Service Science

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

Integrated Vehicle Routing and Service Scheduling Under Time and Cancellation Uncertainties with Application in Nonemergency Medical Transportation

Xian Yu, Siqian Shen, Huizhu Wang

To cite this article:
Xian Yu, Siqian Shen, Huizhu Wang (2021) Integrated Vehicle Routing and Service Scheduling Under Time and Cancellation Uncertainties with Application in Nonemergency Medical Transportation. Service Science 13(3):172-191. https://doi.org/10.1287/serv.2021.0277

http://pubsonline.informs.org/journal/serv

SERVICE SCIENCE
Vol. 13, No. 3, September 2021, pp. 172–191
ISSN 2164-3962 (print), ISSN 2164-3970 (online)

Integrated Vehicle Routing and Service Scheduling Under Time and Cancellation Uncertainties with Application in Nonemergency Medical Transportation

Xian Yu,[a] Siqian Shen,[a,*] Huizhu Wang[b]

[a] Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, Michigan 48109; [b] Ford Motor Company, Dearborn, Michigan 48126
* Corresponding author

Contact: yuxian@umich.edu, https://orcid.org/0000-0003-1059-5303 (XY); siqian@umich.edu, https://orcid.org/0000-0002-2854-163X (SS); hwang152@ford.com (HW)

Received: December 15, 2020
Revised: April 17, 2021
Accepted: June 27, 2021
Published Online in Articles in Advance: September 3, 2021

https://doi.org/10.1287/serv.2021.0277

Abstract. In this paper, we consider an integrated vehicle routing and service scheduling problem for serving customers in distributed locations who need pick-up, drop-off, or delivery services. We take into account the random trip time, nonnegligible service time, and possible customer cancellations, under which an ill-designed schedule may lead to undesirable vehicle idleness and customer waiting. We build a stochastic mixed-integer program to minimize the operational cost plus expected penalty cost of customers' waiting time, vehicles' idleness, and overtime. Furthermore, to handle real-time arrived service requests, we develop K-means clustering-based algorithms to dynamically update planned routes and schedules. The algorithms assign customers to vehicles based on similarities and then plan schedules on each vehicle separately. We conduct numerical experiments based on diverse instances generated from census data and data from the Ford Motor Company's GoRide service, to evaluate result sensitivity and to compare the in-sample and out-of-sample performance of different approaches. Managerial insights are provided using numerical results based on different parameter choices and uncertainty settings.

Funding: The authors gratefully acknowledge support for this work by the University of Michigan and Ford Motor Company Alliance program [Grant 000356-UM0222-H]. S. Shen is supported in part by the National Science Foundation [Grant CMMI-1727618].
Supplemental Material: The online appendices are available at https://doi.org/10.1287/serv.2021.0277.

Keywords: pick up and delivery with time windows • appointment scheduling • random cancellation • stochastic integer programming • K-means clustering

1. Introduction

Ford Motor Company's nonemergency medical transportation (NEMT) provides reservation-based pick-up and drop-off services to patients who are elderly, disabled, or have chronic diseases, for traveling to medical requests (see Dickey 2018). The NEMT type of businesses requires a large fleet of vehicles to serve a large number of dispersed patients during peak hours, and these vehicles may be idle when the number of service requests is low. Some patients may cancel existing reservations, resulting in further system idleness. The trip time and service duration could also be random due to hourly traffic conditions and the difficulty of loading/unloading some patients, respectively. In NEMT, most patients reportedly wait for 10 to 20 minutes for their scheduled trips (see Bryant 2019). Through better designed vehicle routes and schedules for NEMT, one can potentially reduce the total number of vehicles in operations and cover new service regions only using the existing vehicles, while maintaining high vehicle utilization rates to attain financial profits and high quality of service.

Through popularizing NEMT types of systems, under-served populations having scarce mobility resources but high needs can potentially have reliable and affordable transportation. Indeed, an NEMT-like system can be extended for medical home care delivery or grocery delivery, in which vehicles take certified nurses or medicines/goods to patients/customers rather than having them travel to hospitals/grocery stores. Amid the COVID-19 pandemic, this type of service is extremely important to self-quarantined COVID-19 patients with mild conditions and also to patients having chronic diseases who need to regularly visit their doctors and thus could have high cross-infection risk. The latter are also the most vulnerable population groups with the highest fatality rate among all COVID-19 infected case due to their weak health conditions (see Centers for Disease Control and Prevention 2020). Providing such a service can alleviate some stress on

existing medical systems (e.g., hospitals, clinics, and urgent care units) that need to focus on treating COVID-19 patients with critical conditions.

For goods, people, and product delivery, existing models and approaches are mainly based on variants of the vehicle routing problem (VRP) (Laporte 1992, 2007). In a capacitated VRP (CVRP), multiple vehicles are dispatched from a single depot to meet all demand from customers and each vehicle seeks a feasible route within its capacity to deliver to or pick up from a set of customers before returning to the depot, so that the total traveling distance of all vehicles is minimized (see, e.g., Toth and Vigo 2002, Ralphs et al. 2003, Fukasawa et al. 2006). The VRP with time windows (VRPTW) studies problems where each customer should be served only within a specified time interval or time window (Bräysy and Gendreau 2005a). In a more specific context, the pick-up and delivery problem with time windows (PDPTW) (see, e.g., Savelsbergh and Sol 1995, Parragh et al. 2008) assumes that each request specifies the size of the load needed to be transported, the locations where it is to be picked up (the origins), and the locations where it is to be delivered (the destinations).

The NEMT settings are the most relevant to PDPTW, except we also take into account nonnegligible service duration at patients' locations, service cancellations, and the uncertainties. Moreover, we consider a hybrid case having both static and dynamic operations, such that the reservation-based system schedules a set of known service requests first, and then accommodates a few requests that may arrive on short notice in real-time operations. This needs to dynamically reschedule existing unfinished services and to compute the new schedule efficiently.

