Incorporating Service Reliability in Multi-depot Vehicle Scheduling: A Chance-Constrained Approach

Margarita P. Castro
Department of Industrial and Systems Engineering, Pontificia Universidad Católica de Chile, Santiago 7820436, Chile
margarita.castro@ing.puc.cl

Merve Bodur
School of Mathematics, University of Edinburgh, Edinburgh EH9 3FD, UK,
merve.bodur@ed.ac.uk

Amer Shalaby
Department of Civil and Mineral Engineering, University of Toronto, Toronto Ontario M5S 1A4, Canada,
amer.shalaby@utoronto.ca

The multi-depot vehicle scheduling problem (MDVSP) is a critical planning challenge for transit agencies. We introduce a novel approach to MDVSP by incorporating service reliability through chance-constrained programming (CCP), targeting the pivotal issue of travel time uncertainty and its impact on transit service quality. Our model guarantees service reliability measured by on-time performance (OTP), a primary metric for transit agencies, and fairness across different service areas. We propose an exact branch-and-cut (B&C) scheme to solve our CCP model. We present several cut-generation procedures that exploit the underlying problem structure and analyze the relationship between the obtained cut families. Additionally, we design a Lagrangian-based heuristic to handle large-scale instances reflective of real-world transit operations. Our approach partitions the set of trips, each subset leading to a subproblem that can be efficiently solved with our B&C algorithm, and then employs a procedure to combine the subproblem solutions to create a vehicle schedule that satisfies all the planning constraints of the MDVSP. Our empirical evaluation demonstrates the superiority of our stochastic variant in achieving cost-effective schedules with reliable OTP guarantees compared to alternatives commonly used by practitioners, as well as the computational benefits of our methodologies.

1. Introduction

Transit agencies face several decision-making problems to provide a high-quality service to their customers at a low cost, such as timetabling, crew scheduling, and vehicle scheduling (Ibarra-Rojas et al. 2015). In particular, vehicle scheduling problems constitute one of the most important problem classes to public transportation companies since they directly impact the service quality (e.g., waiting time of passengers) and the operational costs (i.e., fleet size and vehicle usage). These problems involve assigning timetabled trips to buses stationed at different depots to minimize the total operational costs. One of the most studied problems in the vehicle scheduling literature is the multi-depot vehicle scheduling problem (MDVSP). Its classic variant considers a fleet of homogeneous buses and multiple depots. Each bus is stationed at a particular depot (i.e., its

starting location) and has to return to that depot at the end of the day. Each bus is assigned to a sequence of timetabled trips, where each trip has a start and end location, a scheduled start time, and a duration. Since trips can start and end in different locations, buses might travel empty from one location to the next, referred to as deadhead trips. The goal of the MDVSP is to find a vehicle schedule (i.e., trip assignments to buses) that minimizes the total cost given by the deployed fleet size (i.e., number of buses employed) and the operational cost associated with deadhead trips (e.g., fuel or electricity). In practice, a vehicle schedule corresponds to a single day of operation, which will be employed for several weeks (usually 2 to 3 months) given the seasonal demand fluctuations.

The classic MDVSP has been vastly studied in the literature, given its relevance to practitioners (Ibarra-Rojas et al. 2015) and its computational complexity (Bertossi, Carraresi, and Gallo 1987). There are several problem variants in the literature, including extensions that consider fleets with electric vehicles (Fusco et al. 2013, Chao and Xiaohong 2013, Tang, Lin, and He 2019), and integrating the MDVSP with other transit problems such as crew scheduling (Mesquita et al. 2013) and timetabling (Liu and Shen 2007). However, only few works aim to address one of the main practical challenges of the MDVSP: the uncertainty in the travel times. Among these, the majority either considers dynamically changing the vehicle schedule in the operational stage (Guedes and Borenstein 2018, He, Yang, and Li 2018), which might be highly impractical, or studies simple stochastic variants that penalize expected delays thus obtaining deterministic models (Naumann, Suhl, and Kramkowski 2011, Shen, Xu, and Li 2016, Ricard et al. 2024), also ignoring the proper modeling of delay propagation among trips and their impact on service quality requirements except (Ricard et al. 2024). We further discuss these related works in detail in Section 2.

On the other hand, practitioners usually rely on deterministic approaches where they overestimate travel times to hedge against possible delays, which results in vehicle schedules with higher costs and no guarantee on service quality or reliability. Agencies most commonly use the so-called on-time performance (OTP) measure for reliability. According to this metric, a bus is considered to be on time when it departs (or alternatively arrives) a predetermined bus stop within a certain range of its scheduled departure (or arrival) (European Committee for Standardisation 2002), and OTP corresponds to the percentage of departing on time. The on-time range choice varies across the transit industry, the most commonly used one having been around 1-minute earlier and 5-minute later than the scheduled time (Guenthner and Hamat 1988). The 2018 data is summarized in (TransitCenter 2018) provides the ranges used by top 20 transit agencies in the United States and their weekday OTP ranging from 44% to 75%. In that regard, Figure 1 presents the recent information for Toronto, Ontario, Canada and London, United Kingdom. The Toronto Transit Commission (2024) uses 1-minute early and 5-minute late for the on-time range, and releases a daily report on their website for OTP targets for their services and what is achieved (Figure 1a).

Report for May 15, 2024

Mode:	Our objective:	Our target:	Actual:	How we did:
1 Yonge-University	Deliver a punctual subway service[1]	90%	85%	✗
2 Bloor-Danforth	Deliver a punctual subway service[1]	90%	93%	✓
4 Sheppard	Deliver a punctual subway service[1]	98%	99%	✓
Bus	On time departures from end terminals[3]	90%	80%	✗
Streetcar	On time departures from end terminals[3]	90%	78%	✗
Wheel-Trans	On time service	90%	94%	✓
Elevator	Provide easy access to our customers[2]	98%	99%	✓
Escalator	Provide easy access to our customers[2]	97%	97%	✓

1. % of Service (up to Headway + 3 minutes)
2. % of devices available
3. % of service (end terminal departures between +1 minute early and -5 minutes late) from two business days ago.

(a) TTC Daily Customer Service Report on May 15, 2024.

Quality of Service Indicators for Low Frequency (Timetabled) **Day and Night Routes**

Quarter 04 23/24
06 January 2024 to 31 March 2024

Probability of Departure (%)

Route	% On Time	Q4 22/23 (% On Time)	Non Arrival or Not Linked (%)	8 to 2.5 mins Early (%)	5 to 15 mins Late (%)
20	79.6	80.1	4.1	2.5	13.8
42	68.4	71.5	8.6	2.8	20.2
61	83.2	85.7	3.4	0.8	12.7
107	84.5	76.6	5.2	1.8	8.6
110	79.0	68.1	2.1	2.6	16.3
110R	57.4	51.3	11.2	2.2	29.2
117	81.0	79.2	3.4	1.8	13.8
138	83.5	86.7	3.1	1.2	12.2
146	75.0	78.6	9.8	0.6	14.6
160	71.0	78.0	8.4	1.1	19.5
162	72.1	75.6	8.4	1.8	17.7
166	72.6	79.5	7.6	1.9	18.0
167	85.6	85.8	2.8	0.8	10.8
178	83.4	80.7	3.8	2.1	10.7
187	79.2	77.9	5.0	4.0	11.7
190	77.7	85.7	4.6	3.1	14.6
201	67.5	77.1	8.9	3.1	20.5
N550	82.5	90.4	4.8	2.9	9.8
N551	83.1	86.8	4.7	2.0	10.1
NEL1	94.2	95.1	2.0	3.1	0.7
All	80.8	81.9	4.5	2.1	12.6

1) **Chance of a bus departing on-time** The chance that a bus runs at the advertised time or between two minutes early and up to five minutes late.

2) ***Q4 22/23 (% On Time)*** Denotes the percentage of departing on time (see 1) for the corresponding financial quarter last year.

(b) TfL Route Results for London Bus Services in Quarter 04 23/24.

Figure 1 On-time performance statistics from Toronto and London bus systems.

Transport for London (2024) uses 2-minute early and 5-minute late for the OTP range, and releases a quarterly report on their bus services (Figure 1b).

Motivated by the practical importance of MDVSP and the limitations of the literature, we propose a novel stochastic MDVSP variant that considers stochastic travel times to achieve the desired service requirements defined based on OTP. The main goal is to build a cost-optimal schedule that satisfies the service requirements given by OTP. In that regard, we not only ensure that most trips have to start on time but also incorporate fairness considerations into schedule (e.g., services on different areas must have similar OTP). Moreover, we consider a risk-averse setting to fulfill these service requirements given the high operational reliability and user satisfaction that transit agencies aim for in practice. Thus, we model the problem as a chance-constrained programming (CCP) model, which enforces the service requirements up to a certain probability threshold. From now own, we call this problem the chance-constrained MDVSP (CC-MDVSP).

We propose exact and heuristic methodologies to address the scenario reformulation of CC-MDVSP. Our exact procedure is a branch-and-cut (B&C) scheme that iteratively offers a cost-efficient vehicle schedule and evaluates the service conditions in a set of scenarios given by different travel time realizations. We present several cut generation algorithms based on different problem characteristics and analyze their theoretical properties. In particular, we show that our subproblems

can be solved via a greedy algorithm in polynomial time, and more importantly their solutions can be used in an iterative procedure to efficiently identify minimal infeasible subsystems (MISs) and accordingly cuts, which we call constraint-based MIS cuts, stronger than the traditionally used alternatives. Moreover, we design a heuristic procedure based on our exact methodology to handle realistic large-scale problems. More specifically, we consider a Lagrangian scheme that decomposes the problem into sub-problems with a smaller set of trips, which can be efficiently solved with our exact B&C methodology. We then solve the Lagrangian dual problem that enforces the service requirements across all trips to find a cost-efficient solution for the complete problem.

We perform extensive computational experiments which show that our proposed methodologies return cost-efficient vehicle schedules that fulfill the service requirements. Comparisons with those commonly used by practitioners (i.e., deterministic models with over-estimations of the travel times) illustrate that our techniques find lower-cost schedules with reliable service guarantees.

The remainder of the paper is organized as follows. Section 2 provides a literature review on the MDVSP, its stochastic variants, and related CCP problems. Section 3 describes our novel stochastic MDVSP variant with service requirements and presents our CCP model. Section 4 introduces our scenario reformulation and or the B&C decomposition approach. Section 5 presents several cut generation alternatives, methodological enhancements, and theoretical results. Section 6 describes our Lagrangian-based scheme to handle large-scale problems. Lastly, Section 7 presents the numerical experiment results and we end with some concluding remarks in Section 8.

2. Literature Review

We review the two main bodies of literature that are closely related to our work. Section 2.1 describes relevant works in the MDVSP literature, emphasizing existing exact algorithms for the deterministic and stochastic variants of this problem. Section 2.2 reviews relevant works on CCP, especially ones that handle chance constraints with integer variables.

2.1. Multi Depot Vehicle Scheduling and Extensions

The deterministic MDVSP is well-studied due to its practical application to public transit agencies (Ibarra-Rojas et al. 2015) and its combinatorial structure which makes the problem NP-hard for the case with two or more depots (Bertossi, Carraresi, and Gallo 1987). Several models and optimization techniques have been proposed to tackle this challenging problem (see, e.g., Bunte and Kliewer (2009), Pepin et al. (2009)). Carpaneto et al. (1989) propose the first exact algorithm for MDVSP based on the branch-and-bound procedure and a single-commodity model with sub-tour breaking constraints, which was then improved by Fischetti et al. (2001) via a branch-and-cut approach.

One of the most used models to solve the MDVSP is the time-space formulation (Kliewer, Mellouli, and Suhl 2006) that discretizes time to consider each trip's start and end time points.

This formulation tends to be quite large in the number of variables, but can be solved using a branch-and-price (B&P) algorithm. Kulkarni et al. (2018) present an inventory formulation that also relies on a time-space network. A column generation approach is proposed to solve the linear programming relaxation along with a heuristic procedure to find high-quality integer solutions.

Alternatively, Desrosiers et al. (1995) propose a multi-commodity flow (MCF) model with a cubic number of variables concerning the number of trips and depots. Hadjar, Marcotte, and Soumis (2006) studied the MCF polyhedral structure and proposed several enhancements, including a reduced cost-fixing strategy and a branch-and-cut approach to remove fractional solutions via odd-cycle cuts. We note that the MCF model is more amenable to our variant with stochastic travel times than the time-space formulations because the former implicitly considers time when computing compatible trips, while the latter explicitly uses time points to create the model.

One of the main practical limitations of the deterministic MDVSP is that it ignores the stochastic nature of travel times and, thus, its delayed consequences. Since delays and disruptions in service are a major concern for transit agencies, several works have built deterministic contingency models to fix the schedule during an operational day, such as rescheduling (Li, Mirchandani, and Borenstein 2007, Guedes and Borenstein 2018), real-time recovery (Visentini et al. 2014), damaging disruptions (Uçar, Birbil, and Muter 2017) and trip-shifting strategies (Desfontaines and Desaulniers 2018), among others. However, these techniques require real-time data and a coordinator to make timely decisions, which are very costly for transit agencies, especially if they have to be made most days.

Surprisingly, only few works consider stochastic travel times while creating the vehicle schedule to reduce the need for such contingency options during operation. Naumann, Suhl, and Kramkowski (2011) base their MDVSP formulation on the time-space model of Kliewer, Mellouli, and Suhl (2006). In their approach, delay values for each trip (except depot and deadhead trips) are generated from an exponential distribution to construct a set of delay scenarios. The idea is to add penalties (i.e., additional cost) to the waiting arcs in the network, which connect pairs of trips and have an associated predetermined waiting time. More specifically, given a consecutive trip and waiting arc combination, if the sampled delay of the trip leads to a delay in the subsequent trip, then a quadratic penalty cost is added to the waiting arc cost. Despite having delay scenarios, the resulting formulation becomes a deterministic model where expected delay penalty costs are added to the waiting arcs. Shen, Xu, and Li (2016) and Shen, Xu, and Wu (2017) follow a similar approach, except base their formulation on the MCF model, ending up with a deterministic model where additional costs are added to the trip arcs. Therefore, these approaches fall short in properly modeling the propagation effects of the delays.

Since the delay propagation for consecutive trips can have a massive impact in practice, recent works have considered dynamic approaches that build and repair the schedule during the operational day. He, Yang, and Li (2018) propose an approximated dynamic programming approach to

build schedules daily that minimize cost and reduce delays of consecutive trips. Tang, Lin, and He (2019) introduce a dynamic and static model that incorporates the need to charge a fleet of electric buses. The static model uses a buffer-distance strategy to protect from delays, while the dynamic model periodically reschedules the bus fleet during a day's operations. We note that none of these approaches actually create a robust schedule that performs well in practice without the need to reschedule (i.e., return a reliable schedule).

To the best of our knowledge, the recent work by Ricard et al. (2024) is the only one that aims to create a reliable vehicle schedule using stochastic travel times and consider proper delay propagation. A set partition model is proposed, which enumerates all possible trip sequences, thus is amenable to B&P. Each sequence considers the operation cost plus a delay penalty, calculated using discretized probability mass function to account for delay propagation (Ricard et al. 2022). As in the previously reviewed works, the resulting model is deterministic since the stochastic travel times are only used to compute the expected delays, which is then transformed into a penalty in the objective function. Moreover, their model is sensitive to the weight assigned to the delay penalty cost, and, as such, users need to tune this parameter to obtain schedules with the desired reliability. In contrast, our CC-MDVSP variant includes the service reliability conditions directly in the model via a CCP that optimizes for a risk-averse setting (i.e., ensure for instance 95% of the days with a reliable service, along with fairness among routes). Also, we measure reliability based on the number of trips that start later than their scheduled start time as commonly done in transit agencies (e.g., in The Toronto Transit Commission (2024), Transport for London (2024)), while Ricard et al. (2024) consider metrics focused on the public transit user's perspective.

2.2. Chance Constrained Programming

CCP is a well-studied modeling paradigm in stochastic programming that considers stochastic constraints to be satisfied with a certain probability (Charnes and Cooper 1959). While these constraints can be cast as conic constraints for parameters with Gaussian or similar distribution (Küçükyavuz and Jiang 2021), handling other types of distribution usually requires some sort of sampling and scenario reformulation (Pagnoncelli, Ahmed, and Shapiro 2009).

Ruszczyński (2002) first introduced the scenario reformulation for CCP when considering a discrete distribution. This reformulation replaces the random variables with a set of scenario and binary variables, making the problem non-convex and significantly increasing its size. This procedure can be extended for probability distributions with infinite support using sample average approximation (SAA), i.e., reformulate the problem with a proper number of scenarios and obtain statistically valid lower and upper bounds to the original problem (Luedtke and Ahmed 2008).

While the number of scenarios can be reduced using SAA, the resulting model is still a large-scale mixed-integer linear program (MILP) with a weak linear programming (LP) relaxation given by a

large set of big-M constraints. Luedtke, Ahmed, and Nemhauser (2010) study the structure of the problem when considering only right-hand-side uncertainty and propose an improved relaxation for the case with only continuous variables. There is also significant research on valid inequalities mechanism to strengthen the big-M coefficients (see, e.g., Küçükyavuz (2012), Song, Luedtke, and Küçükyavuz (2014), Ahmed and Xie (2018)).

One of the most common approaches to deal with the scenario-based MILP is B&C (Luedtke 2014). The procedure decomposes the problem into a master problem containing all the deterministic constraints and one subproblem for each scenario. Given a fixed candidate solution, the B&C solves the subproblems for each scenario and returns a cut if the solution violates the CC feasibility set for that scenario. Although general, this procedure is usually tailored for each application depending on the structure of the subproblem and the nature of its variables. There are few applications in the literature that consider CCP problems with integer variables, such as scheduling (Deng and Shen 2016), vehicle routing (Dinh, Fukasawa, and Luedtke 2018) and partial set covering (Wu and Kucukyavuz 2019). These works make problem-specific variations of the B&C decomposition, and all their integer variables are part of the master problem. In contrast, our problem considers integer variables in both the master and subproblem, a case with scarce literature. Thus, we propose specialized procedures to generate cuts given the subproblem non-convexity.

Canessa et al. (2019) propose an algorithm to solve one-stage pure binary linear CCPs using irreducibly infeasible subsystems (IIS). In a B&C framework, they solve the restriction of the problem where all scenario constraints are enforced (except those corresponding to the scenarios that are eliminated by the branching decisions, if any). When such a restriction is infeasible, they generate an IIS using a commercial solver and generate a cut on the scenario variables enforcing that at least one among the identified minimal set should be eliminated. They can generate a similar IIS cut also in the case where the restricted problem is feasible, by adding a constraint bounding the optimal value and turning the restricted problem into one looking for an improving feasible solution. Being only on the scenario variables, their cuts do not capture any relationship between the scenario variables and the original variables of the model. In contrast, to solve a two-stage CCP with integer recourse, our proposed methodology relies on a B&C procedure where we identify IIS (i.e., MIS) based on the recourse (i.e., second-stage) problems and generate cuts linking the original first-stage variables (i.e., vehicle scheduling variables) and the scenario variables. Moreover, leveraging our problem structure, we introduce a polynomial procedure to identify IIS as well as some cut strengthening procedures. Our newly proposed constraint-based MIS (C-MIS) and extended C-MIS cuts can be useful for some other applications.

3. Problem Description and Formulation

We now describe the proposed stochastic variant of the MDVSP that ensures service requirements in a risk-averse setting. In what follows, we use calligraphic font for sets, uppercase letters for the sets' cardinality, and lowercase letters for the sets' elements if possible (e.g., $a \in \mathcal{A}$ is an element of set $\mathcal{A}$ with cardinality $|\mathcal{A}| = A$), and typewriter font for parameters (e.g., $\mathtt{b}$ is a parameter for the problem). Also, we use $\mathbb{P}(\cdot)$ as the probability operator and $\mathbb{E}(\cdot)$ as the expected value operator.

The MDVSP considers a set of I timetabled trips, $\mathcal{I} = \{1, ..., I\}$, and a set of K depots, $\mathcal{K} = \{1, ..., K\}$. Each depot $k \in \mathcal{K}$ has a capacity $\mathtt{b}_k$ that represents the maximum number of buses that can be placed at depot k. As in the standard deterministic variant (e.g., in Carpaneto et al. (1989)), we consider a homogeneous bus fleet (i.e., all buses are the same), so a timetabled trip $i \in \mathcal{I}$ can be assigned to any bus which can be associated with any depot $k \in \mathcal{K}$. Buses must start and end their tour (i.e., sequence of their assigned timetabled trips) at their associated depot.

Let ξ represent the underlying stochastic process associated with this problem; that is, ξ is the set of random variables representing bus travel times. Each timetabled trip $i \in \mathcal{I}$ has a scheduled start time $\mathtt{s}_i$, a start location $\mathtt{l}_i^{\mathtt{s}}$, an end location $\mathtt{l}_i^{\mathtt{e}}$, and a stochastic duration $\mathtt{d}_i(\xi)$ that represents the time to go from its start location $\mathtt{l}_i^{\mathtt{s}}$ to its end location $\mathtt{l}_i^{\mathtt{e}}$.

In addition to the set of timetabled trips $\mathcal{I}$, we define deadhead trips as bus movements to reach the start location of a trip or return to its associated depot. Specifically, a bus associated with depot $k \in \mathcal{K}$ performs a deadhead trip if it travels from: (i) k to the start location of trip $i \in \mathcal{I}$, (ii) the end location of i to the start location of $j \in \mathcal{I}$ $(i \neq j)$, or (iii) the end of location i to depot k. Then, the random variables $\mathtt{t}_{ij}(\xi)$, $\mathtt{t}_{ki}(\xi)$, and $\mathtt{t}_{ik}(\xi)$ represent the stochastic travel time of deadhead trips between timetabled trips $i, j \in \mathcal{I}$ and from/to the depot $k \in \mathcal{K}$, accordingly.

