_{i \in N_0} h_i l_{it} \right) + \sum_{t \in T} \sum_{i \in N} \sigma_i z_{it} \quad (33)$$

Algorithm 1: Hybrid Heuristic

 $s^{best} \leftarrow \emptyset;$
1. Construction phase:

Initialize: Calculate approximate site cost;

Solve the production–distribution subproblem;

Solve the VRPTW subproblem with Solomon insertion heuristic;

Collect initial solution s^{feas} ;**2. Improvement Phase:**Set $s^{best} \leftarrow s^{feas}$; Set $iterations \leftarrow 0$, $iterUnsuccess \leftarrow 0$ and $time \leftarrow 0$; $maxIterations = \theta \times n \times T \times K, \theta = 50$; $maxUnsuccess = \eta \times n \times T \times K, \eta = 0.01$; $maxTime = 300s$, for easy instances; $maxTime = 600s$, otherwise; Initialize operator weights and temperature $T_0 = 0.3$;**while** $iterations \leq maxIterations$ and $iterUnsuccess \leq maxUnsuccess$ and $time \leq maxTime$ **do**

Select one-candidate-move, two-candidate-move, three-candidate-move, whole-route-move, or random recombination probabilistically as in the adaptive large neighborhood search (Adulyasak, Cordeau, and Jans 2014b);

Perform the shift of delivery for each candidate accordingly (Armentano, Shiguemoto, and Lokketangen 2011);

Put the move into tabu list, the tabu tenure is set to the planning horizon;

Recalculate the inventories for the candidates;

Perform Solomon insertion heuristic on the new delivery schedule;

Perform 2-opt*, 2-opt, and Or-opt local search to find a better VRPTW solution;

Solve production and depot inventory problem;

Collect solution s^{curr} ;**if** $f(s^{curr}) < f(s^{best})$ or s^{curr} is accepted by the simulated annealing criterion **then**
 $s^{best} = s^{curr}$;**end**

Update operator weights;

 $iterations \leftarrow iterations + 1$;**end**Output: s^{best}

subject to (2)–(6), (11)–(14), (16) and (17). The initial solutions we get in this way are better than those using the original formulation (Adulyasak, Cordeau, and Jans 2014b; Armentano, Shiguemoto, and Lokketangen 2011).

In the second phase, we adopt a three-step procedure to improve the solutions. In the first step, we pick two, three, or whole-route customers in a period at a time and shift their deliveries to another period. We allow shifting to both the previous and next periods. The candidate customer–time pairs are selected adaptively (Adulyasak, Cordeau, and Jans 2014b). The transferring amounts of shipment are calculated accordingly (Armentano, Shiguemoto, and Lokketangen 2011). We apply a simulated annealing criterion to decide whether a local search solution should be accepted. The worse solution is accepted with probability $\exp((-100/T)(f(s^{curr}) - f(s^{best}))/f(s^{best}))$, with $T = T_k = 0.095T_{k-1}$. The

best move so far is recorded in the tabu list. The tabu tenure equals the planning horizon of the PRPTW. We incorporate tabu search because we want to avoid cycles when selecting candidate moves, and we add simulated annealing because we want to incorporate some randomness in the search process. The computational results show that the incorporation of tabu search and simulated annealing results in increase of the upper bound by 9.85% and 3.23% at most, respectively.

In the second step, we recalculate the inventories when all the shifts of shipment in the first step are carried out. We then solve the VRPTW for each period by taking the shipment quantities as demands for customers. The VRPTWs are solved first by the Solomon insertion heuristic (Solomon 1987). 2-opt, Or-opt, and 2-opt* local search heuristics (Toth and Daniele 2014) are then used to improve the VRPTW solutions.

In the third step, we solve the production and inventory problem at the depot by solving the corresponding mathematical programming model directly.

The above three-step search procedure terminates when it reaches maximum iterations or maximum time limits, or when the iteration without updating the incumbent solution reaches a limit. The resulting solution is provided as the initial solution to the branch-and-cut algorithm.

5. Computational results

The experiments of this section were run using a laptop with Intel Core 2 Duo CPU P8600 of 2.40 GHz under Ubuntu 14.04 (64 bits) with 4GB RAM. We implemented the branch-and-cut algorithm based on C++ API of IBM ILOG CPLEX version 12 release 6.

5.1. Instance generation

We adapt the Solomon benchmark (Solomon 1987) for the VRPTW to generate instances for the PRPTW. The coordinate, demand d_i and service time of each customer, vehicle capacity Q , and number of vehicles available K are taken from the Solomon benchmark. We assume that $d_{it} = d_i$. We apply the time window of each customer directly for the first period. For later periods, the time windows are shifted to start from the end of previous period, i.e. the latest time a vehicle should return to the depot in the previous period. Other parameters for generating instances are listed in Table 1.