In this paper, we combine appointment scheduling models with PDPTW while taking into account several types of uncertainties. Two settings of problems are considered: (i) in the static setting, we assume that all service requests are known before planning, based on which we make an initial schedule and route plan; (ii) in the dynamic setting, customers arrive during real-time operations before their requested pick-up time. All the origin-destination (O-D) pairs and time windows are known at the time of reserving or announcing service requests; however, both the service duration and travel time could be stochastic. We also incorporate random cancellations such that customers who request services have a certain chance of not showing up.

We model the static problem as a two-stage stochastic mixed-integer linear program (TS-MILP) with the objective to minimize the total operational cost of dispatching and routing vehicles, plus the expected penalty cost of customer waiting, vehicle idleness, and overtime for ensuring high quality of service. Specifically, in the first stage, we assign all customers to different vehicles and also make routing decisions for each vehicle. After observing random service duration/travel time/customer cancellations, we formulate a linear program to calculate customers' waiting time, vehicles' idle time, and overtime. A rolling horizon method is proposed to extend the TS-MILP to solve dynamic vehicle routing and service scheduling for real-time operations. Solving the TS-MILP could suffer from incapability of attaining optimal solutions at scale. Notice that due to the nonzero service time and uncertainties of multiple parameters, we cannot employ the traditional branch-and-price algorithm for VRP variants for solving the TS-MILP. To speed up computation for realistic problem sizes, we develop several heuristic approaches that are all based on the K-means clustering algorithm (Jain 2010). They first group all the customers into K clusters based on their O-D pair similarities, assign one vehicle to each cluster, and then make a schedule for each vehicle separately. Some heuristics also perform a swapping step after getting an initial clustering result based on customers' time windows to distribute them more evenly to each vehicle. We are able to compute a real-world data set from Ford as well as large-scale instances via the integration of the aforementioned optimization and data clustering techniques.

The remainder of the paper is organized as follows. In Section 2, we review the most relevant papers on variants of VRP and appointment scheduling. In Section 3, we describe stochastic optimization models of the static and dynamic problems. In Section 4, we develop clustering-based heuristics to improve the efficiency of the dynamic approach for serving large-scale regions. In Section 5, numerical studies are conducted using instances generated based on the real data of Ford Motor Company's GoRide service. We demonstrate the efficacy of our approaches under diverse settings and reveal managerial insights for different uncertainty realizations. Section 6 concludes the paper and states future research directions.

2. Literature Review

Our problem is closely related to VRPTW (see, e.g., Desrochers et al. 1992; Bräysy and Gendreau 2005a, b), the dial-a-ride problem (DARP) (see, e.g., Cordeau and Laporte 2007, Berbeglia et al. 2012), the pick-up and delivery problem (PDP) (see, e.g., Savelsbergh and Sol 1995, Berbeglia et al. 2010), and appointment scheduling problems (see, e.g., Gupta and Denton 2008, Erdogan and Denton 2013, Berg et al. 2014, Deng and Shen 2016, Jiang et al. 2017). We refer to Laporte (1992, 2007) for classical models for VRP and the related exact algorithms, classical heuristics, and metaheuristics. Cordeau and Laporte (2007) review the literature of DARP, demonstrate the main features of the problem, and provide a summary of the most

important models and algorithms. Savelsbergh and Sol (1995) distinguish PDP from standard VRP and present a survey of its models and solution approaches, with a primary focus on deterministic problems. Berbeglia et al. (2010) and Pillac et al. (2013) provide thorough reviews of dynamic PDP and VRP, respectively, where objects or people have to be served in real-time. Cömert et al. (2017) consider VRP with hard time windows (VRPHTW) using a cluster-first route-second hierarchical approach, where the authors first assign customers to vehicles using different clustering algorithms and then solve a VRPHTW as an MILP. Although the idea is similar to our work, the authors do not take into account any parameter uncertainty. In the context of dynamic and stochastic VRP, Bertsimas and Van Ryzin (1991, 1993) and Bertsimas and Simchi-Levi (1996) develop and review the greedy heuristics for solving VRP with stochastic dynamically arriving demand, mainly based on queueing theories. On the other hand, Dror et al. (1989) review the stochastic programming models for VRPTW with stochastic demand and propose a new solution frame using a Markov decision process (MDP). Powell (1996), Powell et al. (2000), and Simao et al. (2009) apply the MDP to the truckload assignment problem and develop a deterministic myopic method, a stochastic dynamic model, and approximate dynamic programming for estimating future cost functions, respectively. Bent and Van Hentenryck (2004) consider online stochastic multiple vehicle routing with time windows in which customers arrive dynamically and the goal is to maximize the number of customers served. The authors propose a multiple scenario approach (MSA) that continuously generates routing plans for scenarios including known and future requests. Later, Bent and Van Hentenryck (2007) propose to include customer waiting and relocation in the online algorithm to achieve better results. More recently, Bertsimas et al. (2019) specifically consider online vehicle routing solved by ride-sharing companies and propose an optimization framework and an efficient algorithm to allow solving the problem on demand at a large scale, demonstrated by numerical results using real New York City taxi data.

For appointment scheduling, Gupta and Denton (2008) present a comprehensive review on models and methods used in healthcare systems. Pinedo (2012) summarizes all deterministic/stochastic scheduling models and general-purpose procedures of dealing with scheduling problems in practice. Zacharias and Pinedo (2014) study an appointment scheduling problem with customers' no-show behavior and overbooking. Denton and Gupta (2003) model a two-stage stochastic linear program to optimize appointment times for a sequence of jobs with uncertain duration, and Erdogan and Denton (2013) extend their results to handle no-shows and to the multistage dynamic setting. Berg et al. (2014) formulate a two-stage stochastic mixed-integer program for optimizing booking and appointment scheduling under uncertainty of procedure times and patient attendance.

Recently, Jiang et al. (2017) consider a single-server appointment scheduling problem with random no-shows and service duration. They derive mixed-integer nonlinear programming reformulations, valid inequalities, and convex-hull representations under specially structured ambiguity sets for distributionally robust appointment scheduling. Deng and Shen (2016) investigate a multiserver scheduling problem with random service duration and minimize the cost of operating servers, subject to a joint chance constraint limiting the risk of a server running overtime.