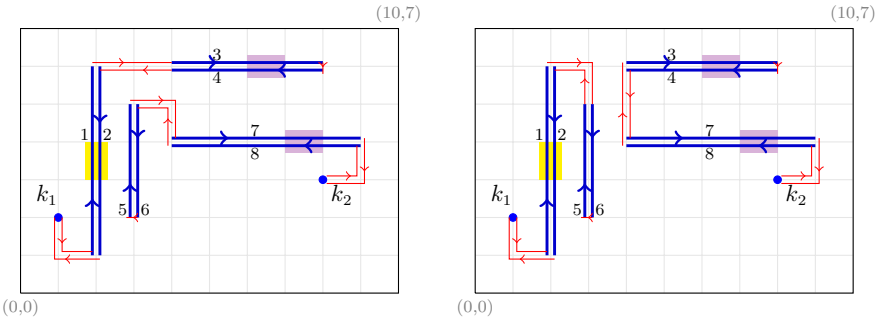
Our CC-MDVSP variant seeks a feasible assignment of timetabled trips to buses such that each trip is assigned to a single bus and the scheduled starting time is met as closely as possible. Thus, each bus assignment corresponds to a sequence of timetabled trips such that there is one deadhead trip between each timetabled trip and one deadhead trip to leave and return to the depot. Then, a solution is a vehicle schedule representing the sequence of timetabled trips for each deployed bus. Since we consider a homogeneous bus fleet, it is sufficient to associate trip sequences with depots.

The objective of the problem is to minimize operational costs. Given that all timetabled trips are assigned to buses, operational costs are associated with deadhead trips. In that regard, $\mathtt{c}_{ij} \geq 0$ for trips $i, j \in \mathcal{I}$ corresponds to the operational cost of scheduling trip i right before trip j on the same bus, which represents, for instance, the fuel consumption of the deadhead trip. Analogously, $\mathtt{c}_{ki} \geq 0$ and $\mathtt{c}_{ik} \geq 0$ represent the cost of scheduling $i \in \mathcal{I}$ as the first and last trip of a bus associated with depot $k \in \mathcal{K}$, respectively. We note that these depot-related deadhead trips typically include the cost of deploying a bus (e.g., the driver's salary) and the operational cost associated with the

travel distance (e.g., fuel cost). Example 1 presents a small instance of the deterministic MDVSP that illustrates the previously described concepts.

EXAMPLE 1. Consider a deterministic MDVSP instance depicted in Figure 2. Each 10×7 grid corresponds to the same MDVSP instance with two depots, $k_1, k_2 \in \mathcal{K}$, positioned at coordinates (1,2) and (8,3), respectively. Each thick blue line corresponds to a timetabled trip, where the arrow points towards the end location and the number right next to it represents its index $i \in \mathcal{I} = \{1, \ldots, 8\}$. For example, trip 1 starts at location $\mathtt{l}_1^{\mathtt{s}} = (2,1)$ and ends in $\mathtt{l}_1^{\mathtt{e}} = (2,6)$. Red thin lines correspond to deadhead trips for a specific vehicle schedule. For instance, the line connecting k_1 and $\mathtt{l}_1^{\mathtt{s}}$ in both grids corresponds to the deadhead trips from the first depot to the start of trip 1.

The timetable in Figure 2 shows the start time and average duration of each trip measured in units of time. In the deterministic setting, we consider each cell distance to represent one unit of time and one unit of operational cost. Moreover, deadhead trips associated with depots have an additional fixed cost of 2, representing bus deployment. For example, the deadhead trip from k_1 to trip 1 has an average duration $\bar{t}_{k_1,1} = 2$ and cost $c_{k_1,1} = 4$. Both grids represent feasible vehicle schedules for the deterministic setting with average times; that is, each trip $i \in \mathcal{I}$ can start at the required $\mathtt{s}_i$ time. Each solution employs two buses (one per depot), and both solutions have the same total cost of 20 (16 units of operational cost and 8 units for deploying the 2 buses). □



i	$\mathtt{s}_i$	$\bar{\mathtt{d}}_i$	i	$\mathtt{s}_i$	$\bar{\mathtt{d}}_i$
1	5	5	5	15.5	3
2	23	5	6	12	3
3	12	4	7	23	5
4	16	4	8	2	5

Figure 2 Example for the deterministic and CC-MDVSP. Both grids represent the same problem instance with different vehicle schedules each. The right-hand side shows the average timetable information.

Our stochastic CC-MDVSP considers all the aforementioned characteristics of the MDVSP, which we call the *planning* portion of the problem. In addition, we consider a set of *service (reliability) requirements* that need to be fulfilled in a risk-averse setting (i.e., achieve the requirements most of the days), which correspond to the *operational* portion of the problem. Motivated by practical considerations (e.g., from The Toronto Transit Commission (2024)), we consider an on-time performance (OTP) metric for the start time of each timetabled trip as well as fairness metrics for related timetabled trips:

- *OTP.* We say that trip $i \in \mathcal{I}$ *starts on time* if the execution start time is inside the time window $[\mathtt{s}_i - \mathtt{lb}, \mathtt{s}_i + \mathtt{ub}]$, with given $\mathtt{lb}, \mathtt{ub} \geq 0$. Then, we say that a vehicle schedule *fulfills the OTP requirements* if at least $\mathtt{f}^{\mathtt{trip}} = \lfloor I \cdot \delta^{\mathtt{trip}} \rfloor$ trips start on time, where $\delta^{\mathtt{trip}} \in (0, 1]$ represents the minimum proportion of the timetabled trips that have to start on time. We note that due to practical considerations, any trip i is not allowed to start earlier than $\mathtt{s}_i - \mathtt{lb}$, even if the assigned bus arrives earlier; however, it can start later than $\mathtt{s}_i + \mathtt{ub}$, which would be then identified as *delayed*. On the other hand, we assume that the very first trip of each bus (i.e., the one scheduled right after leaving a depot) starts as early as possible, namely at $\mathtt{s}_i - \mathtt{lb}$ for such a trip $i \in \mathcal{I}$.
- *Fairness.* These conditions ensure that timetabled trips associated with different routes (i.e., a set of trips with common start and end locations) have similar OTP. Specifically, we consider a set of R routes $\mathcal{R} = \{1, ..., R\}$, where a route $r \in \mathcal{R}$ corresponds to a subset of trips $\mathcal{I}(r) \subseteq \mathcal{I}$ with a common start and end location. We note that every trip is part of a single route, thus, sets $\mathcal{I}(r)$ for $r \in \mathcal{R}$ constitute a partition of $\mathcal{I}$. Then, the fairness requirements of a route $r \in \mathcal{R}$ impose that a proportion of the trips in $\mathcal{I}(r)$ start on time, that is, at least $\mathtt{f}_r^{\mathtt{route}} = \lfloor I^r \cdot \delta^{\mathtt{route}} \rfloor$ trips start on time for $I^r = |\mathcal{I}(r)|$ and a given $\delta^{\mathtt{route}} \in (0, 1]$. If this condition is satisfied for each route, then we say that a vehicle schedule *fulfills the fairness requirements*.

We model these service requirements using a joint chance constraint to represent the risk-averse behavior of transit agencies, that is, fulfill the service requirements for most business days. In particular, we impose that all service requirements must be achieved with a probability of at least $1 - \epsilon$, where $\epsilon \in (0, 1)$ is the risk tolerance or, more specifically, the probability threshold of not fulfilling at least one service condition (i.e., violating one of the OTP and fairness requirements).

Similarly to Ricard et al. (2024), we consider a simple model that incorporates these service requirements, which only considers the direct and indirect trips' delays (i.e., delay propagation). In addition, our CC-MDVSP variant considers a recourse action in which bus drivers can reduce their travel times to avoid violating the service requirements. For example, in many public transit systems, drivers might slightly increase the speed to reduce travel time, a practice commonly known as *expressing* (Eberlein 1997). To model this action, we consider $\mathtt{e}_i$ as the maximum units of time reduction that can be made for a timetabled trip $i \in \mathcal{I}$ to help start subsequent trips on time and, in turn, achieve the OTP.

Example 2. We now extend Example 1 to illustrate the aforementioned metrics for the CC-MDVSP. We consider that trips with inverse directions correspond to the same route (e.g., trips 1 and 2 are on the same route); thus, this instance has 4 routes with 2 timetable trips each. For simplicity, the stochastic setting has two scenarios with equal probability and $\epsilon = 0.5$, that is, at least one scenario has to fulfill the service requirements to satisfy the CC. We set parameters

$\mathtt{lb} = \mathtt{ub} = 0$, $\mathtt{e}_i = 0$ for all $i \in \mathcal{I}$, $\delta^{\mathtt{trip}} = 0.85$ and $\delta^{\mathtt{route}} = 0.5$. Thus, a vehicle schedule in a specific scenario fulfills the service requirements if at least $\mathtt{f}^{\mathtt{trip}} = 7$ trips start on time and each route has at most $\mathtt{f}^{\mathtt{route}} = 1$ delayed trip.

Figure 2 illustrates the traveling times of each scenario using different color schemes: scenario 1 in yellow and scenario 2 in purple. Both scenarios consider average traveling times for trips traversing plane grid edges and an increase of 0.5 units of time if they cross a colored edge associated with each scenario. For example, in scenario 1, trips 1 and 2 have both an increase of 0.5 units of times (i.e., a duration of 5.5 each), but in scenario 2 their duration remains the same.

The left schedule of Figure 2 violates CC. In particular, trips 3 and 4 are delayed in the first scenario, thus, violating the OTP and fairness constraint. Also, trips 2 and 4 are delayed in the second scenario, which violates the OTP constraint. In contrast, the right schedule is a feasible schedule for the CC-MDVSP since the schedule satisfies the service requirement for both scenarios. Specifically, only one trip is delayed in each scenario (i.e., trip 6 in scenario 1 and trip 4 in scenario 2), which satisfies both the OTP and fairness constraints. □

3.1. Multi-commodity Flow Formulation

We now present a model of the CC-MDVSP. In what follows, we use lowercase bold letters for vectors of decision variables and sub-indices to refer to each variable (e.g., $\boldsymbol{x} = (x_1, x_2, x_3)^\top \in \mathbb{R}^3$ is a vector of continuous decision variables).

As discussed in Section 2, several deterministic MDVSP models can be considered for an extension to our chance-constrained setting. Most of these formulations rely on a time-space network flow model (e.g., (Kliewer, Mellouli, and Suhl 2006) and (Kulkarni et al. 2018)) that employ trips' durations to define nodes in the network. Since our problem considers stochastic travel times, this would require building a different time-space network for each possible realization of ξ. Alternatively, the MCF formulation (Desrosiers et al. 1995) relies on a network that relates timetabled trips that can be sequentially scheduled to the same bus and does not explicitly consider the travel times. Therefore, we propose an MCF formulation for the CC-MDVSP. This formulation represents the problem with a network $\mathcal{G} = (\mathcal{N}, \mathcal{A})$, where $\mathcal{N}$ is the set of nodes and $\mathcal{A}$ is the set of arcs. We consider one node for each timetable trip $i \in \mathcal{I}$ and two nodes for each depot $k \in \mathcal{K}$, that is, $\mathcal{N} = \mathcal{I} \cup \mathcal{K}^{\mathtt{s}} \cup \mathcal{K}^{\mathtt{e}}$, where $\mathcal{K}^{\mathtt{s}}$ and $\mathcal{K}^{\mathtt{e}}$ are copies of $\mathcal{K}$ that represent the start and end depots of a vehicle schedule, respectively.

The set of arcs is given by the compatibility set of timetabled trips $\mathcal{C}$, that is, the set of trip pairs that can be scheduled in the same bus. In the deterministic MDVSP, we say that two trips $i, j \in \mathcal{I}$ are compatible if the scheduled end time of trip i plus the travel time of the deadhead trip from i to j is less than or equal to the scheduled start time of trip j. Thus, the compatibility set

is $\mathcal{C}^{\mathtt{det}} = \{(i,j) \in \mathcal{I} \times \mathcal{I} : \mathtt{s}_i + \mathtt{d}_i + \mathtt{t}_{ij} \leq \mathtt{s}_j\}$. However, travel times in the CC-MDVSP are random variables, so we consider a general representation of $\mathcal{C}$ using $\tilde{\mathtt{d}}_i$ and $\tilde{\mathtt{t}}_{ij}$ as estimates of the travel times for $i,j \in \mathcal{I}$. For example, a conservative approach sets $\tilde{\mathtt{d}}_i = 0$ or $\tilde{\mathtt{d}}_i = \min\{\mathtt{d}_i(\xi)\}$ (where the minimum is taken over the support of ξ), while an average approach considers $\tilde{\mathtt{d}}_i = \mathbb{E}[\mathtt{d}_i(\xi)]$, as in the deterministic MDVSP literature; similarly treating the $\tilde{\mathtt{t}}$ parameters. Thus, the *planning compatibility set* for the CC-MDVSP is:

$$\mathcal{C} = \{(i,j) \in \mathcal{I} \times \mathcal{I} : \mathtt{s}_i + \tilde{\mathtt{d}}_i + \tilde{\mathtt{t}}_{ij} \leq \mathtt{s}_j\}.$$

whose member pairs we refer to as *planning compatible*. Then, the set of arcs in the network is given by $\mathcal{A} = \mathcal{C} \cup \{(k,i) : i \in \mathcal{I}, k \in \mathcal{K}^{\mathtt{s}}\} \cup \{(i,k) : i \in \mathcal{I}, k \in \mathcal{K}^{\mathtt{e}}\}$, that is, there is a directed arc: (i) for each pair of compatible trips, (ii) from each start depot to every trip, and (iii) from each trip to every end depot. Note that set $\mathcal{A}$ has cardinality $A = |\mathcal{C}| + 2KI$, and it considers that all trips can be directly linked to a depot.

EXAMPLE 3. Figure 3 shows the network $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ for Example 2 using the planning compatibility set $\mathcal{C}$ with average times. Gray nodes represent the depot copies (i.e., with $k_1^s, k_2^s \in \mathcal{K}^{\mathtt{s}}$ and $k_1^e, k_2^e \in \mathcal{K}^{\mathtt{e}}$) and black nodes correspond to timetable trips. Solid arrows link node trips that are in the planning compatibility set. Dashed arrows represent pull-out and pull-in arcs (Kliewer, Mellouli, and Suhl 2006), that is, deadhead trips from and to the depots, respectively. We represent a subset of all pull-in and pull-out arcs in the network to avoid overcrowding the graph. The network also illustrates the schedule in the left grid of Figure 2 with shaded arcs. □

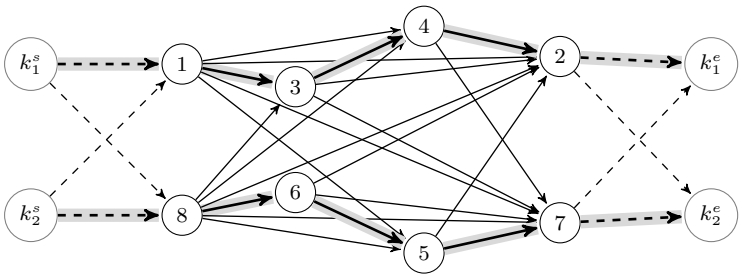


Figure 3 Network representation for the CC-MDVSP instance described in Examples 1 and 2.

We note that the planning compatibility set $\mathcal{C}$ only defines the set of allowed consecutive trip pairs based on the travel time estimates, whereas the proposed chance-constrained model decides which trip sequences are indeed feasible based on the travel time realizations (i.e., the operational portion of the model). As previously mentioned, this set $\mathcal{C}$ can always be built in a conservative manner so that no feasible schedule is cut off. However, building $\mathcal{C}$ via a less conservative but "safe" approach (i.e., not removing any feasible schedule) would reduce the number of arcs in the network

and, as such, help solve the optimization model more efficiently. Also, this definition of $\mathcal{C}$ allows us to easily incorporate any other compatibility considerations/preferences.

The MCF model creates timetabled trip sequences (i.e., one sequence per bus) and assigns them to depots so that all trips are covered exactly once. The main decision vector is $\boldsymbol{x} \in \{0,1\}^{A\times K}$, where $x_{ijk} \in \{0,1\}$ represents visiting node $i \in \mathcal{N}$ right before node $j \in \mathcal{N}$, with a bus associated to depot $k \in \mathcal{K}$. Then, our MCF model for the CC-MDVSP is given by:

$$\min \sum_{k\in\mathcal{K}} \sum_{(i,j)\in\mathcal{A}} \mathsf{c}_{ijk} x_{ijk} \tag{CC-MCF}$$
$$\text{s.t. } \boldsymbol{x} \in \mathcal{X}$$
$$\mathbb{P}\{\boldsymbol{x} \in \mathcal{F}(\xi)\} \geq 1-\epsilon. \tag{1}$$

The objective function of (CC-MCF) represents the travel costs, where, for any given $k \in \mathcal{K}$, $\mathsf{c}_{ijk} = \mathsf{c}_{ij}$ for $(i,j) \in \mathcal{C}$, $\mathsf{c}_{kik} = \mathsf{c}_{ki}$, and $\mathsf{c}_{ikk} = \mathsf{c}_{ik}$ for $i \in \mathcal{I}$. Set $\mathcal{X}$ corresponds to the planning constraints representing the candidate vehicle schedules. This set is described by the deterministic MCF constraints over the network $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ as

$$\mathcal{X} = \Big\{ \boldsymbol{x} \in \{0,1\}^{A\times K} :$$
$$\sum_{k\in\mathcal{K}} \sum_{i:(i,j)\in\mathcal{A}} x_{ijk} = 1, \qquad \forall j \in \mathcal{I}, \tag{2a}$$
$$\sum_{i\in\mathcal{I}} x_{kik} \leq \mathsf{b}_k, \qquad \forall k \in \mathcal{K}, \tag{2b}$$
$$\sum_{j:(j,i)\in\mathcal{A}} x_{jik} - \sum_{j:(i,j)\in\mathcal{A}} x_{ijk} = 0, \qquad \forall i \in \mathcal{I},\ k \in \mathcal{K}, \tag{2c}$$
$$x_{k'ik} = x_{ik'k} = 0 \qquad \forall k, k' \in \mathcal{K},\ k \neq k',\ i \in \mathcal{I} \Big\}. \tag{2d}$$

Constraints (2a) ensure that each trip is scheduled exactly once. Constraints (2b) represent the capacity of the depots, that is, the number of buses (i.e., timetabled trip sequences) that can be assigned to a given depot. Constraints (2c) correspond to flow balance equations and constraints (2d) ensure that the start and end depot nodes coincide with the associated depot.

Lastly, constraint (1) corresponds to the operation portion of the problem, that is, the joint chance constraint that models the service reliability. We consider three additional decision variables to model the service reliability requirements set $\mathcal{F}(\xi)$. For each trip $i \in \mathcal{I}$, variable $v_i \in \{0,1\}$ indicates if trip i starts on time, $y_i \geq 0$ represents the start time of the trip, and $u_i \in [0, \mathsf{e}_i]$ corresponds to the amount of time used by the bus to decrease the duration of the trip. Then, the set of operational constraints is

$$\mathcal{F}(\xi) = \Big\{ \boldsymbol{x} \in \{0,1\}^{A\times K} : \exists\, (\boldsymbol{v}, \boldsymbol{y}, \boldsymbol{u}) \text{ such that}$$

$$\sum_{i\in\mathcal{I}} v_i \geq \mathtt{f}^{\mathtt{trip}}, \tag{3a}$$

$$\sum_{i\in\mathcal{I}(r)} v_i \geq \mathtt{f}_r^{\mathtt{route}}, \qquad \forall r\in\mathcal{R}, \tag{3b}$$

$$y_j + \mathtt{d}_j(\xi) + \mathtt{t}_{ji}(\xi) - u_j - \mathtt{M}_{ji}^{\mathtt{start}}\left(1-\sum_{k\in\mathcal{K}} x_{jik}\right) \leq y_i, \qquad \forall (j,i)\in\mathcal{C}, \tag{3c}$$

$$\mathtt{t}_{ki}(\xi) - \mathtt{M}_{ki}^{\mathtt{start}}\,(1-x_{kik}) \leq y_i, \qquad \forall i\in\mathcal{I}, k\in\mathcal{K}, \tag{3d}$$

$$y_i \leq \mathtt{s}_i + \mathtt{ub}\cdot v_i + \mathtt{M}_i^{\mathtt{OTP}}(1-v_i), \qquad \forall i\in\mathcal{I}, \tag{3e}$$

$$\mathtt{s}_i - \mathtt{lb} \leq y_i, \qquad \forall i\in\mathcal{I}, \tag{3f}$$

$$v_i\in\{0,1\},\ y_i\geq 0,\ u_i\in[0,\mathtt{e}_i] \qquad \forall i\in\mathcal{I}\Big\}. \tag{3g}$$

Constraints (3a) and (3b) enforce the service requirements, that is, the minimum number of trips that start on time across all trips and for each route, respectively. Constraints (3c) and (3d) model the start time of each trip given the set of scheduled decisions. Constraints (3e) relate the start time of each trip with their respective on-time decision variables, and constraints (3f) ensure that start times are not earlier than what is mandated. Note that parameters $\mathtt{M}_{ji}^{\mathtt{start}}, \mathtt{M}_{ki}^{\mathtt{start}}, \mathtt{M}_i^{\mathtt{OTP}} > 0$ for $i,j\in\mathcal{I}$ and $k\in\mathcal{K}$ are sufficiently large values chosen to model the logical implications of constraints (3c)-(3e) properly, respectively. We refer the reader to Section 5.4 for a discussion on tight big-M values for this formulation. Lastly, we assume that all trips scheduled first on a sequence (i.e., right after leaving the depot) always start on time, which makes (3d) redundant, but we leave them in the model for completeness. The validity of this assumption arises in real transit agencies that can adjust their drivers' schedules to guarantee that they always start the first trip on time.

