The Consistent Vehicle Routing Problem considering path consistency in a road network

Citation for published version (APA):

Document license:
TAVERNE

DOI:
10.1016/j.trb.2021.09.005

Document status and date:
Published: 01/11/2021

Document Version:
Publisher’s PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:
• A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher’s website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the “Taverne” license above, please follow below link for the End User Agreement:
www.tue.nl/taverne

Take down policy
If you believe that this document breaches copyright please contact us at:
openaccess@tue.nl
providing details and we will investigate your claim.
The consistent vehicle routing problem considering path
consistency in a road network

Yu Yao\textsuperscript{a,}\textsuperscript{*}, Tom Van Woensel\textsuperscript{b}, Lucas P. Veeenturf\textsuperscript{c}, Pengli Mo\textsuperscript{d}

\textsuperscript{a} College of Civil and Transportation Engineering, Hohai University, Nanjing, China
\textsuperscript{b} School of Industrial Engineering, Eindhoven University of Technology, Eindhoven, Netherlands
\textsuperscript{c} Rotterdam School of Management, Erasmus University, Rotterdam, Netherlands
\textsuperscript{d} State Key Laboratory of Rail Traffic Control and Safety, Beijing Jiaotong University, Beijing, China

Abstract

This paper investigates a new variant of the consistent vehicle routing problem considering path consistency based on the underlying road network (ConVRP\textsubscript{RN}). It is the problem of determining a set of consistent paths for vehicles within a certain period (e.g., a week). We propose a flexible strategy to achieve path consistency, providing a discount for the cost of roads that are traversed every day. The objective is to minimize the total travel costs considering the discount for the paths' consistent parts. We formulate the problem as a set partitioning model and a general arc-flow model, and then solve the problem by a branch-price-and-cut algorithm. The algorithm is based on a two-layer network, including an underlying road-network graph (lower layer) and a serial of customer-based graphs (upper layer). Furthermore, an innovative aggregation technique is proposed to accelerate the algorithm, which is specifically designed for ConVRP\textsubscript{RN}. Finally, we construct a road network based on the real road-network structure of West Jordan and demonstrate the effectiveness of our approach on a set of numerical experiments. The impact of consistency discount is also analyzed to provide information and inspiration for tactical decision making.

1. Introduction

Vehicle routing problems (VRPs) are often used as essential building blocks for handling city logistics (delivery) problems. In recent years, both academia and practitioners indicate that the consistency elements, e.g., the same service provider and the same service time windows, are important to consider, giving rise to the Consistent Vehicle Routing Problem (ConVRP) (Groër et al., 2009; Kovacs et al., 2014).

In this study, we consider the consistency from the drivers' point of view. In a distribution system, the reliability and efficiency of the service are highly related to drivers' familiarity with the paths, routes, schedules, and delivery areas (Quirion-Blais and Chen, 2020; Zhong et al., 2007; Schneider, 2016). However, in most cases, especially in large cities with complex road networks, it is almost impossible for drivers to be familiar with all streets. Therefore, we focus on the ConVRP considering path consistency, which aims to increase the driver familiarity by finding a set of consistent paths for each vehicle within several days. Customers' demands within the planning horizon (e.g., a week) are assumed to be known. This situation has practical applications in many B2B logistics distribution systems. For instance, in the retail supply chain, the stores are resupplied regularly.

* Corresponding author.
E-mail address: yuoyu1@bjtu.edu.cn (Y. Yao).

https://doi.org/10.1016/j.trb.2021.09.005
Received 11 January 2021; Received in revised form 15 July 2021; Accepted 11 September 2021
Available online 24 September 2021
0191-2615/© 2021 Elsevier Ltd. All rights reserved.
If all the vehicles are forced to follow the exactly same paths every day, it will lead to long detours and reduce delivery efficiency. Therefore, we propose a flexible path consistency strategy, which encourages vehicles to perform similar paths every day by providing a discount for the travel cost of the consistent road sections on the paths. The value of the discount adjusts the consistency degree of the paths. When the discount is larger, the consistent segments are performed with lower travel costs such that the consistency degree will increase, and vice versa. When the value of the discount equals 0, the problem is equivalent to solving VRPs in a series of independent scenarios (days).

Another critical issue is how to evaluate path consistency. Most of the existing studies about ConVRP are based on customer-based graphs. A fundamental assumption is that, for every two points of interest (e.g., customers and depots), the best path between them is determined in advance. However, adopting a customer-based graph misses the information of the road network structure, such that the path consistency cannot be evaluated directly. Therefore, we consider our problem based on a road-network graph, which mimics the real-world road network structure.

Similar to Huang et al. (2017), we define a route as the order of customers served in the customer-based graph and a path as the order of nodes followed in the road-network graph. This study focuses on path consistency based on the structure of the underlying road network, i.e., ConVRP\(_{RN}\). Let us compare the customer-based graph and the road-network graph when considering the path consistency. Fig. 1 depicts a simple road network example in Exhibit (a) and two realizations (e.g., weekdays) with different customers. A single vehicle is used to serve customers every day. It is not easy to evaluate the path consistency over the different days (scenarios) on the customer-based graph, as shown in Exhibit (b). On the road-network graph, we observe that the red bold parts of both days’ paths are consistent, as shown in Exhibit (c).

In practice, the traffic conditions are quite different on different segments of the road network. For instance, traffic conditions on arterial roads are more dynamic and complex, requiring higher familiarity with these roads. The traffic conditions on residential roads are relatively stable during the day. Therefore, in our consistency strategy, the mentioned consistency discount is only provided for the arterial roads on paths.

The ConVRP is difficult to solve as it is an interdependence problem (Drexl, 2012), i.e., a change in the route/path of one scenario (day) may have effects on other scenarios (days). Most researchers adopt heuristic methods to solve the person-oriented ConVRP, for instance, districting approaches (Haughton, 2002; Schneider et al., 2015) and a priori strategies (Sungur et al., 2010; Kovacs et al., 2014). A limited number of studies use exact algorithms, noticeable studies include Spliet and Gabor (2015) and Goeke et al. (2019), which develop a branch-price-and-cut algorithm and a column generation framework for solving the ConVRP. Our problem is more complex as it is considered based on the road network. Furthermore, the path consistency considered in this study only exists in the paths of the same vehicle in different scenarios, such that the corresponding relationship of vehicles in different scenarios needs to be considered. Therefore, adopting a standard branch-price-and-cut algorithm is inefficient. To find the optimal solutions for our problem, we develop a dedicated branch-price-and-cut algorithm based on a two-layer network and employ a newly introduced aggregation technique.

The algorithm can solve instances with up to five workdays and 35 customers (per day) in a road network with 113 nodes, and 310 links within three hours. Especially, adopting the aggregation technique reduces 84.55% of the computational time.
The contributions of this paper are summarized as follows.

1. We propose a consistent VRP considering road network information and imposing path consistency. Path consistency is a new variant of the ConVRP defined on a road network. A path-based set partitioning model and a general arc-flow model are presented for this problem.

2. We present a branch-price-and-cut-based algorithm to solve the ConVRP$_{RN}$. An innovative aggregation technique is used to accelerate the algorithm. Specifically, we consider a simpler aggregated version of restricted master problems first and restore them to standard restricted master problems after obtaining integer solutions of the former. An assignment model is developed to split the aggregated variables and sets.

3. We construct a coupled two-layer network for the ConVRP$_{RN}$. The lower layer is a road-network graph providing the information on the road network. The upper layer is a serial of customer-based graphs, which is dynamically updated during the column generation’s iterations and used for pricing, valid inequality separation, and branching procedures. By adopting this two-layer network, we consider the road network structure within the branch-price-and-cut framework.

4. We use a set of numerical experiments to provide insight into the performance of the valid inequalities, aggregation technique, and branch-price-and-cut algorithm. The effect of the consistency discount value on travel distance and consistency degree is analyzed, which provides information and reference for decision-makers. We also analyze side benefits in other consistent dimensions when considering path consistency.

The remainder of the paper is organized as follows. In Section 2, we present a literature review. In Section 3, we provide a formal description of the ConVRP$_{RN}$ and its mathematical model formulation. The two-layer network and branch-price-and-cut algorithm are presented in Section 4. In Section 5, we construct road networks and design a set of experiments. Section 6 provides the results and analysis of the computational experiments. Section 7 concludes this paper.

2. Literature review

This section reviews the literature on consistent vehicle routing problems (Section 2.1) and vehicle routing problems on road networks (Section 2.2).

2.1. Consistent vehicle routing problem

The concept of consistent vehicle routing problem (ConVRP) is first coined in the literature by Groër et al. (2009). A survey article is provided by Kovacs et al. (2014). From a driver’s point of view, consistency can be obtained in the following dimensions: customer consistency, arrival time consistency, route consistency, and districting consistency. These dimensions are not entirely independent. Considering one of them usually leads to consistency in other dimensions as a side benefit (Kovacs et al., 2014).

Customer consistency is achieved by assigning the same driver to the same customers. Arrival time consistency refers to visiting customers at more or less similar times each day. Groër et al. (2009) require the same drivers to visit the same customers at roughly the same time each day and denote this problem as the ConVRP. The solution approach is based on a template concept, and daily vehicle routes are derived from template routes. As the consistency constraints in Groër et al. (2009) might be too strict for some applications, Kovacs et al. (2015) extend and relax the ConVRP to the generalized consistent vehicle routing problem, where some requirements, such as single driver–customer pair and hard time window constraints, are modified to better satisfy the needs of real-world applications. A flexible large neighborhood search is developed to obtain solutions. Goeke et al. (2019) propose an exact method for the ConVRP considering driver consistency and arrival-time consistency. They develop a column generation framework and adopt a large neighborhood search as an upper bounding procedure. Different from previous studies about standard VRPs where in column generation procedures each variable represents a feasible route, in their study, they use each variable to represent the set of routes assigned to a vehicle over the planning horizon to obtain tighter lower bounds. The large neighborhood search within the column generation framework can also be used as a stand-alone heuristic. Rodríguez-Martín et al. (2019) consider the driver consistency in the periodic vehicle routing problem, where each customer has an associated set of allowable visit schedules. They propose an integer linear programming formulation for the problem and solve it by a branch-and-cut algorithm.

Recently, an increasing number of researches pay attention to the ConVRP and further combine it with other VRP extensions. Campelo et al. (2019) focus on the consistency of customer-route assignment and pairs it with the operational and service level constraints, that is, a consistent Vehicle Routing Problem with service level agreements. They design a fix-and-optimize based approach to tackle this problem and validate it in a real case study of a pharmaceutical distribution company. Mancini et al. (2021) examine a collaborative consistent vehicle routing problem with workload balance, which focuses on time window and service consistency. In their problem, carriers are allowed to collaborate and share customers among each other to increase efficiency and profit, and workload balance is also taken into consideration. The authors propose a metaheuristic and an iterated local search algorithm to solve the problem.

Time window assignment vehicle routing problem (TWAVRP), proposed by Spliet and Gabor (2015), focuses on making the delivery for each customer within a specific time interval in different scenarios. In this problem, a customer requires service in each scenario but does not need to be served by the same driver. The goal is to assign time windows and minimize the expected transportation costs. Spliet and Gabor (2015) develop a branch-price-and-cut algorithm to solve this problem to optimality. Spliet et al. (2018) further consider this problem with time-dependent travel times. Neves-Moreira et al. (2018) study a discrete time window assignment vehicle routing problem. Some real-world contexts, such as product dependent time windows, multiple product
delivers, and split deliveries are also considered in their study. A fix-and-optimize based approach is proposed and applied to a set of real-world instances.

The route and region consistency are achieved by repeatedly assigning each driver to the same route or service region repeatedly (Zhong et al., 2007; Schneider et al., 2015). For route consistency, early studies usually design a set of a priori routes in advance (Beasley, 1984) and then use different re-optimization ways to update the preplanned routes to match better the actual demand realization (Bertsimas, 1992). Sungur et al. (2010) consider a courier delivery problem with uncertain customer demands. They propose an insertion-based solution heuristic to find a master plan route and daily routes that maximize the similarity of routes. Precisely, the similarity between routes is measured by counting the number of customers on the daily route within a given distance of any customer on the master plan route.