For home healthcare (HHC) routing and scheduling, Fikar and Hirsch (2017) present a comprehensive review with a focus on the various problem settings and solution approaches. Among them, Heching et al. (2019) use a logic-based Benders' decomposition (LBBD) to solve the assignment-scheduling problem and propose several subproblem relaxations to speed up the computation. Most literature consider static information, that is, all data are known in advance and no uncertainty in the parameter is considered. Several papers deal with parameter uncertainties and among them, Lanzarone and Matta (2014) and Carello and Lanzarone (2014) consider demand uncertainty. They formulate the nurse-to-patient assignment problem as a robust optimization model and propose both analytical and heuristic-based approaches. Yuan et al. (2015) study random service time and propose column generation (CG) and several heuristics to solve a stochastic program for optimizing routing and scheduling decisions. The random service time is also considered in Zhan and Wan (2018) and Zhan et al. (2021), where the former formulates a scenario-based mixed-integer program (SBMIP) and develops an algorithm based on tabu search to efficiently solve the problem, and the latter proposes an L-shaped method with valid inequalities to speed up the solution process. An easy-to-implement heuristic based on a modified traveling salesman problem (MTSP) is also developed in Zhan et al. (2021) for solving large-scale instances. We compare our work with the literature in Table 1, where we also present the maximum size of instances that each paper solves in the last two columns with $|J|$ and $|I|$ being the number of vehicles and customers, respectively.

2.1. Main Contributions of This Paper

To our best knowledge, this paper is the first to combine PDPTW with appointment scheduling and incorporates various uncertainty sources, including time-related uncertainty and cancellations. Our stochastic optimization model comprehensively captures

Table 1. Comparison Between Our Study and HHC Routing and Scheduling Literature

Paper	Problem description	Stochasticity	Dynamic	Exact	Heuristics	$\|J\|$	$\|I\|$
Our work	PDPTW + daily planning	Random travel time/service duration/customer cancellations + stochastic program	Rolling	MILP	K-means clustering	40	450 p/d
Zhan and Wan (2018)	VRP + daily planning	Service duration	—	SBMIP	Tabu search	7	40 p/d
Zhan et al. (2021)	VRP + daily planning	Service duration	—	L-shaped	MTSP	1	10 p/d
Heching et al. (2019)	VRPTW + weekly planning	—	Rolling	LBBD	Subproblem relaxation	20	60 p/w
Allaoua et al. (2013)	VRPTW + daily planning	—	—	ILP	Set Partitioning	9	30 p/d
Rasmussen et al. (2012)	VRPTW + daily planning	—	—	B&P	Clustering	15	150 p/d
Lanzarone and Matta (2014)	Nurse-to-patient assignment	Random demand + robust optimization	Rolling	Analytical	Policy based on sorting	8	40 p/w
Bennett and Erera (2011)	VRP + weekly planning	—	Rolling	—	Distance-based insertion,capacity-based insertion	1	1.5 p/d
Yuan et al. (2015)	VRPTW + daily planning	Random service times + stochastic program	—	CG	Greedy heuristic,variable neighborhood descent algorithm	60	50 p/d
Nickel et al. (2012)	VRPTW + weekly planning	—	Greedy-based insertion	—	Constraint programming heuristic + adaptive large neighborhood search	12	95 p/w
Cappanera and Scutellà (2015)	VRPTW + weekly planning	—	—	ILP	—	11	162 p/w
Bard et al. (2014)	VRPTW + weekly planning	—	—	B&P&C	Rolling horizon method	20	650 p/w

Note. B&P, branch and price; B&P&C, branch and price and cut; ILP, integer linear program; p/d, per day; p/w, per week.

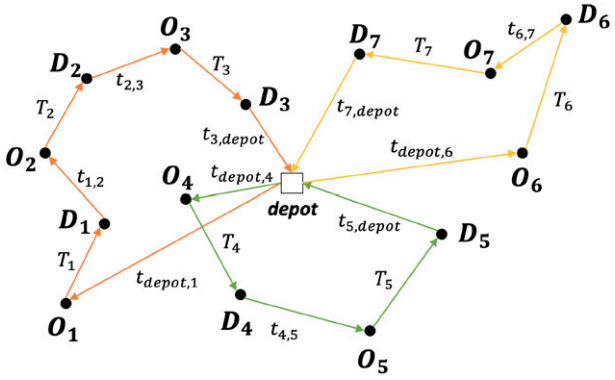
a wide spectrum of decisions made in related applications, ranging from vehicle routing and service sequencing, to specific vehicle arrival and departure times for appointment scheduling. We decompose the model into two stages, such that the second-stage problem can efficiently compute waiting time, idle time, and overtime of an optimal schedule for each vehicle via a linear programming model. Moreover, using the clustering-based heuristics, we are able to quickly compute high-quality solutions for large-scale instances with 40 vehicles and 450 demand requests, within several seconds.

3. Optimization Models

We first consider a reservation-based transportation system, where customers need to provide trip information including O-D pairs and pick-up time windows when reserving their trips. After gathering the information of all the trips, the operating coordinator needs to assign them to a fleet of vehicles and make an initial schedule while considering the uncertainty in the service duration/travel time/customer cancellations. We illustrate the problem in Figure 1, where we have one depot (denoted by a square), three vehicles (denoted by different colored lines), and seven customers with O-D pairs (O_i, D_i), $i = 1, \ldots, 7$ (denoted by black dots). The arrows indicate an optimal solution of vehicle routing decisions, based on which we can make corresponding scheduling decisions.

All the time-related notations (t and T) marked on the figure are explained in Section 3.1 and modeled by random variables. Moreover, each customer who made a reservation may not show up with a certain chance, which we can also model using random variables having 0 or 1 realized values. Next, we define our notation in Section 3.1, and present formulations of static and dynamic problems in Sections 3.2 and 3.3, respectively.

