Highlights

The Consistent Vehicle Routing Problem with Stochastic Customers and Demands

Aldair Alvarez, Jean-François Cordeau, Raf Jans

- We introduce the consistent vehicle routing problem with stochastic customers and demands.
- A sample average approximation (SAA) approach is presented for the problem.
- We introduce a branch-and-cut and a Benders decomposition method to solve the sample problems in our SAA algorithm.

The Consistent Vehicle Routing Problem with Stochastic Customers and Demands

Aldair Alvarez, Jean-François Cordeau, Raf Jans

HEC Montréal and GERAD, 3000 Chemin de la Côte-Sainte-Catherine, Montréal, Canada, H3T 2A7

Abstract

This paper introduces the consistent vehicle routing problem with stochastic customers and demands. We consider driver consistency as customer-driver assignments that remain fixed when the realizations of the random variables are observed. We study the problem in a two-stage scenario-based stochastic programming framework. In the first stage, customers are assigned to drivers, while in the second stage, customers are selected and delivery routes are designed for each of the scenarios. We assume that the realization of the random variables becomes known before the vehicles depart from the depot. The routes are then optimized according to the observed customers and their demands. The first-stage driver-customer assignments can violate the consistency requirement, which is modeled as a desired maximum number of drivers assigned to each customer. This is modeled as a soft constraint with a penalty in the objective function. It is hence possible to assign multiple drivers to a specific customer in the first stage. In the second stage, a customer can only be visited by one of the preassigned drivers. Our problem, therefore, consists in finding assignments that minimize the consistency violation penalties and the expected routing costs and the penalties for unserved customers when the uncertain parameters are revealed. We present a mathematical formulation and a sample average approximation (SAA) approach for the problem. We introduce a branch-and-cut and a Benders decomposition method to solve the sample problems in our SAA algorithm. Computational experiments show that SAA allows finding good-quality solutions for instances with large sets of scenarios. We also analyze the cost-consistency trade-offs and the impact of the uncertainty on the problem. In particular, we observe that consistency can be promoted through a flexible approach that does not compromise excessively on other operational metrics. Furthermore, we analyze the impact of not considering the problem uncertainties during the planning stage.

Keywords: consistent vehicle routing problem, sample average approximation, Benders decomposition, customer uncertainty

1. Introduction

In this paper, we study the consistent vehicle routing problem (ConVRP) with stochastic customers and demands. We study the problem in a two-stage stochastic programming setting where, in the

Email addresses: aldair.alvarez@hec.ca (Aldair Alvarez), jean-francois.cordeau@hec.ca (Jean-François Cordeau), raf.jans@hec.ca (Raf Jans)

first stage, a decision-maker assigns drivers to potential customers before the actual requests and the corresponding demands are known. After the realization of the random variables, the decision-maker selects which customers to visit and designs delivery routes respecting both selected driver-customer assignments and the vehicle capacities. Our problem setting assumes that some customers may be left unserved. In such a case, a penalty is incurred. Moreover, we assume that the outcome of the uncertain information is known before the vehicles depart from the depot. As a result, the routes can be optimized based on the observed and selected customers and their demands. As pointed out by Ledvina et al. (2022), current information technologies allow collecting this type of information before the departure of the vehicles. The first-stage driver-customer assignments can violate the consistency requirement, which is modeled as a desired maximum number of drivers assigned to each customer. This is modeled as a soft constraint with a penalty in the objective function. It is hence possible to assign multiple drivers to a specific customer in the first stage. In the second stage, in each scenario a customer can only be visited by one of the preassigned drivers. In our problem, therefore, consistency is imposed in the form of assignments that minimize the consistency violation penalties and the expected routing costs and the penalties for unserved customers when the uncertain parameters are revealed. Uncertainties are considered using scenario-based stochastic programming (SP), in which each scenario represents the realization of both a set of customer requests as well as their associated demands. Note that in our setting the routing decisions are in the second stage, which complicates the problem since one must solve a variant of the vehicle routing problem for each scenario.

This problem setting may appear in contexts in which the decision-maker does not know if and when in the planning horizon the customers will demand service. However, because of previous requests, the location of the customers is known in advance and it is also possible to infer their demand distribution. This may be the case for couriers and pickup and delivery providers (Sungur et al., 2010; Ledvina et al., 2022) as well as home service providers (Song et al., 2020). These types of companies usually have databases from past requests and serve customers with distinct, irregular demand patterns. Therefore, the historical data can be used as a proxy of potential realizations of the uncertainty faced by the company (for instance, in the form of scenarios). Furthermore, companies might also be interested in routing plans in which service consistency is as important as efficiency.

Our problem setting includes two important features for decision-making in vehicle routing: consistency (Groër et al., 2009) and data uncertainty (Gendreau et al., 2016). In this context, *consistency* is defined as the degree to which certain solution components remain relatively stable over time. This feature is particularly relevant in today's competitive business environment as companies seek to offer personalized services. For instance, carriers can try to promote consistency to each customer in the form of visits by a limited number of different drivers. This facilitates the customization of the service, potentially increasing customer satisfaction levels as well as the company's revenue. From the standpoint of the drivers, visiting a relatively stable group of customers on a regular basis has the potential to improve the efficiency of the delivery operations as drivers become familiar with a region, the conditions of a smaller road network and a limited set of customers.

The second feature is data uncertainty. In many cases, critical input data is not known with certainty during the planning stage. This brings uncertainty and challenges the decision-making process.

Moreover, using point forecasts in the planning stage often leads to poor performance in the execution stage, which further highlights the importance of taking uncertainty into account for operations planning, particularly in the context of the vehicle routing problem (VRP).

In this study, we consider the ConVRP with stochastic customers and demands in a two-stage decision process. In the first stage of the problem, the assignments of drivers to customers must be determined. In the second stage, customers are selected, and routes are designed based on the realization of the random variables. Penalties for not serving observed customers are incurred in this stage, in addition to routing costs. The first-stage assignments remain fixed in the second stage over all the scenarios to ensure driver consistency for the planned visits. These assignments are guided by a predefined parameter indicating the targeted maximum number of different drivers assigned to each customer. However, violations of this target are allowed and penalized in the objective function. This flexibility is permitted to represent the real-world practice in which, in some applications, violations may be acceptable according to the decision-maker's preference. Hence, the objective is to define consistent assignments in the first stage that minimize the violations of the consistency target and the expected total transportation cost as well as the sum of penalties for unserved customers in the second stage.

1.1. Literature Review and Our Contributions

The problem that we study relates directly to the deterministic ConVRP, introduced by Groër et al. (2009) in the context of a multiday VRP. In the deterministic ConVRP, two forms of consistency are usually considered: *driver consistency* (Braekers and Kovacs, 2016), in which customers have to be visited by the same driver each day they require service, and *arrival time consistency* (Kovacs et al., 2014a), in which the different visits to each of the customers should happen at approximately the same time. Extensions of this problem include considering consistency as an objective instead of a constraint (Smilowitz et al., 2013), allowing a limited number of drivers to visit each customer (instead of a single one) in the driver consistency context (Kovacs et al., 2015), and other forms of consistency (Yao et al., 2021; Rodríguez-Martín and Yaman, 2022).

These studies, however, assume that all the information is known with certainty beforehand and, therefore, carry out the assignment of customers to drivers as well as the routing of the daily visits simultaneously. Kovacs et al. (2014a) provide a review of studies considering consistency in vehicle routing problems, primarily in the deterministic setting. Other papers have considered consistency in deterministic supply chain optimization problems such as the inventory routing problem (Diabat et al., 2021; Coelho et al., 2012) as well as the production routing problem (Alvarez et al., 2022).

Studies on the ConVRP under uncertainty have mainly focused on using scenario-based SP. In this approach, a set of scenarios is used to describe the potential realizations of the random variables according to their corresponding probability distribution. For instance, Sungur et al. (2010) address a case with stochastic customers and service time uncertainty. In this work, consistency is favored by generating a master routing plan that minimizes, among other objectives, the adjustment cost of the daily routing plan for the observed customer requests. The authors use a set of scenarios to represent the uncertainty of the customers' presence, including the vehicle routing as a second-stage decision.

The problem addressed in this study considers time windows, uncapacitated vehicles and constraints on the maximum duration of the routes.

Spliet and Dekker (2016) study a ConVRP with uncertain demands. In their problem, drivers are assigned to customers before demand is known so that the expected routing cost over all the demand scenarios is minimized. Consistency is modeled by setting constraints imposing that each driver visits a minimum fraction of their assigned customers in each scenario. In their study, however, the authors do not penalize potential inconsistencies and their computational experiments focus mostly on cases with only three scenarios. Song et al. (2020) also employ scenario-based SP for a case with customer uncertainty. In their problem, the first stage assigns known (regular) customers to drivers while the second stage selects new (observed) customers and assigns them to the planned routes of each scenario, such that the expected total profit over all the scenarios is maximized. This approach ensures consistency for regular customers only. The authors introduce their study in the context of the team-orienteering problem and, as such, they consider time windows, uncapacitated vehicles, and maximum route duration constraints. In both of these works, routing is a second-stage decision.

Finally, Ledvina et al. (2022) address a case with uncertain demands incorporating consistency using concepts from manufacturing process flexibility (Jordan and Graves, 1995). In particular, they study the effect of overlapping routing strategies in the form of (partially) redundant customer-driver assignments. First, customers are assigned to the so-called primary route of a driver and then, when the actual demands are observed, the surplus vehicle capacity (if any) is used to visit customers on the extended route assigned to the driver. Consistency is thus promoted by defining *primary* driver-customer assignments but permitting the visit to customers outside the primary set of the driver.

Our research differs from the previous works in both problem setting and solution methods. First, we employ a flexible approach in which the number of different drivers assigned to each customer is an adjustable parameter while favoring consistency by penalizing the violation of a consistency target. In addition, we use scenario-based SP with large sets of scenarios which allows for finding stable customer-driver assignments by explicitly taking into account routing decisions under potential realizations of the uncertain variables.