Note that Solomon benchmarks contain three classes of problems, namely, clustered (C), random (R), and randomly clustered (RC). Each has two subclasses. Random instances turn out to be the hardest to solve. We draw two instances from each subclass of different sizes in the following computational experiments. We test from 25 to 100 customers, 3 periods. For each problem size and instance, we generate 5 random cases. This process results in 300 instances totally. Details of these instances are provided in the online supplement.

5.2. Data preprocessing

After generating the instance, we tighten the time windows of each customer first. Then we calculate the reaching arc set R_i^- and the reachable arc set R_i^+ for each customer. By definition (Lysgaard 2006), the reaching arc set R_i^- is the minimum arc set on any feasible path $(0, \dots, i)$. The reachable arc set R_i^+ is defined as the minimum arc set on any feasible

Table 1. Generation of instances.

Opening time at the plant and customers in period $t > 1$	$a_{it} = a_{i,t-1} + b_{00}, \forall i \in N_0, \forall t > 1$
Closing time at the plant and customers in period $t > 1$	$b_{it} = b_{i,t-1} + b_{00}, \forall i \in N_0, \forall t > 1$
The transportation cost c_{ij} from node i to node j with coordinates (χ_i, ψ_i) and (χ_j, ψ_j)	$c_{ij} = \text{round}(\sqrt{(\chi_i - \chi_j)^2 + (\psi_i - \psi_j)^2})$
Production capacity at the plant	$C = \sum_t \sum_i d_i$
Inventory capacity at the plant	$L_0 = C/2$
Inventory capacity at customers	$L_i = T * d_i$
Initial inventory at customers	$l_{i0} \in [1, T - 1] * d_i$
Initial inventory at the plant	$l_{00} = 0$
Unit cost of production	$u \in [0.2, 2]$
Unit inventory holding cost at customers	$h_i \in [0.1, 1]$
Unit inventory holding cost at the plant	$h_0 = 0.1$
Production set-up cost at the plant	$f = 1000h_0$

path $(i, \dots, 0)$. Subsequently, we can identify an incompatible node pair (i, j) by confirming the arcs between the node pairs are neither in R_j^- nor in R_i^- . We eliminate edge variables between incompatible node pairs.

We extend the time window reduction techniques (Bard, Kontoravdis, and Yu 2002; Kallehauge, Boland, and Madsen 2007) to a multi-period version which is described as follows.

5.2.1. Opening time adjustment

We use the following extended equations to adjust opening time of customer k whose reaching or reachable sets (Lysgaard 2006) are not empty, respectively, at time t .

$$a_{kt} := \max \left\{ a_{kt}, \min_{(i,k) \in A} \{a_{it} + c_{ik}\} \right\}, \quad \forall t \in T, \forall k \in N_0, \delta_t^-(k) \neq \emptyset \quad (34)$$

$$a_{kt} := \max \left\{ a_{kt}, \min \{b_{kt}, \min_{(k,j) \in A} \{a_{jt} - c_{kj}\}\} \right\}, \quad \forall t \in T, \forall k \in N_0, \delta_t^+(k) \neq \emptyset \quad (35)$$

5.2.2. Closing time adjustment

Likewise, we use the following extended equations to adjust closing time of customer k whose reaching or reachable sets are not empty, respectively, at time t .

$$b_{kt} := \min \left\{ b_{kt}, \max \{a_{kt}, \max_{(i,k) \in A} \{b_{it} + c_{ik}\}\} \right\}, \quad \forall t \in T, \forall k \in N_0, \delta_t^-(k) \neq \emptyset \quad (36)$$

$$b_{kt} := \min \left\{ b_{kt}, \max_{(k,j) \in A} \{b_{jt} - c_{kj}\} \right\}, \quad \forall t \in T, \forall k \in N_0, \delta_t^+(k) \neq \emptyset \quad (37)$$

5.3. Results and discussion

The experiments on the effects of the valid inequalities are conducted on the instances with $n = 25 \rightarrow 100$ and $|T| = 3$. The average results are shown in Tables 2 and 3.

Table 2. Effects of valid inequalities on average lower bounds at the root node.