Figure 1. A Single-Depot Vehicle Routing and Appointment Scheduling Problem



3.1. Problem Description and Notation

We use I, J, K to denote the sets of customers, vehicles, and service slots in each vehicle, respectively. (Each service slot can only fulfill one request and a slot can be used only when we have used up all the earlier slots in the same vehicle.) For notation simplicity, we assume that all vehicles have the same number of slots, $|K|$, as the maximum number of service requests a vehicle can serve within the operational time frame. Note that this assumption is made without loss of generality, as if no request is assigned to any remaining slots on a vehicle, the vehicle returns to the depot and therefore the depot will occupy all the remaining slots. Let O_i, D_i be the origin and destination of the trip requested by customer i, $[\underline{a}_i, \bar{a}_i]$ be the requested pick-up time window of customer i, L_j and c_j be the total operating time and operational cost of vehicle j for all $i \in I$ and $j \in J$. For example, if the operational time frame is between 4 a.m. and 7 p.m., we set $L_j = 60$ minutes per hour $\times$ 15 hours $= 900$ minutes. Denote c^w, c^u, c^o as the unit penalty costs of customers' waiting time, vehicles' idle time, and overtime, respectively. (For notation simplicity, we assume that these costs are the same across all customers or vehicles. They can easily be differentiated for individual customers or vehicles without affecting our models later.) Let N be the set of locations, containing all the origins, destinations, and the depot, that is, $N = \{O_i, D_i\}_{i \in I} \cup \{\text{depot}\}$.

Parameter ξ denotes the overall vector of uncertain parameters and let P be its probability distribution. Without loss of generality, we consider discrete distribution P and a finite set of realizations of the random vector ξ. In practice, if the true distribution P of random variable ξ is continuous, we apply the Monte Carlo sampling approach to replace P with an empirical distribution constructed by $|\Omega|$ scenarios, with each scenario $\omega \in \Omega$ having an equal probability $p^\omega = 1/|\Omega|$. The resulting problem with the constructed scenarios is called the sample average approximation (SAA) problem (see Kleywegt et al. 2002). Specifically, for each scenario $\omega \in \Omega$, we denote $\tau_{n_1,n_2}(\omega)$ as the travel time between n_1 and n_2 for all $n_1, n_2 \in N$, $\hat{T}_{O_i}(\omega)$, $\hat{T}_{D_i}(\omega)$ as the service duration at customer i's origin and destination, respectively, and $T_i(\omega)$ as the total time for serving customer i, where $T_i(\omega) = \hat{T}_{O_i}(\omega) + \tau_{O_i,D_i}(\omega) + \hat{T}_{D_i}(\omega)$. Also, for all $i_1, i_2 \in I$, let $t_{i_1,i_2}(\omega)$ be the transition time from customer i_1 to customer i_2, and for all $i \in I$, let $t_{\text{depot},i}(\omega)$, $t_{i,\text{depot}}(\omega)$ be the transition time from the depot to customer i and from customer i to the depot, respectively. Finally, we consider $q_i(\omega)$ as a random service cancellation outcome of customer i, which equals 1 if customer i shows up when an assigned vehicle arrives, and 0 otherwise, for all $i \in I$. The overall random vector is $\xi = (T, \hat{T}, \tau, t, q)$. (Throughout the

paper, we use bold symbols to denote the vector form of a decision variable or a parameter.)

We define binary variables $x_j, y_{ij}, u^j_{i,k}, u^j_{\text{depot},k} \in \{0,1\}$ for all $i \in I, j \in J, k \in K$ such that $x_j = 1$ if we operate vehicle j, $y_{ij} = 1$ if we assign customer i to vehicle j, $u^j_{i,k} = 1$ if customer i is assigned to the kth slot of vehicle j, and $u^j_{\text{depot},k} = 1$ if the depot is assigned to the kth slot of vehicle j, respectively. We also define continuous variables $r^j_k \geq 0$ for all $k \in K, j \in J$ as the planned start time of kth request on vehicle j. Then, in the second stage, for each scenario $\omega \in \Omega$, variables $w^j_k(\omega)$ represent the customer's waiting time at the beginning of the kth slot on vehicle j, $l^j_k(\omega)$ the vehicle's idle time at the end of the kth slot on vehicle j, $l^j_0(\omega)$ the vehicle's idle time before the first request starts on vehicle j, $W_j(\omega)$ the overtime of vehicle j, and $V_j(\omega)$ the total travel time of vehicle j for all $j \in J, k \in K$.

Figure 2 depicts the relationship between the decision variables and parameters. Given a planned schedule (r^j_k) with observed service duration ($T_i(\omega)$) and transition time ($t_{i_1,i_2}(\omega)$), one can easily calculate the waiting ($w^j_k(\omega)$), idling ($l^j_k(\omega)$), and overtime ($W_j(\omega)$) for each scenario ω based on this figure. We formalize it in (2a)–(2f), which can be transformed to a set of linear constraints.

3.2. Static Vehicle Routing and Service Scheduling

We formulate a TS-MILP to model the vehicle routing and service scheduling problem, where the first-stage problem decides which vehicles to operate (x_j), the assignment of each customer to vehicles (y_{ij}), the relative slots in each vehicle ($u^j_{i,k}$), and the planned start time on each vehicle (r^j_k). Let $Q(\boldsymbol{u}, \boldsymbol{r}, \omega)$ be the total penalty cost of waiting, idling, and overtime given first-stage decision variables $\boldsymbol{u}, \boldsymbol{r}$ and the realization of uncertainty ξ in scenario ω. Then, the SAA reformulation can be represented as follows:

$$\min \quad \sum_{j\in J} c_j x_j + \sum_{\omega\in\Omega} p^\omega Q(\boldsymbol{u}, \boldsymbol{r}, \omega) \tag{1a}$$

$$\text{subject to} \quad \sum_{j\in J} y_{ij} = 1, \quad \forall i \in I, \tag{1b}$$

$$y_{ij} \leq x_j, \quad \forall i \in I, j \in J, \tag{1c}$$

$$\sum_{k\in K} u^j_{i,k} = y_{ij}, \quad \forall i \in I, j \in J, \tag{1d}$$

$$\sum_{i\in I} u^j_{i,k} + u^j_{\text{depot},k} = x_j, \quad \forall k \in K, j \in J, \tag{1e}$$

$$\sum_{i\in I} u^j_{i,k+1} \leq \sum_{i\in I} u^j_{i,k}, \quad \forall k = 1, \ldots, |K| - 1, j \in J, \tag{1f}$$