There is also a close relationship between our work and the stochastic VRP. In this area, most of the literature considers routing as a first-stage decision and assumes that the presence of customers or their demand is known only upon arrival at their location, applying a specified recourse policy when failure occurs (Gendreau et al., 2016). This type of approach is largely used for the case with stochastic demand and guarantees consistency as the customer-driver assignments are maintained. However, in certain industries, the outcome of the uncertain parameters may be known before the vehicles start their routes, in particular with current information technologies. This assumption has been used in the literature of the VRP with stochastic customers (Bertsimas, 1992; Sungur et al., 2010). In this case, a full reoptimization of the routes can be applied daily. However, this approach really complicates the problem and would potentially fail to provide consistency in the solutions. This drawback can be addressed, for instance, by selecting a priori customer-driver assignments, which is our approach. This has been explored in the context of the ConVRP only by Spliet and Dekker (2016).

Our contributions in this paper include the following. First, we introduce and study the ConVRP

with stochastic customers and demands with generalized consistency requirements. Our approach allows controlling the level of the targeted consistency by setting a parameter indicating the maximum desired deviation from a perfectly consistent solution. The importance of abiding by this target is also flexible and controlled in the objective function. Second, we show that imposing consistency with a flexible approach in the form of soft constraints allows finding solutions with adequate consistency metrics without compromising excessively on other operational performance metrics. Third, we show the value of the stochastic solutions with respect to two expected value approaches. In particular, the results reveal that the value of the stochastic solutions changes when the probability of occurrence of the customers increases. Finally, we develop several methods for solving the problem (and the sample problems) and compare their performance under different configurations and enhancements.

1.2. Paper Organization

The remainder of the paper is organized as follows. Section 2 describes the problem and introduces a mathematical formulation for it. Section 3 presents our solution approaches. In particular we describe a sample average approximation method as well the branch-and-cut and Benders decomposition algorithms used as components within the SAA. Section 4 presents the computational experiments to assess the value of taking consistency and uncertainty into account in our context and, finally, Section 5 concludes the paper.

2. Problem Definition

We formulate the ConVRP with stochastic customers and demands as a two-stage SP problem. For this purpose, we introduce the following notation. Consider a set of potential customers $\mathcal{C}$, a fleet of homogeneous vehicles of capacity Q, denoted by set $\mathcal{K}$. The vehicles are based at a depot, denoted by node 0, and all their routes depart from and return to this depot. We assume that each vehicle is associated with a single driver and therefore we will use these terms interchangeably. The node set $\mathcal{N} = \{0\} \cup \mathcal{C}$ represents all the possible locations. Let Ω be the finite set of all the scenarios. Scenarios embed the joint realization of the two uncertain variables. First, the customer presence random variable defines whether or not a customer exists in the scenario. Then, an independent random variable represents the demand of the present customers.

The probability of occurrence of each scenario $\omega \in \Omega$ is given by ρ_ω, with $\rho_\omega > 0$ and $\sum_{\omega \in \Omega} \rho_\omega = 1$. In each scenario $\omega \in \Omega$, $\mathcal{C}_\omega$ denotes the set of customers that have a strictly positive demand in the scenario, with $\mathcal{N}_\omega = \{0\} \cup \mathcal{C}_\omega$. For each customer $i \in \mathcal{C}_\omega$, $d_{i\omega}$ denotes its demand and b_i the penalty for not serving it. This penalty can be seen as an outsourcing cost or a compensation paid to the customer when its demand is not met. Notice that not serving a customer can also be interpreted as a customer that will be served on another day. In that sense, the cost associated to skipping customers could also be regarded as the penalty for customers who does not receive their order on time. In practice, the possibility of leaving some customers unserved brings added flexibility to the system, which is a desirable feature when managing complex supply chain systems, in particular in uncertain environments.

The problem is defined on a complete undirected graph $\mathcal{G} = \{\mathcal{N}, \mathcal{E}\}$, where $\mathcal{E} = \{(i,j)\colon i,j \in \mathcal{N}, i < j\}$ is the set of edges. There is also a set of edges $\mathcal{E}_\omega = \{(i,j)\colon i,j \in \mathcal{N}_\omega, i < j\}$ for each scenario ω. Moreover, a travel cost c_{ij} is incurred for traversing edge $(i,j) \in \mathcal{E}$. It is assumed that the travel costs satisfy the triangle inequality.

In order to consider consistency, let $D > 0$ be a parameter indicating the consistency target of the decision-maker. The value of D indicates the maximum number of different drivers that should be assigned to each customer in the first stage. This target may be larger than one and accounts for the decision-maker's tolerance with respect to a perfectly consistent solution. This type of approach follows the literature on the ConVRP with generalized requirements (Kovacs et al., 2015; Wang et al., 2022). In our problem, we allow violations of the consistency target D, penalizing with the unit cost o_i the assignments exceeding the target for every customer $i \in \mathcal{C}$. This flexible approach also aims at representing real-world situations in which some level of violation of the consistency target may be acceptable according to the preferences of the decision-maker.

Note that in our problem, the set $\mathcal{C}$ can be seen as a representation of all the observed customers in the company's database. Likewise, the set of scenarios Ω can be seen as historical data of the company. As such, we consider the observations of past days (scenarios) as realizations of the uncertainty in the problem. This approach was also used by Sungur et al. (2010) in the context of a courier delivery problem under customer uncertainty.

In the problem, there are two decision stages. The first one is tactical and consists of assigning customers to drivers for long-term planning. Then, in the second stage, potential operational decisions are considered by planning vehicle routes respecting the selected assignments. The objective is to minimize the penalties for violating the consistency target in the first-stage assignments plus the expected second-stage routing costs and total penalties for not serving potential customers.

2.1. A Two-Stage Stochastic Programming Model

To model the problem as a two-stage SP model, consider the following decision variables:

y_{ik}: a binary variable equal to one if and only if customer i is assigned to vehicle k;

s_k: a binary variable equal to one if and only if at least one customer is assigned to vehicle k;

$z_{ik\omega}$: a binary variable equal to one if and only if customer i is visited by vehicle k in scenario ω;

$x_{ijk\omega}$: an integer variable indicating the number of times vehicle k traverses edge (i,j) under scenario ω;

λ_i: a continuous variable measuring the violation level of the driver consistency target D for customer i.

Under the assumption that Ω represents all the possible scenarios, the problem can be formulated as follows:

$$\min \sum_{i \in \mathcal{C}} o_i \lambda_i + \sum_{\omega \in \Omega} \rho_\omega \left(\sum_{(i,j) \in \mathcal{E}_\omega} \sum_{k \in \mathcal{K}} c_{ij} x_{ijk\omega} + \sum_{i \in \mathcal{C}_\omega} b_i (1 - \sum_{k \in \mathcal{K}} z_{ik\omega}) \right) \tag{1}$$

$$\text{s.t.} \quad \sum_{k \in \mathcal{K}} y_{ik} \leq D + \lambda_i \qquad i \in \mathcal{C}, \tag{2}$$

$$\sum_{k\in\mathcal{K}} z_{ik\omega} \leq 1 \qquad \omega\in\Omega,\, i\in\mathcal{C}_\omega, \tag{3}$$

$$\sum_{(j,i)\in\mathcal{E}_\omega} x_{jik\omega} + \sum_{(i,j)\in\mathcal{E}_\omega} x_{ijk\omega} = 2z_{ik\omega} \qquad \omega\in\Omega,\, i\in\mathcal{C}_\omega,\, k\in\mathcal{K}, \tag{4}$$

$$\sum_{j\in\mathcal{C}_\omega} x_{0jk\omega} \leq 2 \qquad \omega\in\Omega,\, k\in\mathcal{K}, \tag{5}$$

$$z_{ik\omega} \leq y_{ik} \qquad \omega\in\Omega,\, i\in\mathcal{C}_\omega,\, k\in\mathcal{K}, \tag{6}$$

$$\sum_{i\in\mathcal{C}_\omega} d_{i\omega} z_{ik\omega} \leq Q \qquad \omega\in\Omega,\, k\in\mathcal{K}, \tag{7}$$

$$\sum_{i\in\mathcal{B}} \sum_{\substack{j\in\mathcal{B}:\\ i<j}} x_{ijk\omega} \leq \sum_{i\in\mathcal{B}\setminus\{\ell\}} z_{ik\omega} \qquad \omega\in\Omega,\, \forall\mathcal{B}\subseteq\mathcal{C}_\omega\colon |\mathcal{B}|\geq 2,\, k\in\mathcal{K},\, \ell\in\mathcal{B}, \tag{8}$$

$$\sum_{i\in\mathcal{C}} y_{ik} \leq |\mathcal{C}| s_k \qquad k\in\mathcal{K}, \tag{9}$$

$$\lambda_i \geq 0 \qquad i\in\mathcal{C}, \tag{10}$$

$$y_{ik} \in \{0,1\} \qquad i\in\mathcal{C},\, k\in\mathcal{K}, \tag{11}$$

$$s_k \in \{0,1\} \qquad k\in\mathcal{K}, \tag{12}$$

$$z_{ik\omega} \in \{0,1\} \qquad \omega\in\Omega,\, i\in\mathcal{C}_\omega,\, k\in\mathcal{K}, \tag{13}$$

$$x_{ijk\omega} \in \{0,1,2\} \qquad \omega\in\Omega,\, (i,j)\in\mathcal{E}_\omega\colon i=0,\, k\in\mathcal{K}, \tag{14}$$

$$x_{ijk\omega} \in \{0,1\} \qquad \omega\in\Omega,\, (i,j)\in\mathcal{E}_\omega\colon i>0,\, k\in\mathcal{K}. \tag{15}$$

The objective function (1) minimizes the sum of the first-stage consistency violation penalties plus the expected second-stage cost, given by the sum of the routing costs and penalties for not serving potential customers over all the scenarios. Constraints (2) capture the violations of the consistency target if the customers are assigned to more than D drivers in the first stage. Constraints (3) enforce the customers are visited by at most one vehicle every time they appear in the scenarios. Constraints (4) are the degree constraints, which guarantee the flow conservation of the vehicles assigned to the customers. Constraints (5) enforce that in each scenario and for every vehicle, there are at most two customer nodes adjacent to the depot. These constraints ensure that each vehicle is used at most in one route in each scenario.