Instance	With 1' (%)	With 1 (%)	With 2 (%)	With 3 (%)	With 4 (%)	With 5 (%)	All (%)
C.25-t3	−0.02	−0.04	2.84	0.18	0.07	1.72	8.18
R.25-t3	0.00	0.00	6.14	0.52	0.00	3.70	14.97
RC.25-t3	0.00	0.00	6.36	0.52	0.00	3.51	15.37
C.30-t3	3.23	7.01	4.07	0.05	0.13	3.41	13.73
R.30-t3	0.00	0.00	8.81	0.62	0.03	6.20	23.84
RC.30-t3	0.00	0.00	11.20	0.60	0.02	6.36	23.95
C.40-t3	1.72	4.71	1.98	0.22	0.02	2.58	8.26
R.40-t3	2.19	4.78	8.21	0.68	−0.18	3.55	15.41
RC.40-t3	3.26	4.68	8.06	0.87	−0.18	3.36	14.75
C.50-t3	2.92	9.58	5.55	0.17	−0.10	4.69	19.14
R.50-t3	7.42	11.12	10.88	0.04	0.06	4.56	29.58
RC.50-t3	5.52	10.91	17.36	0.06	0.20	4.40	29.96
C.100-t3	4.32	7.10	2.91	−0.03	−0.03	4.00	14.45
R.100-t3	5.59	9.76	8.68	0.08	−0.08	4.49	23.16
RC.100-t3	4.08	9.74	12.36	0.09	−0.06	4.39	24.35

Notes: The comparing base is the lower bound (LB) obtained from the LP relaxation of the MILP for the PRPTW, including only subtour elimination and incompatible path inequalities. Column *With 1'* indicates adding inventory-related inequalities (Archetti et al. 2011) only upon the base LB. Column *With 1* indicates adding generalized (I, S) inequalities only upon the base LB. Column *With 2* indicates adding reachability inequalities only upon the base LB. Column *With 3* indicates adding ATSP-TW inequalities only upon the base LB. Column *With 4* indicates adding tournament inequalities only upon the base LB. Column *With 5* indicates adding logical inequalities only upon the base LB. Column *All* indicates adding all five families of cuts upon the base LB.

Table 3. Effects of valid inequalities on CPU seconds at the root node.

Instance	With 1' (Δ CPU)	With 1 (Δ CPU)	With 2 (Δ CPU)	With 3 (Δ CPU)	With 4 (Δ CPU)	With 5 (Δ CPU)	All (Δ CPU)
C.25-t3	0.00	0.01	0.43	0.24	−0.04	−0.19	0.37
R.25-t3	0.01	0.01	0.02	0.15	0.00	−0.21	0.53
RC.25-t3	0.05	0.17	0.08	0.35	0.42	0.26	0.78
C.30-t3	0.02	0.17	1.70	1.69	0.03	−0.23	3.59
R.30-t3	0.00	0.01	0.00	0.14	0.01	−0.09	1.11
RC.30-t3	0.08	0.03	−0.29	−0.06	0.79	0.55	2.34
C.40-t3	0.06	0.06	1.10	4.14	−0.27	−0.20	6.41
R.40-t3	1.17	1.80	0.35	0.70	−0.23	−0.50	2.91
RC.40-t3	0.83	1.79	0.72	0.78	0.33	−0.65	3.90
C.50-t3	0.94	2.22	6.72	8.80	0.25	−0.84	8.02
R.50-t3	0.59	0.59	−0.68	−0.51	0.38	−1.84	2.56
RC.50-t3	0.74	0.91	−0.80	−0.24	4.41	2.16	8.09
C.100-t3	−2.51	−7.66	43.35	5.52	3.53	−28.91	64.60
R.100-t3	15.16	24.56	6.08	−6.75	−8.61	−46.68	159.08
RC.100-t3	7.98	24.55	5.91	−6.99	62.85	8.24	232.47

Notes: The comparing base is the lower bound (LB) obtained from the LP relaxation of the MILP for the PRPTW, including only subtour elimination and incompatible path inequalities. Column *With 1'* indicates adding inventory-related inequalities (Archetti et al. 2011) only upon the base LB. Column *With 1* indicates adding generalized (I, S) inequalities only upon the base LB. Column *With 2* indicates adding reachability inequalities only upon the base LB. Column *With 3* indicates adding ATSP-TW inequalities only upon the base LB. Column *With 4* indicates adding tournament inequalities only upon the base LB. Column *With 5* indicates adding logical inequalities only upon the base LB. Column *All* indicates adding all five families of cuts upon the base LB.

5.3.1. The effectiveness of the valid inequalities

We turn off the proposed heuristic procedure and all the CPLEX cuts and heuristics when we test the effectiveness of valid inequalities on average lower bounds at the root node. The lower bound of the reference state is calculated as the objective value of LP relaxation of the MILP for the PRPTW, including subtour elimination and incompatible path inequalities.

The inclusion of subtour elimination and incompatible path inequalities ensures that the lower bounds are truly linked with feasible solutions.