4. Branch-and-Cut Decomposition Scheme

This section introduces the methodology employed to address the (CC-MCF) model. We use the commonly employed SAA approach that transforms the CCP model into a large-scale MILP model (Luedtke and Ahmed 2008) to obtain near-optimal solutions to the CC-MDVSP. Given the size of the model and its well-known weak LP relaxation (Luedtke, Ahmed, and Nemhauser 2010), we propose a B&C approach to handle these issues due to its success in other applications with continuous recourse variables (Luedtke 2014).

Due to the existence of binary recourse variables, there is no readily suitable B&C approach for our problem. Therefore, we devise problem-specific cut-generation procedures to solve the resulting SAA model. Before describing our cut families in Section 5, this section provides the SAA formulation and an overview of the decomposition framework. Moreover, we show that our (scenario) subproblems can be optimally solved via a polynomial-time greedy algorithm, which is key when designing some of our cut-generation variants. We also devise valid inequalities for each scenario realization to strengthen the master problem relaxation.

4.1. SAA Formulation

The SAA formulation for (CC-MCF) considers a set of S scenarios $\mathcal{S}$ obtained by sampling ξ, $\{\xi^s\}_{s\in\mathcal{S}}$, where each scenario $s\in\mathcal{S}$ has probability $1/S$. In what follows, we use superscript notation to represent the specific value of a stochastic parameter omitting the underlying ξ^s parametrization for brevity (e.g., $\mathtt{d}_i^s = \mathtt{d}_i(\xi^s)$ represents the duration of trip $i\in\mathcal{I}$ in scenario $s\in\mathcal{S}$).

For the SAA model, we introduce a new set of binary variables $\boldsymbol{z}\in\{0,1\}^S$, where $z_s = 0$ if the service requirements are *met* for scenario $s\in\mathcal{S}$ (i.e., $\boldsymbol{x}\in\mathcal{F}(\xi^s)$) and $z_s = 1$ if the requirements can be violated. Then, the scenario-based formulation is given by:

$$\min \sum_{k\in\mathcal{K}}\sum_{(i,j)\in\mathcal{A}} \mathtt{c}_{ijk}x_{ijk} \tag{SAA-MCF}$$
$$\text{s.t. } \boldsymbol{x}\in\mathcal{X}$$
$$\sum_{s\in\mathcal{S}} z_s \le \lfloor S\epsilon \rfloor, \tag{4a}$$
$$(\boldsymbol{x}, z_s)\in\mathcal{F}^s, \qquad \forall s\in\mathcal{S}, \tag{4b}$$
$$z_s\in\{0,1\} \qquad \forall s\in\mathcal{S}.$$

Model (SAA-MCF) maintains the objective function and the set of planning constraints of (CC-MCF), but replaces the CC (1) (i.e., the operational portion of the problem) with a set of linear constraints for each scenario. Specifically, constraint (4a) enforces the maximum number of scenarios that can violate the service requirements. Set $\mathcal{F}^s$ corresponds to the service requirements $\mathcal{F}(\xi)$ for each scenario $s\in\mathcal{S}$, which is given by

$$\mathcal{F}^s = \Big\{(\boldsymbol{x},z)\in\{0,1\}^{A\times K}\times\{0,1\} : \exists\,(\boldsymbol{v},\boldsymbol{y},\boldsymbol{u}) \text{ such that}$$
$$\sum_{i\in\mathcal{I}} v_i \ge \mathtt{f}^{\mathtt{trip}}(1-z), \tag{5a}$$
$$\sum_{i\in\mathcal{I}(r)} v_i \ge \mathtt{f}_r^{\mathtt{route}}(1-z), \qquad \forall r\in\mathcal{R}, \tag{5b}$$
$$y_j + \mathtt{d}_j^s + \mathtt{t}_{ji}^s - u_j - \mathtt{M}_{ji}^{\mathtt{start}}\left(1-\sum_{k\in\mathcal{K}} x_{jik}\right) \le y_i, \qquad \forall (j,i)\in\mathcal{C}, \tag{5c}$$
$$\mathtt{t}_{ki}^s - \mathtt{M}_{ki}^{\mathtt{start}}(1-x_{kik}) \le y_i, \qquad \forall i\in\mathcal{I}, k\in\mathcal{K}, \tag{5d}$$
$$y_i \le \mathtt{s}_i + \mathtt{ub}v_i + \mathtt{M}_i^{\mathtt{OTP}}(1-v_i), \qquad \forall i\in\mathcal{I}, \tag{5e}$$
$$\mathtt{s}_i - \mathtt{lb} \le y_i, \qquad \forall i\in\mathcal{I}, \tag{5f}$$
$$v_i\in\{0,1\},\ y_i\ge 0,\ u_i\in[0,\mathtt{e}_i] \qquad \forall i\in\mathcal{I}\Big\}. \tag{5g}$$

The difference between $\mathcal{F}^s$ and $\mathcal{F}(\xi)$ is the inclusion of the scenario variable $z\in\{0,1\}$ and the realization of the random variables for scenario $s\in\mathcal{S}$. In particular, the OTP and fairness constraints

(5a) and (5b) are active if the service requirements for a particular scenario can be satisfied (i.e., $z = 0$). The remaining constraints (5c)-(5g) are analogous to (3c)-(3g) for a scenario realization.

It is well-known that an SAA model gives good approximations for its corresponding CCP model for a limited number of scenarios, and its optimal value converges to the CCP optimal value as the number of scenarios grows (Luedtke and Ahmed 2008). Thus, given a sufficiently large number of scenarios, (SAA-MCF) is guaranteed to yield a near-optimal solution for (CC-MCF).

4.2. Decomposition Framework

Model (SAA-MCF) is a large-scale problem due to scenario copies of variables z, $\boldsymbol{v}$, $\boldsymbol{y}$, $\boldsymbol{u}$, and their corresponding set of constraints $\mathcal{F}^s$ for each $s \in \mathcal{S}$. We propose a B&C framework that divides the problem into a master problem for all planning constraints and one subproblem for each scenario representing the operational constraints and variables.

Our master problem considers all the vehicle scheduling decisions $\boldsymbol{x} \in \{0,1\}^{A \times K}$ and the scenario variables $\boldsymbol{z} \in \{0,1\}^{S}$ indicating whether or not the service requirements are met for each scenario. Thus, the master problem is given by

$$\min \sum_{k \in \mathcal{K}} \sum_{(i,j) \in \mathcal{A}} \mathsf{c}_{ijk} x_{ijk} \qquad \text{(SAA-Master)}$$

$$\text{s.t. } \boldsymbol{x} \in \mathcal{X},$$

$$\sum_{s \in \mathcal{S}} z_s \leq \lfloor S\epsilon \rfloor,$$

$$\sum_{k \in \mathcal{K}} \sum_{(i,j) \in \mathcal{A}} \theta_{ijk} x_{ijk} \leq \theta_0 z_s + \theta_1 (1 - z_s), \qquad \forall (\boldsymbol{\theta}, \theta_0, \theta_1) \in \Theta^s,\ s \in \mathcal{S}, \tag{6a}$$

$$\sum_{(i,j) \in \mathcal{A}} \lambda_{ij} \sum_{k \in \mathcal{K}} x_{ijk} \leq \lambda_0 - 1 + z_s, \qquad \forall (\boldsymbol{\lambda}, \lambda_0) \in \Lambda^s,\ s \in \mathcal{S}, \tag{6b}$$

$$z_s \in \{0,1\}, \qquad \forall s \in \mathcal{S}.$$

Note that (SAA-Master) includes all the MCF planning constraints $\mathcal{X}$, which are common for all scenarios, and inequality (4a) that links all the scenario variables. The model also includes a set of valid inequalities (6a) that relate the scheduling decisions with each scenario variable, which we formally present in Section 4.4. Lastly, (6b) corresponds to the set of constraints added during the B&C algorithm, which are discussed in Section 5. Note that Θ^s and Λ^s are the set of coefficients[1] for constraints (6a) and (6b) for each scenario $s \in \mathcal{S}$, respectively.

The master problem considers all the variables and constraints linking different scenarios, which allows us to create one subproblem for each scenario. The primary goal of solving a subproblem is to check if a candidate vehicle schedule $\hat{\boldsymbol{x}} \in \mathcal{X}$ meets the service requirements for a specific

[1] These sets also determine the constants in the constraints but are referred to as coefficients for conciseness.

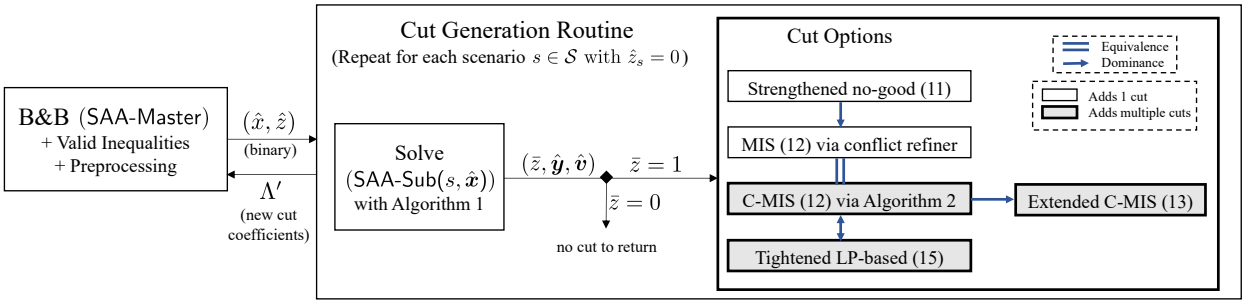


Figure 4 **Diagram of the decomposition scheme and the relationship between the proposed cuts.**

scenario. To do so, we consider subproblem (SAA-Sub$(s, \hat{\boldsymbol{x}})$) for each scenario $s \in \mathcal{S}$ and a fixed master solution $\hat{\boldsymbol{x}}$:

$$\bar{z} \in \underset{z \in \{0,1\}}{\arg\min} \{z : (\hat{\boldsymbol{x}}, z) \in \mathcal{F}^s\}. \qquad \text{(SAA-Sub}(s, \hat{\boldsymbol{x}})\text{)}$$

The model includes all the variables and constraints in $\mathcal{F}^s$ and minimizes the scenario variable z to indicate if $\hat{\boldsymbol{x}}$ meets the service requirements for $s \in \mathcal{S}$ when the optimal value is zero.

Figure 4 illustrates the main components of our B&C procedure. We perform a branch-and-bound (B&B) search over (SAA-Master) until we find a binary feasible solution $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{z}})$. This solution is passed to our cut generation routine. This routine iterates over all the scenarios where the master problem indicates that the vehicle schedule meets the service requirements (i.e., $\hat{z}_s = 0$) and checks if that is the case by solving (SAA-Sub$(s, \hat{\boldsymbol{x}})$) using the greedy algorithm (i.e., Algorithm 1 described in Section 4.3). If the optimal value of the subproblem $\bar{z}$ is one (i.e., at least one of the requirements cannot be met), then we generate one or more cuts for that scenario and add the cut coefficients to the corresponding set $\Lambda^{s\prime}$. Our procedure iterates over all scenarios searching for as many cuts as possible, but alternatives can be considered (e.g., add at most one cut per iteration). Lastly, we return the new set of cut coefficients Λ' and add them to the master problem as $\Lambda = \Lambda \cup \Lambda'$.

Figure 4 also presents several components of our procedure described in the following sections. For instance, the master problem can consider preprocessing steps commonly used in the determinist MDVSP literature (e.g., variable fixing and odd-cycle cuts (Hadjar, Marcotte, and Soumis 2006, Groiez et al. 2013)) and our valid inequalities detailed in Section 4.4. Lastly, it summarizes our proposed cut-generation alternatives and their relationship, which are presented in Section 5.

4.3. Subproblem Greedy Algorithm

Solving subproblem (SAA-Sub$(s, \hat{\boldsymbol{x}})$) for each candidate vehicle schedule $\hat{\boldsymbol{x}}$ and scenario $s \in \mathcal{S}$ is one of the main components of our cut generation routine. Any MILP solver can directly solve this subproblem, but it could be computationally expensive as the number of timetabled trips grows. In what follows, we present a polynomial-time greedy algorithm that returns an optimal solution to (SAA-Sub$(s, \hat{\boldsymbol{x}})$). This greedy procedure is used to solve (SAA-Sub$(s, \hat{\boldsymbol{x}})$) as illustrated in Figure 4.

Moreover, this procedure is a crucial component to some of our proposed cut-generation routines, as explained in Section 5.

To explain the greedy algorithm and other components of our methodology, we represent the schedule $\hat{\boldsymbol{x}}$ as a set of buses $\mathcal{B}(\hat{\boldsymbol{x}}) = \{\mathcal{B}_1, ..., \mathcal{B}_B\}$ (i.e., trips sequences), where B corresponds to the total number of buses in the schedule. Specifically, $\mathcal{B}(\hat{\boldsymbol{x}})$ is a partition of $\mathcal{I}$ such that each $\mathcal{B}_b$ with $b \in \{1, ..., B\}$ contains all the trips in the same trip sequence given by $\hat{\boldsymbol{x}}$. For each bus b, $\mathcal{B}_b$ is an ordered set with respect to the trip's schedule $\hat{\boldsymbol{x}}$ (e.g., the first trip in $\mathcal{B}_b$ is the first trip in the schedule of bus b). For example, the schedule in the left grid of Figure 2 contains two buses given by $\mathcal{B}_1 = (1, 3, 4, 2)$ and $\mathcal{B}_2 = (8, 6, 5, 7)$.

Algorithm 1 Greedy Algorithm for (SAA-Sub$(s, \hat{\boldsymbol{x}})$)

1: **procedure** GREEDY(s, $\mathcal{B}(\hat{\boldsymbol{x}})$)
2: Initialize $u_i^* = \mathtt{e}_i$ for all $i \in \mathcal{I}$
3: **for** $b \in \{1, ..., B\}$ **do**
4: **for** $p \in \{1, ..., |\mathcal{B}_b|\}$ **do**
5: Let i be the index of the p-th trip in $\mathcal{B}_b$
6: **if** $p = 1$ (i.e., i is the first trip in $\mathcal{B}_b$) **then** $y_i^* = \mathtt{s}_i - \mathtt{lb}$ and $v_i^* = 1$
7: **else**

$$y_i^* = \max\left\{\mathtt{s}_i - \mathtt{lb},\, y_j^* + \mathtt{d}_j^s + \mathtt{t}_{ji}^s - u_j^*\right\} \tag{7}$$

8: Set $v_i^* = 1$ if $y_i^* \in [\mathtt{s}_i - \mathtt{lb}, \mathtt{s}_i + \mathtt{ub}]$ and $v_i^* = 0$ otherwise
9: Set j as the last trip iterated (i.e., $j = i$)
10: **if** $\sum_{i \in \mathcal{I}(r)} v_i^* \geq \mathtt{f}_r^{\mathtt{route}}$ for all $r \in \mathcal{R}$ and $\sum_{i \in \mathcal{I}} v_i^* \geq \mathtt{f}^{\mathtt{trip}}$ **then** $z^* = 0$
11: **else** $z^* = 1$
12: **return** $(z^*, \boldsymbol{y}^*, \boldsymbol{v}^*, \boldsymbol{u}^*)$

Algorithm 1 shows our greedy procedure that finds the earliest start time of each timetabled trip for a given vehicle schedule $\hat{\boldsymbol{x}}$. It first fixes the expressing variables to their maximum value to consider the minimum total duration of each trip (line 2). We then iterate over the trips of each bus following their schedule order (lines 3-5) to guarantee that all predecessor trips' start times are computed before a given trip. If a trip is the first in the bus, it starts as early as possible and is always on time (line 6). Otherwise, (7) computes the earliest start time considering the previous trip in $\mathcal{B}_b$ and the scheduled start time. These times are then used to determine if a trip arrives on time (line 8). Lastly, we check if the OTP and fairness constraints are met, set the value of z^* accordingly, and return the solution (lines 9-10). This algorithm has a computational complexity of $O(I)$ because we iterate over each trip only once to compute the earliest start times $\boldsymbol{y}^*$ and the on-time variables $\boldsymbol{v}^*$. As stated in Proposition 1, whose proof is given in Appendix A.1, this algorithm returns an optimal solution for (SAA-Sub$(s, \hat{\boldsymbol{x}})$) for a given $\hat{\boldsymbol{x}}$ and $s \in \mathcal{S}$.

PROPOSITION 1. *For a master solution $\hat{\boldsymbol{x}}$ and scenario $s \in \mathcal{S}$, Algorithm 1 returns an optimal solution for* (SAA-Sub$(s, \hat{\boldsymbol{x}})$)*, where $\boldsymbol{y}^*$ is the earliest start times of each timetabled trip.*

4.4. Master Problem Valid Inequalities

As previously mentioned, we consider a set of valid inequalities (6a) to strengthen the master problem (SAA-Master) by relating the scheduling variables $\boldsymbol{x}$ with the scenario variables z_s for each $s \in \mathcal{S}$. We now describe these inequalities and show their validity.

The general idea is to use the scenario realizations to determine the set of trip pairs that will lead to delays. We first compute the operational compatibility set for each scenario $s \in \mathcal{S}$ as $\mathcal{C}^s = \{(i,j) \in \mathcal{I} \times \mathcal{I} : \mathtt{s}_i - \mathtt{lb} + \mathtt{d}_i^s + \mathtt{t}_{ij}^s - \mathtt{e}_i \leq \mathtt{s}_j + \mathtt{ub}\}$. This definition states that if we schedule trips $(i,j) \notin \mathcal{C}^s$ on the same bus, trip j will be delayed in scenario $s \in \mathcal{S}$. We use this observation to derive constraints that relate trip pairs that lead to known delays for any given scenario:

$$\sum_{j \in \mathcal{I}} \sum_{i:(i,j) \in \mathcal{C} \setminus \mathcal{C}^s} \sum_{k \in \mathcal{K}} x_{ijk} \leq z_s I_s + (1 - z_s)\mathtt{f}^{\mathtt{trip}}, \qquad \forall s \in \mathcal{S}. \tag{8}$$

where $I_s = |\{j \in \mathcal{I} : \exists i \in \mathcal{I} \text{ such that } (i,j) \in \mathcal{C} \setminus \mathcal{C}^s\}|$ is the maximum number of trips that can be delayed in s because one of its possible predecessors. Note that set $\mathcal{C} \setminus \mathcal{C}^s$ corresponds to all trip pairs that are feasible in the planning phase but lead to delays in scenario $s \in \mathcal{S}$. Thus, for each scenario $s \in \mathcal{S}$, inequality (8) considers all trips j that have a predecessor in $\mathcal{C}$ that will lead to a delay in scenario s. Since the depot does not affect the compatibility set and each trip can only be associated with one depot, we can consider all possible depot assignments. Thus, the left-hand side of (8) counts the number of delayed trips, and the right-hand side enforces $z_s = 1$ if the number of delayed trips surpasses the OTP requirements.

Similarly, we create inequalities that represent the fairness requirements for each route:

$$\sum_{j \in \mathcal{I}(r)} \sum_{i:(i,j) \in \mathcal{C} \setminus \mathcal{C}^s} \sum_{k \in \mathcal{K}} x_{ijk} \leq z_s I_s^r + (1 - z_s)\mathtt{f}_r^{\mathtt{route}}, \qquad \forall s \in \mathcal{S}, r \in \mathcal{R}, \tag{9}$$

where $I_s^r = |\{j \in \mathcal{I}(r) : \exists i \in \mathcal{I} \text{ such that } (i,j) \in \mathcal{C} \setminus \mathcal{C}^s\}|$ has an analogous mean to I_s.

The main difference between (8) and (9) is that the latter considers delays of trips associated with a specific route. Proposition 2 states the validity of the inequalities and their coefficient values considering the general form of (6a). The proof follows from the construction of the compatibility sets, the OTP definitions for each scenario, and that each trip must be visited exactly once (i.e., constraint (2a)).

PROPOSITION 2. *Inequalities (8) are valid for* (SAA-Master) *with coefficients:*

$$\theta_{ijk} = \begin{cases} 1, & (i,j) \in \mathcal{C} \setminus \mathcal{C}^s,\ k \in \mathcal{K}, \\ 0, & \textit{otherwise}, \end{cases} \qquad \theta_0 = I, \qquad \theta_1 = f^{trip}.$$

The analogous result is true for (9).

Lastly, we note that valid inequalities (8) and (9) do not dominate each other because each family of inequalities considers a different set of trips (i.e., either $\mathcal{I}$ or $\mathcal{I}(r)$ for some $r \in \mathcal{R}$). In fact, (8) can consider trips for different routes, while (9) only considers possible delays of trips of a particular route. Therefore, both families of inequalities can be useful to strengthen (SAA-Master).

5. Cut Generation Procedures

We now describe several cut generation procedures for our B&C decomposition framework. As previously mentioned, subproblem (SAA-Sub($s, \hat{\boldsymbol{x}}$)) is non-convex, so we cannot employ out-of-the-box Benders cuts as proposed by Luedtke (2014). Therefore, we develop our own cut-generation algorithms that leverage the underlying structure of the problem.