Constraints (6) link the first-stage assignments to the second-stage routing variables, ensuring visits to the customers can only be carried out by the preassigned drivers. Constraints (7) guarantee the vehicle capacity satisfaction while constraints (8) are the subtour elimination constraints (SECs). Constraints (9) activate the aggregated assignment variables s when at least one customer is assigned to the vehicles. Finally, the domain of the decision variables is defined by constraints (10)-(15). Note that variables s_k are not necessary to represent the problem. However, they are useful to enhance the performance of the branch-and-cut method used to solve the problem, as will be explained in Section 3.2.1.

3. Solution Methodology

It is well know that two-stage stochastic programs, in particular with integer variables in both stages, can be very difficult to solve. This difficulty tends to grow significantly with the size of the scenario set Ω. We resort, therefore, to the sample average approximation (SAA) method (Kleywegt et al., 2002) to solve our problem. In this section, we first provide an overview of our SAA and then introduce the methods we use to solve the different sample problems that appear in each iteration of the SAA method.

3.1. Sample Average Approximation

In the SAA method, one solves a series of *sample problems* resulting from replacing the large set Ω with a small sample of scenarios N, such that $|N| \ll |\Omega|$. This step is replicated M times to obtain as many candidate first-stage solutions. Each of these solutions is evaluated using a large set of scenarios, providing a feasible solution for the complete problem in each iteration, from which we keep the best one as the incumbent solution. Normally, when the full set Ω is not known or is too large to be enumerated, the evaluation is done on a large subset $\Omega'' \subseteq \Omega$. However, since in our problem setting we assume that we know the complete set of scenarios, we evaluate solutions over the set Ω in our SAA approach. In addition to upper bounds, SAA provides a statistical estimate for the lower bound on the optimal value of the complete problem and, therefore, an estimate of the optimality gap of the incumbent solution. The SAA method can be described as follows:

1. Choose a sample size $|N|$ and the number of replications M of the method.
2. For $m = 1, \ldots, M$:
   (a) Generate a scenario set $\Omega^m \subset \Omega$ by randomly selecting $|N|$ scenarios from Ω. Solve the resulting sample problem, obtaining the corresponding objective value and solution, denoted respectively by ν_{Ω^m} and Z^m (consisting of the solution vectors $\bar{\boldsymbol{y}}, \bar{\boldsymbol{s}}, \bar{\boldsymbol{z}}, \bar{\boldsymbol{x}}, \bar{\boldsymbol{\lambda}}$).
   (b) Use the solution Z^m to try to obtain a feasible solution for the complete problem, containing all the scenarios Ω. For this, fix the first stage solution $\bar{\boldsymbol{y}}$ and solve the resulting $|\Omega|$ scenario subproblems independently. If all the scenario problems are feasible, then their joint solutions are a feasible solution for the complete problem. The objective value $\nu_\Omega(Z^m)$ of this feasible solution can be calculated as

$$\nu_\Omega(Z^m) = \sum_{i \in \mathcal{C}} o_i \bar{\lambda}_i + \sum_{\omega \in \Omega} \rho_\omega \left(\sum_{(i,j) \in \mathcal{E}_\omega} \sum_{k \in \mathcal{K}} c_{ij} \tilde{x}_{ijk\omega} + \sum_{i \in \mathcal{C}_\omega} b_i (1 - \sum_{k \in \mathcal{K}} \tilde{z}_{ik\omega}) \right)$$

   where $\tilde{x}_{ijk\omega}$ and $\tilde{z}_{ik\omega}$ are the optimal values of the routing and visit variables, respectively, when fixing the values of λ_i according to the solution Z^m. In the general method, if one of the scenario problems is infeasible, $\nu_\Omega(Z^m)$ is set to ∞. However, in our setting the subproblems are always feasible.

3. Get the best feasible solution as $Z^* = Z^{m'}$, with $m' = \arg\min_{m\in\{1,\dots,M\}}\{\nu_\Omega(Z^m)\}$, whose objective value is $\nu_\Omega(Z^*)$ and whose variance is

$$\sigma^2_{\nu_\Omega(Z^*)} = \frac{1}{|\Omega|(|\Omega|-1)}\sum_{\omega\in\Omega}(G_\omega(Z^*) - \nu_\Omega(Z^*))^2,$$

where $G_\omega(Z^*)$ is the total objective value obtained by solving the subproblem of scenario ω with the first stage variables fixed at Z^*.

4. Calculate the average of the objective values of the sample problems

$$\bar{\nu}_\Omega = \frac{1}{M}\sum_{m=1}^{M}\nu_{\Omega^m},$$

which is a statistical lower bound on the optimal value of the complete problem, in addition to its variance

$$\sigma^2_{\bar{\nu}_\Omega} = \frac{1}{M(M-1)}\sum_{m=1}^{M}(\nu_{\Omega^m} - \bar{\nu}_\Omega)^2.$$

These statistics are computed over all the feasible sample problems. Note that it is not always possible to solve to optimality the sample problems within an SAA algorithm. In such a case, we can compute the statistical lower bound $\bar{\nu}_\Omega$ and its variance $\sigma^2_{\bar{\nu}_\Omega}$ using a lower bound on the optimal value of the sample problems. We have employed this strategy in our implementation.

5. Compute the SAA gap

$$\nu_\Omega(Z^*) - \bar{\nu}_\Omega$$

and its variance

$$\sigma^2_{\nu_\Omega(Z^*)} + \sigma^2_{\bar{\nu}_\Omega}.$$

In general, the performance of SAA depends on the parameter choices (sample problem size $|N|$ and number of replications M) and the method used to solve the sample problems. On the one hand, smaller $|N|$ values result in problems that may be solved more easily while larger samples may lead to first-stage decisions resulting in potentially better solutions when evaluated on the large set of scenarios Ω. On the other hand, ideally, one uses effective exact methods in each replication to optimally solve the sample problem. However, even sample problems with small scenario sets may be hard to solve, and one can resort to early stopping criteria for exact methods or using heuristic algorithms. In the following, we describe two different exact methods proposed to solve the sample problems within our SAA.

3.2. Solving the Sample Problems

In this section we describe a branch-and-cut (BC) and a Benders decomposition (BD) method to solve the resulting sample problems appearing in each replication of the SAA. Both methods can also be used as a stand-alone method to solve the complete problem, i.e., considering the set of scenarios Ω. These methods are based on the formulation (1)-(15), and their details are presented in the following.

3.2.1. Branch-and-Cut Method

Our BC method is based on the dynamic addition of the SECs (8) to the model. These constraints are therefore initially dropped from the formulation and an exact routine separates them. Specifically, in our BC method we employ an exact separation algorithm based on the solution of minimum $s-t$ cut problems over the support graph of the branch-and-bound solutions. Consider the following notation. At any node of the tree, let $\bar{x}$ and $\bar{z}$ denote the solution of the vehicle flow and visit variables, respectively. We build an undirected support graph for each scenario ω and vehicle k, by creating a node n_i for each node $i \in \mathcal{N}_\omega$ in the original graph with $\bar{z}_{ik\omega} > 0$. The weight of each edge (n_i, n_j) such that $i < j$ is set to $\bar{x}_{ijk\omega}$.

We solve a minimum $s-t$ cut problem for each customer node n_i in the constructed graph, setting the node n_0 (corresponding to the depot in the original graph) as the source node s, and n_i as the sink node t. A violated SEC is identified if the capacity of the minimum cut is less than $2\bar{z}_{ik\omega}$. This type of separation was originally proposed by Padberg and Rinaldi (1990) and has been successfully applied in the literature (e.g., Adulyasak et al., 2014; Diabat et al., 2021; Alvarez et al., 2022). In our implementation, using the node partition $\mathcal{B}$ containing the sink (customer) node, we check the SEC for every scenario $\omega \in \Omega$ and vehicle $k \in \mathcal{K}$, adding it if a violation is observed. Note that Equation (8) is defined for any node ℓ in $\in \mathcal{B}$, which we choose as $\ell = \arg\max_{i \in \mathcal{B}}\{\bar{z}_{ik\omega}\}$. We solve the minimum cut problems with the algorithm available in the Concorde solver (Applegate et al., 2018). We separate the SECs at the root node of the tree to improve the linear relaxation of the model, and every time an integer solution is found.