The proposed valid inequalities are appended as follows. First, we add generalized (I, S) inequalities only upon the reference state. The results show that generalized (I, S) inequalities turn out to be very effective when customer number is over 50 for all three instance classes. Average lower bounds increase by 11.12%. As a comparison, we also test the effectiveness of related valid inequalities (26) (Archetti et al. 2011). Average lower bounds increase by 7.42%. Thus the proposed generalized (I, S) inequalities outperform previously related valid cuts.

Second, we add strengthened reachability inequalities only upon the reference state. The strengthened reachability inequalities also turn out to be highly effective for all three instance classes, especially when customer number is above 50. Average lower bounds increase as high as 17.36%.

Third, we add ATSP-TW inequalities only upon the reference state. The generalized ATSP-TW inequalities turn out to be effective for most of the cases, although average lower bounds increase no more than 0.87%.

Next, we add tournament inequalities only upon the reference state. The generalized tournament inequalities turn out to be only modest effective, because average lower bounds increase no more than 0.2%. The reason might be that the effectiveness of tournament inequalities is offset by the existence of incompatible path inequalities in the reference state.

Then, we add the newly proposed logical inequalities upon the reference state. The new logical inequalities turn out to be effective in all three instance classes. Average lower bounds increase 6.36% at most.

Finally, we add all the valid inequalities upon the reference state. The computational results show that, for the clustered class, average lower bounds can increase by 19.14% at most. For the random class, average lower bounds can increase by 29.58% at most. For the randomly clustered class, average lower bounds can increase by 29.96% at most.

From Table 3, the computational times are sometimes decreased. The overall increases in computational times are negligible compared with the increase of lower bounds. Thus the computational results show that the valid inequalities we propose are very effective on average.

5.3.2. The performance of the branch-and-cut algorithm

We check the performance of the proposed branch-and-cut algorithm on multiple instances. We turn on CPLEX cuts and heuristics by default. The computational results are presented in Tables 4–7. Column *Heuristic Time* indicates CPU seconds spent by the hybrid heuristic. Column *Root Time* indicates time spent at the root node. Root time contains heuristic time. Column *Branch Time* indicates time spent by branching. Column *B&C time* indicates the total time spent at the root node and in branching. Column *Heur UB* indicates heuristic upper bounds obtained by the hybrid heuristic. Column *Root LB* indicates the lower bounds obtained after solving the root node. Column *ObjValue* indicates the final upper bounds of the feasible solutions. Column *Gap%* indicates the final gaps between the *ObjValue* and the final lower bounds. Column *CPLEX Cuts #* indicates the number of cuts added by CPLEX. Column *User Cuts #* indicates the number of cuts added by the proposed

Table 4. Performance of the proposed branch-and-cut algorithms under formulations I and II.

Instance	Formulation I		Formulation II	
	Gap (%)	CPU (seconds)	Gap (%)	CPU (seconds)
C.25.t3	0	28	0	232
R.25.t3	0	8	0	358
RC.25.t3	0	14	0	876
C.30.t3	0	2302	0	389
R.30.t3	1.5	3870	1.1	1600
RC.30.t3	3.4	14,400	0.0	1243
C.40.t3	0	3922	0	1187
R.40.t3	1	3837	1	1438
RC.40.t3	3.9	14,400	0.0	1006
C.50.t3	0.80	3851	0.5	1378
R.50.t3	0	1119	0.5	2257
RC.50.t3	1.74	14,400	1.0	3600
C.100.t3		Out of memory	0.64	1843
R.100.t3		Out of memory	3.30	3296
RC.100.t3		Out of memory	0.81	3600

Table 5. Performance of the proposed branch-and-cut algorithm with formulation II.

Instance #	Heuristic time	Root time	Branch time	B&C time	Heur UB (%)	Root LB (%)	Gap (%)	CPLEX Cuts#	User Cuts#	Nodes #
C.25.t3	170	175	57	232	0.64	0.75	0	146	96	6973
R.25.t3	218	223	136	358	1.44	1.63	0	153	104	16,892
RC.25.t3	600	608	271	876	1.76	1.00	0	227	156	21,573
C.30.t3	376	379	10	389	2.20	0.24	0	165	60	795
R.30.t3	495	496	1104	1600	2.68	2.67	1.09	179	549	102,802
RC.30.t3	600	616	578	1243	2.51	2.10	0	215	161	25,179
C.40.t3	433	440	747	1187	1.67	0.54	0.13	253	132	19,814
R.40.t3	528	533	905	1438	1.65	1.84	1.10	218	244	26,966
RC.40.t3	600	603	355	1006	8.59	1.23	0	182	47	2755
C.50.t3	477	479	899	1378	5.72	1.34	0.52	205	113	20,147
R.50.t3	625	628	1424	2257	8.82	4.43	0.51	331	304	13,663
RC.50.t3	600	602	1961	3600	6.17	2.04	1.00	404	281	50,820
C.100.t3	676	888	955	1843	2.65	1.02	0.64	370	125	2662
R.100.t3	753	803	2102	3296	9.89	3.33	3	560	316	6647
RC.100.t3	692	838	1196	3600	8.30	1.57	1	440	292	7639

Table 6. Performance of the proposed branch-and-cut algorithm with formulation II under different inventory holding cost.