Also related to the work presented in this paper is that of Quirion-Blais and Chen (2020). They study a VRPTW with drivers' experience. Our study is different from their paper in the following three aspects. Firstly, Quirion-Blais and Chen (2020) focus on the utilization of drivers' experience and view the consistency as an additional benefit. In our study, the main objective is to obtain consistent paths and thereby improving drivers' familiarity of certain paths. Secondly, Quirion-Blais and Chen (2020) consider the problem based on a customer-based graph and evaluate the similarity of routes based on the sequences of consecutive customers. In our study, detailed paths (the order of nodes followed in the road-network graph) of vehicles are considered and used to evaluate the consistency. Thirdly, Quirion-Blais and Chen (2020) design a case-based reasoning (CBR) approach to solve the VRPTW with drivers' experience, in which the core idea is to store past experience obtained from historical data and to reuse it to design future solutions. In our study, we develop a branch-price-and-cut algorithm, to solve the proposed problem with multiple scenarios. No historical data is provided before solving the problem.

There are also some studies considering the region consistency by restricting drivers to certain service territories. Then drivers could be familiar with their regional roads. Two-stage approaches are usually adopted to solve this kind of problems. For instance, Haugland et al. (2007) formulate the problem as a two-stage stochastic program with recourse, where the districting decisions are made in the first stage, and then VRPs are solved for each district in the second stage. Zhong et al. (2007) propose a two-stage method with strategic and operational phases for solving dispatching problems. Schneider et al. (2015) further consider time window constraints. Although our study also aims to improve drivers familiarity of routes/paths, we adopt an alternative direction to achieve consistency, that is, considering the path consistency directly in the road network.

2.2. Routing on road networks

In most literature on VRP, the models and algorithms are based on the assumption that the paths between every two nodes are defined in advance, i.e., on a customer-based graph (Huang et al., 2017). However, on a real-life road network, each road segment is labeled with several attributes: distance, travel time, cost, and type. The customer-based graph cannot adequately represent these attributes. Therefore, working with road-network graphs is proposed to address this issue. The related works are split up into three classes. The first class of approaches represents the road network with a multi-graph, which contain a serial of non-dominated paths between two nodes (Garaix et al., 2010). Researchers apply the multi-graph representation on different variants of VRP (Huang et al., 2017) and develop both exact (Ben Ticha et al., 2019) and heuristic algorithms (Lai et al., 2016). Ticha et al. (2017) investigate the VRPTW based on a multi-graph and show that the multi-graph representation has a positive impact on the solution quality. Recently, Soriano et al. (2020) consider an inconsistent VRP based on a multi-graph. In their problem, a set of routes with diversified arrival times and no en-route waiting time is obtained to reduce risk and exposure in the cash-in-transit transportation. They adopt a multi-graph to achieve extra flexibility with the detours and solve the problem by an adaptive neighborhood search approach.

Another class of approaches tackles the problem directly on a road-network graph, which mimics a real-world network. Letchford et al. (2014) reformulate the problem proposed by Garaix et al. (2010) on a road-network graph and develop a dynamic programming algorithm for finding pricing routes. Ben Ticha et al. (2019) proposes a branching scheme for the VRPTW based on a road-based network. They also provide a comparison between the multi-graph and road-network, which shows that the branch-and-price algorithm is more efficient in the multi-graph than in the road-network. Mahmoudi and Zhou (2016) consider a time-dependent pickup and delivery problem with time window on real-world road networks. The model is formulated based on a three-dimensional state-space–time network and then solved by a Lagrangian relaxation solution framework.

The last class of approaches considers the problem based on road-network graphs and transform them into customer-based graphs by aggregating different attributes on road segments (Huang et al., 2006; Zografos and Androustopoulos, 2008).

Unlike the mentioned studies in Section 2.1, we focus on another important consistency element, the path consistency in the road network. A two-layer network is constructed to handle the road-network information within the branch-price-and-cut framework. To the best of our knowledge, no similar study has been published. We summarize the detailed characteristics of the most closely related studies discussed above in Table 1.

3. Problem statement and model formulation

This section describes the ConVRP$_{RN}$ and introduces the notation used throughout the paper. We also present a set partitioning formulation of the ConVRP$_{RN}$.
According to the distance, and the second term represents the discount of the consistent parts of the paths. Constraints (2) represent 

\[
\begin{align*}
\text{Subject to:} & \\
\sum_{k \in K} \sum_{p \in P_k^s} a_{ip} x_{ipk}^s = 1, \forall i \in N^s, s \in S & \\
\sum_{p \in P_k^s} b_{ip} x_{ipk}^s - y_{ijk} & \geq 0, \forall (i, j) \in A^s, s \in S, k \in K
\end{align*}
\]

 otherwise. The set partitioning model is expressed as (1)–(5).

The objective function (1) is to minimize the total travel costs with the consistency discount. The first term is the travel costs according to the distance, and the second term represents the discount of the consistent parts of the paths. Constraints (2) represent

<table>
<thead>
<tr>
<th>Paper</th>
<th>Consistent dimension</th>
<th>Road network</th>
<th>Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Beasley (1984)</td>
<td>Route</td>
<td>No</td>
<td>Heuristics</td>
</tr>
<tr>
<td>Zhong et al. (2007)</td>
<td>Region</td>
<td>No</td>
<td>Tabu search</td>
</tr>
<tr>
<td>Groër et al. (2009)</td>
<td>Person, arrival time</td>
<td>No</td>
<td>Heuristics</td>
</tr>
<tr>
<td>Sungur et al. (2010)</td>
<td>Route</td>
<td>No</td>
<td>Heuristics</td>
</tr>
<tr>
<td>Kovacs et al. (2015)</td>
<td>Person, arrival time</td>
<td>No</td>
<td>Large neighborhood search</td>
</tr>
<tr>
<td>Schneider et al. (2015)</td>
<td>Region</td>
<td>No</td>
<td>Heuristics</td>
</tr>
<tr>
<td>Spath and Gabor (2015)</td>
<td>Arrival time</td>
<td>No</td>
<td>Branch price and cut</td>
</tr>
<tr>
<td>Goek et al. (2019)</td>
<td>Person, arrival time</td>
<td>No</td>
<td>Column generation</td>
</tr>
<tr>
<td>Rodríguez-Martín et al. (2019)</td>
<td>Person</td>
<td>No</td>
<td>Branch and cut</td>
</tr>
<tr>
<td>Campelo et al. (2019)</td>
<td>Person</td>
<td>No</td>
<td>Fix-and-optimize based approach</td>
</tr>
<tr>
<td>Mancini et al. (2021)</td>
<td>Arrival time, person</td>
<td>No</td>
<td>Iterated local search</td>
</tr>
<tr>
<td>Quirion-Blais and Chen (2020)</td>
<td>Route</td>
<td>No</td>
<td>Case-based reasoning approach</td>
</tr>
<tr>
<td>Huang et al. (2017)</td>
<td>Road-network graph</td>
<td>Multigraph</td>
<td>Lagrangian relaxation</td>
</tr>
<tr>
<td>Ben Ticha et al. (2019)</td>
<td>–</td>
<td>–</td>
<td>Solver</td>
</tr>
<tr>
<td>Mancini et al. (2021)</td>
<td>Arrival time, person</td>
<td>No</td>
<td>Iterated local search</td>
</tr>
<tr>
<td>Soriano et al. (2020)</td>
<td>Arrival time (inconsistency)</td>
<td>Multigraph</td>
<td>Adaptive neighborhood search</td>
</tr>
<tr>
<td>This paper</td>
<td>Path</td>
<td>Two-layer network</td>
<td>Branch price and cut</td>
</tr>
</tbody>
</table>

### 3.1. Problem statement

Consider a directed graph \( G = (N, A) \) representing a road network. Arcs in set \( A \) represent road segments, which include arterial roads and residential roads. Specifically, the set of arterial roads is denoted as \( A^s \). Let \( c_{ij} \geq 0 \) be the travel cost for traversing arc \((i, j) \in A \). Nodes in set \( N \) represent the depot, customer nodes, road connections, and intersections. Let \( S \) be a set of scenarios (i.e., working days), where each scenario represents a specific demand realization. For each scenario \( s \in S \), \( N^s \subset N \) represents the set of customer nodes in this scenario. Demand quantity \( q^s_i \) is associated with each customer \( i \in N^s \). A fleet of \( |K| \) vehicles with a capacity of \( Q \) is available in each scenario. If a segment of arterial roads is traversed by a certain vehicle in each scenario (i.e., every day), we denote this part of the path as consistent, and its travel cost receives a discount of \( \theta \). For instance, the bold lines in Fig. 1(c) represent the consistent parts of paths.

The ConVRP\(_{RN} \) is to find a set of feasible paths for vehicles in each scenario (on each day) \( s \in S \) that each customer present in the scenario is served exactly once. A feasible path should satisfy the following conditions: (1) the total demand of served customers on a path is within the vehicle capacity, and (2) the path starts from and ends at the depot. The objective is to minimize the total travel costs considering the discount brought by path consistency. To make the problem easier to solve, the non-elementary paths are allowed, but they will not appear in the optimal solution as each customer need to be served exactly once.

For the reader’s convenience, we provide Table 2 listing the notation used in this paper.

### 3.2. Model

We formulate a path-based set partitioning model for ConVRP\(_{RN} \). We use \( P_k^s \) to represent the feasible path set for vehicle \( k \) on day \( s \). For each path \( p \in P_k^s \), a cost \( c_p \) is associated with it. The parameter \( a_{ip} \) represent the number of times that customer \( i \) is served in path \( p \); \( b_{ip} \) represents the number of times that arc \((i, j) \) is traversed in path \( p \). The binary variable \( x_{ipk}^s \) = 1 if path \( p \) is used by vehicle \( k \) on day \( s \), and \( x_{ipk}^s = 0 \) otherwise. The binary variable \( y_{ijk} \) = 1 if vehicle \( k \) traverses arc \((i, j) \in A^s \) on each day; and \( y_{ijk} = 0 \) otherwise. The set partitioning model is expressed as (1)–(5).

\[
\begin{align*}
\min \sum_{i \in S} \sum_{k \in K} \sum_{p \in P_k^s} c_p x_{ipk}^s - \theta \cdot |S| \sum_{(i, j) \in A^s} \sum_{k \in K} c_{ij} y_{ijk} \\
\text{Subject to:} & \\
\sum_{k \in K} \sum_{p \in P_k^s} a_{ip} x_{ipk}^s = 1, \forall i \in N^s, s \in S & \\
\sum_{p \in P_k^s} b_{ip} x_{ipk}^s - y_{ijk} & \geq 0, \forall (i, j) \in A^s, s \in S, k \in K & \\
x_{ipk}^s & \in \{0, 1\}, \forall p \in P_k^s, s \in S, k \in K & \\
y_{ijk} & \in \{0, 1\}, \forall (i, j) \in A^s, k \in K
\end{align*}
\]

The objective function (1) is to minimize the total travel costs with the consistency discount. The first term is the travel costs according to the distance, and the second term represents the discount of the consistent parts of the paths. Constraints (2) represent...
3.3. Range of \( \theta \) value

In both the path-based model and the arc-flow model, the consistent discount \( \theta \) is an important parameter which adjusts the consistency degree of the resulting paths. Here we give the meaningful range for setting the discount parameter.

**Proposition.** In a situation with \( |S| \) scenarios, the meaningful range for setting the discount parameter \( \theta \) is \([0, 1 - \frac{1}{|S|}]\). The sufficient condition to achieve 100% path consistency is that the value of the discount parameter \( \theta \) is greater than or equal to \( 1 - \frac{1}{|S|} \).

**Proof.** A vehicle will perform a detour instead of taking the shortest path if the detour helps to form consistent parts with the paths in other scenarios, and the saving cost brought by the consistency discount is greater than increased cost caused by the detour.

Consider an extreme example with \( |S| \) scenarios, where only one scenario has customer demands to serve. If each scenario is optimized independently, the results would be that only one scenario needs vehicles to perform paths. When considering the consistency, the vehicles in each scenario will repeat the same paths if Eq. (6) holds.

\[
\theta x_{ij} \geq (|S| - 1)(1 - \theta)x_{ij}, \forall (i,j) \in A^*
\]  

(6)

The right-hand side of the Eq. (6) represents the cost savings achieved by the consistency discount, and the left-hand side of the Eq. (6) represents the cost increase caused by performing detour paths.

After simplification, this equation could be expressed as Eq. (7), which shows that the value of \( \theta \) for achieving 100% consistency only depends on the number of scenarios considered in the instances.

\[
\theta \geq 1 - \frac{1}{|S|}
\]  

(7)
Y. Yao et al.