In order to further enhance the performance of our BC algorithm, we incorporated some additional features. First, we included two sets of symmetry-breaking constraints (SBCs). The presence of symmetries can slow down the performance of the BC method since many redundant solutions may exist. For instance, we may alter the customer-driver assignments of a solution by only permuting the values of the vehicle index without changing the objective value. It is therefore essential to tackle symmetries in this type of formulation. The first set of SBCs, (16), imposes that, for the first scenario, vehicle k can only be used if vehicle $k-1$ is also used. These constraints are defined on a single scenario only (it could be any scenario) given that if we impose them for more than one scenario we might cut off the optimal solution. This could happen as these SBCs require the usage of vehicles of lower indices which, for some scenarios, might need to remain idle for consistency purposes. Next, constraint (17) orders the vehicles according to their aggregated assignment, indicating that vehicle k can be assigned to customers only if a vehicle with a smaller index was assigned as well. These two sets of constraints can be written as:

$$\sum_{j \in \mathcal{C}_\omega} x_{0jk\omega} \leq \sum_{j \in \mathcal{C}_\omega} x_{0j,k-1,\omega} \qquad \omega = 1,\, k \in \mathcal{K}\colon k > 1, \tag{16}$$

$$s_k \leq s_{k-1} \qquad k \in \mathcal{K}\colon k > 1. \tag{17}$$

We also impose constraints (18) which are trivial valid inequalities (VIs) that force every customer

to be assigned to at least one driver:

$$\sum_{k \in \mathcal{K}} y_{ik} \geq 1 \qquad\qquad i \in \mathcal{C}. \tag{18}$$

These VIs are valid since we have $D > 0$. We also use a primal heuristic to further enhance the BC method. In particular, given an integer solution (with respect to all the integer variables of the model), we can use the values of the y variable ($\bar{y}$) to try to generate a feasible solution for the complete problem. For this purpose, we solve each scenario subproblem individually for the assignment given by $\bar{y}$. If the cost of the resulting scenario solutions plus the first-stage cost of the current integer solution ($\sum_{i \in \mathcal{C}} o_i \bar{\lambda}_i$) is better than the incumbent cost, we have found a new best solution which becomes the incumbent in the BC tree. The heuristic is invoked every time an integer solution is found in the tree. Notice that this procedure can be executed regardless of whether the integer solution is feasible or not with respect to the SECs.

3.2.2. Benders Decomposition Method

The second algorithm we use to solve the sample problems is based on the Benders decomposition method (Benders, 1962). In a BD algorithm, the structure of the original problem is exploited to partition it into a *master problem* (MP) and a series of *subproblems*, which typically are easier to solve than the original problem. When applied to SP problems, the method is commonly referred to as the L-shaped method (Laporte and Louveaux, 1993; Rahmaniani et al., 2017). The application of a BD algorithm for our problem derives from the fact that when we fix the first-stage solution, it is possible to separate the $|\Omega|$ subproblems and solve them independently.

In our implementation, the MP is derived from the relaxation of the complicating constraints (3)-(8), related to the VRP part of the problem. The MP is, therefore, an assignment problem, defined as follows:

$$\min \sum_{i \in \mathcal{C}} o_i \lambda_i + \sum_{\omega \in \Omega} \rho_\omega \theta_\omega \tag{19}$$

$$\begin{aligned} \text{s.t.} \quad & (2), (9) - (12), \\ & \theta_\omega \geq 0 \qquad\qquad \omega \in \Omega, \\ & \textit{Optimality cuts}, \end{aligned} \tag{20}$$

where θ_ω is an auxiliary variable that captures the second-stage objective value under scenario ω. The optimality cuts impose the bounds derived in the subproblems to ensure that variables θ do not underestimate the second-stage costs. These cuts are generated by solving the subproblems, which correspond to prize-collecting VRPs with preassigned drivers. In particular, for a MP solution $\bar{y}$, we have the following subproblem for each scenario $\omega \in \Omega$:

$$\theta_\omega(\bar{y}) = \min \sum_{(i,j) \in \mathcal{E}_\omega} \sum_{k \in \mathcal{K}} c_{ij} x_{ijk} + \sum_{i \in \mathcal{C}_\omega} b_i (1 - \sum_{k \in \mathcal{K}} z_{ik}) \tag{21}$$

$$\text{s.t.} \quad \sum_{k \in \mathcal{K}} z_{ik} \leq 1 \qquad\qquad i \in \mathcal{C}_\omega, \tag{22}$$

$$\sum_{(j,i)\in\mathcal{E}_\omega} x_{jik} + \sum_{(i,j)\in\mathcal{E}_\omega} x_{ijk} = 2z_{ik} \qquad i \in \mathcal{C}_\omega,\ k \in \mathcal{K}, \tag{23}$$

$$\sum_{j\in\mathcal{C}_\omega} x_{0jk} \le 2 \qquad k \in \mathcal{K}, \tag{24}$$

$$z_{ik} \le \bar{y}_{ik} \qquad i \in \mathcal{C}_\omega,\ k \in \mathcal{K}. \tag{25}$$

$$\sum_{i\in\mathcal{C}_\omega} d_{i\omega} z_{ik} \le Q \qquad k \in \mathcal{K}, \tag{26}$$

$$\sum_{i\in\mathcal{B}} \sum_{\substack{j\in\mathcal{B}:\\ i<j}} x_{ijk} \le \sum_{i\in\mathcal{B}\setminus\{\ell\}} z_{ik} \qquad \forall \mathcal{B} \subseteq \mathcal{C}_\omega\colon |\mathcal{B}| \ge 2,\ k \in \mathcal{K},\ \ell \in \mathcal{B}, \tag{27}$$

$$z_{ik} \in \{0,1\} \qquad i \in \mathcal{C}_\omega,\ k \in \mathcal{K}, \tag{28}$$

$$x_{ijk} \in \{0,1,2\} \qquad (i,j) \in \mathcal{E}_\omega\colon i = 0,\ k \in \mathcal{K}, \tag{29}$$

$$x_{ijk} \in \{0,1\} \qquad (i,j) \in \mathcal{E}_\omega\colon i > 0,\ k \in \mathcal{K}, \tag{30}$$

where all the variables and constraints maintain the same interpretation as in Section 2.1 but dropping the scenario index ω.

Note that the subproblems are VRPs and not traveling salesman problems (TSPs) since, in the MP, each customer can be assigned to more than one driver as per constraints (2). However, these subproblems are typically constrained to a significant degree by the linking constraints (25) and we can, in general, solve them efficiently via the BC algorithm.

Our BD algorithm was implemented in a branch-and-check fashion in which the MP is solved only once and the Benders cuts are generated on the fly. For this purpose, we solve the MP using a BC algorithm in which every time a feasible solution $(\bar{y}^*, \bar{\theta}^*)$ is found, we solve the scenario subproblems to verify the feasibility of the solution. Specifically, given the current candidate MP solution and the subproblem optimal objective function values $\theta_\omega(\bar{y}^*)$ (if any), if we have $\bar{\theta}^*_\omega < \theta_\omega(\bar{y}^*)$ then $\bar{\theta}^*_\omega$ underestimates the second-stage cost for scenario ω. In this case, we add the following optimality cut (Laporte and Louveaux, 1993):

$$\theta_\omega \ge \theta_\omega(\bar{y}^*) - \theta_\omega(\bar{y}^*)\left(\sum_{i\in\mathcal{C}} \sum_{\substack{k\in\mathcal{K}:\\ \bar{y}^*_{ik}=1}} (1 - y_{ik}) + \sum_{i\in\mathcal{C}} \sum_{\substack{k\in\mathcal{K}:\\ \bar{y}^*_{ik}=0}} y_{ik} \right), \tag{31}$$

which ensures that the θ variable will assume the value $\theta_\omega(\bar{y}^*)$ when the MP solution is $\bar{y}^*$. These cuts are verified for every scenario subproblem ω.

It is well-known that these cuts can be weak, which does not favor the performance of the BD method. We have, therefore, tried to enhance the BD efficiency by using initial cuts to strengthen the initial MP. For this purpose we have added bounds of the form

$$\theta_\omega \ge \sum_{i\in\mathcal{C}_\omega} \min\left\{ b_i, \min_{\substack{j\in\mathcal{N}_\omega:\\ j<i}} \{c_{ji}\}, \min_{\substack{j\in\mathcal{N}_\omega:\\ i<j}} \{c_{ij}\} \right\}, \tag{32}$$

which impose a lower bound on the cost of each subproblem. This lower bound corresponds to the sum of the minimum between the minimum travel cost to reach each customer and the skipping cost. We also use the SBCs and VIs of the BC method to enhance the BD method.

Finally, in our implementation, we always evaluate $\theta_\omega(\bar{y}^*)$ for all the subproblems, even if a violated optimality cut (31) is found early in this process. This allows us to always find feasible solutions to the original problem by combining the subproblem solutions with the corresponding first-stage solution.

4. Computational Experiments

This section aims at demonstrating the value of considering uncertainty in the context of the ConVRP. We also aim to further demonstrate the value of consistency for logistics and transportation, in particular when uncertainty is involved. We first discuss the instances in Section 4.1. Next, we assess the performance of the solution methods, whose results and comparison are presented in Section 4.2. Then, in Sections 4.3 and 4.4, we focus on analyzing the impact of taking the consistency requirements and uncertainty into account and the trade-offs arising from their inclusion.

All the algorithms were coded in C++, using CPLEX 22.1 as MIP and LP solver. We set the relative MIP optimality gap tolerance to 10^{-5} and the rest of the parameters of the solver were kept at their default values. The experiments were executed on 2.4 GHz processors, using a single thread and a RAM limit of 32GB.

4.1. Problem Instances

Four main sets of instances have been used in the ConVRP literature. Since these datasets have been introduced for the deterministic case, they present planning horizons of three to five days. Moreover, all the customers have a fixed service frequency, i.e., each customer appears on each of the days with a given probability. The first two sets were proposed by Groër et al. (2009) and contain small and large instances. Set A contains 10 instances, five of them with 10 customers and the remaining five with 12 customers. Data set B, in turn, contains 12 instances with 50 to 199 customers. In these two sets the visit frequency is 70%. Set C was introduced by Kovacs et al. (2014b) and extends set B by considering visit frequencies of 50% and 90%. Finally, data set D was presented by Goeke et al. (2019) and builds on set C to consider medium-sized instances. As such, this set has six instances with 20 and 30 customers (more instances are derived by the combination with other parameters concerning the deterministic ConVRP).

For our experiments, we construct instances from those of sets A, B and D since they include small-, medium- and large-size instances regarding the number of customers. We considered the 12 instances of set A, the seven instances of set B with up to 100 customers, and all the six instances of set D. From each of these 23 instances, we can then generate instances by defining a set of scenarios Ω indicating the observed customers and their demands.