Instance #	h	Heuristic time	Root time	Branch time	B&C time	Heur UB%	Root LB%	Gap (%)	CPLEX Cuts#	User Cuts#	Nodes #
C.25.t3	[0.1,1]	170.3	174.5	57	232	0.64	0.75	0	146	96	6973
R.25.t3	[0.1,1]	218.0	222.5	136	358	1.44	1.63	0	153	104	16,892
RC.25.t3	[0.1,1]	600.0	607.6	271	876	1.76	1.00	0	227	156	21,573
C.25.t3	[0.01,0.1]	234.0	235.0	40	275	2.06	1.49	0	105	55	3573
R.25.t3	[0.01,0.1]	224.3	229.3	853	1082	3.47	2.63	0.37	159	584	51,071
RC.25.t3	[0.01,0.1]	149.9	87.8	1238	1391	3.49	2.18	0	193	513	83,746

branch-and-cut algorithm. Column *Nodes #* indicates the number of nodes processed by the algorithm.

First, we compare the performance of the branch-and-cut algorithm with formulations I and II in Table 4. For instances with less than 50 customers, the branch-and-cut algorithm with formulation I is faster than the algorithm with formulation II for most cases. However,

Table 7. Performance of the proposed branch-and-cut algorithm with formulation II under different planning horizons.

Instance #	Heuristic time	B&C time	Heur UB%	Root LB%	Gap (%)	CPLEX Cuts#	User Cuts#	Nodes #
C.25.t3	126	126	1.00	0.47	0	60	17	20
R.25.t3	72	75	3.95	0.68	0	90	42	153
RC.25.t3	685	1894	5.66	2.55	0.32	305	175	32,711
C.50.t3	300	302	17.17	1.10	0	112	13	5
R.50.t3	651	2115	10.64	8.06	0.24	311	406	17,164
RC.50.t3	600	3600	7.36	2.24	0.81	507	228	74,194
C.25.t6	1341	2311	1.51	0.77	0.23	193	51	40,422
R.25.t6	620	2160	1.72	1.26	0.68	475	211	31,255
RC.25.t6	1207	3600	1.05	1.50	1.01	440	268	62,981
C.50.t6	702	3600	12.98	1.32	1.09	475	80	17,078
R.50.t6	753	3600	4.32	5.67	5.44	838	353	10,098
RC.50.t6	844	3600	9.13	4.61	9.52	621	326	13,893

for some instances, especially those which cannot be solved by both algorithms, the proposed branch-and-cut algorithm can obtain a better lower bound under formulation II. For instances over 50 customers, the branch-and-cut algorithm with formulation I runs out of memory. Whereas the proposed branch-and-cut algorithm with formulation II can still solve some of the instances to optimality.

The details on the performance of the proposed branch-and-cut algorithm with formulation II are given in Tables 5–7. For the basic configuration, we solve all instances on 25 customers, 3 periods in 1 hour. The instances with 30 customers are solved for 11 out of 12 cases. The unsolved instances have average gaps 4.3%. The instances with 40 customers are solved for 10 out of 12 cases. The unsolved instances have average gaps 4.4%. The instances with 50 customers are solved for 5 out of 12 instances. The unsolved instances have average gaps 2.1%. The instances with 100 customers are solved for 4 out of 12 instances. The unsolved instances have average gaps 9.87%.

We also check the performance of the branch-and-cut algorithm under different inventory holding costs. As shown in Table 6, the branch-and-cut algorithms perform almost equally under these two situations, although one of the instances with smaller inventory cost is not solved within 1 hour. The average gap is 1.48%. The hybrid heuristic has met with difficulty in this instance to find a good enough solution.

Finally, we check the performance of the branch-and-cut algorithm under different planning horizons. As the planning horizon increases to $|T| = 6$, the branch-and-cut algorithm solves two instances in 1 hour. The unsolved instances have average gaps at most 2% when customer size is less than 50.

6. Case study

This section presents an implementation of the proposed model on the production and distribution operations of a food company operating in the Nanjing city, China. We first describe the case study, then present our results.