Fig. 2. Example of a two-layer network.

Obviously, this conclusion also applies to other general instances with relaxed assumptions.

4. Solution approach

To solve the proposed model, we develop a Branch-Price-and-Cut-based solution framework, i.e., a branch-and-bound algorithm in which at each node of the searching tree a linear programming (LP) relaxation is solved using column generation and valid inequalities are added to strengthen the relaxation. The interested reader is referred to Barnhart et al. (2000) and Desrosiers and Lübbecke (2010) for details about Branch-Price-and-Cut. To avoid inefficiency when applying the branch-price-and-cut algorithm directly on the road network (Ben Ticha et al., 2019), we construct a two-layer network, where the upper layer is used for the pricing sub-problem, separation of valid inequalities, and branching, and the lower layer provides the detailed information on the road network. To improve the performance of the algorithm, we also propose an aggregation technique. Roughly stated, this technique solves the linear relaxation of a simpler aggregated restricted master problem (A-RMP) instead of a standard restricted master problem (RMP) in some branch-and-bound nodes. The algorithm framework is presented as follows.

- **Step 1.** Solve the linear relaxation of a RMP (see Section 4.2) or an A-RMP (see Section 4.3).
- **Step 2.** Solve the pricing sub-problems to find new columns with negative reduced cost (see Section 4.2). If any columns are found, add them to the RMP/A-RMP and go to step 1. Otherwise, go to step 3.
- **Step 3.** Generate valid inequalities (see Section 4.4) to strengthen the linear relaxation. If any violated inequalities are found, add them to the RMP/A-RMP and go to Step 1. Otherwise, go to step 4.
- **Step 4.** If an integer solution of the RMP is obtained, the current node is fathomed. Otherwise, perform the branching rules and add the child nodes to the set of unprocessed branch nodes. Mark the current node as processed node and use the branch-and-bound framework to explore unprocessed branch node by repeating the above steps.

4.1. Two-layer network construction

We first construct a two-layer network, as shown in Fig. 2. The lower layer is a road-network graph $G = (N, A)$ as described in Section 3.1. The upper layer gives dynamically generated customer-based graphs. Specifically, for each vehicle $k$ on each day $s$, we generate a customer-based graph $G^t_k = (N^t_k, A^t_k)$. The node set $N^t_k$ only consists of the depot node $o$ and customer nodes on the day $s$, and $A^t_k$ represents the set of arcs that connect every two nodes in set $N^t_k$.

Unlike most studies using a predefined multi-graph to handle the road network information, our method generates a series of customer-based graphs dynamically. The reason for adopting this dynamic generation method is that in our problem, the cost of the path depends on the attributes of the road network and is affected by the solutions for other days, i.e., path consistency over days. Hence, we update the customer-based graphs in each iteration of the column generation. The details are presented in Section 4.2.

In the two-layer network, the pricing sub-problem, separation of valid inequalities, and branching are performed on the customer-based graphs. The found routes based on the customer-based graphs are projected onto the road-network graph to obtain the corresponding paths.
4.2. Column generation and pricing problem

We use a column generation procedure to solve the linear relaxation of the set partitioning model, (1)–(5), referred to as the master problem. In each iteration, a linear relaxation of the restricted master problem (RMP), where only a subset of columns (paths) considered, is first solved. Then pricing sub-problems are solved for each day \( s \) and each vehicle \( k \) to identify new feasible columns with negative reduced costs. The columns with negative reduced costs are added to the RMP. If no new columns can be found, the column generation procedure terminates. The obtained solution is an optimal solution for the linear relaxation of the RMP and provides a lower bound for the set partitioning problem.

In our implementation, the pricing problem is decomposed into several problems, one for each vehicle each day. Each pricing problem is solved in three steps. Firstly, we update the upper-layer customer-based graph \( G^u_k \) for vehicle \( k \) on day \( s \). Secondly, the Shortest Path Problem with Resource Constraints (SPPRC) with \( ng \)-routes is solved based on the customer-based graph, aiming to find feasible routes with negative reduced costs. Thirdly, the found new routes are converted to paths on the road-network graph \( G = (N, A) \).

Let \( \pi_i^j \) and \( \mu_{jk}^i \) denote the dual variables corresponding to constraints (2) and (3). The modified cost for arc \((i, j) \in A\) on the lower-layer graph (road-network graph) is expressed as Eq. (8).

\[
\tilde{e}_{ij} = \begin{cases} 
  e_{ij} & (i, j) \in A^u \backslash A^s \\
  e_{ij} - \pi_j - \sum_{k} \pi_k^j & (i, j) \in A^s 
\end{cases}
\]  

To update the arcs in the upper-layer graph, we project all the nodes \( i \in N_k^u \) onto the lower-layer road-network graph, and obtain the shortest paths between each two of them based on the modified cost \( \tilde{e}_{ij} \). Let \((i, i_1, i_2, \ldots, i_m, j)\) be the node sequence of the shortest path between node \( i \) and \( j \). For each arc \((i, j) \in A^u_k \) on the upper-layer graph, we update its corresponding path on the lower-layer graph as \( P_j = (i, i_1, i_2, \ldots, i_m, j) \) and its cost as \( \tilde{e}_{ij} = e_{i_j} + \sum_{n} \mu_{i_ni_{n+1}} \).

After updating the upper-layer network, we then solve the SPPRC based on that to find new routes with a negative reduced cost. Let notation \( r \) represent the index of a route on the upper-layer network, and \( A(r) = \{(i_1, i_2), \ldots, (i_{n-1}, i_n)\} \) represents the set of arcs traversed in route \( r \). Let parameter \( a_r \) be the number of times that customer \( i \) is served in route \( r \). The reduced cost for route \( r \) is calculated by expressions (9):

\[
\tilde{e}_i = \sum_{(i, j) \in A(r)} \tilde{e}_{ij} - \sum_{k} \pi_k^j \sum_{(i, j) \in A(r)} \tilde{e}_{ij} = \sum_{(i, j) \in A(r)} \tilde{e}_{ij} - \sum_{k} \pi_k^j i_r
\]  

The modified cost for arc \((i, j) \in A^u_k \) is:

\[
\tilde{e}_{ij} = e_{ij} - \pi_j^i.
\]  

We allow for non-elementary routes when solving the pricing problem, as the Elementary Shortest Path Problem with Resource Constraints (ESPPRC) is still a difficult-to-solve (\( \mathcal{NP} \)-hard) problem (Dror, 1994). Using a non-elementary path does not change the problem’s optimal solution but weakens its LP relaxation. A Compromise between elementary and non-elementary routes involves methods including the SSPRC forbidden k-cycle (Christofides et al., 1981; Imich and Villeneuve, 2006; Fukasawa et al., 2006) and the SSPRC with \( ng \)-routes (Baldacci et al., 2011).

In this study, we find the routes with the negative reduced cost through the dynamic programming \( ng \)-route algorithm proposed by Martello et al. (2014) and use Decremental Search Space Relaxation (DSSR) and completion bounds to speed up the algorithm. The found routes are added to the corresponding route set \( K_k \). This procedure is done on the upper-layer customer-based graph. All the found routes are then converted to paths on the road network by projecting the routes onto the lower-layer road-network. We use a function \( p = R(r, G^u_k, G) \) represents that the path \( p \) is obtained by projecting the route \( r \) on the customer-based graph \( G^u_k \) onto the road-network graph \( G \). A graphical example of the projection is provided in Fig. 3.

The \( ng \)-route relaxation

The \( ng \)-routes are restricted non-elementary routes. For each customer node \( i \in N^u_k \setminus \{o\} \), there is a predefined \( ng \)-set \( N^u_k \) associated with it, which usually includes a certain number of nearest customers of \( i \) (including itself). When building a route \( r \), there is a dynamically computed set \( II(r) \). If a customer \( i \) belongs to this set, it is forbidden to extend route \( r \) to customer \( i \). Let \( V(r) \) be the set of costumers visited in route \( r \), \( V(r) = \{i_1, \ldots, i_{n-1}, i_n\} \). The forbidden set \( II(r) \) is calculated by Eq. (11).

\[
II(r) = \{i_m \in V(r) \setminus \{i_o\} : i_m \in \bigcap_{h=m+1}^{n} N^u_h \} \cup \{i_o\}
\]  

The size of the \( ng \)-sets \( (A_{\text{ng}}) \) is an important parameter that affects the quality of the solution and the complexity of the problem. The larger the \( A_{\text{ng}} \) is, the larger the smallest cycles that may appear in a route, but the complexity of the problem also increases. Let \( g(r) = \sum_{(i, j) \in V} q^j \) be the total quantity of demands which are already delivered to customers in current route \( r \). The label associated with the route \( r \) is expressed as \( L(r) = (i_o, q, II(r), \tilde{e}_i) \), where \( i_o \) is the last node of the route. If a customer node \( i_{n+1} \) is not in \( II(r) \) and satisfy \( g(r) + q_{n+1} \leq Q \), then the extension function to \( i_{n+1} \) is shown as Eq. (12).

\[
L(r') = (i_{n+1}, q, \tilde{e}_i, \sum_{h=n+1}^{\infty} II(r) \cap N^u_{i_{n+1}} \cup \{i_{n+1}\}, \tilde{e}_i + \tilde{e}_{i_{n+1}})
\]  

Starting from an initial label associated with the depot node \( o \), labels are computed based on the extension function (12). To reduce the number of possible routes, a dominance rule is applied to discard unpromising labels. Consider two feasible partial routes
Fig. 3. Example of converting a route to a path.

$r_1$ and $r_2$ ending at a same node, $r_1$ is dominated by $r_2$ if all possible extensions from $r_2$ can also be done from $r_1$ with a lower reduced cost. That means, the following three conditions hold and at least one of them is not equal:

- $q(r_1) \leq q(r_2)$,
- $\hat{c}_{r_1} \leq \hat{c}_{r_2}$, and
- $\Pi(r_1) \subseteq \Pi(r_2)$.

**Dynamic programming ng-route algorithm**

An exact dynamic programming algorithm for ng-route based on the non-dominance principle is presented. The algorithm creates a matrix $\mathcal{M}$ with the size of $(Q + 1) \times |N_s^i|$. Each entry of $\mathcal{M}(q, i)$ is a bucket storing labels of non-dominated routes starting from the depot $o$ and ending at node $i$ with loading quantity $q(r) = q$. The dynamic programming starts from the label in bucket $\mathcal{M}(0, o)$ and then fills the matrix $\mathcal{M}$ by running from $q = 0$ up to $q = Q$.

Note that we only test the dominance rule for the labels in the same bucket, i.e., labels with the same end node $i$ and capacity $q(r)$. Labels using less capacity are unlikely to dominate others using higher capacity. The dynamic programming algorithm is summarized in Algorithm 1.

**Algorithm 1 Dynamic Programming ng-Route Algorithm**

**Input:** customer-based graph $G^i_s$, matrix $\mathcal{M}$, $\Delta_{iq}$, ng-sets $N_j^i$ for each customer $i$

**Output:** the best ng-routes

1: $\mathcal{M}(q, i) \leftarrow \emptyset$, $\forall q \in \{0, 1, \ldots, Q\}, i \in N_s^i$

2: $\mathcal{M}(0, o) \leftarrow \{(o, 0, \{o\}, 0)\}$

3: for $q := 0, 1, 2, \ldots, Q$ do

4: for all $i \in N_s^i$ do

5: for all $L(r_i) \in \mathcal{M}(q, i)$ do

6: for all $j \in N_s^i$ do

7: if $j \notin H(r_i)$ and $q + q_j \leq Q$ then

8: $L(r_j) := (j, q + q_j, H(r_i) \cap N_j^i \cup \{j\}, \hat{c}_{r_i} + \hat{c}_{ij})$

9: insertLabel $\leftarrow \text{true}$

10: for all $L(r_j') \in \mathcal{M}(q + q_j, j)$ do

11: if $L(r_j)$ dominates $L(r_j')$ then

12: delete $L(r_j')$

13: else if $L(r_j')$ dominates $L(r_j)$ then

14: insertLabel $\leftarrow \text{false}$

15: break

16: if insertLabel then

17: $\mathcal{M}(q + q_j, j) \leftarrow \mathcal{M}(q + q_j, j) \cup L(r_j)$

18: return the best routes corresponding to $\mathcal{M}$
Decremental state-space relaxation

The main idea of decremental state-space relaxation (DSSR) is using an iterative method to obtain the ng-routes. A subset of $\mathcal{N}_r$, donated as $\mathcal{R}_r^{\text{ng}}$, is used at iteration $itr$ instead of the ng-set $\mathcal{N}_r$. In particular, $\mathcal{R}_r^0$ is an empty set. At each iteration, Algorithm 1 is executed with $\mathcal{R}_r^{\text{ng}}$ as input instead of $\mathcal{N}_r$. If there exist repeated vertices violating the original ng-set $\mathcal{N}_r$ in the obtained best routes, we add the repeated vertices into $\mathcal{R}_r^{itr+1}$ and execute Algorithm 1 again. The procedure repeats until the ng-routes are found.