The uncertain parameters were assumed to be independent random variables and we applied a Monte Carlo simulation to generate the scenarios. For the case of stochastic customers, each customer has a probability of occurrence α_i. When considering stochastic demands, we used a discrete uniform distribution in the interval $[\bar{d}_i(1-\epsilon), \bar{d}_i(1+\epsilon)]$, $\forall i \in \mathcal{C}_\omega$, where $\epsilon \in [0,1]$ is the demand uncertainty level which we assumed to be 0.50. In this context, $\bar{d}_i$ was taken from the values of the deterministic instances. Specifically, for set A we have $\bar{d}_i = 2\ \forall i \in \mathcal{C}_\omega$, while for sets B and D we set $\bar{d}_i$ as the maximum observed demand for each customer in the deterministic instance. When we generate

scenarios considering both customer presence and demand as random variables, we first draw the realization of the customer presence and then, for those customers that are present, we draw their demand using the discrete uniform distribution described above. The probability of occurrence of each scenario is set as $\rho_\omega = 1/|\Omega|$, $\forall \omega \in \Omega$.

In our experiments, unless stated otherwise, we generated instances by combining values of $|\Omega| \in \{100, 500\}$ and $\alpha_i \in \{0.2, 0.5, 0.8\}$ $\forall i \in \mathcal{C}$. This combination results in 138 instances with up to 100 customers and up to 500 scenarios. For each instance, the number of vehicles was set to $\max\{\lceil 2d_\Omega^{\max}/Q \rceil - 1, D + 1\}$, with $d_\Omega^{\max} = \max_{\omega \in \Omega}\{\sum_{i \in \mathcal{C}_\omega} d_{i\omega}\}$. The first term comes from Chitsaz et al. (2019) and ensures that the instance is feasible with respect to the largest demand scenario. The second term ensures we have enough vehicles regarding the consistency target. The values of the penalty terms were set to $b_i = 3(2c_{0i})$ and $o_i = 0.1(2c_{0i})$ $\forall i \in \mathcal{C}$, while the value of D (targeted maximum number of different drivers) was set to 1.

4.2. Comparison of Solution Approaches

In this section, we aim to compare the performance of the BC and BD, as standalone methods, and the SAA algorithm using the BC and BD to solve the sample problems (SAA-BC and SAA-BD, respectively). The idea is to choose the method we will use for the (upcoming) sections regarding the analysis of the value of considering consistency and uncertainty.

For the SAA approaches, we present the results of the parameter combination (number of samples M and their size $|N|$) that resulted in the best performance. Specifically, for SAA-BC we used the combination $M = 20$ and $|N| = 5$ and for SAA-BD we used $M = 20$ and $|N| = 15$. A time limit of 30 minutes was set for each sample problem, resulting in a maximum time of 10 hours. This time does not include the evaluation time, which is usually under 15 minutes in total. For the BC and BD as standalone methods, we set a time limit of 10 hours. Detailed analyses on the parameter choice for the different methods are presented in Appendix A and Appendix B.

In Table 1, separated by probability of occurrence, we present the following statistics about the different methods: (*i*) The number of optimal solutions found by the method (for the BC and the BD this status is proven during the execution while for the SAA methods this is evaluated with respect to the best lower bound of the BC and the BD); (*ii*) The average total cost of the solutions found by the method; (*iii*) The average relative difference ('Gap to best') of the objective value of the solutions of each method with respect to the best solution found by all the methods simultaneously. The relative differences are computed as $100 \times (z - z^*)/z^*$, where z is the objective value of the solution of the method and z^* is the objective value of the best solution; (*iv*) The average total CPU time of the method, including the evaluation time for the SAA methods, in seconds; (*v*) The average time to find the best solution by the method, in seconds. Additionally, we present the number of instances in each probability of occurrence group ('No. of instances'). We have also included a column ('Total') indicating the average over all the instances except for the counters for which we display sums.

At first glance, we can notice the difficulty in proving the optimality of the solutions when the probability of occurrence increases. This observation is a result of denser scenarios regarding the number of customers. We can observe, nevertheless, that both SAA methods can find optimal solutions

Table 1: Performance comparison of the methods

Statistic	Method	Prob. of occurrence			Total
		20%	50%	80%	
No. of instances		46	46	46	138
No. of optimal*	BC	31	16	0	47
	BD	30	16	0	46
	SAA-BC	30	11	0	41
	SAA-BD	30	14	0	44
Total cost	BC	184.56	1,310.10	2,833.00	1,442.55
	BD	171.32	395.91	588.29	385.17
	SAA-BC	154.88	257.83	346.32	253.01
	SAA-BD	164.08	354.17	501.29	339.84
Gap to best (%)	BC	8.50	160.44	300.63	156.52
	BD	6.43	24.78	36.23	22.48
	SAA-BC	2.24	0.21	0.11	0.85
	SAA-BD	4.74	16.14	21.19	14.02
Total time (s)	BC	12,037	23,952	35,838	23,942.39
	BD	31,352	32,130	32,823	32,101.51
	SAA-BC	7,905	14,385	23,499	15,262.93
	SAA-BD	33,502	36,072	36,090	35,221.36
Time to best (s)	BC	4,098	6,550	7,406	6,017.92
	BD	3,145	5,224	6,940	5,103.41
	SAA-BC	5,049	7,383	9,003	7,145.02
	SAA-BD	6,731	7,923	9,500	8,051.56

in most cases when the BC proves their optimality. It is valuable to mention that all the methods found feasible solutions for all the instances. From the values of 'Total cost' and 'Gap to best' we can also see that, on average, SAA-BC finds the best quality solutions, clearly outperforming the other methods for all the probabilities of occurrence.

Regarding the CPU time required by the methods, it is possible to see that it increases with the probability of occurrence. SAA-BC presents the lowest average time for all the cases since it is able to solve more (small) samples to optimality, finishing early for several instances. A further analysis of the results also revealed that the lower bounds of the BC and BD are rather weak even for small-sized instances. This is a result of the well-known relatively weak LP relaxation of standard vehicle flow formulations for VRPs. Furthermore, the 'Time to best' values reveal that all the methods find their best solutions relatively early in the search process, regardless of the probability of occurrence in the scenarios.

Table 2 displays, for every method, the average relative difference of the solutions' objective value to the best solutions ('Gap to best') separated according to the number of scenarios $|\Omega|$ for every

dataset. We also show the number of instances in each group and, in the last row, the average over all instances. In this table it is possible to observe that SAA-BC consistently provides good quality solutions when compared to the other methods, regardless of the test set and the number of scenarios of the instances. As expected, set B is the most challenging one for all the methods as its instances are the largest in terms of the number of customers.

After this analysis, it is possible to conclude that SAA-BC consistently provides good quality solutions and outperforms the other methods regarding the analyzed statistics. For instance, the optimality gap of the solutions found by SAA-BC is less than 0.2% if we consider the 47 instances for which BC proved their optimality. We have, therefore, selected SAA-BC as the main method for the analyses presented in the rest of this study.

Table 2: Gap to best solution (%) for the different instance classes and number of scenarios

$\|\Omega\|$	Set	No. of instances	Method			
			BC	BD	SAA-BC	SAA-BD
100	A	30	0.26	0.75	0.20	0.02
	B	21	393.69	58.15	4.23	42.25
	D	18	0.83	12.31	0.29	7.09
500	A	30	0.74	2.47	0.16	0.09
	B	21	582.16	58.26	0.37	37.25
	D	18	59.02	18.86	0.27	7.48
Total		138	156.52	22.48	0.85	14.02

In order to provide a base for comparison, we show in Table 3 a summary of the characteristics of the solutions found by SAA-BC. In particular, the table presents, separated by the size of the scenario set and the probability of occurrence, the following statistics: the average total cost, the average transportation and skipping cost and the penalties for violating the consistency target as a percentage of the total cost; the average number of clusters, the violation level of the consistency target, and the percentage of skipped customers. The percentage of skipped customers is computed as the total number of skipped customers (calculated over all scenarios) divided by total number of customers with strictly positive demand (calculated over all scenarios) Note also that a cluster in our context is a vehicle to which at least one customer was assigned.

The violation level of the consistency target is measured using the metric proposed by Diabat et al. (2021) and extended in Alvarez et al. (2022) in the context of integrated inventory and production routing problems. For this purpose, let $\bar{y}_i^k$ be the solution for variables y_i^k (whether driver k is assigned to customer i). The driver consistency metric κ is as follows:

$$\kappa^D = \frac{\sum_{i \in \mathcal{C}} \max\left\{\sum_{k \in \mathcal{K}} \bar{y}_i^k - D, 0\right\}}{|\mathcal{C}|} \times 100. \tag{33}$$

The value of κ^D computes the average deviation from the driver consistency target D. If every

customer i is assigned to at most D drivers, then the numerator is 0, and then κ^D is equal to 0 as well. This case indicates a solution not deviating from the consistency target. A nonzero value indicates the average violation of the target per customer. Notice that κ^D considers a deviation from the target only when the number of drivers assigned to the customers exceeds D. Note also that in Equation (33) the average deviation is multiplied by 100 in order to increase its significance and facilitate its interpretation, since otherwise κ may take relatively small fractional values.

In the table, we can observe how the cost of the solutions increases with the probability of occurrence. This is mostly due to the increase in transportation costs in the solutions, as more customers are observed in the scenarios. The relative (and absolute) skipping costs and penalties also tend to increase with the probability of occurrence. However, it is important to highlight that the fraction of skipped customers remains relatively low (around 1% overall) as well as the observed violation levels (7.37 on average). This observation implies that to find solutions with relatively low violations of the driver consistency target it is not necessary to compromise the service level by skipping a significant number of customers.