6.1. Description and data

In the Nanjing city, China, most consumer products are distributed through chain stores of power retailers. Suguo company is among the biggest power retailers. Suguo covers the

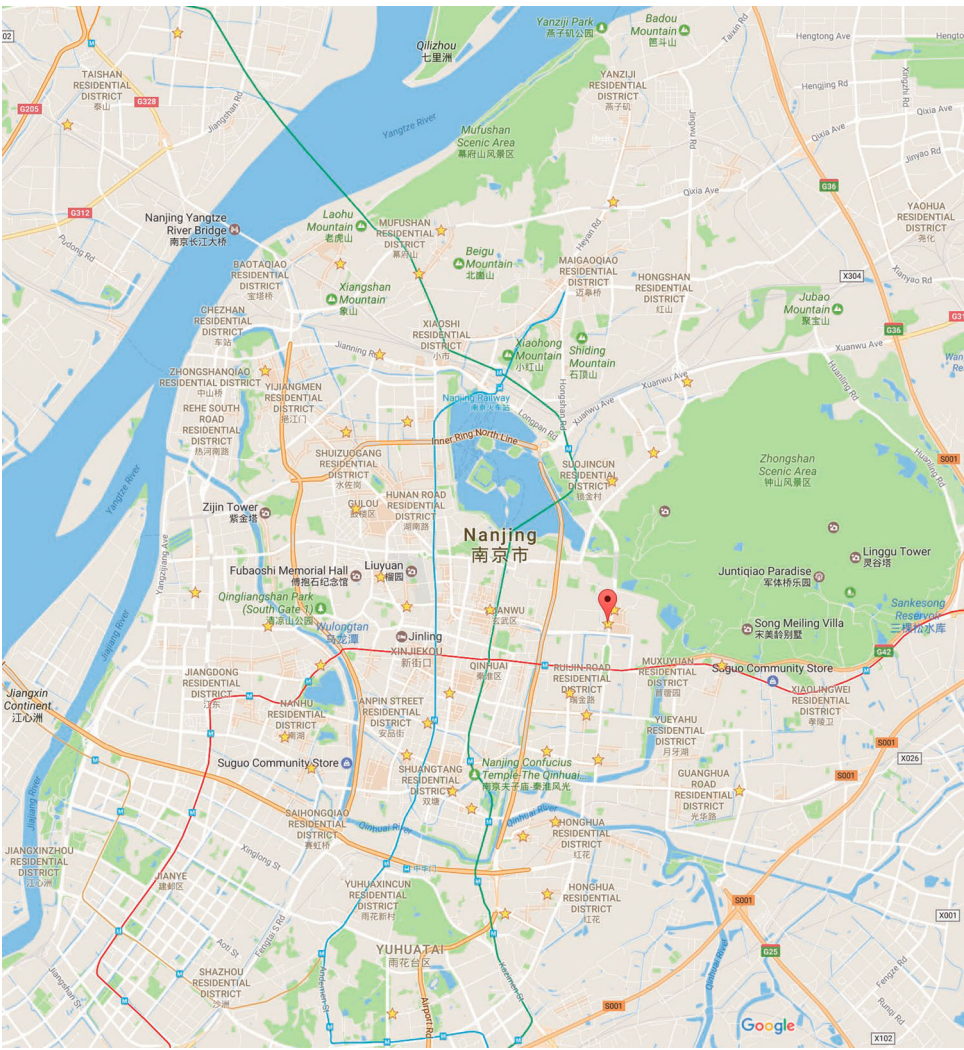


Figure 1. The representation of the retailer networks.

city with hundreds of large and small stores. Suguo community stores are large and thus the target store of the food company. The food company, situated on the north side of the Yangtze river, provides a certain kind of fresh meat product to Suguo community stores which mainly located on the south side of the Yangtze river. The retailer network is illustrated in Figure 1, in which the retailers are represented by yellow stars. The shortest paths between these sites are calculated using Google Maps. Detailed information can be found in the online supplement.

In a typical operations scenario, the food company has only 200 kg of fresh meat products in stock at the beginning of planning period. The 40 Suguo community stores have only limited products in stock at the beginning of planning horizon. According to the purchasing statistics, the Suguo community stores have demand estimates for the product for the following week. The vehicle used for deliveries can carry 8 tons of meat products and traverse

the city at an average speed of 40 kilometers per hour. The cost to distribute these products is 30 RMB dollars per kilometer. Other parameters are included in the online supplement.

Since the city of Nanjing has a population of nearly 10 million, traffic condition is getting worse. Consequently, the retailer stores can only be served within relative narrow time windows. The food company will decide how many items to produce, and the routing plans which respect the time windows.