Completion bounds

To further speed up the algorithm, we adopt a completion bounds technique, which can terminate the extension of routes with no chance leading to a negative reduced cost in advance. The completion bounds are calculated at each iteration of DSSR. Let $\mathcal{T}_q^*(q,i)$ be the reduced cost of the best route starting from node $i$ and ending at the depot with a total demand of $q$ at iteration $itr$. The completion bound of bucket $\mathcal{M}(q,i)$ at iteration $itr$ is calculated as Eq. (13).

$$\hat{T}_{q,i}(q,i) = \min_{q' \leq q} \{\mathcal{T}_{itr}(q',i)\}$$ (13)

These bounds are then used at iteration $(itr + 1)$. Specifically, the label $L(r)$ at iteration $(itr + 1)$ can be extended to node $i_{itr+1}$ only if following condition holds:

$$\hat{c}_r + \hat{T}_{q,i}(Q - q(r),i_{itr+1}) < 0.$$ (14)

In other words, if the left hand of Eq. (14) is equal to or larger than 0, this extension is discarded as it cannot lead to any route with negative reduced cost.

Heuristic pricing

Even with the DSSR and completion bounds techniques described above, it still requires a relatively long time to solve the ng-route pricing problem exactly. We accelerate the column generation procedure by adopting a faster heuristic algorithm, a truncated version of the proposed exact algorithm. In the heuristic algorithm, each bucket $\mathcal{M}(q,i)$ only stores a fixed (small) number of labels, e.g., only the labels with the least reduced costs are stored. Note that revisiting customers is forbidden in the heuristic algorithm. The exact algorithm is called only when heuristics can find no new routes with a negative reduced cost.

4.3. Aggregation technique

In our problem, vehicles cannot share one column pool as standard VRPs. The reasons include (1) the customers and demands vary from day to day, and (2) the indexes for vehicles are needed as the path consistency exists only when a segment is traversed by a particular vehicle (not different vehicles) every day. Therefore, each vehicle for each day needs its column pool $P_k^s$ theoretically (See also the model in Section 3.2).

However, using a specialized column pool for each vehicle each day leads to repeated calculation and increased computational time. To address this issue, we propose an aggregation technique. An aggregated version of RMP is defined and donated as A-RMP, which only considers the aggregated version of consistency constraints, i.e., using Eqs. (15) and (16) instead of Eqs. (3) and (5).

$$\sum_{p \in P_s} b_{ijp} \bar{x}_{ij}^p - y_{ij} \geq 0, \forall (i, j) \in A^*$$ (15)

$$y_{ij} \in Z^+, \forall (i, j) \in A^*$$ (16)

In the linear relaxation of the A-RMP, dual variables $\mu_{ij}$ are associated with the constraints (15). Note that the index $k$ of dual variables is removed compared to that in the standard RMP. The pricing problem is decomposed to a serial of problems for each day. For each of them, we construct a customer-based graph $G^i = (N^i, A^i)$ and update it at each iteration of the column generation. Other steps for the pricing problems are similar to that in Section 4.2. The new paths found are added to the column pool $P_k^s$, which is shared by different vehicles. When solving the A-RMP’s linear relaxation, only $|S|$ pricing sub-problems need to be solved at each iteration of the column generation, compared to $|S| \times |k|$ in the RMP.

During the branch-price-and-cut procedure, the linear relaxation of the A-RMP is solved first, and the linear relaxation of the RMP is solved only when the integer solution of the linear relaxation of the A-RMP is already obtained. Specifically, when an integer solution for the linear relaxation of the A-RMP is obtained at a branch-and-bound node, we check if this solution can convert to an integer feasible solution for the RMP. We use a simple MILP model, Eqs. (17)–(25), to separate the aggregated sets and variables. The $P_k^s$ is the set of paths that are used on day $s$ according to the solution of A-RMP, i.e., $x_{ij}^s = 1, \forall p \in P_k^s, s \in S$. Constraints (18) ensure the total number of vehicles that each path assigned to is 1. Constraints (19) ensure that the total number of paths that each vehicle performing is 1. Constraints (20) and (21) assign the consistent parts of paths to certain vehicles. The $\phi_{pk}^s$ is an auxiliary binary variable. As the objective is to minimize the total value of $\phi_{pk}^s$, the value of $\phi_{pk}^s$ will be equal to the nearest integer rounded up from $x_{pk}^s$.

$$\min \sum_{p \in S} \sum_{k \in K} \sum_{p \in P_k^s} \phi_{pk}^s$$ (17)
\[
\sum_{k \in K} \lambda^s_{pk} = 1, \forall p \in P^{rs}, s \in S \tag{18}
\]

\[
\sum_{p \in P^{rs}} \lambda^s_{pk} = 1, \forall k \in K, s \in S \tag{19}
\]

\[
\sum_{k \in K} y_{ijk} = y_{ij}, \forall (i, j) \in A^s \tag{20}
\]

\[
\sum_{p \in P^{rs}} b_{ijl} \lambda^s_{pk} \geq y_{ijk}, \forall (i, j) \in A^s, k \in K, s \in S \tag{21}
\]

\[
\phi^s_{p} \geq \lambda^s_{pk}, \forall p \in P^{rs}, k \in K, s \in S \tag{22}
\]

\[
\phi^s_{p} = \{0, 1\}, \forall p \in P^{rs}, k \in K, s \in S \tag{23}
\]

\[
y_{ijk} \in \mathbb{R}^+, \forall (i, j) \in A^s, k \in K \tag{24}
\]

\[
\lambda^s_{pk} \in \mathbb{R}^+, \forall p \in P^{rs}, k \in K, s \in S \tag{25}
\]

If all variables \( \lambda^s_{pk} \) are integers in the solution, it means the integer solution of A-RMP could be converted to an integer solution to the RMP. Otherwise, it indicates that at least one route is assigned to more than one vehicle. Then, further branching is needed, which is discussed in Section 4.5.

By using this technique, we do not need to solve the linear relaxation of the RMP at each branch-and-bound node. Instead, it is only solved after the integer solutions of the A-RMP’s corresponding linear relaxation are found. Thus, the number of column generation iterations and computational time is reduced.

### 4.4. Valid inequalities

To strengthen the linear relaxation bound of the proposed set partitioning model, we further add round capacity inequalities (RCIs). RCIs ensure that all subsets of the demand are served by sufficient vehicles. The separation routines for RCIs are performed based on the customer-based graph. For each day \( s \), let \( W \) be a subset of \( N^s \); \( \delta^s(W) \) be the set of arcs having exactly one end-node inside \( W \); \( q^s(W) \) be the total demand of set \( W \); and \( e_{ijr} \) be a parameter which equals 1 if arc \((i, j) \in A^s\) is traversed by route \( r \), equals 0 otherwise. The value of variable related to routes \((\lambda^s_{rk} \text{ and } \lambda^s_{sj})\) equals the corresponding value for paths, as shown in Eqs. (26) and (27), where the path \( p = \bar{P}(r, G^s_k, G) \) represents the projection of route \( r \) on the road-network graph (see Section 4.2). The RCI is then expressed as Eq. (28) and its corresponding dual variables are given as \( \gamma^s_W \).

\[
\lambda^s_{rk} = \lambda^s_{sj}, \forall k \in K, s \in S, p \in P^{s}_k, p = \bar{P}(r, G^s_k, G) \tag{26}
\]

\[
\lambda^s_{jr} = \lambda^s_{pr}, \forall s \in S, p \in P^{s}, p = \bar{P}(r, G^s_k, G) \tag{27}
\]

\[
\sum_{r \in R^s_k} \sum_{(i,j) \in \delta^s(W)} e_{ijr} \lambda^s_{jr} \geq 2 \cdot \left\lfloor \frac{q^s(W)}{Q} \right\rfloor, \forall W \subseteq N^s, s \in S \tag{28}
\]

Note that RCIs do not alter the structure of pricing sub-problems. Their dual variables could be transferred directly on the modified arc costs, i.e., all we need to do is to change the expression of \( \hat{e}_{ij} \) from Eq. (10) to Eq. (29).

\[
\hat{e}_{ij} = e_{ij} - \hat{e}_{ij} - \sum_{(i,j) \in \delta^s(W)} \hat{e}_{ij} \gamma^s_W \tag{29}
\]

For the sake of computational tractability, we use a greedy heuristic algorithm inspired by Augerat et al. (1998) to separate the RCIs. On the graph \( G^s = (N^s, A^s) \), each arc \((i, j) \in A^s\) is associated with a weight \( \hat{e}_{ij} \), which is calculated by \( \hat{e}_{ij} = \sum_{r \in R^s} e_{ijr} \lambda^s_{jr} \text{ or } \hat{e}_{ij} = \sum_{r \in R^s} \sum_{k \in K} e_{ijr} \lambda^s_{rk} \). For any subset \( F \) of \( A^s \), we use \( \bar{x}(F) \) to represent the summation of the \( \hat{e}_{ij} \) for all \((i, j) \in F \).

The details of the RCI separation are described as follows.
1. We construct a support graph for each day. The support graph is a sub-graph of $G^i$ and donated as $G^{i*} = (N^i, A^{i*})$, where $A^{i*} = \{(i, j) \in A^i : \bar{x}_{ij} > 0\}$.

2. We shrink the support graph to decrease the size of the separation problem. In particular, the customer node $i$ and $j$ can be shrunk to a super vertex $v$ if $\bar{x}_{ij} + \bar{x}_{ji} = 1$. The demand of $v$ is calculated by $q_i^v = q_i^v + q_j^v$, and the weight of arc $(v, u)$ is defined as $\bar{x}_{iu} = \bar{x}_{iu} + \bar{x}_{ju}$. We iteratively choose nodes (except the depot) and shrink them to single super vertices until no more node pairs can be shrunk.

3. A greedy heuristic is used to construct the candidate customer sets $W$. Each set $W$ is developed from an initial set with one node or super vertex. Then a node $s_{n+1}$ different from the depot is added to set $W$ iteratively until the set contains all the nodes. The selected node is the one with maximal $\bar{x}(\delta(W : \{s_{n+1}\}))$, where $\delta(W : \{s_{n+1}\})$ represents the set of arcs with one side in $W$ and another side is $s_{n+1}$.

4. All the sets $W$ obtained during the iteration are the candidate customer sets and used to check Eq. (28). If a set $W$ violates the constraint, one valid RCI is found.

4.5. Branching strategy

The column generation and valid inequalities separation procedures are used to obtain the lower bounds of the problem. By embedding them within a branch-and-bound scheme, it can be used to solve an integer program. The scheme defines how to subdivide the feasible area associated with the node in the branch-and-bound tree when the linear relaxation solution of the RMP associated with the node is fractional. The subdivision produces two “child nodes” for the “parent node”.

As stated in Section 4.3, there are two RMP versions, the A-RMP based on $G^i$ and the normal RMP based on $G^i_k$. We define an aggregation flag, $A_{\text{FLAG}} = [0, 1]$, for each branch-and-bound node. $A_{\text{FLAG}} = 1$ when no integral solution to A-RMP has been found in the current branch-and-bound node and its parent node; otherwise, it is 0. This flag’s initial value is set as one, and its parent node inherits the value of the child node. Next, we present the branching rules used in our scheme.

Let $\bar{x}_{ij}^s$ be the flow on arc $(i, j) \in G^i_k$ and $\bar{x}_{ij}^k$ be the flow on arc $(i, j) \in G^i$, $\bar{x}_{ij}^k = \sum_{s \in S} \bar{x}_{ij}^s$. In our implementation, we apply the following three different branching rules.