Table 3: Summary of the attributes of the solutions of SAA-BC

$\lvert\Omega\rvert$	Prob. of occurrence	No. of instances	Total cost	Travel cost (%)	Consistency penalty (%)	Skipping cost (%)	No. of clusters	Violation level	Skipped customers (%)
100	20%	23	152.99	96.55	0.07	3.38	1.70	0.19	0.98
	50%	23	257.70	92.08	1.93	5.99	3.04	5.82	1.33
	80%	23	341.33	93.08	4.15	2.77	4.04	16.70	0.44
500	20%	23	156.77	95.95	0.24	3.81	1.74	0.36	1.14
	50%	23	257.97	91.86	1.93	6.21	3.26	6.87	1.27
	80%	23	351.31	92.32	3.28	4.41	4.30	14.27	0.72
Total		138	253.01	93.64	1.93	4.43	3.01	7.37	0.98

4.3. The Cost of Consistency

This section aims to assess the cost of consistency in our context. In the deterministic ConVRP literature, it is well-known that, in general, consistency can be improved without sacrificing too much on standard operational metrics such as the total travel cost (Smilowitz et al., 2013). We aim to verify if this observation remains valid in the stochastic case. For our purposes, we compare the solutions of the stochastic program (generated via SAA-BC) with solutions generated from two different approaches. The first approach corresponds to a case in which we completely ignore the consistency requirements, optimizing the (daily) scenarios separately. In the second case, we analyze the impact of a higher value for the consistency target D. For each case, we provide statistics and analyses regarding the corresponding comparison.

4.3.1. Decoupled Approach

In the first case, we are interested in knowing the impact of ignoring consistency in our stochastic ConVRP. That implies not considering constraints (2) in the original formulation and its cost in the objective function (1). We refer to this case as the *decoupled* approach, which results in a stochastic

program completely separable by scenario. As such, we can solve the scenario problems individually. However, to avoid solutions with artificially high consistency violations (since we solve the scenario problems without interdependencies), the drivers of the resulting routes can be reassigned while minimizing the consistency violations and preserving the structure of the routes. This can be done with the reassignment formulation presented in Appendix C, which takes as input a set of routes for every scenario. The result of the formulation is the same set of routes with (potentially) new drivers assigned to each of them so that the consistency violation is minimized. After applying this procedure, we can assess the consistency of the resulting solution and compare it with that of the original stochastic program. Notice that the decoupled approach provides a best-case scenario for the second-stage costs.

An analysis in this regard is presented in Table 4 which displays, separated by the probability of occurrence, statistics for the stochastic program ('SP') and the decoupled approach. We present for each group the average total cost of the solutions and the average of each of the objective function components. Moreover, we also display the average number of clusters, violation level of the driver consistency constraints, and percentage of skipped customers in the solutions. This analysis considers only those instances for which we were able to compute the solution for the decoupled approach within 10 hours and, furthermore, the reassignment formulation was solved to optimality within 30 minutes. The analysis thus considers the solutions of 88 instances in total, of which 34, 28 and 26 correspond to the case with 20%, 50% and 80% of probability of occurrence, respectively.

In the table, it is possible to observe that the decoupled approach leads to solutions with higher total costs. As expected, the cost difference comes mainly from the consistency penalties, even after optimally reassigning the drivers. The observed violations of the consistency target are relatively high in the decoupled approach when compared to the base case. In particular, for the case with 80% of probability of occurrence, the violation level is more than four times higher than that of the stochastic program.

We can also observe that the routing costs present a slight decrease when applying the decoupled approach since its objective function only considers the trade-off between routing and skipping costs. All the customers are, therefore, always visited in the decoupled approach since skipping is never a profitable option, which also results in more vehicles being used (i.e., more clusters) in the decoupled approach. This observation also highlights the gains in flexibility brought by considering skipping as a recourse in the stochastic ConVRP context. In practice, these skips can be seen as delivery outsourcing or as a compensation paid to the customer and allow to maintain relatively low violation levels of the consistency target. These results also show that promoting consistency with a flexible approach in the form of soft constraints allows the finding of solutions with adequate consistency metrics without compromising excessively on other performance metrics such as routing costs or penalties for skipping customers.

4.3.2. Increasing the Consistency Target

In the second case, we investigate the effects of considering a higher consistency target value. For this purpose, we have solved the instances with the SAA-BC method but considering $D = 2$ and $D = 3$. The results obtained with this configuration are compared with respect to those obtained with $D = 1$.

Table 4: Comparison of the stochastic program with a decoupled approach

Approach	Prob. of occurrence	Total cost	Travel cost	Consistency penalty	Skipping cost	No. of clusters	Violation level (κ^D)	Skipped customers (%)
SP	20%	71.07	67.81	0.04	3.22	1.15	0.10	0.64
	50%	83.07	79.48	0.59	3.00	1.68	5.72	1.19
	80%	94.79	91.29	1.93	1.57	2.19	16.45	0.33
Decoupled	20%	89.05	67.15	21.90	0.00	1.76	19.86	0.00
	50%	103.58	76.50	27.07	0.00	2.14	58.80	0.00
	80%	116.32	87.37	28.95	0.00	2.27	76.32	0.00

Table 5 presents, for each value of D, the average total cost of the solutions as well as the average of its individual components. Moreover, the table displays the average number of clusters as well as the violation level (κ^D) and percentage of skipped customers of the solutions.

The results reveal that the added flexibility of considering a more relaxed consistency target leads to solutions with lower costs. This cost reduction is achieved by further reducing the fraction of customers that are skipped and the level of deviation from the consistency target. These solutions have on average more clusters, i.e., more vehicles are used in them. When we set $D > 1$ the routing costs account for almost the total cost of the solutions. This may be due to the reduction of the trade-off between the consistency costs and the operational costs when the target is loose. In that case, the problem reduces almost to the individual optimization of the scenario problems for which we aim at reducing their routing and skipping costs.

In the solutions with $D = 2$, the average violation level with respect to a target of one driver per customer (κ^1) is 76.96, in contrast to the value of 7.37 observed when $D = 1$. Analogously, when we set a target of $D = 3$ and evaluate the violation level with respect to a target of one driver per customer we obtain a value of 146.89. These results further highlight that we can adequately promote consistency with a flexible soft-constrained approach by accounting for the cost of such flexibility. No experiments for larger values of D are included since for $D = 3$ the consistency constraints are almost inactive.

Table 5: Results for different consistency targets

Consistency target D	Total cost	Travel cost	Consistency penalty	Skipping cost	No. of clusters	Violation level (κ^D)	Skipped customers (%)
1	253.01	229.07	8.04	15.90	3.01	7.37	0.98
2	232.94	231.07	1.42	0.45	4.42	0.49	0.07
3	222.58	222.40	0.00	0.18	5.23	0.00	0.04

4.4. The Value of the Stochastic Solutions

In this section, we evaluate the value of the stochastic solutions (VSS). This evaluation is performed by comparing the solutions of the stochastic program with those obtained with an expected value (EV) approach. In the EV problem, instead of the scenario-based SP formulation, we use a single scenario

in which the random variables take the value of their corresponding expectation. We then evaluate the expected cost of using the solution of the EV problem (EEV). For this purpose, we solve the scenario-based SP problem with its first-stage decisions fixed to the values of the solution of the EV problem. VSS is then computed as the difference between the optimal value of the EEV problem and the SP model, i.e., VSS = EEV − SP.

VSS measures the potential advantages of using the SP approach with respect to a case in which the uncertainty is ignored in the planning stage. Note that to evaluate the expected cost of using the EV solution (EEV problem) we only need to solve $|\Omega|$ scenario subproblems (with preassigned drivers) independently. We set a total time budget of 10 hours, with a time limit of one hour for each individual problem (including the EV problem). We present the results of the instances for which all the problems were solved to optimality to avoid reporting artificially high VSS.

In this context, we present in Table 6 some statistics regarding this analysis, separated by number of scenarios $|\Omega|$ and probability of occurrence. In particular we computed the relative VSS with respect to the objective value of the SP solutions, the relative change of the travel costs, the change in percentage of skipped customers and the relative change of the number of clusters. The last three statistics are computed for the EEV solution relative to the SP solution. Additionally, the third column shows the number of instances considered for the statistics of each group. It is worth mentioning that in the EEV solutions, the consistency requirement is always met. In the EV problem approach, since we have a single scenario, there is no incentive to assign more than one driver to each customer given that customers can be visited by at most one vehicle in every scenario.

The results show that significant cost increases can be incurred when ignoring the uncertainties in our context. In particular, it is worth highlighting that the average VSS represents approximately 20% of the objective value for 80% of probability of occurrence (for the instances with 100 scenarios). It is possible to see that the average VSS increases with the density of the scenarios. These observations justify the use of the SP model since otherwise significant cost increases would be incurred.

Furthermore, the relative changes of the travel cost reveal that the EEV solutions yield plans with lower routing costs in the execution phase. This is achieved, however, by skipping a significantly larger number of customers (compared to the SP solutions). This may be explained by the creation of fewer and larger clusters, i.e., the assignment of fewer vehicles to the customers in the planning (first) stage in the EV problem. By ignoring the uncertainties in the EEV problem in the form of a planning stage with a single scenario with (relatively) small demands, the customers are grouped together in fewer, larger clusters. This results in turn in the impossibility of visiting all the customers in the execution stage due to capacity constraints. These results highlight the need to adequately consider uncertainty when planning for operations with consistency requirements.