6.2. Results

Before we apply the proposed model and solution method, the company made use of a heuristic solution, which results in a cost of 400,655 RMB dollars for the planning week. Using our model and algorithm, we can provide an optimized solution with a cost of 354,065 RMB dollars within seconds, resulting a 11.6% of decrease.

7. Conclusion

We have introduced, modeled, and analyzed the production routing problem with time windows (PRPTW), a generalization of the multi-vehicle production and inventory routing problem. We have developed a new model for the PRPTW which incorporates time windows and lot-sizing ingredients of the PRPTW. We also have introduced several families of valid inequalities: logical, lot-sizing, and vehicle routing types (specifically asymmetric traveling salesman problem with time windows and vehicle routing problem with time windows). These inequalities greatly increase the lower bounds at the root of the branch-and-cut tree and reduce the CPU times for finding optimal PRPTW solutions. Numerical results from the case study also show that the algorithm has the potential to reduce the total costs.

Several generalizations are possible for the PRPTW. One worth mentioning here is the possibility of incorporating start-up times when there are multiple types of products. Another would be to consider the problem in more complex supply chain environments like multi-level production and routing problems with time windows. The branch-and-cut algorithm can be further improved by incorporating column generation techniques to reduce the CPU times when dealing with a larger number of time periods. A comparative study of the strong branching rule and different bottleneck tolerance based rules (Goldengorin et al. 2006) might be beneficial to reduce the CPU times for our branch-and-cut algorithm.

Acknowledgments

We would like to thank the anonymous reviewers for their constructive and insightful suggestions. The programming help from Dr. Jack Gao is also acknowledged.

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

Y. Qiu's work is supported by National Natural Science Foundation of China (NSFC) under Grant No. 71571092. Y. Qiu also acknowledges the support from Jiangsu Overseas Research & Training Program for University Prominent Young & Middle-aged Teachers and Presidents during her stay at UF. Y. Qiu's work is also supported in part by General Research Project for Humanities and Social Sciences from Chinese Ministry of Education under Grant No. 11YJCZH137 and the Priority Academic Program Development of Jiangsu Higher Education Institutions (PAPD).

ORCID

Yuzhuo Qiu  <http://orcid.org/0000-0002-9772-1546>

References

- Absi, N., C. Archetti, S. Dauzere-Peres, and D. Feillet. 2015. "A Two-Phase Iterative Heuristic Approach for the Production Routing Problem." *Transportation Science* 49 (4): 784–795.
- Achterberg, T., T. Koch, and A. Martin. 2005. "Branching Rules Revisited." *Operations Research Letters* 33 (1): 42–54.
- Adulyasak, Y., J.-F. Cordeau, and R. Jans. 2014a. "Formulations and Branch-and-Cut Algorithms for Multivehicle Production and Inventory Routing Problems." *INFORMS Journal on Computing* 26 (1): 103–120.
- Adulyasak, Y., J.-F. Cordeau, and R. Jans. 2014b. "Optimization-Based Adaptive Large Neighborhood Search for the Production Routing Problem." *Transportation Science* 48 (1): 20–45.
- Adulyasak, Y., J.-F. Cordeau, and R. Jans. 2015a. "Benders Decomposition for Production Routing Under Demand Uncertainty." *Operations Research* 63 (4): 851–867.
- Adulyasak, Y., J.-F. Cordeau, and R. Jans. 2015b. "The Production Routing Problem: A Review of Formulations and Solution Algorithms." *Computers & Operations Research* 55, 141–152.
- Archetti, C., L. Bertazzi, G. Laporte, and M. G. Speranza. 2007. "A Branch-and-Cut Algorithm for a Vendor-Managed Inventory-Routing Problem." *Transportation Science* 41 (3): 382–391.
- Archetti, C., L. Bertazzi, G. Paletta, and M. G. Speranza. 2011. "Analysis of the Maximum Level Policy in a Production–Distribution System." *Computers & Operations Research* 38 (12): 1731–1746.
- Armentano, V. A., A. L. Shigemoto, and A. Lokketangen. 2011. "Tabu Search with Path Relinking for an Integrated Production–Distribution Problem." *Computers & Operations Research* 38 (8): 1199–1209.
- Bard, J. F., G. Kontoravdis, and G. Yu. 2002. "A Branch-and-Cut Procedure for the Vehicle Routing Problem with Time Windows." *Transportation Science* 36 (2): 250–269.
- Bard, J. F., and N. Nananukul. 2009a. "Heuristics for a Multi-Period Inventory Routing Problem with Production Decisions." *Computers & Industrial Engineering* 57 (3): 713–723.
- Bard, J. F., and N. Nananukul. 2009b. "The Integrated Production–Inventory–Distribution–Routing Problem." *Journal of Scheduling* 12 (3): 257–280.
- Bard, J. F., and N. Nananukul. 2010. "A Branch-and-Price Algorithm for an Integrated Production and Inventory Routing Problem." *Computers & Operations Research* 37 (12): 2202–2217.
- Boudia, M., M. A. O. Louly, and C. Prins. 2007. "A Reactive Grasp and Path Relinking for a Combined Production–Distribution Problem." *Computers & Operations Research* 34 (11): 3402–3419.
- Boudia, M., and C. Prins. 2009. "A Memetic Algorithm with Dynamic Population Management for an Integrated Production–Distribution Problem." *European Journal of Operational Research* 195 (3): 703–715.
- Chandra, P., and M. L. Fisher. 1994. "Coordination of Production and Distribution Planning." *European Journal of Operational Research* 72 (3): 503–517.
- Chow, J. Y., and A. E. Nurumbetova. 2015. "A Multi-Day Activity-Based Inventory Routing Model with Space-Time-Needs Constraints." *Transportmetrica A: Transport Science* 11 (3): 243–269.
- Díaz-Madroño, M., D. Peidro, and J. Mula. 2015. "A Review of Tactical Optimization Models for Integrated Production and Transport Routing Planning Decisions." *Computers & Industrial Engineering* 88, 518–535.