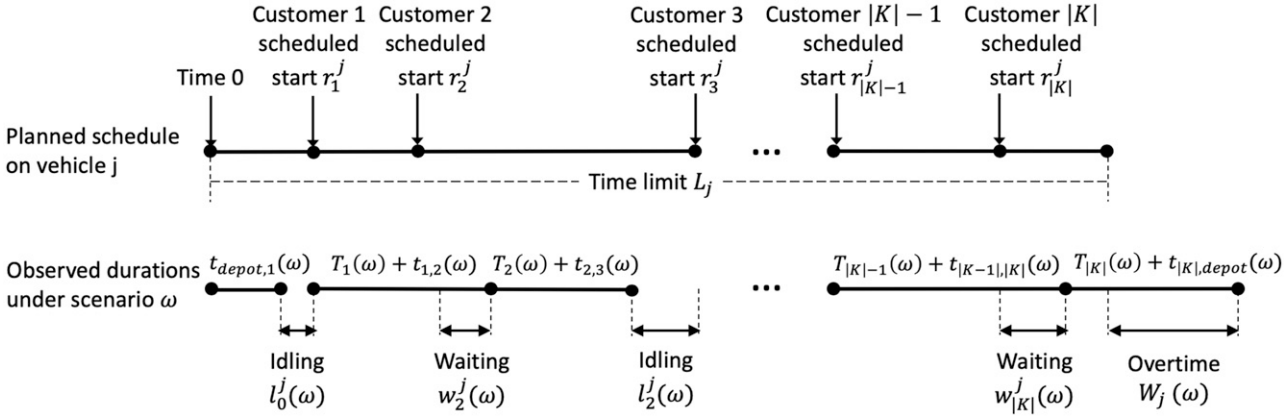
$$\sum_{i\in I} \underline{a}_i u^j_{i,k} + \left(1 - \sum_{i\in I} u^j_{i,k}\right) L_j \leq r^j_k \leq \sum_{i\in I} \bar{a}_i u^j_{i,k} + \left(1 - \sum_{i\in I} u^j_{i,k}\right) L_j, \quad \forall k \in K, j \in J, \tag{1g}$$

$$r^j_{k+1} \geq r^j_k, \quad \forall k = 1, \ldots, |K| - 1, j \in J, \tag{1h}$$

$$x_j \in \{0,1\}, y_{ij} \in \{0,1\}, u^j_{i,k} \in \{0,1\}, r^j_k \geq 0, \quad \forall i \in I, k \in K, j \in J. \tag{1i}$$

The objective (1a) minimizes the total operational cost and an expected second-stage cost $Q(\boldsymbol{u}, \boldsymbol{r}, \omega)$. Constraints (1b)–(1d) ensure that every request is assigned to a slot on an operating vehicle. Constraints (1e) assign at most one request to each slot on each vehicle, and if there are no requests assigned to this slot, the vehicle returns to the depot. Constraints (1f) prohibit assigning a request to a slot on a vehicle if an earlier slot is vacant. Constraints (1g) ensure that each request on vehicle j starts within its requested time window, and if no requests are assigned to vehicle j at the kth slot, r^j_k is set as the time limit L_j for vehicle j.

Figure 2. Relationship Between Planned Schedule and Realized Waiting Time, Idle Time, and Overtime Depending on Observed Time Duration in One Scenario for a Vehicle



Constraints (1h) ensure the requests' start times are in line with their order.

In the second stage under scenario ω, the waiting time $w_1^j(\omega)$, $w_{k+1}^j(\omega)$, idle time $l_0^j(\omega)$, $l_k^j(\omega)$, and overtime $W_j(\omega)$ for all $k = 1, \ldots, |K|-1$ and $j \in J$ can be respectively measured by

$$w_1^j(\omega) = \max\left\{0, \sum_{i\in I} t_{\text{depot},i}(\omega)u_{i,1}^j - r_1^j\right\}, \tag{2a}$$

$$l_0^j(\omega) = \max\left\{0, r_1^j - \sum_{i\in I} t_{\text{depot},i}(\omega)u_{i,1}^j\right\}, \tag{2b}$$

$$w_{k+1}^j(\omega) = \max\{0, r_k^j + w_k^j(\omega) + \sum_{i\in I} q_i(\omega)T_i(\omega)u_{i,k}^j + \sum_{i_1,i_2\in I\bigcup\{\text{depot}\}} t_{i_1,i_2}(\omega)u_{i_1,k}^j u_{i_2,k+1}^j - r_{k+1}^j\}, \tag{2c}$$

$$l_k^j(\omega) = \max\left\{0, r_{k+1}^j - \left(r_k^j + w_k^j(\omega) + \sum_{i\in I} q_i(\omega)T_i(\omega)u_{i,k}^j + \sum_{i_1,i_2\in I\bigcup\{\text{depot}\}} t_{i_1,i_2}(\omega)u_{i_1,k}^j u_{i_2,k+1}^j\right)\right\}, \tag{2d}$$

$$W_j(\omega) = \max\left\{0, r_{|K|}^j + w_{|K|}^j(\omega) + \sum_{i\in I} q_i(\omega)T_i(\omega)u_{i,|K|}^j + \sum_{i\in I} t_{i,\text{depot}}(\omega)u_{i,|K|}^j - L_j\right\}, \tag{2e}$$

$$l_{|K|}^j(\omega) = \max\left\{0, L_j - \left(r_{|K|}^j + w_{|K|}^j(\omega) + \sum_{i\in I} q_i(\omega)T_i(\omega)u_{i,|K|}^j + \sum_{i\in I} t_{i,\text{depot}}(\omega)u_{i,|K|}^j\right)\right\}. \tag{2f}$$