- Rule 1: Branching on the number of vehicles used in the solution.
- Rule 2: Branching on the aggregated variable $\bar{x}_{ij}^k$ on the graph $G^i$.
- Rule 3: Branching on the variable $\bar{x}_{ij}^s$ on the graph $G^i_k$.

The first branching rule is to add/change fleet size constraints when the number of vehicles used in the solution is fractional. In the branching rule 2, we select an arc $(i, j)$ on graph $G^i$ such that $0 < \bar{x}_{ij}^k < 1$. Then two child nodes are derived that in one the arc cannot belong to the solution ($\bar{x}_{ij}^k = 0$); and in the other one the arc is enforced in the solution ($\bar{x}_{ij}^k = 1$). A similar procedure for branching rule 3 is performed on the arcs in $G^i_k$. If there are several arcs with fractional flow, we choose the arc of which the value of $\bar{x}_{ij}^s$ or $\bar{x}_{ij}^k$ is closest to 0.5.

The rule 1 usually has stronger impact on the solution that it is given higher priority compared to other two rules. Therefore, we perform rule 2 or 3 after obtaining a solution with integer number of used vehicles. When exploring a branch-and-bound node with integer number of used vehicles, there are three possible situations.

1. If $A_{\text{FLAG}} = 1$ and there exists any $\bar{x}_{ij}^s$ being fractional, we perform the branching rule 2.
2. If $A_{\text{FLAG}} = 1$ and the flows on all arcs of graph $G^i, \forall s \in S$ are integer, we solve the assignment model (17)–(25) to separate the aggregated variables. If the solution to the assignment model is also integral, then a feasible solution of the RMP is obtained and this branch is fathomed. Otherwise, we set the $A_{\text{FLAG}} = 0$, calculate the arc flow by $\bar{x}_{ij}^k = \sum_{s \in S} \epsilon_{ij}^s \bar{x}_{ij}^s$, generate copied route set for each vehicle $k$ on day $s$ by $R_k^s = R^s$, and perform the branching rule 3.
3. If $A_{\text{FLAG}} = 0$ and there exists any $\bar{x}_{ij}^k$ being fractional, the branching rule 3 is performed.

During the exploration of the branch-and-bound tree, the node with the lowest lower bound is selected. Upper bounds are updated when integer solutions to the linear relaxation of RMP are found. We also solve the RMP (or A-RMP) as an integer program after the column generation and valid inequalities separation at each node. The upper bound is updated when the obtained integer solution is smaller than the current upper bound.

5. Experimental design

We construct the road network based on the urban area of West Jordan, Utah, the United States, as shown in Fig. 4(a). The road segments in this area are classified into five categories, from high to low level. They are the expressway, principal arterial, major arterial, minor arterial, and zone connector. In our application, we consider the first three types as the arterial roads and the last two as the residential roads described in Section 3.1. To test our algorithm performance on different instance sizes, we generate three road networks, as shown in Fig. 4(b)–(d). The details of them are presented in Table 3, where the columns “|N|” and “|A|” show the total number of nodes and arcs in the road network, and “|C|” shows the total number of candidate customer nodes.

We define several different sizes for the scenarios (days), those are the small(S) road network with 7 customers, the medium(M) road network with 10, 15, and 20 customers, and the large(L) road network with 10, 15, 20, and 35 customers. The number of days considered in each instance varies from 3 to 5, representing 5 workdays in one week. For the sake of clarity, we name the instances
based on their settings. For instance, “M-10-5(1)’’ represents the instance based on the medium road network and considering five days with ten customers on each day. For each setting of the problem size, we generate five instances with different customers and demands, so the number in the brackets, e.g.”(1)”, is the instance’s index. To have instances with road network information which are closer to real-world applications, the way of generating the data set in our study has some differences with that in previous studies (Groër et al., 2009; Spliet and Gabor, 2015). In our test network, all the customer nodes are real locations with demands. So we take all the nodes as candidates and randomly choose a certain number of customers to generate different scenarios. Note that by this instance generation approach, it can happen that some customer candidates might not appear at all at any day. Each customer’s demand on each day is drawn from a normal distribution with an expectation of 50 and a variance of 10 and rounding it to a positive integer. The capacity of vehicles is set as 300, and the available fleet size is set as infinity.

6. Experiment results

In this section, we present and discuss the results of our numerical experiments. The branch-price-and-cut algorithm is implemented in PYTHON on an Intel Core i7 CPU, 2.20 GHz with 16 GB of memory. CPLEX 12.6 is used as the LP and MIP solver. For all experiments, the time limit is set equal to 3 h.

6.1. Performance of the valid inequalities

To analyze the impact of the valid inequalities on our algorithm, we test the algorithm without and with valid inequalities on a set of instances, respectively. We choose the instances based on the medium network and with ten customers each day. The consistency discount equals 0.1. Table 4 reports the results of experiments. The column labeled “Inst.” shows the setting and index of instances. The column “T.Time(s)” presents the computational time in seconds. The columns “LP Gap(%)”, “R. Gap(%)”, and
Table 4
Comparison of the algorithm without and with valid inequalities.

<table>
<thead>
<tr>
<th>Inst.</th>
<th>Without valid inequalities</th>
<th>With valid inequalities</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>T.Time (s)</td>
<td>LP Gap (%)</td>
</tr>
<tr>
<td>M-10-3(1)</td>
<td>10800.00</td>
<td>2.53</td>
</tr>
<tr>
<td>M-10-3(2)</td>
<td>2033.48</td>
<td>1.76</td>
</tr>
<tr>
<td>M-10-3(3)</td>
<td>10800.00</td>
<td>2.49</td>
</tr>
<tr>
<td>M-10-3(4)</td>
<td>10800.00</td>
<td>3.18</td>
</tr>
<tr>
<td>M-10-3(5)</td>
<td>91.86</td>
<td>0.99</td>
</tr>
<tr>
<td>M-10-4(1)</td>
<td>10800.00</td>
<td>2.78</td>
</tr>
<tr>
<td>M-10-4(2)</td>
<td>10800.00</td>
<td>1.58</td>
</tr>
<tr>
<td>M-10-4(3)</td>
<td>10800.00</td>
<td>1.42</td>
</tr>
<tr>
<td>M-10-4(4)</td>
<td>10800.00</td>
<td>3.45</td>
</tr>
<tr>
<td>M-10-4(5)</td>
<td>366.08</td>
<td>1.45</td>
</tr>
<tr>
<td>M-10-5(1)</td>
<td>10800.00</td>
<td>2.28</td>
</tr>
<tr>
<td>M-10-5(2)</td>
<td>10800.00</td>
<td>2.20</td>
</tr>
<tr>
<td>M-10-5(3)</td>
<td>10800.00</td>
<td>1.58</td>
</tr>
<tr>
<td>M-10-5(4)</td>
<td>10800.00</td>
<td>3.56</td>
</tr>
<tr>
<td>M-10-5(5)</td>
<td>6082.09</td>
<td>1.17</td>
</tr>
</tbody>
</table>

Table 5
Comparison of the algorithm without and with the aggregation technique.

<table>
<thead>
<tr>
<th>Inst.</th>
<th>Without aggregation technique</th>
<th>With aggregation technique</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>T.Time (s)</td>
<td>Gap (%)</td>
</tr>
<tr>
<td>L-10-3(1)</td>
<td>610.69</td>
<td>0</td>
</tr>
<tr>
<td>L-10-3(2)</td>
<td>136.66</td>
<td>0</td>
</tr>
<tr>
<td>L-10-3(3)</td>
<td>877.46</td>
<td>0</td>
</tr>
<tr>
<td>L-10-3(4)</td>
<td>496.30</td>
<td>0</td>
</tr>
<tr>
<td>L-10-3(5)</td>
<td>650.46</td>
<td>0</td>
</tr>
<tr>
<td>L-10-4(1)</td>
<td>928.62</td>
<td>0</td>
</tr>
<tr>
<td>L-10-4(2)</td>
<td>465.63</td>
<td>0</td>
</tr>
<tr>
<td>L-10-4(3)</td>
<td>904.97</td>
<td>0</td>
</tr>
<tr>
<td>L-10-4(4)</td>
<td>1434.47</td>
<td>0</td>
</tr>
<tr>
<td>L-10-4(5)</td>
<td>9725.24</td>
<td>0</td>
</tr>
<tr>
<td>L-10-5(1)</td>
<td>2574.05</td>
<td>0</td>
</tr>
<tr>
<td>L-10-5(2)</td>
<td>1107.08</td>
<td>0</td>
</tr>
<tr>
<td>L-10-5(3)</td>
<td>4938.93</td>
<td>0</td>
</tr>
<tr>
<td>L-10-5(4)</td>
<td>9873.20</td>
<td>0</td>
</tr>
<tr>
<td>L-10-5(5)</td>
<td>5860.22</td>
<td>0</td>
</tr>
</tbody>
</table>

“Gap(%)” show, respectively, the gap between the value of the LP relaxation and the best-found solution, the gap after adding RCIs, and the gap between lower bound and best-found solution when the algorithm terminates. All gaps in the table are expressed in percentages. The columns “Node” presents the number of nodes processed in the branch-and-bound tree. Finally, the column labeled “RCI” shows the total number of found valid RCIs.

In Table 4, we observe that the algorithm without valid inequalities (i.e., the branch-and-price algorithm) can only solve four out of 15 instances to optimality, with the average computation time of 2143.38 s. By including the valid inequalities, the average computation time of these four instances drops to 26.27 s, which is only 1.23% of the computation time before adding the valid inequalities; the corresponding average branch-and-bound tree size decreases from 1177.5 to 1.25. The other 11 instances are also solved to optimality within 3 h by the algorithm with valid inequalities. On average, 18.2 valid inequalities are found by the round capacity separation method. The average LP gap of 2.16% is improved to 0.08% (R.gap) after adding RCIs.

6.2. Performance of the aggregation technique

We also conduct computational experiments to investigate the performance of the aggregation technique. In Table 5, we report the experiments’ results using the branch-price-and-cut algorithm without and with the proposed aggregation technique. We perform the algorithms on the instances with ten customers based on the large network and set the discount value as 0.1.

Both the algorithm without and with the aggregation technique can solve all of the 15 instances to optimality. However, the average computational time drops from 2705.60 s to 417.93 s by utilizing the aggregation technique, which is a decrease of 84.55%. At the same time, the average size of branch-and-bound trees is reduced from 10.33 to 5.4. The aggregation technique also reduces the average computation time at each branch-and-bound node.
In this subsection, we solve the arc-flow model (see Appendix) by using CPLEX and compare the results with that of the proposed branch-price-and-cut algorithm. A group of instances with up to 20 customers (per day), 5 days, and $\theta = 0.1$ are tested. For comparison, the time limit for both CPLEX and the branch-price-and-cut algorithm is set as 3 h. The detailed results and computing time are shown in Table 6. The results show that CPLEX can only find the optimal solution for instances with 7 customers, and the computational time is hundreds of times longer than that of the branch-price-and-cut algorithm. The proposed branch-price-and-cut algorithm is able to obtain the optimal solution for larger instances (up to 15 customers per day). When the number of customers reaches 20, both CPLEX and the branch-price-and-cut algorithm cannot obtain the optimal solution. However, the final solution gaps of the branch-price-and-cut algorithm are much smaller compared to that of CPLEX.

Table 6
Comparison of the solutions solved by CPLEX and Branch-price-and-cut Algorithm.

<table>
<thead>
<tr>
<th>Inst.</th>
<th>CPLEX T.Time (s)</th>
<th>Gap (%)</th>
<th>Branch-price-and-cut T.Time (s)</th>
<th>Gap (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>S-7-3(1)</td>
<td>63.94</td>
<td>0</td>
<td>0.62</td>
<td>0</td>
</tr>
<tr>
<td>S-7-4(1)</td>
<td>215.07</td>
<td>0</td>
<td>0.78</td>
<td>0</td>
</tr>
<tr>
<td>S-7-5(1)</td>
<td>442.22</td>
<td>0</td>
<td>2.76</td>
<td>0</td>
</tr>
<tr>
<td>M-10-3(1)</td>
<td>10 800.00</td>
<td>4.36</td>
<td>24.04</td>
<td>0</td>
</tr>
<tr>
<td>M-10-4(1)</td>
<td>10 800.00</td>
<td>5.33</td>
<td>47.70</td>
<td>0</td>
</tr>
<tr>
<td>M-10-5(1)</td>
<td>10 800.00</td>
<td>5.49</td>
<td>99.44</td>
<td>0</td>
</tr>
<tr>
<td>M-15-3(1)</td>
<td>10 800.00</td>
<td>6.11</td>
<td>297.79</td>
<td>0</td>
</tr>
<tr>
<td>M-15-4(1)</td>
<td>10 800.00</td>
<td>6.42</td>
<td>1006.28</td>
<td>0</td>
</tr>
<tr>
<td>M-15-5(1)</td>
<td>10 800.00</td>
<td>5.51</td>
<td>1804.37</td>
<td>0.59</td>
</tr>
<tr>
<td>M-20-3(1)</td>
<td>10 800.00</td>
<td>10.98</td>
<td>297.79</td>
<td>0.63</td>
</tr>
<tr>
<td>M-20-4(1)</td>
<td>10 800.00</td>
<td>11.81</td>
<td>1804.37</td>
<td>1.08</td>
</tr>
</tbody>
</table>

Table 7
Branch-price-and-cut results with consistency discount as 0.1.