A naive cut generation alternative is to use no-good cuts, which are common in the logic-based Benders Decomposition (LBBD) literature (Hooker and Ottosson 2003). We can adapt these cuts for our B&C decomposition to relate a vehicle schedule with the scenario variables representing the fulfillment of the service requirements. In particular, the no-good cut for scenario $s \in \mathcal{S}$ in which a master solution $\hat{\boldsymbol{x}}$ does not fulfill the service requirements is

$$\sum_{(i,j,k)\in\overline{\mathcal{V}}^1} x_{ijk} + \sum_{(i,j,k)\in\overline{\mathcal{V}}^0} (1 - x_{ijk}) \leq \overline{V} - 1 + z_s, \tag{10}$$

where sets $\overline{\mathcal{V}}^0 = \{(i,j,k) \in \mathcal{A} \times \mathcal{K} : \hat{x}_{ijk} = 0\}$ and $\overline{\mathcal{V}}^1 = \{(i,j,k) \in \mathcal{A} \times \mathcal{K} : \hat{x}_{ijk} = 1\}$ represent the variable indices fixed to zero and one, respectively, for the vehicle schedule $\hat{\boldsymbol{x}}$, and $\overline{V} = |\overline{\mathcal{V}}^1 \cup \overline{\mathcal{V}}^0|$ is the size of the vehicle schedule assignment.

We can improve the standard no-good cut (10) by considering the structure of (SAA-MCF). First, it is sufficient to include the variables associated with the actual sequence assignments (i.e., set $\overline{\mathcal{V}}^1$) because all trips have to be sequenced exactly once and the remaining variables are known to be zero, as stated in (2a). Second, the depot associated with a trip sequence is irrelevant in practice to calculate the OTP and fairness requirements because the first trip in a bus always starts as early as possible. Thus, we can consider any possible depot assignment for the selected sequences. Then, a stronger no-good cut for $\hat{\boldsymbol{x}}$ and scenario $s \in \mathcal{S}$ is

$$\sum_{(i,j)\in\mathcal{V}} \sum_{k\in\mathcal{K}} x_{ijk} \leq V - 1 + z_s, \tag{11}$$

where $\mathcal{V} = \{(i,j) \in \mathcal{C} : \exists k \in \mathcal{K} \text{ such that } \hat{x}_{ijk} = 1\}$ corresponds to the set of trip pairs sequenced in solution $\hat{\boldsymbol{x}}$ ignoring the depot assignments, and $V = |\mathcal{V}|$ is the number of trips pairs. It is clear that (11) is stronger than (10) because the latter considers all possible depot assignments to the trip pairs and also uses a smaller right-hand side constant. Nonetheless, cut (11) is still quite weak

for our decomposition framework because it considers a single sequencing assignment (i.e., $\hat{x}_{ij\cdot}$ solution) from a set of exponentially many options.

In what follows, we present three different procedures to generate stronger cuts that consider a subset of trip pairs (i.e., partial trip sequences) that lead to vehicle schedules with unmet service requirements. Namely, Sections 5.1 and 5.2 present infeasible set (IS) cuts with two different approaches to find the associated set: (i) a conflict refinement procedure from commercial solvers to find a minimal IS (MIS) and (ii) an iterative procedure based on the greedy algorithm solution to find what we call a *constraint-based MIS* (C-MIS). Section 5.3 shows how to strengthen the C-MIS cuts by exploiting the procedure to find C-MISs. Section 5.4 shows how to find cuts using the LP relaxation of the subproblem (SAA-Sub$(s, \hat{\boldsymbol{x}})$) and the solution found by our greedy algorithm. We also show that there is a strong relationship between the C-MIS cuts and the LP-based cuts. Lastly, Section 5.5 shows how we can improve the B&C procedure given the specific structure of our cuts. Figure 4 summarizes all the proposed cut-generation alternatives, the relationship between each other, and the number of different cuts that each one can provide.

5.1. MIS Cuts via Conflict Refinement

A common practice in the literature is to strengthen no-good cuts by considering the minimal set of variable assignments that lead to an infeasible solution (Rahmaniani et al. 2017). Specifically, for a given master solution $\hat{\boldsymbol{x}}$ and scenario $s \in \mathcal{S}$, an IS is any subset of trip pairs $\mathcal{V}^{\text{IS}} \subseteq \mathcal{V}$ such that any schedule with this subset of trip pairs fails to fulfill the service requirements (i.e., violate (1)). An MIS is an IS $\mathcal{V}^{\text{MIS}} \subseteq \mathcal{V}$ such that no subset of $\mathcal{V}^{\text{MIS}}$ is also an IS. Then, an MIS cut is given by

$$\sum_{(i,j)\in\mathcal{V}^{\text{MIS}}} \sum_{k\in\mathcal{K}} x_{ijk} \leq |\mathcal{V}^{\text{MIS}}| - 1 + z_s. \tag{12}$$

One way to find an MIS is to use the conflict refinement tools of commercial solvers, such as CPLEX or Gurobi. Specifically, for a given $\hat{\boldsymbol{x}}$ that does not meet the service requirements in scenario $s \in \mathcal{S}$, we modify subproblem (SAA-Sub$(s, \hat{\boldsymbol{x}})$) by fixing variable $z = 0$ and, thus, making the problem infeasible. We then run the conflict refinement procedure to find the minimum set of constraints associated with $\hat{\boldsymbol{x}}$ that leads to a conflict (i.e., constraints (5c)). Finally, $\mathcal{V}^{\text{MIS}}$ is given by the trip pairs associated with the conflicting constraints.

The MIS cuts (12) are guaranteed to remove the same or more solutions than (11) because $\mathcal{V}^{\text{MIS}} \subseteq \mathcal{V}$. One issue of this approach is its computational time because it needs to solve a MILP and run the conflict refinement procedure for each candidate's master problem solution and scenario. Moreover, there might be multiple MISs for a given $\hat{\boldsymbol{x}}$ and $s \in \mathcal{S}$, but the conflict refinement procedure only returns one.

5.2. C-MIS Cuts Using Greedy Solution

Given the drawbacks of using MILP-based conflict refinement to find an MIS and the lack of alternative efficient procedures, we develop a polynomial-time procedure to find C-MISs using the solution of Algorithm 1. We define a C-MIS as an MIS obtained when we relax (SAA-Sub($s, \hat{\boldsymbol{x}}$)) by considering a single violated service requirement constraint (i.e., one inequality of (5a) and (5b)). To build a C-MIS, we develop a two-step algorithm that: (i) identifies a set of delayed trips that constitute the base to build a C-MIS, and (ii) finds a subsequence of trips in the master candidate schedule that leads to those delays.

In what follows, consider a master solution $\hat{\boldsymbol{x}}$ that violates the service requirements for a scenario $s \in \mathcal{S}$ and its corresponding solution $(z^*, \boldsymbol{y}^*, \boldsymbol{v}^*, \boldsymbol{u}^*)$ from Algorithm 1. We use the notation $\mathcal{D} = \{i \in \mathcal{I} : v_i^* = 0\}$ to represent the set of trips that are delayed in scenario s, and the bus notation $\mathcal{B}(\hat{\boldsymbol{x}}) = \{\mathcal{B}_1, ..., \mathcal{B}_B\}$ to represent a schedule $\hat{\boldsymbol{x}}$ (i.e., the trip sequences of each bus).

Step (i). To create a C-MIS, we first identify one service requirement that is violated (i.e., one inequality in (5a) and (5b)). We then create a minimal subset of trips $\mathcal{D}^{\texttt{C-MIS}} \subseteq \mathcal{D}$ that violates such requirement and also satisfies a special *predecessor condition* detailed in what follows. If we choose (5a), $\mathcal{D}^{\texttt{C-MIS}}$ is the minimal number of delayed trips that violate the constraint when $z = 0$, that is, $\mathcal{D}^{\texttt{C-MIS}} \subseteq \mathcal{D}$ such that $|\mathcal{D}^{\texttt{C-MIS}}| = I - \texttt{f}^{\texttt{trip}} + 1$. Analogously, if the chosen condition is (5b) for some route $r \in \mathcal{R}$, we consider $\mathcal{D}^r = \mathcal{D} \cap \mathcal{I}(r)$ and $\mathcal{D}^{\texttt{C-MIS}} \subseteq \mathcal{D}^r$ such that $|\mathcal{D}^{\texttt{C-MIS}}| = I_r - \texttt{f}_r^{\texttt{route}} + 1$. Among these $\mathcal{D}^{\texttt{C-MIS}}$ options, we consider those that satisfy the following predecessor condition: for each trip $i \in \mathcal{D}^{\texttt{C-MIS}}$, any delayed predecessor should also be in $\mathcal{D}^{\texttt{C-MIS}}$, that is, if $i \in \mathcal{D}^{\texttt{C-MIS}}$ with $i \in \mathcal{B}_b$ then $j \in \mathcal{D} \cap \mathcal{B}_b$ with $y_j^* \leq y_i^*$ is also in $\mathcal{D}^{\texttt{C-MIS}}$. The analogous condition is imposed when choosing (5b) for some $r \in \mathcal{R}$ by replacing $\mathcal{D}$ with $\mathcal{D}^r$. This predecessor condition is crucial to guarantee that the resulting IS is minimal, as shown in Proposition 3.

Step (ii). Given $\mathcal{D}^{\texttt{C-MIS}}$, we create a C-MIS $\mathcal{V}^{\texttt{C-MIS}}$ by selecting a trip subsequence from the schedule $\hat{\boldsymbol{x}}$ for each trip in $\mathcal{D}^{\texttt{C-MIS}}$ that *explain its delay*. We say that a trip subsequence of $p > 1$ trips $(i_1, ...i_{p-1}, j)$ explains the delay of trip $j \in \mathcal{D}^{\texttt{C-MIS}}$ if j would be delayed even when trip i_1 start as early as possible (i.e., $\texttt{s}_{i_1} - \texttt{lb}$). Also, the subsequence $(i_1, ...i_{p-1}, j)$ is minimal if it explains the delay of j, but the subsequence without trip i_1 (i.e., $(i_2, ...i_{p-1}, j)$) does not.

Algorithm 2 details the procedure to build a C-MIS for a given violated constraint $\texttt{con} \in \{(5a), \{(5b)\}_{r \in \mathcal{R}}\}$ and find minimal subsequences that explain the delays of trips in the corresponding $\mathcal{D}^{\texttt{C-MIS}}$. The procedure iterates over the scheduled buses and selects the last delayed trip of a bus that belongs to $\mathcal{D}^{\texttt{C-MIS}}$, trip $i \in \mathcal{D}^{\texttt{C-MIS}}$ (line 4). For trip i to be delayed, its start time should be strictly greater than $\texttt{s}_i + \texttt{ub}$, so we consider the earliest delayed start time to be $s_i + \texttt{ub} + \epsilon^{\texttt{tol}}$ (line 5), where $\epsilon^{\texttt{tol}} > 0$ is a predefined tolerance (e.g., $\epsilon^{\texttt{tol}} = 0.1$ or $\epsilon^{\texttt{tol}} = 0.01$). Since a predecessor trip must cause this delay, we select the previous trip on the bus, trip j, add the trip pair (j, i) to

the C-MIS, and then check if the predecessor is the sole source of the delay (lines 7-8). If i is still delayed even if j starts as early as possible (i.e., $s_j - \texttt{lb}$), then j is sufficient to explain the delay of i, and there is no need to check other predecessor trips (lines 9-10). Otherwise, j has to start later than $\texttt{s}_j - \texttt{lb}$ to cause the delay for i, and we calculate the earliest possible start time of j for i to be delayed (lines 12). Lastly, if j belongs to $\mathcal{D}^{\texttt{C-MIS}}$, we also need to explain the daily of j, thus, we also consider the earliest start time of j (i.e., $\texttt{s}_j + \texttt{ub} + \epsilon^{\texttt{tol}}$) (line 14). The algorithm iterates until we explain the delay for all trips in $\mathcal{D}^{\texttt{C-MIS}}$.

Algorithm 2 C-MIS construction based on the greedy algorithm solution

1: **procedure** C-MIS_GREEDY($s, \mathcal{B}(\hat{\boldsymbol{x}})$, $\boldsymbol{y}^*$, $\boldsymbol{v}^*$, $\texttt{con} \in \{(5a), \{(5b)\}_{r\in\mathcal{R}}\}$)
2: Build a $\mathcal{D}^{\texttt{C-MIS}}$ based on con as explained in *Step (i)* of Section 5.2
3: **for** $b \in \{1, ..., B\}$ such that $\mathcal{B}_b \cap \mathcal{D}^{\texttt{C-MIS}} \neq \emptyset$ **do**
4: Select trip $i \in \mathcal{B}_b \cap \mathcal{D}^{\texttt{C-MIS}}$ with the latest start time and set $\mathcal{D}^{\texttt{C-MIS}} = \mathcal{D}^{\texttt{C-MIS}} \setminus \{i\}$
5: $\texttt{start\_time\_to\_explain} = \texttt{s}_i + \texttt{ub} + \epsilon^{\texttt{tol}}$
6: **while** $\texttt{start\_time\_to\_explain} > 0$ **do**
7: Select the trip $j \in \mathcal{B}_b$ that is scheduled right before trip i.
8: Add trip pair to C-MIS: $\mathcal{V}^{\texttt{C-MIS}} = \mathcal{V}^{\texttt{C-MIS}} \cup \{(j, i)\}$
9: **if** $\texttt{s}_j - \texttt{lb} + \texttt{d}^s_j + \texttt{t}^s_{ji} \geq \texttt{start\_time\_to\_explain}$ **then** # Trip j explains the delay of i
10: $\texttt{start\_time\_to\_explain} = 0$
11: **else** # Explain the delay of j
12: $\texttt{start\_time\_to\_explain} = \texttt{start\_time\_to\_explain} - (\texttt{d}^s_j + \texttt{t}^s_{ji} - \texttt{e}_j)$.
13: **if** $j \in \mathcal{D}^{\texttt{C-MIS}}$ **then**
14: $\texttt{start\_time\_to\_explain} = \max\{\texttt{start\_time\_to\_explain}, \texttt{s}_j + \texttt{ub} + \epsilon^{\texttt{tol}}\}$
15: Set $i = j$.
16: Go back to line 4 if $\mathcal{B}_b \cap \mathcal{D}^{\texttt{C-MIS}} \neq \emptyset$

Example 4 illustrates some of the specific considerations of Algorithm 2. Also, Proposition 3 (proved in Appendix A.2) shows that the resulting set $\mathcal{V}^{\texttt{C-MIS}}$ from Algorithm 2 is indeed a C-MIS for a given $\hat{\boldsymbol{x}}$.

EXAMPLE 4. Consider the trip sequence depicted in Figure 5 . The graph illustrates the first 6 trips on the sequence, where shaded nodes correspond to delayed trips (i.e., $4, 6 \in \mathcal{D}^{\texttt{C-MIS}}$). The number above the arrows correspond to the duration of the predecessor trips and the travel time between trips for an scenario $s \in \mathcal{S}$ (e.g., $\texttt{d}^s_2 + \texttt{t}^s_{23} = 15$ for arc between trips 2 and 3). As illustrated in the graph, the number and the interval above each node trip $i \in \mathcal{I}$ represents the earliest start time of such trip y^*_i and its corresponding time window for it to be on-time (i.e., $[\texttt{s}_i - \texttt{lb}, \texttt{s}_i + \texttt{ub}]$). The numbers below the nodes are the start_time_to_explain used in Algorithm 2. We consider $\texttt{e}_i = 0$ for all $i \in \mathcal{I}$, $\epsilon^{\texttt{tol}} = 1$, $\texttt{lb} = 1$ and $\texttt{ub} = 3$ for this example.

To find the minimal subsequence that explains the delays of trips 4 and 6 $\in \mathcal{D}^{\texttt{C-MIS}}$, we start by backtracking from trip 6. The minimum start time for this trip to be delayed is 96, thus,

$\texttt{start\_time\_to\_explain} = 96$ (line 5 of Algorithm 2). We now take its immediate predecessor, trip 5, and check if it the sole source of the delay of trip 6. Note that if trip 5 start as early as possible (i.e., $\mathtt{s}_5 - \mathtt{lb} = 72$), then trip 6 would be on time because $y_6 = \mathtt{s}_5 - \mathtt{lb} + \mathtt{d}^s_5 + \mathtt{t}^s_{5,6} = 72 + 23 = 95 \leq \mathtt{s}_6 + \mathtt{ub}$. Therefore, trip 5 starts slightly later for trip 6 to be delay, that is, it must start at time $\texttt{start\_time\_to\_explain} = 73$ or later (line 12 of Algorithm 2). Once again, we take the predecessor of trip 5 (i.e., trip 4) and check if it can explain the start time of 5. Trip 4 has to start at time 59 for trip 5 to start at time 73, which also coincide with the earliest start time of trip 4 to be delay, thus, $\texttt{start\_time\_to\_explain} = 59$ (line 14 of Algorithm 2). Lastly, we analyze the predecessor of trip 4 (i.e., trip 3), and we see that if trip 3 starts as early as possible (i.e., $\mathtt{s}_3 - \mathtt{lb} = 38$), then trip 4 starts at time 59, which coincides with $\texttt{start\_time\_to\_explain}$. Therefore, the subsequence (3,4,5,6) explains the delays of trips 4 and 6 $\in \mathcal{D}^{\texttt{C-MIS}}$ at scenario $s \in \mathcal{S}$. This subsequence is minimal because removing trip 3 from it will result of trips 4 and 6 to arrive on-time. □

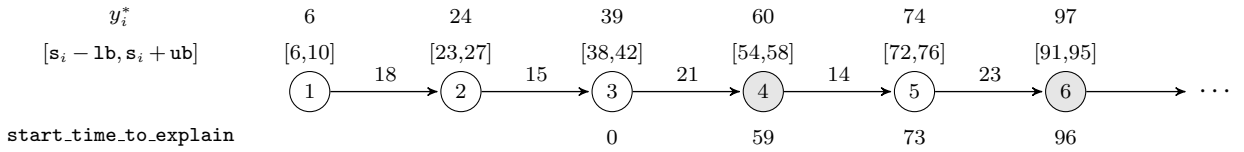


Figure 5 **Graphical example on how to compute a C-MIS using Algorithm 2.**

PROPOSITION 3. *Consider a vehicle schedule $\mathcal{B}(\hat{x})$ and a scenario $s \in \mathcal{S}$ such that at least one of the service requirements is violated. Then, for any such constraint $\texttt{con} \in \{(5a), \{(5b)\}_{r \in \mathcal{R}}\}$, Algorithm 2 builds a C-MIS $\mathcal{V}^{C\text{-}MIS}$.*

Algorithm 2 has two main advantages when compared to the conflict refinement procedure explained in Section 5.1. First, it is a polynomial-time algorithm since it iterates two times over the set of trips (i.e., for choosing $\mathcal{D}^{\texttt{C-MIS}}$ and for explaining their delays). Second, the procedure allows us to obtain multiple C-MISs by choosing different violated service requirements and $\mathcal{D}^{\texttt{C-MIS}}$. Our implementation takes advantage of this fact in that it creates a C-MIS for each violated service constraint (i.e., (5a) and (5b)), and returns their corresponding the violated cuts. Lastly, the resulting cut is equivalent to (12) since we only change the procedure for obtaining a C-MIS.

As a final remark, we note that $\mathcal{V}^{\texttt{C-MIS}}$ might not be an MIS for the subproblem of scenario $s \in \mathcal{S}$ if there are multiple service requirement violations, because it only considers a single service requirement constraint at a time. Nevertheless, our implementation constructs a C-MIS for each violated service requirement constraint for a given scenario $s \in \mathcal{S}$ and $\hat{\boldsymbol{x}}$, and we empirically observed that at least one of them is indeed an MIS.

5.3. Extended C-MIS Cuts

We now present a procedure to find stronger cuts by extending a C-MIS. The main idea is to find alternative subsequences of trips that also explain the delay of the trips in $\mathcal{D}^{\texttt{C-MIS}}$. Note that the trip pairs in $\mathcal{V}^{\texttt{C-MIS}}$ can be represented as disjoint trip subsequences $(i_1, ..., i_p)$ where p is the size of a particular sequence and (i, j) are consecutive trips in a subsequence if and only if $(i, j) \in \mathcal{V}^{\texttt{C-MIS}}$.

Consider a subsequence $(i_1, ..., i_p)$ of $\mathcal{V}^{\texttt{C-MIS}}$ with $i_p \in \mathcal{D}^{\texttt{C-MIS}}$. To create alternative subsequences that also explain the delay of i_p, we seek for replacements of the initial trip i_1 that will also explain the delay of i_p and any other delayed trip such subsequence. Formally, we look for trips $i'_1 \in \mathcal{I}$ such that $(i'_1, i_2) \in \mathcal{C}$, i'_1 does not appear in any trip pair in $\mathcal{V}^{\texttt{C-MIS}}$, and the new sequence $(i'_1, i_2, ..., i_p)$ explains the delay of i_p and potentially other trips in $\mathcal{D}^{\texttt{C-MIS}}$ that appear in $(i_2, ..., i_{p-1})$. If any such trip i'_1 exists, we add the trip pair (i'_1, i_2) to a set of additional pairs $\mathcal{V}^{\texttt{Extra}}$. We then create an extended C-MIS cut that considers all trip pairs in $\mathcal{V}^{\texttt{C-MIS}} \cup \mathcal{V}^{\texttt{Extra}}$, as shown in (13). Proposition 4 shows the validity of the cut and its dominance relationship to (12).