In (2a)–(2f), the actual transition time $t_{i_1,i_2}(\omega)$ between customer i_1 and customer i_2 depends on the service cancellation probability of customer i_1, that is, if i_1 cancels the service, then the vehicle will travel from i_1's origin to i_2's origin directly; otherwise, the vehicle travels from i_1's destination to i_2's origin. The calculations of the random travel time are given by

$$t_{i_1,i_2}(\omega) = q_{i_1}(\omega)\tau_{D_{i_1},O_{i_2}}(\omega) + (1 - q_{i_1}(\omega))\tau_{O_{i_1},O_{i_2}}(\omega),$$

for all $i_1, i_2 \in I$. Specially, for each $i \in I$, we have

$$t_{\text{depot},i}(\omega) = \tau_{\text{depot},O_i}(\omega),$$
$$t_{i,\text{depot}}(\omega) = q_i(\omega)\tau_{D_i,\text{depot}}(\omega) + (1 - q_i(\omega))\tau_{O_i,\text{depot}}(\omega).$$

We illustrate different cases of waiting and idle time in Figure 3, where we assume that on vehicle j, the kth and $(k+1)$th slots are assigned to customers i_1 and i_2, respectively. The upper figure shows the scenario when we have idle time at the end of the kth slot, whereas the lower figure shows the scenario when we have waiting time at the beginning of $(k+1)$th slot. Note that the differences between $w_{k+1}^j(\omega)$ and $l_k^j(\omega)$ for all $k = 0, \ldots, |K|-1$ and the difference between $W_j(\omega)$ and $l_k^j(\omega)$ are always constants. That is, in scenario ω_1 (see upper figure in Figure 3), we have

$$w_{k+1}^j(\omega) - l_k^i(\omega) = -l_k^i(\omega) = r_k^j + w_k^j(\omega) + T_{i_1}(\omega) + t_{i_1,i_2}(\omega) - r_{k+1}^j,$$

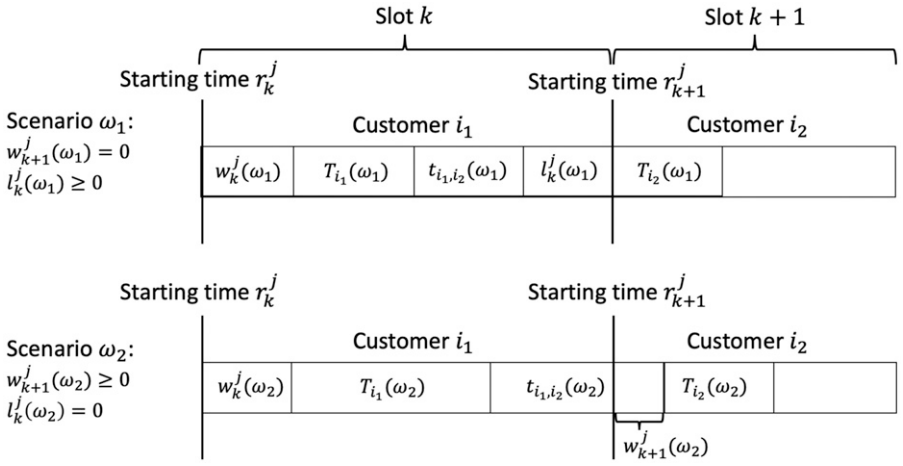
and in scenario ω_2 (see lower figure in Figure 3), we have

$$w_{k+1}^j(\omega) - l_k^i(\omega) = w_{k+1}^j(\omega) = r_k^j + w_k^j(\omega) + T_{i_1}(\omega) + t_{i_1,i_2}(\omega) - r_{k+1}^j,$$

both of which share the same right-hand side. One can argue for the difference between $W_j(\omega)$ and $l_k^j(\omega)$ using the same logic. This leads to the following formulation of the second-stage problem.

For a given VRP solution $\boldsymbol{u}, \boldsymbol{r}$ and scenario $\omega \in \Omega$, we formulate $Q(\boldsymbol{u}, \boldsymbol{r}, \omega) =$

Figure 3. Illustration of Waiting and Idle Time of Slots k and $k+1$ on Vehicle j



$$\min \quad \sum_{j\in J}\left(\sum_{k\in K}(c^w w_k^j(\omega)+c^l l_k^j(\omega))+c^o W_j(\omega)+c^d V_j(\omega)\right) \tag{3a}$$

$$\text{subject to } w_1^j(\omega)-l_0^j(\omega)=\sum_{i\in I} t_{\text{depot},i}(\omega)u_{i,1}^j-r_1^j,\ \forall j\in J, \tag{3b}$$

$$w_{k+1}^j(\omega)-l_k^j(\omega)-w_k^j(\omega)=r_k^j+\sum_{i\in I}q_i(\omega)T_i(\omega)u_{i,k}^j + \sum_{i_1,i_2\in I\bigcup\{\text{depot}\}} t_{i_1,i_2}(\omega)u_{i_1,k}^j u_{i_2,k+1}^j-r_{k+1}^j,$$
$$\forall k=1,\ldots,|K|-1, j\in J, \tag{3c}$$

$$W_j(\omega)-l_{|K|}^j(\omega)-w_{|K|}^j(\omega)=r_{|K|}^j+\sum_{i\in I}q_i(\omega)T_i(\omega)u_{i,|K|}^j + \sum_{i\in I}t_{i,\text{depot}}(\omega)u_{i,|K|}^j-L_j,\ \forall j\in J, \tag{3d}$$

$$V_j(\omega)=\left(\sum_{k\in K}\sum_{i\in I}q_i(\omega)T_i(\omega)u_{i,k}^j+\sum_{i\in I}t_{\text{depot},i}(\omega)u_{i,1}^j + \sum_{k=1}^{|K|-1}\sum_{i_1,i_2\in I\bigcup\{\text{depot}\}} t_{i_1,i_2}(\omega)u_{i_1,k}^j u_{i_2,k+1}^j + \sum_{i\in I}t_{i,\text{depot}}(\omega)u_{i,K}^j\right),\ \forall j\in J, \tag{3e}$$

$$w_k^j(\omega)\ge 0, l_0^j(\omega)\ge 0, l_k^j(\omega)\ge 0, W_j(\omega)\ge 0, V_j(\omega)\ge 0,\ \forall k\in K. \tag{3f}$$