<table>
<thead>
<tr>
<th>Inst.</th>
<th>T.Time (s)</th>
<th>#Opt.</th>
<th>LP Gap (%)</th>
<th>R. Gap (%)</th>
<th>Gap (%)</th>
<th>Nodes</th>
<th>RCI</th>
<th>#Veh.</th>
</tr>
</thead>
<tbody>
<tr>
<td>S-07–3</td>
<td>1.72</td>
<td>5/5</td>
<td>0.62</td>
<td>0.07</td>
<td>0</td>
<td>1.4</td>
<td>0.8</td>
<td>2</td>
</tr>
<tr>
<td>S-07–4</td>
<td>3.18</td>
<td>5/5</td>
<td>0.58</td>
<td>0</td>
<td>0</td>
<td>1.4</td>
<td>1.2</td>
<td>2</td>
</tr>
<tr>
<td>S-07–5</td>
<td>5.45</td>
<td>5/5</td>
<td>0.86</td>
<td>0.10</td>
<td>0</td>
<td>1.6</td>
<td>2.4</td>
<td>2</td>
</tr>
<tr>
<td>M-10–3</td>
<td>29.83</td>
<td>5/5</td>
<td>2.19</td>
<td>0.10</td>
<td>0</td>
<td>5.8</td>
<td>13.6</td>
<td>2</td>
</tr>
<tr>
<td>M-10–4</td>
<td>46.66</td>
<td>5/5</td>
<td>2.14</td>
<td>0.06</td>
<td>0</td>
<td>3.6</td>
<td>16.4</td>
<td>2</td>
</tr>
<tr>
<td>M-10–5</td>
<td>95.80</td>
<td>5/5</td>
<td>2.12</td>
<td>0.08</td>
<td>0</td>
<td>3.4</td>
<td>24.6</td>
<td>2</td>
</tr>
<tr>
<td>M-15–3</td>
<td>517.27</td>
<td>5/5</td>
<td>1.51</td>
<td>0.39</td>
<td>0</td>
<td>34.6</td>
<td>25.8</td>
<td>3</td>
</tr>
<tr>
<td>M-15–4</td>
<td>2071.98</td>
<td>5/5</td>
<td>1.43</td>
<td>0.26</td>
<td>0</td>
<td>19</td>
<td>44.8</td>
<td>3</td>
</tr>
<tr>
<td>M-15–5</td>
<td>4698.11</td>
<td>4/5</td>
<td>1.40</td>
<td>0.20</td>
<td>0.04</td>
<td>63.6</td>
<td>77.4</td>
<td>3</td>
</tr>
<tr>
<td>M-20-3</td>
<td>6913.74</td>
<td>2/5</td>
<td>3.29</td>
<td>0.69</td>
<td>0.49</td>
<td>261</td>
<td>395</td>
<td>4</td>
</tr>
<tr>
<td>M-20-4</td>
<td>10 800.00</td>
<td>0/5</td>
<td>4.29</td>
<td>1.51</td>
<td>1.23</td>
<td>241.4</td>
<td>377.8</td>
<td>4</td>
</tr>
<tr>
<td>M-20-5</td>
<td>10 800.00</td>
<td>0/5</td>
<td>4.41</td>
<td>1.82</td>
<td>1.61</td>
<td>165.2</td>
<td>370.6</td>
<td>4</td>
</tr>
<tr>
<td>L-10–3</td>
<td>114.59</td>
<td>5/5</td>
<td>1.42</td>
<td>0.29</td>
<td>0</td>
<td>2.4</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>L-10–4</td>
<td>297.31</td>
<td>5/5</td>
<td>1.47</td>
<td>0.45</td>
<td>0</td>
<td>8.2</td>
<td>7.4</td>
<td>2</td>
</tr>
<tr>
<td>L-10–5</td>
<td>1021.89</td>
<td>5/5</td>
<td>1.32</td>
<td>0.32</td>
<td>0</td>
<td>5.6</td>
<td>8.8</td>
<td>2</td>
</tr>
<tr>
<td>L-15–3</td>
<td>444.63</td>
<td>5/5</td>
<td>1.03</td>
<td>0.33</td>
<td>0</td>
<td>40.4</td>
<td>17.2</td>
<td>3</td>
</tr>
<tr>
<td>L-15–4</td>
<td>3341.32</td>
<td>4/5</td>
<td>1.08</td>
<td>0.29</td>
<td>0.06</td>
<td>37.4</td>
<td>19</td>
<td>3</td>
</tr>
<tr>
<td>L-15–5</td>
<td>3844.28</td>
<td>3/5</td>
<td>1.11</td>
<td>0.30</td>
<td>0.07</td>
<td>33.2</td>
<td>39.8</td>
<td>3</td>
</tr>
<tr>
<td>L-20-3</td>
<td>10 800.00</td>
<td>0/5</td>
<td>2.65</td>
<td>0.92</td>
<td>0.55</td>
<td>197.4</td>
<td>270</td>
<td>4</td>
</tr>
<tr>
<td>L-20-4</td>
<td>10 800.00</td>
<td>0/5</td>
<td>2.39</td>
<td>0.87</td>
<td>0.68</td>
<td>136.6</td>
<td>326.4</td>
<td>4</td>
</tr>
<tr>
<td>L-20-5</td>
<td>10 800.00</td>
<td>0/5</td>
<td>2.57</td>
<td>0.96</td>
<td>0.83</td>
<td>96</td>
<td>165.6</td>
<td>4</td>
</tr>
<tr>
<td>L-35-3</td>
<td>10 800.00</td>
<td>0/5</td>
<td>3.02</td>
<td>1.72</td>
<td>1.64</td>
<td>85.4</td>
<td>427.8</td>
<td>7</td>
</tr>
<tr>
<td>L-35-4</td>
<td>10 800.00</td>
<td>0/5</td>
<td>3.50</td>
<td>2.07</td>
<td>2.02</td>
<td>62.4</td>
<td>442</td>
<td>7</td>
</tr>
<tr>
<td>L-35-5</td>
<td>10 800.00</td>
<td>0/5</td>
<td>4.36</td>
<td>2.81</td>
<td>2.76</td>
<td>41</td>
<td>434.6</td>
<td>7</td>
</tr>
</tbody>
</table>

6.3. Comparison between branch-price-and-cut algorithm and CPLEX

In this subsection, we solve the arc-flow model (see Appendix) by using CPLEX and compare the results with that of the proposed branch-price-and-cut algorithm. A group of instances with up to 20 customers (per day), 5 days, and $\theta = 0.1$ are tested. For comparison, the time limit for both CPLEX and the branch-price-and-cut algorithm is set as 3 h. The detailed results and computing time are shown in Table 6. The results show that CPLEX can only find the optimal solution for instances with 7 customers, and the computational time is hundreds of times longer than that of the branch-price-and-cut algorithm. The proposed branch-price-and-cut algorithm is able to obtain the optimal solution for larger instances (up to 15 customers per day). When the number of customers reaches 20, both CPLEX and the branch-price-and-cut algorithm cannot obtain the optimal solution. However, the final solution gaps of the branch-price-and-cut algorithm are much smaller compared to that of CPLEX.
Table 8
Branch-price-and-cut results with consistency discount as 0.2.

<table>
<thead>
<tr>
<th>Inst.</th>
<th>T.Time (s)</th>
<th>#Opt.</th>
<th>LP Gap (%)</th>
<th>R. Gap (%)</th>
<th>Gap (%)</th>
<th>Nodes</th>
<th>RCI</th>
</tr>
</thead>
<tbody>
<tr>
<td>S-07-3</td>
<td>2.23</td>
<td>5/5</td>
<td>0.64</td>
<td>0.15</td>
<td>0</td>
<td>1.4</td>
<td>0.8</td>
</tr>
<tr>
<td>S-07-4</td>
<td>1.89</td>
<td>5/5</td>
<td>0.55</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1.4</td>
</tr>
<tr>
<td>S-07-5</td>
<td>3.07</td>
<td>5/5</td>
<td>0.72</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2.2</td>
</tr>
<tr>
<td>M-10-3</td>
<td>30.15</td>
<td>5/5</td>
<td>2.15</td>
<td>0.17</td>
<td>0</td>
<td>1.8</td>
<td>10.4</td>
</tr>
<tr>
<td>M-10-4</td>
<td>177.74</td>
<td>5/5</td>
<td>1.90</td>
<td>0.12</td>
<td>0</td>
<td>6.4</td>
<td>15</td>
</tr>
<tr>
<td>M-10-5</td>
<td>164.76</td>
<td>5/5</td>
<td>1.74</td>
<td>0.06</td>
<td>0</td>
<td>2</td>
<td>20</td>
</tr>
<tr>
<td>M-15-3</td>
<td>177.30</td>
<td>5/5</td>
<td>1.48</td>
<td>0.31</td>
<td>0</td>
<td>3</td>
<td>21.4</td>
</tr>
<tr>
<td>M-15-4</td>
<td>436.25</td>
<td>5/5</td>
<td>1.36</td>
<td>0.07</td>
<td>0</td>
<td>2.8</td>
<td>32.8</td>
</tr>
<tr>
<td>M-15-5</td>
<td>679.25</td>
<td>5/5</td>
<td>1.18</td>
<td>0.15</td>
<td>0</td>
<td>5.2</td>
<td>44.6</td>
</tr>
<tr>
<td>M-20-3</td>
<td>8653.23</td>
<td>1/5</td>
<td>3.42</td>
<td>0.84</td>
<td>0.68</td>
<td>161.6</td>
<td>275.4</td>
</tr>
<tr>
<td>M-20-4</td>
<td>10 800.00</td>
<td>0/5</td>
<td>4.75</td>
<td>2.07</td>
<td>1.90</td>
<td>137.2</td>
<td>296</td>
</tr>
<tr>
<td>M-20-5</td>
<td>10 800.00</td>
<td>0/5</td>
<td>5.17</td>
<td>2.52</td>
<td>2.42</td>
<td>81.8</td>
<td>285.8</td>
</tr>
<tr>
<td>L-10-3</td>
<td>213.34</td>
<td>5/5</td>
<td>1.49</td>
<td>0.32</td>
<td>0</td>
<td>4.6</td>
<td>7</td>
</tr>
<tr>
<td>L-10-4</td>
<td>921.18</td>
<td>5/5</td>
<td>1.44</td>
<td>0.14</td>
<td>0</td>
<td>6.6</td>
<td>11</td>
</tr>
<tr>
<td>L-10-5</td>
<td>3045.52</td>
<td>5/5</td>
<td>1.22</td>
<td>0.21</td>
<td>0</td>
<td>13.4</td>
<td>10</td>
</tr>
<tr>
<td>L-15-3</td>
<td>3009.98</td>
<td>5/5</td>
<td>0.98</td>
<td>0.25</td>
<td>0</td>
<td>63.8</td>
<td>44.4</td>
</tr>
<tr>
<td>L-15-4</td>
<td>7274.82</td>
<td>3/5</td>
<td>1.33</td>
<td>0.46</td>
<td>0.17</td>
<td>58</td>
<td>41</td>
</tr>
<tr>
<td>L-15-5</td>
<td>9420.59</td>
<td>3/5</td>
<td>1.37</td>
<td>0.42</td>
<td>0.15</td>
<td>67.8</td>
<td>42</td>
</tr>
<tr>
<td>L-20-3</td>
<td>10 800.00</td>
<td>0/5</td>
<td>2.73</td>
<td>1.20</td>
<td>0.85</td>
<td>157.6</td>
<td>269.4</td>
</tr>
<tr>
<td>L-20-4</td>
<td>10 800.00</td>
<td>0/5</td>
<td>2.89</td>
<td>1.58</td>
<td>1.34</td>
<td>71.8</td>
<td>213.2</td>
</tr>
<tr>
<td>L-20-5</td>
<td>10 800.00</td>
<td>0/5</td>
<td>3.40</td>
<td>2.13</td>
<td>1.97</td>
<td>50</td>
<td>161</td>
</tr>
<tr>
<td>L-35-3</td>
<td>10 800.00</td>
<td>0/5</td>
<td>5.41</td>
<td>4.13</td>
<td>4.06</td>
<td>61.4</td>
<td>360.8</td>
</tr>
<tr>
<td>L-35-4</td>
<td>10 800.00</td>
<td>0/5</td>
<td>9.09</td>
<td>7.72</td>
<td>7.68</td>
<td>36.2</td>
<td>308.2</td>
</tr>
<tr>
<td>L-35-5</td>
<td>10 800.00</td>
<td>0/5</td>
<td>11.80</td>
<td>10.47</td>
<td>10.42</td>
<td>37.2</td>
<td>336.7</td>
</tr>
</tbody>
</table>

6.4. Results on all instances

Next, we show the experiment results of our branch-price-and-cut algorithm on all instances. There are 480 instances tested in total, where each setting is repeated five times, and the value of \( \theta \) is set as 0.1, 0.2, 0.3, and 0.4. The results are reported in Tables 7–10, for the different values of \( \theta \) respectively. In these tables, each line shows the average results for the 5 instances with the same setting. The column "#Opt." shows the number of optimally solved instances out of the five considered instances. The column "#Veh." in Table 7 shows the number of vehicles used per day.