PROPOSITION 4. *Consider a vehicle schedule $\mathcal{B}(\hat{x})$, a scenario $s \in \mathcal{S}$ such that at least one of the service requirements $\texttt{con} \in \{(5a), \{(5b)\}_{r \in \mathcal{R}}\}$ is violated, and a C-MIS $\mathcal{V}^{C\text{-}MIS}$ built using $\texttt{con}$. Then, the following constraint is valid and dominates (12) with $\mathcal{V}^{MIS} = \mathcal{V}^{C\text{-}MIS}$,*

$$\sum_{(i,j) \in \mathcal{V}^{C\text{-}MIS} \cup \mathcal{V}^{Extra}} \sum_{k \in \mathcal{K}} x_{ijk} \leq |\mathcal{V}^{C\text{-}MIS}| - 1 + z_s, \tag{13}$$

where $\mathcal{V}^{Extra}$ is constructed as previously explained.

Proof. The validity of the cut follows from the procedure for finding additional trip pairs $\mathcal{V}^{\texttt{Extra}}$ and that each trip has only one predecessor (i.e., inequality (2a)). To prove dominance, note that (13) and (12) with $\mathcal{V}^{\texttt{MIS}} = \mathcal{V}^{\texttt{C-MIS}}$ have the same right-hand side but (13) considers more variables by incorporating all the alternative pairs in $\mathcal{V}^{\texttt{Extra}}$. Thus, the extended version removes the same schedules that (12), but it also removes the alternative schedules built with $\mathcal{V}^{\texttt{Extra}}$. □

5.4. LP-based Benders Cuts

We now present a procedure to find cuts for each integer master solution $\hat{\boldsymbol{x}}$ and an infeasible scenario $s \in \mathcal{S}$ using a strengthened version of the linear programming (LP) relaxation of (SAA-Sub$(s, \hat{\boldsymbol{x}})$) and show that the resulting cuts are equivalent to our C-MIS cuts.

While we can directly use the LP relaxation to derive cuts, preliminary experiments show that such relaxation is quite weak and, in most cases, does not produce a violated cut when a schedule does not meet the service requirements. Thus, we present a procedure that tightens the coefficients of the LP relaxation and introduces additional inequalities using the optimal solution of Algorithm 1 to close the integrality gap and return valid cuts for the master problem.

We now show how to strengthen the LP relaxation of (SAA-Sub$(s, \hat{\boldsymbol{x}})$) utilizing the optimal solution $(z^*, \boldsymbol{y}^*, \boldsymbol{v}^*, \boldsymbol{u}^*)$ obtained with Algorithm 1 to indirectly enforce integrality on variables $\boldsymbol{v}$ and z in its optimal solution. First, the value of v_i for each $i \in \mathcal{I}$ is enforced by constraint (5e) and is directly impacted by the big-M coefficient $\mathtt{M}_i^{\mathtt{OTP}}$. While $\mathtt{M}_i^{\mathtt{OTP}}$ can be arbitrarily large for the MILP formulation, we need it to be as small as possible to force $v_i = 0$ whenever $y_i^* > \mathtt{s}_i + \mathtt{ub}$ for $i \in \mathcal{I}$. Thus, we set $\mathtt{M}_i^{\mathtt{OTP}} = y_i^* - \mathtt{s}_i$ for every particular $\hat{\boldsymbol{x}}$ and s to ensure the integrality of v_i in an optimal solution of the strengthened LP.

To enforce integrality on z, we include an additional constraint that relates the set of delayed trips with z. Specifically, we pick one constraint in $\mathtt{con} \in \{(5a), \{(5b)\}_{r \in \mathcal{R}}\}$ that is violated at the solution obtained by Algorithm 1 and construct a $\mathcal{D}^{\mathtt{C\text{-}MIS}}$ using the procedure detailed in *Step (i)* of Section 5.2. Using this set, we add the following inequality to the LP relaxation of the subproblem:

$$\sum_{i \in \mathcal{D}^{\mathtt{C\text{-}MIS}}} (1 - v_i) \leq z + |\mathcal{D}^{\mathtt{C\text{-}MIS}}| - 1. \tag{14}$$

Note that in any optimal solution $v_i = 0$ for all $i \in \mathcal{D}^{\mathtt{C\text{-}MIS}}$ because of the tightened $\mathtt{M}_i^{\mathtt{OTP}}$ coefficients, so (14) forces $z = 1$ in such solutions.

Lastly, to guarantee the validity of the resulting cuts and prove its relationship to (12) with its corresponding $\mathcal{V}^{\mathtt{C\text{-}MIS}}$ (i.e., the $\mathcal{V}^{\mathtt{C\text{-}MIS}}$ returned by Algorithm 2 for the $\mathcal{D}^{\mathtt{C\text{-}MIS}}$ use in (14)), we also tighten the big-M coefficients $\mathtt{M}_{ji}^{\mathtt{start}}$ for all $(j, i) \in \mathcal{C}$ as follows:

$$\mathtt{M}_{ji}^{\mathtt{start}} = \begin{cases} y_i^*, & \forall (j,i) \in \mathcal{V} \\ \text{large enough number}, & \text{otherwise.} \end{cases}$$

Note that these tighten coefficients only appear in (5c) and are valid for the model. Also, they do not affect the integrality of the variables $\boldsymbol{v}$ and z in the optimal solution, but are important for our proof. To summarize, the tightened LP subproblem is given by

$$\min_{z \in [0,1]} \left\{ z : \, (\hat{\boldsymbol{x}}, z) \in \overline{\mathcal{F}}^s \cap \{(14)\} \right\}, \tag{SAA-Sub-LP$(s, \hat{\boldsymbol{x}})$}$$

where $\overline{\mathcal{F}}^s$ is the LP relaxation of $\mathcal{F}^s$ with the tightened $\mathtt{M}_i^{\mathtt{OTP}}$ and $\mathtt{M}_{ij}^{\mathtt{start}}$ coefficients for every $i \in \mathcal{I}$ and $(i, j) \in \mathcal{C}$, respectively. Appendix A.4 shows in detail (SAA-Sub-LP$(s, \hat{\boldsymbol{x}})$) and its dual.

Given the tightened LP model, we employ the dual of (SAA-Sub-LP$(s, \hat{\boldsymbol{x}})$) to derive the following Bender's cut

$$\sum_{(i,j) \in \mathcal{C}} \alpha_{ij} \mathtt{M}_{ij}^{\mathtt{start}} \sum_{k \in \mathcal{K}} x_{ijk} \leq \sum_{(i,j) \in \mathcal{C}} \alpha_{ij} \mathtt{M}_{ij}^{\mathtt{start}} \sum_{k \in \mathcal{K}} \hat{x}_{ijk} - 1 + z_s, \tag{15}$$

where $\boldsymbol{\alpha} \geq 0$ are the dual variables associated with constraints (5c). Proposition 5 states the validity of the cut and shows that there exists a dual solution such that (15) is equivalent to the C-MIS cut (12) associated with $\mathcal{V}^{\mathtt{C\text{-}MIS}}$. Appendix A.4 details the proof of the proposition and shows how to construct such dual solution.

PROPOSITION 5. *Consider the master solution $\hat{\boldsymbol{x}}$ that violates one of the service requirements $\texttt{con} \in \{(5a), \{(5b)\}_{r \in \mathcal{R}}\}$ for a scenario $s \in \mathcal{S}$, and a $\mathcal{D}^{\texttt{C-MIS}}$ constructed using $\texttt{con}$. Then, there exists a dual optimal solution of (SAA-Sub-LP$(s, \hat{\boldsymbol{x}})$) such that (15) and the C-MIS cut (12) for the corresponding $\mathcal{V}^{\texttt{C-MIS}}$ are identical.*

While Proposition 5 states how to obtain a C-MIS cut using (SAA-Sub-LP$(s, \hat{\boldsymbol{x}})$), preliminary results show that it is computationally more efficient to obtain such cuts using Algorithm 2. Therefore, we omit this alternative from our empirical results in Section 7. Nonetheless, we believe that this insightful result could be beneficial for other CCP problems where obtaining a C-MIS (or MIS) could be computationally challenging.

5.5. B&C Modification Given Cuts Structure

We present a modification to the B&C procedure presented in Section 4 that takes advantage of the specific structure of our cut variants. In particular, we propose relaxing the integrality constraints on $\boldsymbol{z}$ when solving the master problem (SAA-Master). With this change, our decomposition scheme depicted in Figure 4 enters the cut generation routine for each scenario $s \in \mathcal{S}$ when $\hat{z}_s < 1$ in the master problem, in contrast to only when $\hat{z}_s = 0$. Proposition 6 states the validity of this modification, whose proof (in Appendix A.5) follows from the structure of our proposed cuts.

PROPOSITION 6. *The B&C decomposition scheme with the previously mentioned modification converges in a finite number of iterations and returns an optimal schedule to* (SAA-MCF).

We note that relaxing the integrality constraints of $\boldsymbol{z}$ is quite general and can be applied to other CCP problems where the generated cuts have a similar form to (6b), i.e., they are tight at the master solution producing them and enforce the integrality of $\boldsymbol{z}$. Our computational results show that this modification can reduce the solution time and increase the number of instances solved because it significantly decreases the size of the B&C tree by avoiding branching on $\boldsymbol{z}$.

6. Lagrangian-based Approach for Large-scale Instances

The branch-and-cut procedure presented in Section 4 can obtain optimal solutions for the scenario reformulation of our problem. However, our experimental results show that it can only handle problem instances with a small number of trips in a reasonable amount of time (i.e., solve instances with 80 trips and four depots in less than two hours). In contrast, real-world instances of the MDVSP based on small and medium-sized cities can consider hundreds or thousands of trips. Therefore, we need a procedure that can find vehicle schedules that satisfy the service requirements for large-size instances.

To do so, we propose a Lagrangian-based technique that leverages our exact procedure to find low-cost vehicle schedules that satisfy the service requirements. The main idea is to partition the

set of trips and create a subproblem for each partition that can be efficiently solved with our B&C decomposition scheme. We can then devise a procedure to combine the scheduling solution of each subproblem and create a vehicle schedule that satisfies all the planning constraints of the original problem. To enforce the operational constraints modeled with the CC, we utilize Lagrangian penalties to ensure that the number of scenarios that violate the service requirements is small across all partition subproblems. As a result, our methodology finds a low-cost solution that is feasible for the original problem or violates the CC in a small amount (i.e., a few scenarios).

Next, we detail the main components of our procedure. Section 6.1 presents the proposed strategies to partition the set of trips and the resulting subproblems. Then, Section 6.2 introduces our Lagrangian decomposition scheme to enforce the CC, and Section 6.3 shows how we solve the resulting Lagrangian dual problem and obtain a vehicle schedule with the required characteristics.

6.1. Trips Partitioning and Subproblems

Our procedure starts by partitioning the set of trips $\mathcal{I}$ into small-size groups to create one CC-MDVSP subproblem for each partition. Constructing a trip partition that results in a minimum-cost schedule is a type of bin-packing problem and, as such, an NP-hard problem. Thus, this work considers a heuristic approach for partitioning trips that lead to low-cost vehicle schedules.

Since the trip partition directly affects the cost of the resulting schedule, we want to partition the trips such that several trips in a group can be scheduled on the same bus to minimize the number of buses in the scheduling solution of a subproblem (i.e., the largest cost component of the problem). To do so, we solve a deterministic MDVSP (e.g., using average times and ignoring service requirements) and use the resulting solution to assign trips to the different groups such that trips in the same bus are assigned to the same group. Algorithm 3 details this assignment where $\mathtt{m}^{\mathtt{gr}}$ is the minimum size of a group, $\mathcal{B}(\hat{\boldsymbol{x}})$ is the scheduling solution of the deterministic MDVSP, and $\mathcal{I}^{\mathtt{gr}} = \{\mathcal{I}_1, ..., \mathcal{I}_P\}$ is the trip partition, where $\mathcal{P} = \{1, ..., P\}$ is the set of indices of the partition of size $P = |\mathcal{P}|$. Other heuristics or exact strategies can also be considered, which we leave as future work directions.

Algorithm 3 Trip partitioning based on deterministic solution

1: **procedure** PARTITIONTRIPS($\mathcal{B}(\hat{\boldsymbol{x}})$, $\mathtt{m}^{\mathtt{gr}}$)
2: $\quad\mathcal{I}^{\mathtt{gr}} = \mathcal{I}^{\mathtt{aux}} = \emptyset$
3: $\quad$**for** bus $\mathcal{B} \in \mathcal{B}(\hat{\boldsymbol{x}})$ **do**
4: $\quad\quad$Add trips to the current group: $\mathcal{I}^{\mathtt{aux}} = \mathcal{I}^{\mathtt{aux}} \cup \mathcal{B}$
5: $\quad\quad$**if** $\mathcal{I}^{\mathtt{aux}} \geq \mathtt{m}^{\mathtt{gr}}$ **then**
6: $\quad\quad\quad$Add group to trip partition: $\mathcal{I}^{\mathtt{gr}} = \mathcal{I}^{\mathtt{gr}} \cup \mathcal{I}^{\mathtt{aux}}$, and reset group: $\mathcal{I}^{\mathtt{aux}} = \emptyset$
$\quad$**return** trip partition $\mathcal{I}^{\mathtt{gr}}$

Algorithm 3 creates a partition of $\mathcal{I}$ such that all trips are assigned to a single group. Then, our Lagrangian-based procedure uses $\mathcal{I}^{\text{gr}}$ to create a subproblem for each group $p \in \mathcal{P}$ given by

$$\min \sum_{k \in \mathcal{K}} \sum_{(i,j) \in \mathcal{A}^p} \mathsf{c}_{ij} x^p_{ijk} \qquad \text{(SAA-MCF}^p\text{)}$$

$$\text{s.t. } \boldsymbol{x}^p \in \mathcal{X}^p$$

$$\sum_{s \in \mathcal{S}} z^p_s \leq \lfloor S\epsilon \rfloor, \tag{16a}$$

$$(\boldsymbol{x}^p, z^p_s) \in \mathcal{F}^{p,s}, \qquad \forall s \in \mathcal{S}, \tag{16b}$$

$$z^p_s \in \{0,1\} \qquad \forall s \in \mathcal{S}.$$

This formulation is identical to (SAA-MCF) but adjusts the variables and constraints to consider only the trips in $\mathcal{I}^p$. Specifically, we use superscript p to denote the modified elements that only consider trips in $\mathcal{I}^p$, that is, the compatible set $\mathcal{C}^p$, the set of arcs $\mathcal{A}^p$ and the operational constraints $\mathcal{F}^{p,s}$ for each scenario $s \in \mathcal{S}$, also add it to the decision variables to distinguish the subproblems. Note that sets $\mathcal{K}$ and $\mathcal{S}$ are the same for each subproblem.

Once each subproblem is solved, we combine the schedules of (SAA-MCF^p) for each $p \in \mathcal{P}$ to create a vehicle schedule that satisfies all the planning constraints $\mathcal{X}$ of the original problem. Given that trip groups are pair-wise disjoint, constraints $\mathcal{X}^p$ for each $p \in \mathcal{P}$ will enforce the trips' coverage and the flow balance constraints (i.e., (2a), (2c)-(2d) considering all trips). The only constraint that might be violated is the maximum capacity of the depots (2b). In such a case, we heuristically reassign the depot for some trip sequences in order to fulfill the capacity requirement. Specifically, we iteratively seek for a trip sequence $\mathcal{B}_b = (i_1, ..., i_n)$ (where n is the number of trips) assigned to an overloaded depot k' and reassign it to a depot k'' with available capacity that minimized the cost of the depot reallocation, that is, $k'' \in \arg\min_{k \in \mathcal{K} \setminus \{k'\}} \{c_{ki_1} + c_{i_n k} : k \text{ has available capacity}\}$.

Therefore, we can use this procedure (i.e., partition the trips, solve the subproblems, and reassign the depots) to create a vehicle schedule that fulfills all the operational constraints of the problem. Unfortunately, this technique will not necessarily enforce the CC of the original problem because (SAA-MCF^p) enforces individual CCs for each partition $p \in \mathcal{P}$ and, thus, ignores the joint behavior. The following section addresses this issue with a Lagrangian decomposition approach.

6.2. Lagrangian Decomposition

A reason why satisfying the individual CC for each subproblem (SAA-MCF^p) does not enforce the common CC of the original problem is that each subproblem can unmet the service requirements in different scenarios. This discrepancy usually results in more scenarios with unmet service conditions for the combined vehicle schedule and, thus, a violation of the common CC. Inspired by this observation, we propose a Lagrangian decomposition scheme that limits the set of scenarios with

unmet service requirements across all subproblems via Lagrangian penalties. Specifically, we propose model (SAA-MCF-Joint) that combines subproblems (SAA-MCF^p) for each $p \in \mathcal{P}$ and relates the scenario variables for each partition.

$$\begin{aligned}
\min \sum_{p\in\mathcal{P}}\sum_{k\in\mathcal{K}}\sum_{(i,j)\in\mathcal{A}^p} \mathsf{c}_{ij} x^p_{ijk} & && \text{(SAA-MCF-Joint)}\\
\text{s.t. } \boldsymbol{x}^p \in \mathcal{X}^p, & \quad && \forall p \in \mathcal{P}\\
\sum_{s\in\mathcal{S}} z^p_s \le \lfloor S\epsilon \rfloor & \quad && \forall p \in \mathcal{P}, && \text{(17a)}\\
(\boldsymbol{x}^p, z^p_s) \in \mathcal{F}^{p,s} & \quad && \forall s \in \mathcal{S}, p \in \mathcal{P},\\
z^1_s \ge \frac{1}{P-1}\sum_{p=2}^{P} z^p_s & \quad && \forall s \in \mathcal{S}, && \text{(17b)}\\
z^p_s \in \{0,1\} & \quad && \forall s \in \mathcal{S}, p \in \mathcal{P}.
\end{aligned}$$

This model includes all the constraints and variables of (SAA-MCF^p) for each group and links the scenario variables through constraint (17b), where z_{sp} is the indicator variable associated with scenario $s \in \mathcal{S}$ and group $p \in \mathcal{P}$. Specifically, (17b) forces a scenario variable in the first group to be turn-on if the service requirements are unmet in any other group. Since the number of scenarios in the first group with unmet service conditions is restricted to be less than $\lfloor S\epsilon \rfloor$ (17a), we guarantee that the total number of scenarios with unmet service requirements across all subproblems is also bounded by that amount.

We develop a Lagrangian decomposition scheme based on (SAA-MCF-Joint) that dualizes (17b) to create sub-problems for each trip group. The resulting objective function is given by:

$$\sum_{p\in\mathcal{P}}\sum_{k\in\mathcal{K}}\sum_{(i,j)\in\mathcal{A}^p} \mathsf{c}_{ij} x^p_{ijk} + \sum_{s\in\mathcal{S}} \mu_s \left(-(P-1) z^1_s + \sum_{p=2}^{P} z^p_s \right),$$

where $\mu_s \ge 0$ for $s \in \mathcal{S}$ are the Lagrangian penalties associated with (17b). Thus, the resulting Lagrangian dual problem is

$$\max_{\boldsymbol{\mu}} \left\{ \sum_{p\in\mathcal{P}} \mathsf{L}^p(\boldsymbol{\mu}) : \boldsymbol{\mu} \in \mathbb{R}^S_+ \right\} \qquad \text{(LagrDual)}$$

where we have one Lagrangian subproblem for each group $p \in \mathcal{P}$ given by

$$\mathsf{L}^p(\boldsymbol{\mu}) = \min_{\boldsymbol{x},\boldsymbol{z}} \left\{ \sum_{k\in\mathcal{K}}\sum_{(i,j)\in\mathcal{A}^p} \mathsf{c}_{ij} x_{ijk} + \mathsf{C}^p \sum_{s\in\mathcal{S}} \mu_s z_s : \boldsymbol{x} \in \mathcal{X}^p, (16a)-(16b), \boldsymbol{z} \in \{0,1\}^S \right\}$$

with constant $\mathsf{C}^1 = -(P-1)$ and $\mathsf{C}^p = 1$ for all $p \in \mathcal{P} \setminus \{1\}$.

This decomposition allows us to find vehicle schedules with a small number of scenarios with unmet service requirements by solving the Lagrangian dual problem. However, this procedure might not necessarily enforce the CC in (SAA-MCF). In particular, model (SAA-MCF-Joint) enforces the

service requirements for each trip group (i.e., constraints in set $\mathcal{F}^{p,s}$), but the service requirements in (SAA-MCF) (i.e., $\mathcal{F}^{s}$) consider all trips together. Thus, there could be a scenario where the service requirements are met for all groups individually, but the service requirements for all trips together might be unmet. Our preliminary experiments show that this behavior could happen but usually leads to only a slight violation of the CC.

6.3. Solving the Lagrangian Dual Problem

There is a vast literature on methodologies to solve Lagrangian dual problems, such as sub-gradient and cutting plane method (Fisher 2004, Frangioni 2005). Our problem (LagrDual) has the particularity that each subproblem $\mathsf{L}^p(\boldsymbol{\mu})$ for $p \in \mathcal{P}$ is an NP-hard problem that can take a significant amount of time to solve due to the CC. Therefore, we opt for the bundle method (Lemaréchal 1975, Frangioni 2002) to solve (LagrDual) since it has a fast converge rate (i.e., $\mathcal{O}(\frac{1}{\epsilon^3})$ for a tolerance $\epsilon > 0$) when compared to other alternatives, and thus, requires fewer iterations to converge to the optimal solutions.