We also calculate a different value of the stochastic solutions by considering only the demand uncertainty when defining the EV problem. We refer to this approach as VSS2 and the deterministic problem solution is referred to as EEV2. In particular, in the EV problem we set the expected demand value of each customer $i \in \mathcal{C}$ as its average demand over the scenarios in which the customer is observed Ω_i, i.e., $d_i = \sum_{\omega \in \Omega_i} d_{iw}/|\Omega_i|$ (in the first approach the denominator is $|\Omega|$). The expected demand values in this approach tend to be larger since this approach ignores the uncertain nature of

Table 6: Value of the stochastic solutions

$\|\Omega\|$	Prob. of occurrence	No. of instances	%VSS/OF	% change travel cost	Change in % skipped customers	% change no. of clusters
100	20%	18	0.87	-1.40	0.75	-9.26
	50%	14	2.56	-2.07	1.35	-20.24
	80%	15	20.66	-3.88	4.51	-5.00
500	20%	18	0.82	-1.89	0.79	-10.19
	50%	13	2.65	-2.15	1.84	-32.05
	80%	15	18.37	-3.03	4.09	-6.56
Total		93	7.38	-2.36	2.15	-13.15

the customers' presence. We present in Table 7 an analysis analogous to the one in Table 6.

When comparing these results with those of Table 6, it is possible to observe how the relative VSS increases (on average over all the instances) when more aspects of the uncertainty in the system are ignored. As such, an average cost increase of 14.8% is observed in the VSS2 case, in contrast to the 7.4% obtained in the previous analysis. These values further highlight the importance of considering uncertainty in the planning phase. Moreover, the results show the significant impact of neglecting the uncertainty on the customers' presence in this context.

An important observation about the results of VSS2 is the variation of the results with the probability of occurrence α. In general, the number of clusters tends to be largely overestimated for lower values of α (when the observed scenarios have fewer customers), which constrains to a larger extent the routing decisions and increases the corresponding cost. Large values of α impact also the service level by requiring more skips even without overestimating the number of clusters. This observation may be partially explained by the capacity constraints of the vehicles, which do not allow for serving the observed demands when α is high. On the other hand, for relatively low values of α the number of skips is reduced (compared to the SP solutions), as a result of the creation of clusters with few customers, whose observed demands can be accommodated by the vehicles.

Table 7: Value of the stochastic solutions ignoring the customers' presence uncertainty (VSS2)

$\|\Omega\|$	Prob. of occurrence	No. of instances	%VSS2/OF	% change travel cost	Change in % skipped customers	% change no. of clusters
100	20%	15	20.58	21.23	-0.20	100.00
	50%	14	17.95	26.09	-1.22	53.57
	80%	13	4.29	4.98	0.39	1.92
500	20%	15	22.18	23.35	-0.38	113.33
	50%	14	16.38	25.47	-1.03	35.71
	80%	12	4.39	5.88	0.35	0.00
Total		83	14.83	18.39	-0.37	53.92

Note that VSS and VSS2 indicate the quality of some simplified heuristic solutions obtained by deterministic approaches. As a heuristic, EEV (incorporating both probabilistic occurrence and demand) provides a better heuristic approach compared to EEV2 (where the probability of occurrence is ignored) for cases where $\alpha = 20$ and 50%. However, for cases where $\alpha = 80\%$, the second approach provides a better heuristic solution. Finally, it is worth mentioning that both EEV and EEV2 provide solutions in which each customer is assigned to a single driver only, i.e., solutions that do not violate any consistency target.

4.5. The Impact of Regular Customers

In a final experiment, we explored scenarios with different presence profiles for the customers. For this purpose, we define as *regular* (or *frequent*) customers those with 50% probability of occurrence while *occasional* customers have a probability of occurrence of 20%. In this context, we created instances with different proportions of regular customers (the rest of the customers is set as occasional). In particular, we explored five different proportions of regular customer in the scenario generation, namely 0, 20, 50, 80 and 100%. The results are shown in Table 8 and correspond to the average total cost of the solutions and its individual components, the average number of clusters, the violation level of the consistency target and percentage of skipped customers of the solutions. We also show the average relative VSS for each case, computed as in Section 4.4. Each group in the table contains 46 instances (with 100 and 500 scenarios). However, for the results in the last column we report only the average over those instances for which we can compute the VSS to optimality.

The results show that all the cost components tend to increase with the proportion of regular customers. In particular, the routing costs show significant increases with the proportion of more frequent customers while the percentage of skipped customers remains relatively stable. On the other hand, the results further reveal the difficulty of maintaining good driver consistency metrics when the scenarios have more regular customers. However, the violation levels remain relatively low at a value of 6 for the case with 100% of regular customers. The values of VSS further reveal that the presence of more regular customers increases the importance of adopting approaches to explicitly take into consideration the uncertainty in the problem.

Table 8: Comparison for different proportions of regular and occasional customers

Profile	Total cost	Travel cost	Consistency penalty	Skipping cost	No. of clusters	Violation level	Skipped customers (%)	%VSS/OF
0% regular	154.88	143.26	0.59	11.04	1.72	0.28	1.06	0.84
20% regular	180.49	170.97	0.70	8.82	2.20	0.51	0.87	1.86
50% regular	208.65	198.25	1.37	9.03	2.78	1.82	0.90	2.64
80% regular	239.27	219.42	2.34	17.50	2.98	3.09	1.30	3.21
100% regular	257.83	234.00	6.81	17.03	3.15	6.34	1.30	2.60

5. Conclusions

In this paper, we have introduced and studied the consistent vehicle routing problem under the consideration of stochastic customers and demands. We studied the problem using a two-stage scenario-based stochastic programming approach. The first stage represents the planning phase and consists of assigning drivers to customers. These assignments remain fixed in the operational (second) stage in which, after the realization of the random variables, we define which customers will be visited by which vehicle and the routes that the vehicles will follow, respecting the selected assignments. In our problem, we set a target for the desired maximum number of drivers assigned to each customer. We then promote consistency via a flexible approach that penalizes violations of the consistency target. We have chosen a sample average approximation using a branch-and-cut algorithm to solve the sample problems as the method to evaluate the value of taking consistency into account in our context as well as to explore the cost-consistency trade-offs in the problem.

On the test set used, we have shown that our flexible approach allows finding solutions with adequate consistency requirements without compromising excessively on other performance metrics such as routing costs or penalties for skipping customers. In particular, the violation level of the consistency target can be more than three times higher when we do not optimize considering explicit consistency requirements. We have also verified the negative impacts of not considering the problem uncertainties during the planning stage. Our experiments showed that adopting our stochastic setting allows to avoid cost increases of up to 20% when compared to a deterministic case.

In the future, the problem could be extended to other forms of consistency. Also, studying the impact and interrelation between different consistency requirements could be a promising path for follow-up research. Future research directions to extend this study could also focus on solution approaches for the problem. Alternative formulations and enhancements for the branch-and-cut and Benders decomposition methods are an avenue that could be explored. Moreover, heuristic algorithms like progressive hedging is a worthy research direction.

References

Adulyasak, Y., Cordeau, J.-F., and Jans, R. (2014). Formulations and branch-and-cut algorithms for multivehicle production and inventory routing problems. *INFORMS Journal on Computing*, 26(1):103–120.

Alvarez, A., Cordeau, J.-F., and Jans, R. (2022). The consistent production routing problem. *Networks*, 80(3):356–381.

Applegate, D., Bixby, R., Chvátal, V., and Cook, W. (2018). Concorde TSP solver. `http://www.math.uwaterloo.ca/tsp/concorde.html`. Accessed: 2018-07-20.

Benders, J. F. (1962). Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4:238–252.

Bertsimas, D. J. (1992). A vehicle routing problem with stochastic demand. *Operations Research*, 40(3):574–585.

Braekers, K. and Kovacs, A. A. (2016). A multi-period dial-a-ride problem with driver consistency. *Transportation Research Part B: Methodological*, 94:355–377.

Chitsaz, M., Cordeau, J.-F., and Jans, R. (2019). A unified decomposition matheuristic for assembly, production, and inventory routing. *INFORMS Journal on Computing*, 31(1):134–152.

Coelho, L., Cordeau, J.-F., and Laporte, G. (2012). Consistency in multi-vehicle inventory-routing. *Transportation Research Part C: Emerging Technologies*, 24:270–287.

Diabat, A., Archetti, C., and Najy, W. (2021). The fixed-partition policy inventory routing problem. *Transportation Science*, 55(2):353–370.

Gendreau, M., Jabali, O., and Rei, W. (2016). 50th anniversary invited article-future research directions in stochastic vehicle routing. *Transportation Science*, 50(4):1163–1173.

Goeke, D., Roberti, R., and Schneider, M. (2019). Exact and heuristic solution of the consistent vehicle-routing problem. *Transportation Science*, 53(4):1023–1042.

Groër, C., Golden, B., and Wasil, E. (2009). The consistent vehicle routing problem. *Manufacturing & Service Operations Management*, 11(4):630–643.

Jordan, W. C. and Graves, S. C. (1995). Principles on the benefits of manufacturing process flexibility. *Management Science*, 41(4):577–594.

Kleywegt, A. J., Shapiro, A., and Homem-de Mello, T. (2002). The sample average approximation method for stochastic discrete optimization. *SIAM Journal on Optimization*, 12(2):479–502.

Kovacs, A. A., Golden, B. L., Hartl, R. F., and Parragh, S. N. (2014a). Vehicle routing problems in which consistency considerations are important: A survey. *Networks*, 64(3):192–213.

Kovacs, A. A., Golden, B. L., Hartl, R. F., and Parragh, S. N. (2015). The generalized consistent vehicle routing problem. *Transportation Science*, 49(4):796–816.

Kovacs, A. A., Parragh, S. N., and Hartl, R. F. (2014b). A template-based adaptive large neighborhood search for the consistent vehicle routing problem. *Networks*, 63(1):60–81.

Laporte, G. and Louveaux, F. V. (1993). The integer L-shaped method for stochastic integer programs with complete recourse. *Operations Research Letters*, 13(3):133–142.

Ledvina, K., Qin, H., Simchi-Levi, D., and Wei, Y. (2022). A new approach for vehicle routing with stochastic demand: Combining route assignment with process flexibility. *Operations Research*, 70(5):2655–2673.

Padberg, M. and Rinaldi, G. (1990). Facet identification for the symmetric traveling salesman polytope. *Mathematical programming*, 47(1):219–257.