In our experiments, the problem's scale increases from three dimensions, the road network scale, the number of customers, and the number of days. The increase of any of them leads to a larger size of the problem and leads to longer computation time. We observe from the results that our algorithm is able to obtain high-quality feasible solutions within 3 h for most instances. Specifically, for the instances within 15 customers, 280 out of 300 instances are solved to optimality, and the final average gaps of the other 20 instances are within 0.52%. With the increase of customer size and scenario size, the gaps of solutions become larger. In instance L-35-5 with \( \theta = 0.4 \), the average gap reaches 24.54%. The value of \( \theta \) also affects the efficiency of the algorithm. It is because that more detours (columns) need to be considered with the larger value of \( \theta \).

To illustrate our branch-price-and-cut algorithm's detailed convergence process, we choose the instance L-15-3(1) with \( \theta = 0.3 \) and plot its convergence graph, as shown in Fig. 5. The total branch-and-bound tree size is 15 and the computation time is 1754.92 s. The optimal solution is obtained when exploring the 4th branch-and-bound node with 294.38 s.

6.5. Effect of the consistency discount

The path consistency is achieved at the expense of detours, i.e., vehicles might perform detours instead of the shortest paths when considering the path consistency. The consistency discount \( \theta \) is an important parameter that affects the total travel distance and the consistency degree of the results. In particular, when \( \theta = 0 \), the problem is equivalent to solving a series of independent VRP \( R_N \) without considering the consistency. Therefore, we adopt the solutions of ConVRP \( R_N \) with \( \theta = 0 \) as benchmarks when analyzing the effect of different values of \( \theta \) on the results. We only count the instances within 15 customers, as the optimal solutions of other larger instances are not found.

We first analyze the impact of \( \theta \) on the actual travel distance. In our proposed model, the value of the objective function is the travel cost with discount, and the first term is the actual travel distance. Table 11 shows the actual travel distance information for different values of \( \theta \). The column "\( \theta = 0 \)(km)" shows the travel distance (kilometers) on arterial roads (see column “Art.”), residential roads (see column “Res.”), and in total (see column “Total”) in the benchmark. The columns from “\( \theta = 0.1(\%) \)” to “\( \theta = 0.4(\%) \)” show the percentage of the increased travel distance compared to the benchmark (\( \theta = 0 \)(km)).

We observe that the larger value of \( \theta \) leads to longer traveled distance, and the increased distance is mainly on the arterial roads. The reason for that is as the value of \( \theta \) increases, a more considerable discount for the consistency paths is received, i.e., the benefit derived from the consistency increases.
Table 9
Branch-price-and-cut results with consistency discount as 0.3.

<table>
<thead>
<tr>
<th>Inst.</th>
<th>T.Time (s)</th>
<th>#Opt.</th>
<th>LP Gap (%)</th>
<th>R. Gap (%)</th>
<th>Gap (%)</th>
<th>Nodes</th>
<th>RCI</th>
</tr>
</thead>
<tbody>
<tr>
<td>S-07–3</td>
<td>2.38</td>
<td>5/5</td>
<td>0.68</td>
<td>0.23</td>
<td>0</td>
<td>1.4</td>
<td>0.8</td>
</tr>
<tr>
<td>S-07–4</td>
<td>2.33</td>
<td>5/5</td>
<td>0.41</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0.8</td>
</tr>
<tr>
<td>S-07–5</td>
<td>4.74</td>
<td>5/5</td>
<td>0.57</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>M-10–3</td>
<td>90.25</td>
<td>5/5</td>
<td>1.77</td>
<td>0.07</td>
<td>0</td>
<td>1.6</td>
<td>8.4</td>
</tr>
<tr>
<td>M-10–4</td>
<td>343.83</td>
<td>5/5</td>
<td>1.45</td>
<td>0.15</td>
<td>0</td>
<td>4.2</td>
<td>15</td>
</tr>
<tr>
<td>M-10–5</td>
<td>597.63</td>
<td>5/5</td>
<td>1.35</td>
<td>0.14</td>
<td>0</td>
<td>5.6</td>
<td>16.8</td>
</tr>
<tr>
<td>M-15–3</td>
<td>177.38</td>
<td>5/5</td>
<td>1.20</td>
<td>0.08</td>
<td>0</td>
<td>2</td>
<td>22.4</td>
</tr>
<tr>
<td>M-15–4</td>
<td>237.88</td>
<td>5/5</td>
<td>1.11</td>
<td>0.07</td>
<td>0</td>
<td>1.8</td>
<td>25.2</td>
</tr>
<tr>
<td>M-15–5</td>
<td>2500.06</td>
<td>5/5</td>
<td>0.91</td>
<td>0.15</td>
<td>0</td>
<td>9.6</td>
<td>42</td>
</tr>
<tr>
<td>M-20–3</td>
<td>10800.00</td>
<td>0/5</td>
<td>3.53</td>
<td>0.93</td>
<td>0.83</td>
<td>127.8</td>
<td>162.6</td>
</tr>
<tr>
<td>M-20–4</td>
<td>10800.00</td>
<td>0/5</td>
<td>4.90</td>
<td>2.15</td>
<td>2.02</td>
<td>101.4</td>
<td>172.6</td>
</tr>
<tr>
<td>M-20–5</td>
<td>10800.00</td>
<td>0/5</td>
<td>7.25</td>
<td>4.64</td>
<td>4.56</td>
<td>64.8</td>
<td>223</td>
</tr>
<tr>
<td>L-10–3</td>
<td>398.45</td>
<td>5/5</td>
<td>1.36</td>
<td>0.24</td>
<td>0</td>
<td>4.6</td>
<td>6.2</td>
</tr>
<tr>
<td>L-10–4</td>
<td>1054.88</td>
<td>5/5</td>
<td>1.28</td>
<td>0.21</td>
<td>0</td>
<td>4.2</td>
<td>8.4</td>
</tr>
<tr>
<td>L-10–5</td>
<td>1898.64</td>
<td>5/5</td>
<td>1.03</td>
<td>0.23</td>
<td>0</td>
<td>6</td>
<td>7.6</td>
</tr>
<tr>
<td>L-15–3</td>
<td>2963.46</td>
<td>4/5</td>
<td>1.23</td>
<td>0.33</td>
<td>0.10</td>
<td>42.2</td>
<td>40.4</td>
</tr>
<tr>
<td>L-15–4</td>
<td>5593.10</td>
<td>3/5</td>
<td>1.43</td>
<td>0.53</td>
<td>0.34</td>
<td>39</td>
<td>32.8</td>
</tr>
<tr>
<td>L-15–5</td>
<td>2500.00</td>
<td>5/5</td>
<td>0.91</td>
<td>0.15</td>
<td>0</td>
<td>9.6</td>
<td>42</td>
</tr>
<tr>
<td>L-20–3</td>
<td>10800.00</td>
<td>0/5</td>
<td>3.53</td>
<td>0.93</td>
<td>0.83</td>
<td>127.8</td>
<td>162.6</td>
</tr>
<tr>
<td>L-20–4</td>
<td>10800.00</td>
<td>0/5</td>
<td>4.90</td>
<td>2.15</td>
<td>2.02</td>
<td>101.4</td>
<td>172.6</td>
</tr>
<tr>
<td>L-20–5</td>
<td>10800.00</td>
<td>0/5</td>
<td>7.25</td>
<td>4.64</td>
<td>4.56</td>
<td>64.8</td>
<td>223</td>
</tr>
</tbody>
</table>

Comparing the instances with medium size and large size network in Table 11, we found that the road network scale also affects the price of consistency. Within the same \( \theta \) value and the same number of customers, the price for consistency increases as the road network scale increases. It means that the vehicles need to detour longer to achieve path consistency in the larger network.

In these experiment results, the consistency brings the extra travel distance up to 8.82% (in instance L-10-5), which is acceptable in most situations. In real-life applications, the companies could set the value of \( \theta \) according to their requirements.

Next, we analyze the impact of the value of \( \theta \) on the consistency degree of obtained paths. We evaluate the consistency degree on a sub-network with only the arterial roads. Specifically, the consistency degree is defined as the ratio of the consistent road segments’ distance to the total distance. For a fair comparison, in the instances without consistency consideration, we still set a very small value for \( \theta \) (0.0001) such that the path-to-vehicle assignment is optimized. In Table 12, we show the consistency degree in
percentages with different values of $\theta$. The consistency degree increases as the value of $\theta$ increases and the marginal utility may diminish.

The impact of different $\theta$ value on both the actual travel distance and the consistency degree are summarized in Fig. 6.

To provide an intuitive impression of the consistency, we illustrate the results of Inst. L-15-5(1), which considers five workdays in one week. The instance is based on the large road network and has 15 customers each day. Three vehicles are used on each day. As shown in Fig. 7, we observe that the paths are more and more consistent as the value of $\theta$ increases.

6.6. Side benefits on other consistent dimensions

As mentioned by Kovacs et al. (2014), considering one consistent dimension usually leads to consistency in other dimensions. In our study, the driver–customer pair consistency and arrival time consistency are achieved as side benefits when considering path consistency. For clarification, we take instance M-10-5(1) as an example to show the connection between different consistent dimensions.

Fig. 8(a) and (b) illustrate the optimal results of M-10-5(1) without and with considering the path consistency, respectively. We use blue nodes to represent the customers who are served by different drivers on different days. It is observed that the number of customers served by different drivers decreases from 7 to 2 after considering the path consistency.

Considering path consistency also leads to consistent arrival times as a side benefit. We assume that the speed for each road section is the same (40 km/h), and then we calculate the arrival times for each customer in each scenario. The results are shown in Fig. 9. Red nodes represent the arrival times of customers in the results without path consistency, and blue nodes represent that with considering path consistency. Obviously, after considering path consistency, the span of the earliest and the latest time arriving at the customer is reduced. Specifically, the average span decreases from 6.51 to 2.30, which is a reduction of 64.7%.

---

Table 11

Impact of different $\theta$ values on the actual travel distance.