A bundle method is a variant of the cutting plane method, which adds a quadratic stabilizer to improve convergence. The procedure solves subproblems $\mathsf{L}^p(\hat{\boldsymbol{\mu}})$ for each $p \in \mathcal{P}$ for a given $\hat{\boldsymbol{\mu}} \geq 0$ and creates a joint subgradient associated to the optimal solution $(\hat{\boldsymbol{x}}^p, \hat{\boldsymbol{z}}^p)$ of each subproblem as:

$$g(\hat{\boldsymbol{\mu}})^\top = \left(-(P-1)\hat{z}_1^1 + \sum_{p=2}^{P} \hat{z}_1^p, \ \ldots \ , -(P-1)\hat{z}_S^p + \sum_{p=2}^{P} \hat{z}_S^p \right)$$

This subgradient is then used to create a cutting plane valid for $\sum_{p\in\mathcal{P}} \mathsf{L}^p(\cdot)$, which is added to a quadratic optimization problem that returns the next set of Lagrangian penalties. Specifically, we start with $\boldsymbol{\mu}_0 = (0, \ldots, 0)$ and in each iteration $\ell \geq 0$, we first solve the subproblems $\mathsf{L}^p(\boldsymbol{\mu}^\ell)$ for the current $\boldsymbol{\mu}^\ell$, create subgradient $g(\boldsymbol{\mu}^\ell)$, and add a cutting plane to the following quadratic problem that finds the new set of Lagrangian penalties:

$$\boldsymbol{\mu}^{\ell+1} \in \underset{\boldsymbol{\mu}\in\mathbb{R}^S_+}{\arg\max} \left\{ \theta + \frac{1}{2t}||\boldsymbol{\mu} - \boldsymbol{\mu}^\ell|| : \theta \leq \sum_{p\in\mathcal{P}} \mathsf{L}^p(\boldsymbol{\mu}^l) + g(\boldsymbol{\mu}^l)^\top(\boldsymbol{\mu} - \boldsymbol{\mu}^l), \ \forall l \in \{0, ..., \ell\}, \theta \in \mathbb{R} \right\}.$$

In this problem, variable θ over-approximates the optimal value of (LagrDual), that is, a valid dual bound for the problem. The objective function includes a quadratic stabilizer $\frac{1}{2t}||\boldsymbol{\mu} - \boldsymbol{\mu}^\ell||$ to improve converge, where $t < 1$ is a positive parameter that is updated in each iteration as suggested by Lemaréchal (1975). The procedure ends when we observe a small relative difference between the Lagrangian primal and dual bound (i.e., $\sum_{p\in\mathcal{P}} \mathsf{L}^p(\boldsymbol{\mu}^\ell)$ and the optimal value of θ for iteration ℓ, respectively) or if there is small relative difference between the primal bounds or dual bounds from one iteration to the next (we use a tolerance of 0.001). In addition, we define a maximum number of iterations (i.e., $\ell \leq 100$) due to the expensive computational cost of solving each iteration. However, our experiments show that in most cases the algorithm converges before this limit is achieved.

Our procedure also builds a solution for (SAA-MCF) in each iteration (if possible) and saves the best solution found so far. Specifically, we create a vehicle schedule using the solutions of each subproblem and adjust the depot assignments when necessary, as detailed in Section 6.1. We then evaluate the service requirements $\mathcal{F}^s$ for each scenario $s \in \mathcal{S}$ and check if constraint (4a) is satisfied or not. We first keep the solutions with the smallest violations of (4a) (i.e., fewer scenarios with violated service requirements). If a feasible solution is found, we then only keep the feasible schedules with the lowest cost. We return the best schedule found so far at the end of the procedure. By doing so, we can guarantee that the resulting schedule is either feasible with the smallest cost found so far or infeasible with the smallest violation of the CC.

7. Empirical Evaluation

We now present the numerical experiment results for the B&C scheme and the Lagrangian-based approach presented in Sections 4 and 6, respectively, to solve the CC-MDVSP. In what follows, we first provide details on the test instances and the experimental setup. We then present experimental results comparing the performances of different cut types for our B&C procedure and show the value of the CC variant when compared to the deterministic version of the problem used in practice. Finally, we evaluate the solution quality of our Lagrangian-based approach for large-scale instances and compare them to the solutions found by the deterministic version.

7.1. Experimental Setup

We generate random instances of the problem following the procedure described in (Carpaneto et al. 1989, Kulkarni et al. 2019) with slight modifications to incorporate the stochastic travel times, expressing, and group trips into routes (see Appendix B for further details). We consider instances with $I \in \{50, 60, 70, 80\}$ trips, 10 trips per route, and $K \in \{2, 3, 4\}$ depots. We generate five instances per each I and K configuration for a total of 60 randomly generated instances. All instances, unless specify otherwise, utilize $\epsilon = 0.05$, $\delta^{\texttt{trip}} = 0.9$, and $\delta^{\texttt{route}} = 0.8$ for the CC probability and the service requirements, respectively. We consider 750 scenarios to solve the SAA formulation of the problem and independently sample 2000 scenarios to evaluate the quality of the solutions. We preliminary tested with different values for $S \in \{500, 750, 1000\}$ and concluded that $S = 750$ was suitable for our experiments. Lastly, as commonly assumed in the literature (see, for example, Kulkarni et al. (2018), Ricard et al. (2024), we consider all travel times to be integer values, thus, we use $\epsilon^{\texttt{tol}} = 1$ when computing our C-MIS cuts.

The code for the decomposition algorithms is implemented in `C++` using solver IBM ILOG CPLEX 20.1 with callback functions. All experiments are run on a single thread with a 16GB memory limit using the University of Toronto SciNet server Niagara[2].

[2] See https://docs.scinet.utoronto.ca/index.php/Niagara_Quickstart for further server specifications.

7.2. Efficiency and Value of the B&C Decomposition Scheme

To evaluate our B&C decomposition scheme to solve (SAA-MCF), we compare the different cut generation procedures introduced in Section 5 and evaluate the impact of other algorithmic enhancements, such as the valid inequalities presented in Section 4.4. We also assess the quality of the return schedules and show that the solutions are more robust and have a lower cost than the ones obtained by solving the deterministic MDVSP with over-approximations of the traveling times, a practice commonly used by transit agencies (Kittelson, Quade, and Hunter-Zaworski 2003, Furth and Muller 2007). In these experiments, we use a time limit of 2 hours per instance.

Cut Comparison. We first compare the computational performance of the cut generation alternatives: MIS cuts (12) using the conflict refinement tool of CPLEX (i.e., MIS), the C-MIS cuts and its extended version (i.e., C-MIS and EC-MIS, respectively). Table 1 reports the number of problems solved to optimality, average optimality gap, solving time, and added cuts. Optimality gaps are the ones returned by the solver, and the average gap only considers the instances without optimality proof. The average time and added cuts columns only consider the cases where all alternatives found an optimal solution. Bold numbers correspond to the best values, that is, the largest values in a row for the first set of columns and the smallest values in the remaining set of columns.

Table 1 Cut generation comparison in B&C approach

		# Optimal			% Optimality Gap			Av. Time (sec)			Av. Added Cuts		
I	K	MIS	C-MIS	EC-MIS	MIS	C-MIS	EC-MIS	MIS	C-MIS	EC-MIS	MIS	C-MIS	EC-MIS
50	2	**5**	**5**	**5**	-	-	-	45.9	15.8	**12.3**	1687	**1207**	1254
	3	**5**	**5**	**5**	-	-	-	7.1	2.9	**1.7**	**378**	633	412
	4	**5**	**5**	**5**	-	-	-	32.1	**10.4**	13.1	894	804	**799**
60	2	4	4	4	0.28	0.18	**0.10**	48.5	**14.4**	14.6	1538	1388	**1090**
	3	**5**	**5**	**5**	-	-	-	806.4	821.8	**644.5**	3257	2980	**2652**
	4	**5**	**5**	**5**	-	-	-	530.6	458.7	**317.7**	2382	2508	**2300**
70	2	3	3	3	5.37	**0.69**	3.05	814.0	**256.5**	632.2	3797	**3589**	3859
	3	**5**	**5**	**5**	-	-	-	941.2	965.4	**558.3**	3609	3570	**2321**
	4	**3**	2	**3**	0.26	**0.17**	0.21	**1291.7**	1712.4	1569.4	**2456**	3201	2826
80	2	**3**	**3**	**3**	0.84	0.49	**0.42**	324.9	249.9	**76.3**	2661	2904	**1655**
	3	3	3	**4**	**4.53**	5.14	5.26	125.8	111.7	**77.7**	1578	1886	**1630**
	4	**3**	**3**	**3**	5.40	11.75	**4.88**	**693.9**	1293.2	1383.3	3057	**2941**	3246
Total/Av.		49	48	**50**	3.01	3.07	**2.25**	426.4	429.0	**363.4**	2006	1998	**1743**

Table 1 shows that EC-MIS generally performs best by solving the largest number of instances and obtaining the smallest optimality gaps and solving times. We observe that MIS and C-MIS have similar performances, which is mostly explained because the conflict refinement of CPLEX was run only after our greedy algorithm determined that a cut was needed. Also, it is interesting to see that MIS and C-MIS add similar number of cuts, which shows the benefit of adding multiple C-MIS per

violated subproblem and, thus, reduce the search space. Lastly, EC-MIS adds the smallest amount of cuts in general, which is expected due to the dominance relationship with C-MIS.

Enhancement Comparison. The above results considered enhancements to the basic B&C algorithm, in particular: (i) the valid inequalities in Section 4.4, and (ii) relaxing the integrality of the z variables in Section 5.5. Table 2 shows the importance of both enhancements in our procedure when using EC-MIS cuts (we observe similar results for other cut variants). Here, Nn avoids both enhancements, VI only uses the valid inequalities, ZC considers z as continuous variables, and Bo uses both enhancements. The meaning of the columns and bold numbers is the same as in Table 1. In particular, the average time columns only consider the instances that are solved to optimality across all four variants and, thus, differ from the values in Table 1.

Table 2 B&C enhancement comparison with cuts EC-MIS

		# Optimal				% Optimality Gap				Av. Time (sec)			
I	K	Nn	VI	ZC	Bo	Nn	VI	ZC	Bo	Nn	VI	ZC	Bo
50	2	5	5	5	5	-	-	-	-	12.6	33.7	**9.2**	12.3
	3	5	5	5	5	-	-	-	-	1.7	10.4	**1.5**	1.7
	4	5	5	5	5	-	-	-	-	8.8	23.1	**5.8**	13.1
60	2	4	4	4	4	0.27	0.25	0.23	**0.10**	67.4	48.7	18.9	**14.6**
	3	4	4	4	**5**	0.38	0.17	0.17	-	387.7	244.2	335.8	**165.2**
	4	**5**	4	**5**	**5**	-	0.05	-	-	608.2	532.9	324.1	**287.1**
70	2	**3**	2	**3**	**3**	5.68	5.24	3.06	**3.05**	18.1	140.2	**13.9**	23.3
	3	**5**	4	**5**	**5**	-	0.05	-	-	**59.4**	1839.0	70.3	68.6
	4	2	2	2	**3**	0.20	0.91	**0.15**	0.21	**1565.1**	1923.2	1584.2	1569.4
80	2	**3**	2	**3**	**3**	0.53	0.62	0.64	**0.42**	**4.0**	13.9	5.7	9.1
	3	3	3	3	**4**	**2.83**	10.56	4.97	5.26	37.7	180.7	**32.1**	77.7
	4	**3**	2	**3**	**3**	7.20	5.68	4.97	**4.88**	**39.1**	355.0	74.2	100.6
Total/Average		47	42	47	**50**	2.60	3.28	**2.17**	2.25	189.8	390.5	155.5	**140.8**

Table 2 shows that there is a clear advantage of using both enhancements to improve all metrics, and, surprisingly, this effect does not occur when we try each enhancement on its own. In particular, we see that VI has a negative effect in terms of run time and instances solved, mostly due to the large number of inequalities included (i.e., at most one per scenario and service requirement, for a total of $2000 \sim 5000$ inequalities) and, thus, enlarging the master problem. However, when combined with ZC, the effect of additional constraints is significantly decreased since we are reducing the branching factor of the problem by ignoring the integrality of z. We also try other enhancements presented in the literature for the deterministic MCF model (i.e., odd-cycle inequalities and variable fixing, as shown by Hadjar, Marcotte, and Soumis (2006) and Groiez et al. (2013)), but our results show that these alternatives have none or negative effects in the procedure, which might be explained by the different problem structure between the deterministic MDVSP and the CC-MDVSP.

Solution Quality. Lastly, we compare the quality of the solutions found with our best CC variant (i.e., CC, using EC-MIS and Bo) with two deterministic variants commonly used by practitioners (Kittelson, Quade, and Hunter-Zaworski 2003, Furth and Muller 2007): (i) a model that considers only average travel times (i.e., Mean), and (ii) a conservative approach that uses the 75 percentile to estimate travel times (i.e., 75Per). Table 3 shows the results where the first set of columns presents the average objective value found by each procedure, the second set is the relative difference between models CC and 75Per with respect to Mean (i.e., $\frac{X-\text{Mean}}{\text{Mean}}$ with $X \in \{\text{CC}, \text{75Per}\}$), and the third set presents the percentage of scenarios that satisfy the CC when evaluated over 2000 scenarios that were not considered during the optimization phase.

Table 3 Solution quality comparison with deterministic approaches

		Objective value			% Obj. Diff.		% Scenarios		
I	K	Mean	75Per	CC	75Per	CC	Mean	75Per	CC
50	2	161,110.4	172,332.4	161,250.4	7.0	0.1	83.9	99.4	94.4
	3	170,010.4	179,480.0	170,054.4	5.6	0.0	87.8	99.1	95.5
	4	168,266.8	179,713.2	168,357.6	6.8	0.1	87.2	99.3	95.1
60	2	217,675.6	239,556.4	217,939.7	10.1	0.1	82.2	98.2	94.5
	3	201,747.2	221,111.6	202,062.4	9.6	0.2	76.2	98.1	94.3
	4	225,984.0	241,715.2	226,221.5	7.0	0.1	86.1	98.2	95.0
70	2	225,623.2	250,116.4	229,758.8	10.9	1.8	74.3	97.7	93.7
	3	226,316.4	246,416.4	226,497.8	8.9	0.1	79.7	98.2	93.4
	4	232,159.2	256,424.4	232,975.2	10.5	0.4	71.3	97.5	93.2
80	2	277,649.6	300,261.2	278,568.3	8.1	0.3	76.1	97.3	93.0
	3	259,916.4	280,397.6	264,609.8	7.9	1.8	74.3	97.5	93.8
	4	245,881.2	266,018.4	251,442.8	8.2	2.3	72.4	94.2	93.4
Average:					8.4	0.6	79.3	97.9	94.1

We observe that CC obtains a very low objective value, usually no more than 1% over Mean, and has a percentage of scenarios that satisfy the CC close to the target number of 95%. In contrast, Mean has the lowest cost but only fulfills the OTP conditions in 79% of the scenario on average, and 75Per significantly increases the objective value and usually returns schedules that are too conservative. Therefore, our CC alternative is the best in terms of finding low-cost schedules that can achieve OTP in the range of the number of scenarios desired (i.e., 95%).

7.3. Performance of the Lagrangian-based Approach

To test the Lagrangian-based approach for large-scale instances, we use instances with $I \in \{200, 300\}$, and all the other parameters remain the same. These experiments consider a total time limit of 24 hours, with a 10-minute time limit per subproblem and a maximum of 100 Lagrangian

dual iterations. None of our experiments reached the limits for the total time and Lagrangian iterations. Lastly, we solve each subproblem using the best cut generation and enhancement combination shown in the previous section (i.e., EC-MIS with Bo).

We first evaluate the quality of the solutions for different group sizes where each group $p \in \mathcal{P}$ had a maximum of $I_p \in \{20, 30, 40, 50\}$ trips, (i.e., 4 to 15 groups). As explained in Section 6.1, we solve the deterministic MDVSP (in this case 75Per) to group the trips and ensure that each group has many trips that can be scheduled together. Preliminary results using Mean and random alternatives show significantly worse performance. Table 4 shows the results for the Lagrangian-based approach for the four group sizes (i.e., La20, La30, La40, and La50) when solving for 750 scenarios. The first set of columns shows the average objective value. The second set of columns presents the percentage of scenarios that fulfill the service requirements. The last set of columns shows the total run time in hours. Bold numbers correspond to the lowest values for column sets 1 and 3 in each row, and values between $95 \pm 1\%$ for column set 2.

Table 4 Performance comparison for Lagrangian-based approach with different group sizes

		Objective Value ($\times 10^3$)				% Scenarios OTP				Time (hours)			
I	K	La20	La30	La40	La50	La20	La30	La40	La50	La20	La30	La40	La50
200	2	658.7	657.9	647.8	**639.3**	**95.5**	**95.7**	**95.4**	**95.0**	**1.12**	3.45	1.61	2.72
	3	641.5	635.9	633.2	**623.4**	96.1	**95.7**	**95.9**	**94.6**	**1.40**	4.37	5.22	6.45
	4	638.2	637.0	631.1	**622.5**	96.1	**95.7**	**95.5**	**95.5**	**1.32**	2.88	4.63	6.84
300	2	959.4	948.7	933.8	**922.1**	**94.6**	**94.9**	92.7	91.4	**2.31**	5.45	10.59	16.21
	3	973.2	954.0	951.6	**932.8**	**95.8**	93.9	93.4	93.0	**5.94**	12.07	13.53	19.40
	4	968.4	957.0	943.7	**936.7**	**94.9**	92.9	92.8	92.7	**9.01**	14.98	21.53	16.51
Average		806.6	790.2	779.5	**774.5**	**95.5**	**94.3**	93.7	92.4	**3.52**	9.52	11.35	10.66

We observe intuitive results for instances with $I = 200$, in which La50 achieves the lowest cost schedule satisfying the CC to a reasonable degree (i.e., $95 \pm 0.5\%$), but taking significantly more time than smaller group alternatives. In contrast, when $I = 300$, only La20 finds schedules that fulfill the CC. We believe that the Lagrangian dual problem becomes harder to solve when the number of groups and their size increase; that is, it is harder to find feasible solutions that synchronize the CC across all subproblems. We also note that most of the subproblems are solved to optimality for La20 and La30 (i.e., over 95% of them), while around 75% are optimally solved for La50, which could also explain the behaviour for instances with $I = 300$. However, the optimality gaps are usually small (i.e., around 1-5%) for sub-optimal subproblem solutions.

Table 5 compares the results obtained by the two best Lagrangian-based alternatives in terms of satisfying the CC with the two deterministic alternatives, Mean and 75Per, when evaluated over 2000 scenarios. The columns and bold numbers have the same meaning as in Table 3. First, we see

Table 5 Solution quality comparison for large-scale instances

		Obj.Val	% Obj. Diff.			% Scenarios			
I	K	Mean	75Per	La20	La30	Mean	75Per	La20	La30
200	2	598.6	11.5	12.2	**10.0**	31.3	91.6	**94.5**	**94.4**
	3	575.1	**11.5**	13.2	**11.5**	31.2	91.0	**94.3**	93.7
	4	573.1	12.0	12.5	**11.4**	27.6	93.8	**94.8**	**95.2**
300	2	843.2	**9.0**	16.6	13.8	8.3	80.3	**94.6**	92.5
	3	858.4	**9.0**	15.6	13.4	9.1	77.7	92.9	93.4
	4	853.0	**9.2**	17.0	13.5	10.0	80.6	**94.4**	93.2
Average		716.9	10.4	14.5	12.3	19.6	85.8	**94.2**	93.7

that 75Per, La20, and La30 have similar objective values, all between 10% and 15% increased when compared to Mean. However, there is a significant difference when we compare the percentage of scenarios that satisfy the service requirements. The percentage significantly decreases for Mean and 75Per as the number of trips increases, which illustrates the lack of reliability guarantees for these approaches as the instances become larger (and, thus, more realistic). In contrast, La20 obtains, on average, a percentage close to our 95% target, which the percentage is closer to 94% for La30. Therefore, we can infer that the Lagrangian-based approach can indeed find feasible (or close to feasible) solutions for our CC-MDVSP, especially when the subproblems are small.

8. Conclusion

This paper addresses the multi-depot vehicle scheduling problem (MDVSP) by incorporating travel time uncertainty and service reliability into the model. We propose a novel chance-constrained programming model to guarantee on-time performance (OTP) and ensure fair service distribution across different routes. Our study presents two optimization methods: an exact approach using a branch-and-cut procedure and a heuristic method based on Lagrangian decomposition. Our experimental results demonstrate the effectiveness of our proposed model in achieving reliable OTP. For the exact approach, the proposed cut families and enhancement ideas yield significant improvements in computational efficiency and solution quality, while the optimization-based heuristic method provides high-quality practical solutions for larger instances. Our methodological ideas have the potential to be adopted for some other chance-constrained programs with integer recourse.

For MDVSP, our solutions can significantly reduce the number of buses required while maintaining high service reliability. This translates to substantial cost savings for transit agencies, as minimizing the fleet size without compromising OTP is a critical objective. Additionally, our solutions show that it is possible to maintain reliability without resorting to large slack times, which are often added to schedules in practice to account for travel time variability, thereby improving transit speed and overall service attractiveness. Equity is another important consideration in

our approach. Our model ensures that service improvements are distributed fairly across different routes, supporting the recent equity efforts in transit planning. This proactive consideration of fairness sets our work apart from traditional demand-centric models. Our findings also highlight the potential benefits of the multi-depot scheduling approach in the stochastic setting to achieve more efficient and reliable transit operations, further underscoring the value of our methodology. All in all, addressing some key priorities of transit agencies, namely the goal of delivering a cost-effective, reliable, and equitable service, our work sets a new benchmark in vehicle scheduling and paves the way for future advancements in public transportation planning.

Future work could explore extending our model to include electric vehicles and other sustainability considerations, further enhancing the applicability of our approach to modern transit systems. Additionally, investigating real-time scheduling adjustments in response to dynamic changes in travel times could further improve service reliability and operational efficiency. Last but not least, the combination of stochastic MDVSP with other planning problems, such as timetabling, would be of high interest.