The objective function (3a) minimizes the total penalty of waiting, idleness, overtime and total travel time in scenario ω. Constraints (3b) and (3c) yield either the waiting time of the $(k+1)$th slot or the vehicle's idle time after finishing the kth slot, both of which will have the values as in (2a)–(2d). Similarly, constraints (3d) yield either the overtime $W_j(\omega)$ or the idle time $l_{|K|}^j(\omega)$. Constraints (3e) calculate the total travel time of each vehicle by summing over the service time in each slot, and the travel time between any two adjacent slots. All the waiting, idleness, and overtime variables are nonnegative according to constraints (3f).

The second-stage value function $Q(\boldsymbol{u},\boldsymbol{r},\omega)$ is a nonconvex function with respect to $\boldsymbol{u}$ because of the bilinear term $u_{i_1,k}^j u_{i_2,k+1}^j$ on the right-hand-side of (3c). Given binary-valued $u_{i_1,k}^j$ and $u_{i_2,k+1}^j$, we provide exact reformulations of the bilinear terms $z_{i_1,i_2,k}^j=u_{i_1,k}^j u_{i_2,k+1}^j$ in (3c) using McCormick envelopes:

$$z_{i_1,i_2,k}^j\le u_{i_1,k}^j,\ \forall i_1,i_2\in I\bigcup\{\text{depot}\}, k=1,\ldots,|K|-1, j\in J, \tag{4a}$$

$$z_{i_1,i_2,k}^j\le u_{i_2,k+1}^j,\ \forall i_1,i_2\in I\bigcup\{\text{depot}\}, k=1,\ldots,|K|-1, j\in J, \tag{4b}$$

$$z_{i_1,i_2,k}^j\ge u_{i_1,k}^j+u_{i_2,k+1}^j-1,\ \forall i_1,i_2\in I\bigcup\{\text{depot}\}, k=1,\ldots,|K|-1, j\in J. \tag{4c}$$

We add variables $z_{i_1,i_2,k}^j,\ \forall i_1,i_2\in I\cup\{\text{depot}\}, k=1,\ldots,|K|-1, j\in J$ and constraints (4a)–(4c) into the first-stage problem. As a result, the second-stage value function is now a convex function in terms of the first-stage decisions $\boldsymbol{u}$, $\boldsymbol{r}$, $\boldsymbol{z}$, and we denote it as $Q(\boldsymbol{u},\boldsymbol{r},\boldsymbol{z},\omega)$ to replace the original $Q(\boldsymbol{u},\boldsymbol{r},\omega)$ in Equations (1a)–(1i).

In online Appendix A, we provide modeling details of three extensions to model (1) for accommodating various practical issues, including allowing ride pooling and multiple customers sharing one ride (extension I), enforcing deadlines for dropping off customers (extension II), and allowing vehicles dispatched from multiple depots (extension III).

3.3. Dynamic Vehicle Routing and Service Scheduling

Model (1) provides initial routes and schedules for each vehicle on a day-to-day basis. However, in the NEMT application, trip schedulers often observe that customers request trips in a short notice. A common mechanism for handling dynamic demand arrivals is to use a rolling horizon-based approach, in which plans are made using all known information within a planning horizon, but decisions are not finalized until necessitated by a deadline. At each execution of the algorithm, the planning horizon is rolledforward to include more information, and we resolve the problem and implement some decisions with updated input data and parameters. Next, we elaborate how to extend our models in a rolling horizon framework to handle real-time service requests.

We can optimize model (1) with updated parameters each time when a new request becomes known, and assume that no new service request will be considered while executing the algorithm. Specifically, when a new request shows up at time s, we assume the following sequence of events. First, we update all vehicles' current status, including the time when they become available, $\hat{r}_s^j$, and the corresponding locations when they become available, $\hat{O}_s^j$, for all $j\in J$. There are five possible cases when a new request can occur (i.e., time s) and we specify their corresponding $(\hat{r}_s^j, \hat{O}_s^j)$-values in Figure 4 and as follows:

- Cases 1 and 2: When the assigned customer i_1 is waiting for vehicle j or has already boarded, we let the vehicle finish its current request and then become available. In these two cases, we set $\hat{r}_s^j=r_k^j+w_k^j(\omega_1)+q_{i_1}(\omega_1)T_{i_1}(\omega_1)$, $\hat{O}_s^j=D_{i_1}$ if customer i_1 does not cancel the reservation (i.e., $q_{i_1}(\omega_1)=1$), and $\hat{O}_s^j=O_{i_1}$ otherwise.
- Case 3: When the vehicle is traveling from the previous customer's destination to the next customer's origin and $l_k^j(\omega_1)\ge 0$, we set $\hat{r}_s^j=r_k^j+w_k^j(\omega_1)+q_{i_1}(\omega_1)T_{i_1}(\omega_1)+t_{i_1,i_2}(\omega_1)$ and $\hat{O}_s^j=O_{i_2}$.
- Case 4: When the vehicle is idle, we set $\hat{r}_s^j=s$ and $\hat{O}_s^j=O_{i_2}$.
- Case 5: When the vehicle is traveling from the last customer's destination to the next customer's origin and $w_{k+1}^j(\omega_2)\ge 0$, we let the vehicle finish the $(k+1)$th slot's request and set $\hat{r}_s^j=r_{k+1}^j+w_{k+1}^j(\omega_2)+q_{i_2}(\omega_2)$

$T_{i_2}(\omega_2)$, $\hat{O}_s^j = D_{i_2}$ if customer i_2 does not cancel the reservation (i.e., $q_{i_2}(\omega_2) = 1$), and $\hat{O}_s^j = O_{i_2}$ otherwise.