Rahmaniani, R., Crainic, T. G., Gendreau, M., and Rei, W. (2017). The benders decomposition algorithm: A literature review. *European Journal of Operational Research*, 259(3):801–817.

Rodríguez-Martín, I. and Yaman, H. (2022). Periodic vehicle routing problem with driver consistency and service time optimization. *Transportation Research Part B: Methodological*, 166:468–484.

Smilowitz, K., Nowak, M., and Jiang, T. (2013). Workforce management in periodic delivery operations. *Transportation Science*, 47(2):214–230.

Song, Y., Ulmer, M. W., Thomas, B. W., and Wallace, S. W. (2020). Building trust in home services-tochastic team-orienteering with consistency constraints. *Transportation Science*, 54(3):823–838.

Spliet, R. and Dekker, R. (2016). The driver assignment vehicle routing problem. *Networks*, 68(3):212–223.

Sungur, I., Ren, Y., Ordóñez, F., Dessouky, M., and Zhong, H. (2010). A model and algorithm for the courier delivery problem with uncertainty. *Transportation Science*, 44(2):193–205.

Wang, K., Zhen, L., Xia, J., Baldacci, R., and Wang, S. (2022). Routing optimization with generalized consistency requirements. *Transportation Science*, 56(1):223–244.

Yao, Y., Van Woensel, T., Veelenturf, L. P., and Mo, P. (2021). The consistent vehicle routing problem considering path consistency in a road network. *Transportation Research Part B: Methodological*, 153:21–44.

Appendix A. Impact of Methods Enhancements

This section evaluates the enhancements proposed for the branch-and-cut (BC) and Benders decomposition (BD) methods. In this regard, we have compared the full method (Base case) to methods without the individual enhancements. Tables A.9 and A.10 show the results of the analysis for the BC and BD method, respectively. In the tables, we present the number of instances in each group and the number of feasible and optimal solutions found by the methods. In addition, we report the average cost of the solutions and the average CPU time and time to find the best (final) solution, in seconds. 'No Ineq' displays the results without the valid inequalities (16)-(18) for both approaches and for the BC method, 'No primal' shows the results without the primal heuristic. Notice that the primal heuristic is embedded naturally in the BD method (since we solve all the scenario subproblems for evaluation purposes). As such, we did not evaluate a case without this component for the BD. In these experiments, we imposed a time limit of 30 minutes per instance intending to find adequate configurations for the methods used within the SAA approach. For this experiments we considered the same instances of the main manuscript, as well as additional sets with 10, 20 and 50 scenarios.

The results show the positive impacts of considering the enhancements. In particular, we can observe a significant improvement in the quality of the solutions when we include the primal heuristic in the BC method. Moreover, the full version of the BC and BD method prove the optimality of two and three additional solutions within the time limit, respectively. In addition, in the BC method the time to the best solution ('Time to best') is shorter when the enhancements are included while for the BD this time increases. However, this increase is compensated with the significant improvement in the quality of the solutions in the full version of the BD method.

Table A.9: Impact of the enhancements on the BC method

Case	No. of instances	No. of feasible	No. of optimal	Total cost	Total time	Time to best
Base	345	345	144	1,156.51	1,088	269
No Ineq	345	345	142	1,439.51	1,092	317
No primal	345	345	142	1,904.86	1,084	370

Table A.10: Impact of the enhancements on the BD method

Case	No. of instances	No. of feasible	No. of optimal	Total cost	Total time	Time to best
Base	345	345	5	379.24	1,787	362
No Ineq	345	345	2	408.14	1,791	311

Appendix B. Choice of the SAA Parameters

This section shows the results of the SAA-BC and SAA-BD methods for different values of M and $|N|$ and time limits to solve the sample problems ('Sample time'). We explore the usage of 10 and 20

samples (M) and sample sizes ($|N|$) ranging from 5 to 20 scenarios. For each configuration we analyze the average cost of the solutions ('Total cost'), the total CPU time ('Total time') and the time to find the best solution ('Time to best') in seconds, the relative optimality gap of the SAA solutions ('Opt gap') (computed with the formula shown in Section 3.1) as well as the average relative optimality gap of the sample problems ('Sample gap'). Note that the latter values refer to the average of the gaps reported at termination by the solver for the sample problems. The results for SAA-BC and SAA-BD are shown in Tables B.11 and B.12, respectively. It is worth mentioning that for both SAA approaches the i-th sample problem is always the same (for the same sample size) for comparability purposes. The results for SAA-BC show that in general better solutions can be found (and more quickly) when the sample size ($|N|$) decreases. This is a result of the capacity of the BC method to find high-quality solutions for small-sized problems. As expected, it is also possible to observe that in general using more samples (M) increases the chances of finding better solutions. For SAA-BD, Table B.12 shows a rather stable behavior in terms of solution quality, execution time and optimality gaps. The configuration chosen to be used in the main body of the manuscript is marked with '*'.

Table B.11: Performance of the SAA-BC method for different configurations

Sample time	M	$\|N\|$	No. of instances	Total cost	Total time	Time to best	Opt gap (%)	Sample gap (%)
30 min	10	5	138	255.83	7,479	3,736	16.00	10.62
		10	138	563.94	9,342	4,917	19.09	16.53
		15	138	568.77	10,313	4,984	19.73	18.25
		20	138	714.87	10,871	4,695	21.60	20.71
30 min	20	5	138	*253.01	15,263	7,145	16.00	10.83
		10	138	564.68	18,845	8,354	19.14	16.43
		15	138	566.29	20,903	9,654	19.51	18.32
		20	138	712.35	21,798	9,620	21.14	20.57
60 min	10	5	138	253.47	14,269	7,466	15.60	9.97
		10	138	266.40	18,164	10,740	16.76	14.03
		15	138	563.93	20,050	10,476	18.92	17.34
		20	138	579.57	21,213	8,215	19.75	18.63

Appendix C. Model to Improve Driver Consistency

We can try to improve the driver consistency of a given input solution $\bar{x}$ by reassigning the vehicles that carry out each route while maintaining the same routing decisions. Considering the same notation introduced in Section 2.1, the mathematical formulation is as follows:

$$\min \sum_{i \in \mathcal{C}} o_i \lambda_i \tag{C.1}$$

$$\text{s.t.} \quad \sum_{k \in \mathcal{K}} x_{ijk\omega} = \sum_{k \in \mathcal{K}} \bar{x}_{ijk\omega} \qquad \omega \in \Omega,\ (i,j) \in \mathcal{E}_\omega, \tag{C.2}$$

Table B.12: Performance of the SAA-BD method for different configurations

Sample time	M	$\|N\|$	No. of instances	Total cost	Total time	Time to best	Opt gap (%)	Sample gap (%)
30 min	10	5	138	344.69	17,384	5,831	42.41	40.64
		10	138	342.37	17,539	5,302	42.69	41.85
		15	138	341.21	17,612	4,902	43.18	42.63
		20	138	342.48	17,831	5,032	43.84	43.42
30 min	20	5	138	341.99	34,479	10,431	43.61	42.00
		10	138	340.10	34,866	9,915	44.58	43.80
		15	138	*339.84	35,221	8,052	45.53	45.12
		20	138	342.31	35,670	8,039	46.70	46.48
60 min	10	5	138	343.53	34,208	11,974	41.79	40.11
		10	138	341.74	34,571	11,285	42.22	41.42
		15	138	340.76	34,805	9,999	42.78	42.25
		20	138	341.31	35,349	10,092	43.46	43.00

$$\sum_{k\in\mathcal{K}} y_{ik} \le D + \lambda_i \qquad i \in \mathcal{C}, \tag{C.3}$$

$$\sum_{k\in\mathcal{K}} z_{ik\omega} \le 1 \qquad \omega \in \Omega,\ i \in \mathcal{C}_\omega, \tag{C.4}$$

$$\sum_{(j,i)\in\mathcal{E}_\omega} x_{jik\omega} + \sum_{(i,j)\in\mathcal{E}_\omega} x_{ijk\omega} = 2z_{ik\omega} \qquad \omega \in \Omega,\ i \in \mathcal{C}_\omega,\ k \in \mathcal{K}, \tag{C.5}$$

$$\sum_{j\in\mathcal{C}_\omega} x_{0jk\omega} \le 2 \qquad \omega \in \Omega,\ k \in \mathcal{K}, \tag{C.6}$$

$$z_{ik\omega} \le y_{ik} \qquad \omega \in \Omega,\ i \in \mathcal{C}_\omega,\ k \in \mathcal{K}, \tag{C.7}$$

$$\lambda_i \ge 0 \qquad i \in \mathcal{C}, \tag{C.8}$$

$$y_{ik} \in \{0,1\} \qquad i \in \mathcal{C},\ k \in \mathcal{K}, \tag{C.9}$$

$$z_{ik\omega} \in \{0,1\} \qquad \omega \in \Omega,\ i \in \mathcal{C}_\omega,\ k \in \mathcal{K}, \tag{C.10}$$

$$x_{ijk\omega} \in \{0,1,2\} \qquad \omega \in \Omega,\ (i,j) \in \mathcal{E}_\omega\colon i = 0,\ k \in \mathcal{K}, \tag{C.11}$$

$$x_{ijk\omega} \in \{0,1\} \qquad \omega \in \Omega,\ (i,j) \in \mathcal{E}_\omega\colon i > 0,\ k \in \mathcal{K}. \tag{C.12}$$

The objective function (C.1) minimizes the cost of the inconsistencies resulting from the (potentially) new assignments. Constraints (C.2) enforce that every edge traversed in the input solution is also traversed in the output solution, but a reassignment of the drivers is allowed. The remaining constraints serve the same purpose as in the original formulation. Note that this model can be solved using a general-purpose solver, whose performance can be enhanced using the inequalities presented in Section 3.2.1.