References

Ahmed S, Xie W, 2018 *Relaxations and approximations of chance constraints under finite distributions. Mathematical Programming* 170(1):43–65.

Bertossi AA, Carraresi P, Gallo G, 1987 *On some matching problems arising in vehicle scheduling models. Networks* 17(3):271–281.

Bunte S, Kliewer N, 2009 *An overview on vehicle scheduling models. Public Transport* 1(4):299–317.

Canessa G, Gallego JA, Ntaimo L, Pagnoncelli BK, 2019 *An algorithm for binary linear chance-constrained problems using IIS. Computational Optimization and Applications* 72(3):589–608.

Carpaneto G, Dell'Amico M, Fischetti M, Toth P, 1989 *A branch and bound algorithm for the multiple depot vehicle scheduling problem. Networks* 19(5):531–548.

Chao Z, Xiaohong C, 2013 *Optimizing battery electric bus transit vehicle scheduling with battery exchanging: Model and case study. Procedia-Social and Behavioral Sciences* 96:2725–2736.

Charnes A, Cooper WW, 1959 *Chance-constrained programming. Management Science* 6(1):73–79.

Deng Y, Shen S, 2016 *Decomposition algorithms for optimizing multi-server appointment scheduling with chance constraints. Mathematical Programming* 157(1):245–276.

Desfontaines L, Desaulniers G, 2018 *Multiple depot vehicle scheduling with controlled trip shifting. Transportation Research Part B: Methodological* 113:34–53.

Desrosiers J, Dumas Y, Solomon MM, Soumis F, 1995 *Time constrained routing and scheduling. Handbooks in Operations Research and Management Science* 8:35–139.

Dinh T, Fukasawa R, Luedtke J, 2018 *Exact algorithms for the chance-constrained vehicle routing problem. Mathematical Programming* 172(1):105–138.

Eberlein XJ, 1997 *Real-time control strategies in transit operations: Models and analysis. Transportation Research Part A* 1(31):69–70.

European Committee for Standardisation, 2002 *Transportation-logistics and services-public passenger transport-service quality definition, targeting and measurement.*

Fischetti M, Lodi A, Martello S, Toth P, 2001 *A polyhedral approach to simplified crew scheduling and vehicle scheduling problems. Management Science* 47(6):833–850.

Fisher ML, 2004 *The Lagrangian relaxation method for solving integer programming problems. Management Science* 50(12):1861–1871.

Frangioni A, 2002 *Generalized bundle methods. SIAM Journal on Optimization* 13(1):117–156.

Frangioni A, 2005 *About Lagrangian methods in integer optimization. Annals of Operations Research* 139:163–193.

Furth PG, Muller TH, 2007 *Service reliability and optimal running time schedules. Transportation Research Record* 2034(1):55–61.

Fusco G, Alessandrini A, Colombaroni C, Valentini MP, 2013 *A model for transit design with choice of electric charging system. Procedia-Social and Behavioral Sciences* 87:234–249.

Groiez M, Desaulniers G, Hadjar A, Marcotte O, 2013 *Separating valid odd-cycle and odd-set inequalities for the multiple depot vehicle scheduling problem. EURO Journal on Computational Optimization* 1:283–312.

Guedes PC, Borenstein D, 2018 *Real-time multi-depot vehicle type rescheduling problem. Transportation Research Part B: Methodological* 108:217–234.

Guenthner RP, Hamat K, 1988 *Distribution of bus transit on-time performance. Transportation Research Record* 1202:1–8.

Hadjar A, Marcotte O, Soumis F, 2006 *A branch-and-cut algorithm for the multiple depot vehicle scheduling problem. Operations Research* 54(1):130–149.

He F, Yang J, Li M, 2018 *Vehicle scheduling under stochastic trip times: an approximate dynamic programming approach. Transportation Research Part C: Emerging Technologies* 96:144–159.

Hooker JN, Ottosson G, 2003 *Logic-based Benders decomposition. Mathematical Programming* 96(1):33–60.

Ibarra-Rojas OJ, Delgado F, Giesen R, Muñoz JC, 2015 *Planning, operation, and control of bus transport systems: A literature review. Transportation Research Part B: Methodological* 77:38–75.

Kittelson P, Quade K, Hunter-Zaworski K, 2003 *Transit capacity and quality of service manual. Transportation Research Board, National Academy Press, Washington, DC* .

Kliewer N, Mellouli T, Suhl L, 2006 *A time–space network based exact optimization model for multi-depot bus scheduling. European Journal of Operational Research* 175(3):1616–1627.

Küçükyavuz S, 2012 *On mixing sets arising in chance-constrained programming. Mathematical Programming* 132(1):31–56.

Küçükyavuz S, Jiang R, 2021 *Chance-constrained optimization: A review of mixed-integer conic formulations and applications. arXiv preprint arXiv:2101.08746* 2.

Kulkarni S, Krishnamoorthy M, Ranade A, Ernst AT, Patil R, 2018 *A new formulation and a column generation-based heuristic for the multiple depot vehicle scheduling problem. Transportation Research Part B: Methodological* 118:457–487.

Kulkarni S, Krishnamoorthy M, Ranade A, Ernst AT, Patil R, 2019 *A benchmark dataset for the multiple depot vehicle scheduling problem. Data in Brief* 22:484–487.

Lemaréchal C, 1975 *An extension of davidon methods to non differentiable problems*, 95–109 (Berlin, Heidelberg: Springer Berlin Heidelberg).

Li JQ, Mirchandani PB, Borenstein D, 2007 *The vehicle rescheduling problem: Model and algorithms. Networks: An International Journal* 50(3):211–229.

Liu Zg, Shen Js, 2007 *Regional bus operation bi-level programming model integrating timetabling and vehicle scheduling. Systems Engineering-Theory & Practice* 27(11):135–141.

Luedtke J, 2014 *A branch-and-cut decomposition algorithm for solving chance-constrained mathematical programs with finite support. Mathematical Programming* 146(1):219–244.

Luedtke J, Ahmed S, 2008 *A sample approximation approach for optimization with probabilistic constraints. SIAM Journal on Optimization* 19(2):674–699.

Luedtke J, Ahmed S, Nemhauser GL, 2010 *An integer programming approach for linear programs with probabilistic constraints. Mathematical Programming* 122(2):247–272.

Mesquita M, Moz M, Paias A, Pato M, 2013 *A decomposition approach for the integrated vehicle-crew-roster problem with days-off pattern. European Journal of Operational Research* 229(2):318–331.

Naumann M, Suhl L, Kramkowski S, 2011 *A stochastic programming approach for robust vehicle scheduling in public bus transport. Procedia-Social and Behavioral Sciences* 20:826–835.

Pagnoncelli BK, Ahmed S, Shapiro A, 2009 *Sample average approximation method for chance constrained programming: theory and applications. Journal of Optimization Theory and Applications* 142(2):399–416.

Pepin AS, Desaulniers G, Hertz A, Huisman D, 2009 *A comparison of five heuristics for the multiple depot vehicle scheduling problem. Journal of Scheduling* 12(1):17.

Rahman MM, Wirasinghe S, Kattan L, 2018 *Analysis of bus travel time distributions for varying horizons and real-time applications. Transportation Research Part C: Emerging Technologies* 86:453–466.

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

Ricard L, Desaulniers G, Lodi A, Rousseau LM, 2022 *Predicting the probability distribution of bus travel time to measure the reliability of public transport services. Transportation Research Part C: Emerging Technologies* 138:103619.

Ricard L, Desaulniers G, Lodi A, Rousseau LM, 2024 *Increasing schedule reliability in the multiple depot vehicle scheduling problem with stochastic travel time. Omega* 127:103100.

Ruszczyński A, 2002 *Probabilistic programming with discrete distributions and precedence constrained knapsack polyhedra. Mathematical Programming* 93(2):195–215.

Shen Y, Xu J, Li J, 2016 *A probabilistic model for vehicle scheduling based on stochastic trip times. Transportation Research Part B: Methodological* 85:19–31.

Shen Y, Xu J, Wu X, 2017 *Vehicle scheduling based on variable trip times with expected on-time performance. International Transactions in Operational Research* 24(1-2):99–113.

Song Y, Luedtke JR, Küçükyavuz S, 2014 *Chance-constrained binary packing problems. INFORMS Journal on Computing* 26(4):735–747.

Tang X, Lin X, He F, 2019 *Robust scheduling strategies of electric buses under stochastic traffic conditions. Transportation Research Part C: Emerging Technologies* 105:163–182.

The Toronto Transit Commission, 2024 *Daily customer service report.* https://www.ttc.ca/customer-service/daily-customer-service-report, accessed: 2024-05-15.

TransitCenter, 2018 *Your bus is on time. What does that even mean?* https://transitcenter.org/bus-time-even-mean/, accessed: 2024-05-15.

Transport for London, 2024 *Route results for London bus services - Quarter 04 23/24*. https://bus.data.tfl.gov.uk/boroughreports/current-quarter.pdf, accessed: 2024-05-15.

Uçar E, Birbil Şİ, Muter İ, 2017 *Managing disruptions in the multi-depot vehicle scheduling problem. Transportation Research Part B: Methodological* 105:249–269.

Visentini MS, Borenstein D, Li JQ, Mirchandani PB, 2014 *Review of real-time vehicle schedule recovery methods in transportation services. Journal of Scheduling* 17(6):541–567.

Wu HH, Kucukyavuz S, 2019 *Probabilistic partial set covering with an oracle for chance constraints. SIAM Journal on Optimization* 29(1):690–718.

Appendix A: Proofs

A.1. Proof of Proposition 1

PROPOSITION 1. *For a master solution $\hat{\boldsymbol{x}}$ and scenario $s \in \mathcal{S}$, Algorithm 1 returns an optimal solution for* (SAA-Sub$(s, \hat{\boldsymbol{x}})$)*, where $\boldsymbol{y}^*$ is the earliest start times of each timetabled trip.*

Proof. First, by construction, a solution of Algorithm 1 is feasible for (SAA-Sub$(s, \hat{\boldsymbol{x}})$) since it satisfies all the constraints in $\mathcal{F}^s$. To prove that the solution is optimal, we first set the expressing variables to be as large as possible (i.e., $\boldsymbol{u}^* = \mathtt{e}$), which is feasible and allows us to set $\boldsymbol{y}^*$ as the earliest start time for each trip. We use the assumption that the first trip of any bus can start as early as possible, that is, (5d) is redundant and (5f) can be tight for the first trip in each bus. We then compute the start time of each trip considering that it will start as early as possible to avoid unnecessary delays, that is the maximum bounds between inequalities (5f) and (5c), as expressed in (7). Since we iterate the trips in order, (7) is guaranteed to give us the earliest start time of each trip $i \in \mathcal{I}$ that satisfies all the constraints of the problem. We then set $v_i^* = 0$ for all trips $i \in \mathcal{I}$ that present a delay (i.e., $y_i^* > \mathtt{s}_i + \mathtt{ub}$), and, accordingly, z^* takes the smallest possible value given $\boldsymbol{v}^*$. Notice that the solution is indeed optimal because any $\boldsymbol{y}^*$ that is not set to its earliest start time could potentially increase the number of $\boldsymbol{v}^*$ set to 0 and, thus, force z^* to be equal to one. □

A.2. Proof of Proposition 3

PROPOSITION 3. *Consider a vehicle schedule $\mathcal{B}(\hat{x})$ and a scenario $s \in \mathcal{S}$ such that at least one of the service requirements is violated. Then, for any such constraint $\mathtt{con} \in \{(5a), \{(5b)\}_{r \in \mathcal{R}}\}$, Algorithm 2 builds a C-MIS $\mathcal{V}^{\text{C-MIS}}$.*

Proof. For simplicity, we illustrate the proof with $\mathtt{con} = (5a)$ that forces $z = 1$. The proof is analogous if we choose any violated inequality in (5b). We first argue that $\mathcal{V}^{\text{C-MIS}}$ is an IS for $\mathcal{B}(\hat{x})$ and scenario s. By construction, Algorithm 2 chooses a subsequence of predecessors for every $i \in \mathcal{D}^{\text{C-MIS}}$ such that even if the first trip in such subsequence starts as early as possible, trip i will be delayed. Thus, we can guarantee that the chosen subsequences in $\mathcal{V}^{\text{C-MIS}}$ are sufficient to ensure that the trips in $\mathcal{D}^{\text{C-MIS}}$ are delayed and, thus, the service requirement is unfulfilled.

To prove that $\mathcal{V}^{\text{C-MIS}}$ is minimal, we first note that Algorithm 2 explains the delays of trip in $i \in \mathcal{D}$ if and only if $i \in \mathcal{D}^{\text{C-MIS}}$. By design of Algorithm 2, we explain the delays of all trips in $\mathcal{D}^{\text{C-MIS}}$. On the other hand, the predecessor condition of *Step (i)* guarantees that no other delayed trip $j \in \mathcal{D} \setminus \mathcal{D}^{\text{C-MIS}}$ appears in the subsequences chosen by Algorithm 2 because j cannot be a predecessor of any $i \in \mathcal{D}^{\text{C-MIS}}$ and not belong to $\mathcal{D}^{\text{C-MIS}}$. Thus, by construction, there cannot be a trip $j \in \mathcal{D} \setminus \mathcal{D}^{\text{C-MIS}}$ that is also in a pair trip of $\mathcal{V}^{\text{C-MIS}}$.

Second, removing any trip pair in $\mathcal{V}^{\text{C-MIS}}$ breaks the subsequence of predecessors that explain the delay of trips in $\mathcal{D}^{\text{C-MIS}}$. For every $i \in \mathcal{D}^{\text{C-MIS}}$, Algorithm 2 constructs the minimal subsequence of trips that explain the delay of i, so removing any pair in such subsequence will lead to a smaller subsequence where i is not delayed. Since the size of $\mathcal{D}^{\text{C-MIS}}$ is also minimal, having trips in $\mathcal{D}^{\text{C-MIS}}$ that can be on-time will satisfy con, and, thus, make $\mathcal{V}^{\text{C-MIS}}$ not an IS for the restricted subproblem with con as the only service requirement. □

A.3. Primal and Dual Model for Strengthen LP

We now present the LP model with the enhancements described in Section 5.4 to generate Benders cuts, that is, model SAA-Sub-LP($s, \hat{\boldsymbol{x}}$). This model is valid for a given master solution $\hat{\boldsymbol{x}}$ and utilizes the optimal greedy solution $(z, \boldsymbol{y}*, \boldsymbol{v}^*, \boldsymbol{u}^*)$ from Algorithm 1 to close the integrality gap. Recall that the model includes inequality (18c) to enforce the integrality of z and considers $\mathtt{M}_i^{\mathtt{OTP}} = y_i^* - \mathtt{s}_i$ for all $i \in \mathcal{I}$ to force integrality of $\boldsymbol{v}$. To ease exposition, we do not replace the proposed values of $\mathtt{M}_{ij}^{\mathtt{start}}$ for all $(i,j) \in \mathcal{C}$ here and they delay their replacement for the proof in the following section. Also, we assign $u_i = \mathtt{e}_i$ for all $i \in \mathcal{I}$, which is optimal for any given solution. Then, the LP in standard form is:

$$\min z$$

$$\text{s.t.} \sum_{i \in \mathcal{I}} v_i + \mathtt{f}^{\mathtt{trip}} z \geq \mathtt{f}^{\mathtt{trip}}, \tag{18a}$$

$$\sum_{i \in \mathcal{I}(r)} v_i + \mathtt{f}_r^{\mathtt{route}} z \geq \mathtt{f}_r^{\mathtt{route}}, \qquad \forall r \in \mathcal{R}, \tag{18b}$$

$$z + \sum_{i \in \mathcal{D}^{\mathtt{C\text{-}MIS}}} v_i \geq 1, \tag{18c}$$

$$y_i - y_j \geq \mathtt{d}_j^s + \mathtt{t}_{ji}^s - \mathtt{e}_j - \mathtt{M}_{ji}^{\mathtt{start}} \left(1 - \sum_{k \in \mathcal{K}} x_{jik}\right), \qquad \forall (j,i) \in \mathcal{C}, \tag{18d}$$

$$y_k \geq \mathtt{t}_{ki}^s - \mathtt{M}_{ki}^{\mathtt{start}} (1 - x_{kik}), \qquad \forall i \in \mathcal{I}, k \in \mathcal{K}, \tag{18e}$$

$$v_i(\mathtt{ub} - y_i^* + \mathtt{s}_i) - y_i \geq -y_i^*, \qquad \forall i \in \mathcal{I}, \tag{18f}$$

$$y_i \geq \mathtt{s}_i - \mathtt{lb}, \qquad \forall i \in \mathcal{I}, \tag{18g}$$

$$z, v_i, y_i \geq 0, \qquad \forall i \in \mathcal{I}. \tag{18h}$$

The dual model is detailed in what follows. Variables $\sigma^{\mathtt{trip}} \geq 0$, $\sigma_r^{\mathtt{route}} \geq 0$ for $r \in \mathcal{R}$, and $\sigma^{\mathtt{MIS}} \geq 0$ are the dual variables associated with constraints (18a)-(18c), respectively. Similarly, $\alpha_{ji} \geq 0$ for $(j,i) \in \mathcal{C}$ and $\alpha_{ki} \geq 0$ for all $i \in \mathcal{I}, k \in \mathcal{K}$ are the dual variable associated with constraints (18d) and (18e), respectively. Lastly, dual variables $\pi_i \geq 0$ for all $i \in \mathcal{I}$ are associated with (18f) and $\beta_i \geq 0$ for all $i \in \mathcal{I}$ are associated with (18g).

$$\begin{aligned}\max\ & \sigma^{\mathtt{trip}} \mathtt{f}^{\mathtt{trip}} + \sum_{r \in \mathcal{R}} \sigma^{\mathtt{route}} \mathtt{f}_r^{\mathtt{route}} + \sigma^{\mathtt{MIS}} + \sum_{(j,i) \in \mathcal{C}} \alpha_{ji} \left(\mathtt{d}_j^s + \mathtt{t}_{ji}^s - \mathtt{e}_i - \mathtt{M}_{ji}^{\mathtt{start}} \left(1 - \sum_{k \in \mathcal{K}} \hat{x}_{jik}\right)\right) \\ & + \sum_{i \in \mathcal{I}} \sum_{k \in \mathcal{K}} \alpha_{ki} \left(\mathtt{t}_{ki}^s - \mathtt{M}_{ki}^{\mathtt{start}} (1 - \hat{x}_{kik})\right) - \sum_{i \in \mathcal{I}} \pi_i y_i^* + \sum_{i \in \mathcal{I}} \beta_i (\mathtt{s}_i - \mathtt{lb})\end{aligned}$$

$$\text{s.t.} \sum_{(j,i) \in \mathcal{C}} \alpha_{ji} - \sum_{(i,j) \in \mathcal{C}} \alpha_{ij} + \sum_{k \in \mathcal{K}} \alpha_{ki} - \pi_i + \beta_i \geq 0, \qquad \forall i \in \mathcal{I}, \tag{19a}$$

$$\sigma^{\mathtt{trip}} + \sigma_r^{\mathtt{route}} + \pi_i(\mathtt{ub} - y_i^* + \mathtt{s}_i) \geq 0, \qquad \forall i \in \mathcal{I} \setminus \mathcal{D}^{\mathtt{C\text{-}MIS}}, r \in \mathcal{R} : i \in \mathcal{I}(r), \tag{19b}$$

$$\sigma^{\mathtt{trip}} + \sigma_r^{\mathtt{route}} + \sigma^{\mathtt{MIS}} + \pi_i(\mathtt{ub} - y_i^* + \mathtt{s}_i) \geq 0, \qquad \forall i \in \mathcal{D}^{\mathtt{C\text{-}MIS}}, r \in \mathcal{R} : i \in \mathcal{I}(r), \tag{19c}$$

$$\mathtt{f}^{\mathtt{trip}} \sigma^{\mathtt{trip}} + \sum_{r \in \mathcal{R}} \mathtt{f}_r^{\mathtt{route}} \sigma^{\mathtt{route}} + \sigma^{\mathtt{MIS}} \geq 1, \tag{19d}$$

$$\alpha_{ji} \geq 0, \qquad \forall (j,i) \in \mathcal{C},$$

$$\sigma_i^{\mathtt{trip}}, \pi_i, \beta_i \geq 0, \qquad \forall i \in \mathcal{I},$$

$$\sigma_r^{\mathtt{route}} \geq 0, \qquad \forall r \in \mathcal{R},$$

$$\sigma^{\mathtt{MIS}} \geq 0.$$

A.4. Proof of Proposition 5

PROPOSITION 5. *Consider the master solution $\hat{\boldsymbol{x}}$ that violates one of the service requirements $\texttt{con} \in \{(5a), \{(5b)\}_{r\in\mathcal{R}}\}$ for a scenario $s \in \mathcal{S}$, and a $\mathcal{D}^{\texttt{C-MIS}}$ constructed using $\texttt{con}$. Then, there exists a dual optimal solution of (SAA-Sub-LP$(s, \hat{\boldsymbol{x}})$) such that (15) and the C-MIS cut (12) for the corresponding $\mathcal{V}^{\texttt{C-MIS}}$ are identical.*

Proof. In what follows, we use "hat" notation to refer to specific values of the dual variables (e.g., $\hat{\sigma}^{\texttt{MIS}}$ is a specific value assigned to $\sigma^{\texttt{MIS}}$). Recall that we are considering a master solution $\hat{\boldsymbol{x}}$ that violates one of the service requirements $\texttt{con} \in \{(5a), \{(5b)\}_{r\in\mathcal{R}}\}$ for a scenario $s \in \mathcal{S}$, and a $\mathcal{D}^{\texttt{C-MIS}}$ constructed using $\texttt{con}$.