- Fischetti, M., and P. Toth. 1997. "A Polyhedral Approach to the Asymmetric Traveling Salesman Problem." *Management Science* 43 (11): 1520–1536.
- Fumero, F., and C. Vercellis. 1999. "Synchronized Development of Production, Inventory, and Distribution Schedules." *Transportation Science* 33 (3): 330–340.
- Goldengorin, B., G. Jager, and P. Molitor. 2006. "Tolerance Based Contract-or-Patch Heuristic for the Asymmetric TSP." *Lecture Notes in Computer Science* 4235: 86–97.
- Kallehauge, B., N. Boland, and O. B. G. Madsen. 2007. "Path Inequalities for the Vehicle Routing Problem with Time Windows." *Networks* 49 (4): 273–293.
- Kumar, R. S., K. Kondapaneni, V. Dixit, A. Goswami, L. S. Thakur, and M. K. Tiwari. 2016. "Multi-Objective Modeling of Production and Pollution Routing Problem with Time Window: A Self-Learning Particle Swarm Optimization Approach." *Computers & Industrial Engineering* 99, 29–40.
- Lei, L., S. Liu, A. Ruszczyński, and S. Park. 2006. "On the Integrated Production, Inventory, and Distribution Routing Problem." *IIE Transactions* 38 (11): 955–970.
- Lysgaard, J. 2006. "Reachability Cuts for the Vehicle Routing Problem with Time Windows." *European Journal of Operational Research* 175 (1): 210–223.
- Lysgaard, J., A. N. Letchford, and R. W. Eglese. 2004. "A New Branch-and-Cut Algorithm for the Capacitated Vehicle Routing Problem." *Mathematical Programming* 100 (2): 423–445.
- Miller, C. E., A. W. Tucker, and R. A. Zemlin. 1960. "Integer Programming Formulation of Traveling Salesman Problems." *Journal of the ACM (JACM)* 7 (4): 326–329.
- Naddef, D., and G. Rinaldi. 2001. "Branch-and-Cut Algorithms for the Capacitated VRP." In *The Vehicle Routing Problem*, edited by P. Toth, D. Vigo, 53–84. Philadelphia, PA: Society for Industrial and Applied Mathematics.
- Pochet, Y., and L. A. Wolsey. 2006. *Production Planning by Mixed Integer Programming*. New York: Springer-Verlag.
- Qiu, Y., J. Qiao, and P. M. Pardalos. 2017. "A Branch-and-Price Algorithm for Production Routing Problems with Carbon Cap-and-Trade." *Omega* 68, 49–61.
- Savelsbergh, M. W. P. 1985. "Local Search in Routing Problems with time Windows." *Annals of Operations Research* 4 (1): 285–305.
- Solomon, M. M. 1987. "Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints." *Operations Research* 35 (2): 254–265.
- Toth, P., and V. Daniele. 2014. *Vehicle Routing: Problems, Methods, and Applications*. Philadelphia, PA: SIAM.
- You, S. I., J. Y. J. Chow, and S. G. Ritchie. 2016. "Inverse Vehicle Routing for Activity-Based Urban Freight Forecast Modeling and City Logistics." *Transportmetrica A: Transport Science* 12 (7): 650–673.