<table>
<thead>
<tr>
<th>Inst.</th>
<th>$\theta = 0$ (km)</th>
<th>$\theta = 0.1$ (%)</th>
<th>$\theta = 0.2$ (%)</th>
<th>$\theta = 0.3$ (%)</th>
<th>$\theta = 0.4$ (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>S-07–3</td>
<td>50.42</td>
<td>8.00</td>
<td>58.42</td>
<td>4.83</td>
<td>−25.55</td>
</tr>
<tr>
<td>S-07–4</td>
<td>66.07</td>
<td>11.10</td>
<td>77.17</td>
<td>4.31</td>
<td>−21.50</td>
</tr>
<tr>
<td>S-07–5</td>
<td>82.66</td>
<td>13.65</td>
<td>96.31</td>
<td>5.05</td>
<td>−29.98</td>
</tr>
<tr>
<td>M-10–3</td>
<td>66.07</td>
<td>17.29</td>
<td>83.37</td>
<td>0.55</td>
<td>−14.58</td>
</tr>
<tr>
<td>M-10–4</td>
<td>88.40</td>
<td>23.78</td>
<td>112.18</td>
<td>0.58</td>
<td>−13.91</td>
</tr>
<tr>
<td>M-10–5</td>
<td>110.29</td>
<td>30.32</td>
<td>140.62</td>
<td>0.88</td>
<td>−8.35</td>
</tr>
<tr>
<td>M-15–3</td>
<td>92.66</td>
<td>25.84</td>
<td>118.50</td>
<td>3.36</td>
<td>−8.94</td>
</tr>
<tr>
<td>M-15–4</td>
<td>122.84</td>
<td>35.61</td>
<td>158.45</td>
<td>6.46</td>
<td>−9.98</td>
</tr>
<tr>
<td>M-15–5</td>
<td>155.01</td>
<td>43.29</td>
<td>198.30</td>
<td>3.77</td>
<td>−8.98</td>
</tr>
<tr>
<td>Average</td>
<td>101.85</td>
<td>29.02</td>
<td>130.86</td>
<td>3.50</td>
<td>−9.55</td>
</tr>
</tbody>
</table>

Fig. 5. Convergence graph.
Table 12
Impact of different θ values on the consistency degree.

<table>
<thead>
<tr>
<th>Inst.</th>
<th>θ = 0 (%)</th>
<th>θ = 0.1 (%)</th>
<th>θ = 0.2 (%)</th>
<th>θ = 0.3 (%)</th>
<th>θ = 0.4 (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>S-07–3</td>
<td>55.36</td>
<td>84.84</td>
<td>84.84</td>
<td>87.49</td>
<td>89.67</td>
</tr>
<tr>
<td>S-07–4</td>
<td>36.51</td>
<td>80.67</td>
<td>81.86</td>
<td>83.19</td>
<td>84.29</td>
</tr>
<tr>
<td>S-07–5</td>
<td>36.38</td>
<td>77.25</td>
<td>81.23</td>
<td>81.23</td>
<td>82.54</td>
</tr>
<tr>
<td>Average</td>
<td>42.82</td>
<td>80.92</td>
<td>82.64</td>
<td>83.97</td>
<td>85.50</td>
</tr>
<tr>
<td>M-10–3</td>
<td>46.57</td>
<td>62.07</td>
<td>74.13</td>
<td>81.78</td>
<td>84.38</td>
</tr>
<tr>
<td>M-10–4</td>
<td>41.10</td>
<td>61.38</td>
<td>76.85</td>
<td>81.75</td>
<td>82.41</td>
</tr>
<tr>
<td>M-10–5</td>
<td>38.22</td>
<td>56.43</td>
<td>75.66</td>
<td>80.61</td>
<td>83.55</td>
</tr>
<tr>
<td>M-15–3</td>
<td>56.20</td>
<td>82.64</td>
<td>87.67</td>
<td>89.00</td>
<td>90.35</td>
</tr>
<tr>
<td>M-15–4</td>
<td>45.65</td>
<td>80.36</td>
<td>85.34</td>
<td>87.45</td>
<td>87.45</td>
</tr>
<tr>
<td>M-15–5</td>
<td>44.55</td>
<td>75.24</td>
<td>81.99</td>
<td>83.58</td>
<td>83.88</td>
</tr>
<tr>
<td>Average</td>
<td>45.38</td>
<td>69.69</td>
<td>80.27</td>
<td>84.03</td>
<td>85.34</td>
</tr>
<tr>
<td>L-10–3</td>
<td>31.01</td>
<td>53.46</td>
<td>72.88</td>
<td>81.77</td>
<td>84.36</td>
</tr>
<tr>
<td>L-10–4</td>
<td>19.07</td>
<td>46.18</td>
<td>72.54</td>
<td>81.96</td>
<td>82.65</td>
</tr>
<tr>
<td>L-10–5</td>
<td>18.47</td>
<td>50.21</td>
<td>68.66</td>
<td>80.54</td>
<td>83.03</td>
</tr>
<tr>
<td>L-15–3</td>
<td>36.17</td>
<td>59.87</td>
<td>74.28</td>
<td>80.34</td>
<td>83.35</td>
</tr>
<tr>
<td>L-15–4</td>
<td>26.64</td>
<td>56.07</td>
<td>68.28</td>
<td>76.26</td>
<td>80.68</td>
</tr>
<tr>
<td>L-15–5</td>
<td>13.96</td>
<td>51.28</td>
<td>68.73</td>
<td>77.70</td>
<td>78.28</td>
</tr>
<tr>
<td>Average</td>
<td>24.22</td>
<td>52.85</td>
<td>70.90</td>
<td>79.76</td>
<td>82.06</td>
</tr>
</tbody>
</table>

Fig. 6. Travel distance and consistency degree on different θ value.

6.7. Extensions

In this study, we mainly focus on the path consistency on arterial roads, as this type of roads usually has more complex traffic conditions and network structures. However, the proposed consistent strategy allows users to flexibly design on which roads or in which areas the routes need to be consistent. For instance, one could stipulate that only routes in a certain area need to be consistent, or that the routes need to be consistent on all the roads in the network.

Fig. 10 shows three medium networks with different structures. Arcs with red color represent the road sections where the path consistency is considered (𝐴∗ in the model). The optimized results based on these three networks are shown in Fig. 11. There are two observations according to the results. As a first observation, the vehicle paths are relatively flexible in the area where consistency is not considered (compare 11(a) and (b)). Secondly, the vehicles are inclined to pass through the road sections which consider the path consistency (could provide a consistent discount) rather than normal road sections (compare 11(b) and (c)). Based on these two observations, the users could customize the network according to their specific problem characteristics.

7. Conclusions

This study emphasizes the importance of considering path consistency and the underlying road network structure in the vehicle routing problems. Thus, we introduce a new variant of the VRP, the consistent vehicle routing problem considering path consistency in a road network. A flexible consistency strategy is proposed, where a discount is provided for the segments traversed every day to encourage vehicles to take consistent paths. The problem is finding a set of paths for each vehicle on each day such that the total travel cost considering the consistency discount is minimized. We formulate the problem as a set partitioning model and a more straightforward arc-flow model. A branch-price-and-cut algorithm is proposed to solve the problem.
Fig. 7. Results on the large road network with 5 days and 15 customers for each day.
We construct a two-layer network for performing the branch-price-and-cut algorithm. The lower layer is a road network providing the information on the road network structure. The upper layer is a set of customer-graphs for pricing, valid inequalities separation, and branching procedures in the branch-price-and-cut algorithm.

We propose a new aggregation technique and embed it into the branch-price-and-cut framework to reduce the computational time. The technique reduces the algorithm running time by 84.55% and the size of the searching tree by 47.74%. This technique can also provide an idea for other interdependence problems with similar characteristics.

We design experiments with different sizes based on the road network of West Jordan to test our algorithm. Computational results show that our algorithm can solve the instances with up to five workdays and 35 customers (per day) in the network with
Fig. 10. Medium networks with different structures.

Fig. 11. Optimized results of medium networks with different structures.
113 nodes and 310 links. We also analyze the effects of different values of consistency discount on the travel distance and consistency degree, which provides a reference for decision-makers to customize individual problems. Considering path consistency also has side benefits in other consistent dimensions, such as person consistency and arrival time consistency.

There are also some interesting future research directions. For instance, in many cases the customer demands are stochastic and become known only after the routes are established. Hence, one interesting future research topic would be to construct ConVRP\(_{RN}\) as a two-stage stochastic programming problem. Another direction for further study is to consider the inconsistent VRP in a road network, which also has many real-world applications, e.g., cash-in-transit problem.

CRediT authorship contribution statement

Yu Yao: Conceptualization, Methodology, Software, Writing – original draft, Funding acquisition. Tom Van Woensel: Methodology, Writing – review & editing, Supervision. Lucas P. Veelenturf: Methodology, Writing – review & editing, Project administration. Pengli Mo: Validation, Writing – review & editing.

Acknowledgments

This study was supported by Beijing Natural Science Foundation (No. L201016), the National Natural Science Foundation of China (No. 72001020), and the China Postdoctoral Science Foundation (No. 2020M670128). The authors thank the anonymous reviewers for their valuable comments for improving this paper.

Appendix

In this appendix, we provide an arc-flow model for the ConVRP\(_{RN}\). First, we present the following lemma: *In an optimal solution to the VRP\(_{RN}\), no edge will be traversed more than once in either direction*. This Lemma is proved in Letchford and Oukil (2009) and also applies to the ConVRP\(_{RN}\).

Based on this lemma, we define a binary variables \(x_{i,j,k}^s\), which equals to 1 if vehicle \(k\) traverses arc \((i,j)\) on day \(s\); equals to 0 otherwise. Variable \(f_{i,j,k}^s\) represents the total load (if any) that is carried along the arc \((i,j)\) by vehicle \(k\) on day \(s\). Binary variable \(y_{i,j,k}^s\) equals to 1 if vehicle \(k\) traverses arc \((i,j)\) on \(A^+\) on each day; and \(y_{i,j,k}^s = 0\) otherwise. Binary variable \(z_{i,j,k}^s\) equals to 1 if customer \(j\) is served by vehicle \(k\) on day \(s\); equals to 0 otherwise. The ConVRP\(_{RN}\) is modeled as a single-commodity flow formulation as follows.

\[
\min \sum_{s \in S} \sum_{k \in K} \sum_{(i,j) \in A} c_{ij} x_{i,j,k}^s - \theta |S| \sum_{k \in K} \sum_{(i,j) \in A^+} c_{ij} y_{i,j,k}^s
\]

Subject to:

\[
|N^s| \sum_{(i,j) \in A} x_{i,j,k}^s - \sum_{j \in N^s} z_{i,j,k}^s \geq 0, \forall k \in K, s \in S
\]

\[
\sum_{(i,j) \in A} x_{i,j,k}^s = \sum_{(j',i) \in A^+} x_{j',i,k}^s, \forall j \in N, k \in K, s \in S
\]

\[
\sum_{(i,j) \in A} f_{i,j,k}^s - \sum_{(j',i) \in A^+} f_{j',i,k}^s = q_j^s z_{i,j,k}^s, \forall j \in N^s, k \in K, s \in S
\]

\[
\sum_{(i,j) \in A} f_{i,j,k}^s - \sum_{(j',i) \in A^+} f_{j',i,k}^s = 0, \forall j \in N \setminus N^s, k \in K, s \in S
\]

\[
0 \leq f_{i,j,k}^s \leq Q x_{i,j,k}^s, \forall (i,j) \in A, k \in K, s \in S
\]

\[
x_{i,j,k}^s \geq y_{i,j,k}^s, \forall (i,j) \in A^+, k \in K, s \in S
\]

\[
\sum_{k \in K} z_{i,j,k}^s = 1, \forall j \in N^s, s \in S
\]

\[
x_{i,j,k}^s = \{0,1\}, \forall (i,j) \in A, k \in K, s \in S
\]

\[
y_{i,j,k} = \{0,1\}, \forall (i,j) \in A^+, k \in K
\]

\[
z_{i,j,k}^s = \{0,1\}, \forall j \in N^s, k \in K, s \in S
\]
The objective function (30) minimizes the total travel cost over all the days considering the discount for consistency. The first part is the travel cost over all the days, and the second part is the discount for the consistent parts. Constraints (31) represent that a vehicle departs from origin node if it is assigned to serve customers. Constraints (32) are the flow conservation constraints for nodes. Constraints (33) ensure that a certain unit of the commodity is delivered to required nodes by vehicle if that customer is assigned to vehicle. Constraints (34) impose that the amount of commodity on board for each vehicle remains unchanged when passing through non-required nodes. The constraints (35) ensure that the commodity passes along an arc only if the arc appears in the tour and the amount of commodity within vehicle capacity. Constraints (36) represent that an arc can be considered as consistent only if it is traversed by one certain vehicle every day. The constraints (37) impose that each customer is served exactly once. Constraints (38)–(40) are the definition of decision variables.

References