We start the proof by justifying the structure of the Bender's cut (15). First, recall that constraints (18e) are redundant and just appear in the model for completeness (see Section 3 for further details). Thus, we can omit them from our analysis or, equivalently, set $\hat{\alpha}_{ki} = 0$ for all $i \in \mathcal{I}$, $k \in \mathcal{K}$. Then, the optimal objective function value of the dual can be written as follows,

$$\sum_{(i,j)\in\mathcal{C}} \hat{\alpha}_{ij} \mathtt{M}_{ij}^{\texttt{start}} \sum_{k\in\mathcal{K}} \hat{x}_{ijk} + C = 1,$$

where C is constant concerning all other dual variables. Note that this value is equal to 1 since $\hat{\boldsymbol{x}}$ violates one of the service requirements in scenario s. Then, the Bender's cut can be written as follows for master variables $\boldsymbol{x}$ and z_s.

$$z_s \geq \sum_{(i,j)\in\mathcal{C}} \hat{\alpha}_{ij} \mathtt{M}_{ij}^{\texttt{start}} \sum_{k\in\mathcal{K}} x_{ijk} + C = \sum_{(i,j)\in\mathcal{C}} \hat{\alpha}_{ij} \mathtt{M}_{ij}^{\texttt{start}} \sum_{k\in\mathcal{K}} x_{ijk} + 1 - \sum_{(i,j)\in\mathcal{C}} \hat{\alpha}_{ij} \mathtt{M}_{ij}^{\texttt{start}} \sum_{k\in\mathcal{K}} \hat{x}_{ijk}.$$

By ordering the terms in the cut, we get (15), that is,

$$\sum_{(i,j)\in\mathcal{C}} \hat{\alpha}_{ij} \mathtt{M}_{ij}^{\texttt{start}} \sum_{k\in\mathcal{K}} x_{ijk} \leq \sum_{(i,j)\in\mathcal{C}} \hat{\alpha}_{ij} \mathtt{M}_{ij}^{\texttt{start}} \sum_{k\in\mathcal{K}} \hat{x}_{ijk} - 1 + z_s,$$

We now show how to create an optimal dual solution that leads to an C-MIS cut (12) with the corresponding $\mathcal{V}^{\texttt{C-MIS}}$. We consider a master solution $\hat{\boldsymbol{x}}$ that violates the service requirements $\texttt{con} \in \{(5a), \{(5b)\}_{r\in\mathcal{R}}\}$ for a scenario $s \in \mathcal{S}$, the optimal greedy solution $(z, \boldsymbol{y}^*, \boldsymbol{v}^*, \boldsymbol{u}^*)$ from Algorithm 1, a set of delay trips $\mathcal{D}^{\texttt{C-MIS}}$, and the corresponding $\mathcal{V}^{\texttt{C-MIS}}$.

First, recall that we are considering the following big-M coefficients, which are valid for our analysis since they preserve the optimal solution $(z, \boldsymbol{y}^*, \boldsymbol{v}^*, \boldsymbol{u}^*)$.

$$\mathtt{M}_{j,i}^{\texttt{start}} = \begin{cases} y_i^*, & \forall (j,i) \in \mathcal{V}, \\ \text{large number}, & \text{otherwise.} \end{cases}$$

To obtain the desired C-MIS cut, we will show that it is possible to create an optimal dual solution with the following dual variable values:

$$\hat{\alpha}_{ji} = \begin{cases} \dfrac{1}{y_i^*}, & (j,i) \in \mathcal{V}^{\texttt{C-MIS}}, \\ 0, & (j,i) \in \mathcal{C} \setminus \mathcal{V}^{\texttt{C-MIS}}. \end{cases}, \quad \forall (j,i) \in \mathcal{C}$$

Since $y_i > 0$ for all $i \in \mathcal{I}$ due to constraint (18g) and our problem assumptions (see Section 3), values $\hat{\alpha}_{ji}$ for all $(j,i) \in \mathcal{C}$ are well-defined. Also, note that if we replace these dual variables assignments and the big-M

values in (15), we get the desired C-MIS cut. Therefore, the only missing piece is to show that values $\hat{\alpha}_{ji}$ for all $(j,i) \in \mathcal{C}$ lead to an optimal dual solution.

First, we use the complementary slackness conditions to set the values of other dual variables. Since $y_i > 0$ for all $i \in \mathcal{I}$, dual constraint (19a) has to be strictly equal to zero. Consider set $\mathcal{I}^{\text{C-MIS}}$ to be all trips indices that are involved in the C-MIS $\mathcal{V}^{\text{C-MIS}}$, that is, $\mathcal{I}^{\text{C-MIS}} = \{i \in \mathcal{I} : \exists\, j \text{ such that } (j,i) \in \mathcal{V}^{\text{C-MIS}} \text{ or } (i,j) \in \mathcal{V}^{\text{C-MIS}}\}$. Then, since we fixed $\hat{\alpha}_{ji} = 0$ for all $(j,i) \notin \mathcal{V}^{\text{C-MIS}}$, we have that $\pi_i = \beta_i$ for all $i \notin \mathcal{I}^{\text{C-MIS}}$ in order to satisfy (19a), and we set $\hat{\pi}_i = \hat{\beta}_i = 0$ in our constructed dual solution. Also, we set $\hat{\sigma}^{\texttt{trip}} = 0$ and $\hat{\sigma}^{\texttt{route}}_r = 0$ for all $r \in \mathcal{R}$. Lastly, since $z > 0$ for any optimal solution due to (18c), constraint (19d) has to be strictly equal to one and, thus, $\hat{\sigma}^{\text{MIS}} = 1$.

Replacing all the variable assignments into our dual problem and enforcing equality to (19a), we have the following simplified dual problem. To ease notation, we replace all variable assignments previously described except $\hat{\alpha}_{ji}$ for $(j,i) \in \mathcal{V}^{\text{C-MIS}}$. We also replace $\mathtt{d}^s_j + \mathtt{t}^s_{ji} - \mathtt{e}_j$ with $y^*_i - y^*_j$ for all $(j,i) \in \mathcal{V}^{\text{C-MIS}}$ in the objective function since constraint (18d) forces both expressions to be equal in the primal optimal solution given by Algorithm 1.

$$\max 1 + \sum_{(j,i)\in\mathcal{V}^{\text{C-MIS}}} \hat{\alpha}_{ji}(y^*_i - y^*_j) - \sum_{i\in\mathcal{I}^{\text{C-MIS}}} \pi_i y^*_i + \sum_{i\in\mathcal{I}^{\text{C-MIS}}} \beta_i(\mathtt{s}_i - \mathtt{lb})$$

$$\text{s.t.} \sum_{(j,i)\in\mathcal{V}^{\text{C-MIS}}} \hat{\alpha}_{ji} - \sum_{(i,j)\in\mathcal{V}^{\text{C-MIS}}} \hat{\alpha}_{ij} - \pi_i + \beta_i = 0, \qquad \forall i \in \mathcal{I}^{\text{C-MIS}}, \tag{20a}$$

$$\pi_i(\mathtt{s}_i + \mathtt{ub} - y^*_i) \geq 0, \qquad \forall i \in \mathcal{I}^{\text{C-MIS}} \setminus \mathcal{D}^{\text{C-MIS}}, \tag{20b}$$

$$\pi_i \leq \frac{1}{y^*_i - (\mathtt{s}_i + \mathtt{ub})}, \qquad \forall i \in \mathcal{D}^{\text{C-MIS}}, \tag{20c}$$

$$\pi_i, \beta_i \geq 0, \qquad \forall i \in \mathcal{I}.$$

To assign the remaining variables of the models we use (20a) and show that (20b) and (20c) are satisfied. First, note that (20a) has at most one $\hat{\alpha}$ term per summation since every trip $i \in \mathcal{I}$ can have at most one trip as a predecessor and successor, respectively, in any master solution. Thus, (20a) can be written as follows for the general case:

$$\hat{\alpha}_{ji} - \hat{\alpha}_{ij'} - \pi_i + \beta_i = 0, \quad \forall i \in \mathcal{I}^{\text{C-MIS}},\ j, j' \in \mathcal{I} \text{ such that } (j,i), (i,j') \in \mathcal{V}^{\text{C-MIS}}.$$

Then, we partition the trips in $\mathcal{I}^{\text{C-MIS}}$ in three different subsets:

1. Trips in $\mathcal{V}^{\text{C-MIS}}$ that are first in a subsequence, that is,

$$\mathcal{I}^{\text{C-MIS}}_{\texttt{first}} = \{i \in \mathcal{I}^{\text{C-MIS}} : \nexists\ j \in \mathcal{I}^{\text{C-MIS}} \text{ such that } (j,i) \in \mathcal{V}^{\text{C-MIS}}\}$$

2. Trips in $\mathcal{V}^{\text{C-MIS}}$ that are last in a subsequence, that is,

$$\mathcal{I}^{\text{C-MIS}}_{\texttt{last}} = \{i \in \mathcal{I}^{\text{C-MIS}} : \nexists\ j \in \mathcal{I}^{\text{C-MIS}} \text{ such that } (i,j) \in \mathcal{V}^{\text{C-MIS}}\}$$

3. Trips in $\mathcal{V}^{\text{C-MIS}}$ that are in the middle of a subsequence, that is,

$$\mathcal{I}^{\text{C-MIS}}_{\texttt{middle}} = \{i \in \mathcal{I}^{\text{C-MIS}} : \exists\ j, j' \in \mathcal{I}^{\text{C-MIS}} \text{ such that },(j',i),(i,j) \in \mathcal{V}^{\text{C-MIS}}\}$$

Note that trips in $\mathcal{I}^{\texttt{C-MIS}}_{\texttt{first}}$ start as early as possible in the solution of Algorithm 1, so constraint (18f) associated with those indices are loose and, thus, $\hat{\pi}_i = 0$ for all $i \in \mathcal{I}^{\texttt{C-MIS}}_{\texttt{first}}$. Similarly, all trips in $\mathcal{I}^{\texttt{C-MIS}}_{\texttt{last}}$ and $\mathcal{I}^{\texttt{C-MIS}}_{\texttt{middle}}$ start after $\texttt{s}_i - \texttt{lb}$, so the constraints (18g) are loose for such indices and, thus, $\hat{\beta}_i = 0$ for all $i \in \mathcal{I}^{\texttt{C-MIS}}_{\texttt{first}} \cup \mathcal{I}^{\texttt{C-MIS}}_{\texttt{last}}$.Then, replacing these assignments and $\hat{\alpha}_{ji}$ for $(j,i) \in \mathcal{V}^{\texttt{C-MIS}}$ in constraint (20a), we get the following values for the remaining dual variables:

$$\begin{aligned}
\hat{\beta}_j &= \frac{1}{y^*_i}, && \forall j \in \mathcal{I}^{\texttt{C-MIS}}_{\texttt{first}}, (j,i) \in \mathcal{V}^{\texttt{C-MIS}}, \\
\hat{\pi}_j &= \frac{1}{y^*_j}, && \forall j \in \mathcal{I}^{\texttt{C-MIS}}_{\texttt{last}}, \\
\hat{\pi}_j &= \frac{1}{y^*_j} - \frac{1}{y^*_i}, && \forall j \in \mathcal{I}^{\texttt{C-MIS}}_{\texttt{middle}}, (j,i) \in \mathcal{V}^{\texttt{C-MIS}}.
\end{aligned}$$

Note that $\frac{1}{y^*_j} - \frac{1}{y^*_i} \geq 0$ for all $(j,i) \in \mathcal{V}^{\texttt{C-MIS}}$ since j is schedule before i in $\hat{\boldsymbol{x}}$ and, thus, $y^*_j \leq y^*_i$.

We now argue that $\hat{\pi}_j$ for each $j \in \mathcal{I}^{\texttt{C-MIS}}_{\texttt{middle}} \cup \mathcal{I}^{\texttt{C-MIS}}_{\texttt{last}}$ satisfy constraints (20b)-(20c). Note that constraint (20b) is satisfied since our constructed $\hat{\pi}_i \geq 0$ and $\texttt{s}_i + \texttt{ub} \geq y^*_i$ for all trips $i \in \mathcal{I}^{\texttt{C-MIS}}$ without delay (i.e., not in $\mathcal{D}^{\texttt{C-MIS}}$). Constraint (20c) is satisfied for all $i \in \mathcal{I}^{\texttt{C-MIS}}_{\texttt{last}}$ since

$$\frac{1}{y^*_i} \leq \frac{1}{y^*_i - (\texttt{s}_i + \texttt{ub})}$$

as $y^*_i > \texttt{s}_i + \texttt{ub}$ and $\texttt{s}_i + \texttt{ub} \geq 0$ hold. Similarly, with some algebraic manipulation, it can be shown that constraint (20c) is satisfied for all $i \in \mathcal{I}^{\texttt{C-MIS}}_{\texttt{middle}} \cap \mathcal{D}^{\texttt{C-MIS}}$. Therefore, we constructed a feasible dual solution using the desired $\hat{\alpha}_{ij}$ for all $(i,j) \in \mathcal{V}^{\texttt{C-MIS}}$.

Lastly, we replace all of these variable assignments in the objective function to check that we obtain an optimal solution:

$$\begin{aligned}
&1 + \sum_{(j,i) \in \mathcal{V}^{\texttt{C-MIS}}} \hat{\alpha}_{ji}(y^*_i - y^*_j) - \sum_{i \in \mathcal{I}^{\texttt{C-MIS}}} \hat{\pi}_i y^*_i + \sum_{i \in \mathcal{I}^{\texttt{C-MIS}}} \hat{\beta}_i(\texttt{s}_i - \texttt{lb}) \\
&= 1 + \sum_{(j,i) \in \mathcal{V}^{\texttt{C-MIS}}} \hat{\alpha}_{ji}(y^*_i - y^*_j) - \sum_{j \in \mathcal{I}^{\texttt{C-MIS}}_{\texttt{middle}}} \hat{\pi}_j y^*_j - \sum_{j \in \mathcal{I}^{\texttt{C-MIS}}_{\texttt{last}}} \hat{\pi}_j y^*_j + \sum_{j \in \mathcal{I}^{\texttt{C-MIS}}_{\texttt{first}}} \hat{\beta}_j(\texttt{s}_j - \texttt{lb}) \\
&= 1 + \sum_{(j,i) \in \mathcal{V}^{\texttt{C-MIS}}} \left(\frac{y^*_i - y^*_j}{y^*_i} \right) - \sum_{j \in \mathcal{I}^{\texttt{C-MIS}}_{\texttt{middle}}, (j,i) \in \mathcal{V}^{\texttt{C-MIS}}} \left(\frac{1}{y^*_i} - \frac{1}{y^*_j} \right) y^*_i - \sum_{i \in \mathcal{I}^{\texttt{C-MIS}}_{\texttt{last}}} \frac{y^*_j}{y^*_j} + \sum_{i \in \mathcal{I}^{\texttt{C-MIS}}_{\texttt{first}}, (j,i) \in \mathcal{V}^{\texttt{C-MIS}}} \frac{\texttt{s}_j - \texttt{lb}}{y^*_i} \\
&= 1 + \sum_{(j,i) \in \mathcal{V}^{\texttt{C-MIS}}} \left(1 - \frac{y^*_j}{y^*_i} \right) - \sum_{j \in \mathcal{I}^{\texttt{C-MIS}}_{\texttt{middle}}, (j,i) \in \mathcal{V}^{\texttt{C-MIS}}} \left(1 - \frac{y^*_i}{y^*_j} \right) - \sum_{i \in \mathcal{I}^{\texttt{C-MIS}}_{\texttt{last}}} 1 + \sum_{i \in \mathcal{I}^{\texttt{C-MIS}}_{\texttt{first}}, (j,i) \in \mathcal{V}^{\texttt{C-MIS}}} \frac{y^*_j}{y^*_i} \\
&= 1
\end{aligned}$$

Note that $y^*_j = \texttt{s}_j - \texttt{lb}$ for all $j \in \mathcal{I}^{\texttt{C-MIS}}_{\texttt{first}}$, which is used going from lines 3 to 4. Also, all the summations in the fourth line cancel each other, which results in an objective value equal to one. Since the dual objective value is equal to the primal objective value, we can guarantee that this dual solution is optimal. Therefore, we have built an optimal dual solution that leads to a C-MIS cut. The validity of the cuts follows from the validity of the C-MIS cut. □

A.5. Proof of Proposition 6

PROPOSITION 6. *The B&C decomposition scheme with the previously mentioned modification converges in a finite number of iterations and returns an optimal schedule to* (SAA-MCF).

Proof. First, we note that all the proposed cuts introduced in this section have the form of (6b) for any give $s \in \mathcal{S}$, that is,

$$\sum_{(i,j)\in\mathcal{A}} \lambda_{ij} \sum_{k\in\mathcal{K}} x_{ijk} \leq \lambda_0 - 1 + z_s,$$

where $\lambda_{ijk}, \lambda_0 \geq 0$. Due to the master problem constraints (2a) that enforce each trip to be assigned only once, we can guarantee that $\sum_{(i,j)\in\mathcal{A}} \lambda_{ij} \sum_{k\in\mathcal{K}} x_{ijk} \leq \lambda_0$ for any feasible assignment of $\boldsymbol{x}$ in (SAA-Master). Moreover, there exists at least one schedule $\hat{\boldsymbol{x}}$ such that $\sum_{(i,j)\in\mathcal{A}} \lambda_{ij} \sum_{k\in\mathcal{K}} \hat{x}_{ijk} = \lambda_0$, for instance, the schedule that built such constraint. Therefore, there exists a schedule $\hat{\boldsymbol{x}}$ that forces variable z_s to take value one for any generated cut of the form (6b).

Second, for a given $\hat{\boldsymbol{x}}$, the decomposition scheme now solves (SAA-Sub$(s, \hat{\boldsymbol{x}})$) for any $s \in \mathcal{S}$ such that $\hat{z}_s < 1$. Therefore, our procedure will add the necessary constraints to enforce integrality on the appropriate $\boldsymbol{z}$ variables (i.e., the variables associated with scenarios that violate a service requirement) for given candidate schedule $\hat{\boldsymbol{x}}$, which are finitely many. Thus, the modified decomposition scheme will converge in a finite number of iterations, that is, at least one iteration for each candidate schedule that satisfies $\mathcal{X}$. Moreover, the returned schedule $\hat{\boldsymbol{x}}$ is optimal for (SAA-MCF) due to the tightness and validity of the generated cuts (6b). □

Appendix B: Instance Generation

We now detailed how we generate random instances for the CC-MDVSP. We made three slight modifications to the commonly used approach proposed by Carpaneto et al. (1989) for the deterministic MDVSP to consider: (i) trips grouped intro routes, (ii) random travel times, and (iii) expressing. All other problem components (e.g., depot capacities, start time of timetable trips, and costs of deadhead trips) are identical to the approach of Carpaneto et al. (1989). Lastly, as in Carpaneto et al. (1989), we consider all travel times to be integer quantities, thus, we round to the closest integer any fractional number given when sampling random distributions.

Routes. As proposed by Carpaneto et al. (1989), we first generate a set of random locations in a grid to represent a trip's possible start and end locations. Since the CC-MDVSP trips are grouped into routes, we first randomly assign start and end locations to each route and then assign these locations to trips of such route. In particular, each route $r \in \mathcal{R}$ has to locations $\mathtt{l}_1$ and $\mathtt{l}_2$ and each trip $i \in \mathcal{I}^r$ can have one of these locations a starting and/or ending point. For instance, trip $i \in \mathcal{I}^r$ can be a round-trip where $\mathtt{l}_i^{\mathtt{s}} = \mathtt{l}_i^{\mathtt{e}} = \mathtt{l}_1$ (i.e., the trips starts at $\mathtt{l}_1$, goes to $\mathtt{l}_2$, and returns to $\mathtt{l}_1$), while trip $i' \in \mathcal{I}^r$ $(i \neq i')$ can be a one-direction trip with $\mathtt{l}_{i'}^{\mathtt{s}} = \mathtt{l}_2$ and $\mathtt{l}_{i'}^{\mathtt{e}} = \mathtt{l}_1$. Since Carpaneto et al. (1989) distinguishes between short and long trips in terms of duration, we assume that long trips are always round trips while short trips are one-direction trips. We use the same parameters and probabilities used in Carpaneto et al. (1989) to determine the grid size, the number of generated locations, and the proportion of long and short trips.

Random travel times. The approach proposed by Carpaneto et al. (1989) used Euclidean distance to compute travel times of deadhead trips (e.g., $\mathtt{t}_{ij}$ for all $i, j \in \mathcal{C}$) and randomly generate durations of timetable trips using uniform distributions with different parameters depending if the trip is long or short. We use this same approach and distribution to get the average travel time of deadhead trips and trip duration, respectively. To generate scenarios for our CC, we sample using a log-normal distribution as suggested in the literature (see, for example, the work of Rahman, Wirasinghe, and Kattan (2018)). In particular, each time is sampled using a log-normal distribution where its mean is the average time given by the Carpaneto et al. (1989) approach, and the standard deviation is 0.2 times the mean.

Expressing. We consider the maximum expressing time to be a proportion of a trip's average duration (i.e., $\bar{\mathtt{d}}_i$ for all $i \in \mathcal{I}$). If a trip $i \in \mathcal{I}$ has an average duration less than 10, then the maximum expressing is $\mathtt{e}_i = 0$. Otherwise, the maximum expressing is a random integer between 5% and 10% of the average duration, that is, $\mathtt{e}_i \in [0.05\bar{\mathtt{d}}_i, 0.01\bar{\mathtt{d}}_i]$.