Let I_s be the set of all service requests announced prior to time s, excluding the ones that have been either completed or started (e.g., customers are waiting or on board). We then reoptimize the vehicle-customer assignments and corresponding schedules by solving a two-stage stochastic programming model similar to model (1), where the only differences are that we replace all I with I_s and drop the binary variables x_i, meaning that all currently operating vehicles will be in use. The second-stage problem is also similar to model (3), except that we replace all I with I_s and replace constraints (3b) with

$$w_1^j(\omega) - l_0^j(\omega) = \hat{r}_s^j + \sum_{i\in I} t_{\hat{O}_s^j, O_i}(\omega) u_{i,1}^j - r_1^j, \quad \forall j \in J,$$

because vehicle j will start its service at the available time $\hat{r}_s^j$ and the origin of it now becomes $\hat{O}_s^j$, rather than the depot.

We summarize the detailed steps of the rolling horizon method based on the optimization models for dynamic routing and scheduling in Algorithm 1.

Algorithm 1 (Rolling Horizon Method Based on Optimization Models)

1: Solve the initial scheduling-routing problem (1) and obtain an optimal schedule and routing plan $(\bar{\boldsymbol{u}}, \bar{\boldsymbol{r}})$.
2: Sample one out-of-sample scenario ω and implement $\bar{\boldsymbol{u}}, \bar{\boldsymbol{r}}$ based on scenario ω.
3: **while** a new request i^* shows up at time s **do**
4: Initialize $I_s = \emptyset$.
5: **for** each customer i in I **do**
6: **if** the announce time s is earlier than the planned start time of i in $\bar{\boldsymbol{r}}$ **then**
7: Put customer i into I_s.
8: **else**
9: Customer i has been served.
10: **end if**
11: **end for**
12: Put customer i^* into I_s and update $I = I_s$.
13: Gather all vehicles' status $(\hat{r}_s^j, \hat{O}_s^j)$ according to Cases 1–5.
14: Solve a variant of model (1) with input $(\hat{r}_s^j, \hat{O}_s^j, I)$ and obtain an optimal schedule and routing plan $(\bar{\boldsymbol{u}}, \bar{\boldsymbol{r}})$.
15: Sample one out-of-sample scenario ω and implement $\bar{\boldsymbol{u}}, \bar{\boldsymbol{r}}$ based on scenario ω.
16: **end while**

Notice that Algorithm 1 is not restricted to optimization-based models. In fact, as long as there is a way to update scheduling and routing plans, we can always apply the rolling horizon method. For example, we can combine Algorithm 1 with clustering-based heuristics, which we will introduce next.

4. Two-Phase Heuristics Using Data Clustering

Although we can attain solution optimality by solving model (1), a drawback is the scalability of the approach and how quickly we can use it to derive dynamic solutions using a rolling horizon computational framework. We will later show that the optimization models do not scale well, and they are not able to solve small- or medium-sized instances within a two-hour computational time limit. In this section, based on the spatial-temporal features of demand in NEMT types of delivery services, we design vehicle routes and service schedules using machine learning and data classification algorithms. The goal is to improve computational time and to derive easy-to-implement decision policies under diverse sources of uncertainties. The main idea of these heuristics is to break the first-stage assignment-scheduling problem into two steps. In the first step, we cluster $|I|$ customers into $|J|$ groups based on their O-D pairs' similarities using data clustering methods (Jain 2010), such as K-means, K-medoids, and so on. We can also modify and improve the initial solutions by ensuring that customers' time windows do not significantly overlap in the same cluster. Then, we assign each cluster of customers to a vehicle and plan a schedule on each vehicle, which can be solved in parallel, based on sorted time windows of the customers in each corresponding cluster. (Note that once we know the customer-to-vehicle assignment and their service order, deciding an optimal schedule such as vehicle arrival time at each individual customer can be done quickly via solving a small-size linear programming model (3) described in Section 3.2.) We describe details of the two heuristics, K-means and K-means with swap, in Sections 4.1 and 4.2, respectively, for clustering geographically similar customers.

4.1. Heuristic 1: K-Means

For each customer i, we use Google API to extract the latitude and longitude of the origin and destination, denoted by $O_i^{\text{lat}}, O_i^{\text{long}}, D_i^{\text{lat}}, D_i^{\text{long}}$. Then we get a point-by-feature matrix $\{d_i\}_{i=1}^{|I|}$ where each d_i is a four-dimensional real vector representing the geographical information of customer i, that is, $d_i = (O_i^{\text{lat}}, O_i^{\text{long}}, D_i^{\text{lat}}, D_i^{\text{long}})$. Via K-means clustering, we aim to partition the $|I|$ data points into $|J|$ $(\leq |I|)$ sets $S = \{S_1, S_2, \ldots, S_{|J|}\}$ to minimize the within-cluster sum of squares (WCSS). Formally, the problem can be cast as:

$$\begin{aligned} \min_{\boldsymbol{m},\boldsymbol{\mu}} \quad & \sum_{i\in I}\sum_{j\in J} m_{ji} \| d_i - \mu_j \|^2 \\ \text{s.t.} \quad & \sum_{j\in J} m_{ji} = 1, \quad \forall i \in I, \\ & m_{ji} \in \{0,1\}, \quad \forall j \in J, i \in I, \end{aligned} \tag{5}$$

where μ_j is the mean (centroid) of cluster S_j calculated by (7), and $m_{ji} = 1$ if data point d_i belongs to cluster j, and $m_{ji} = 0$ otherwise.

The K-means is a special form of the well-known expectation-maximization (EM) algorithm (Moon 1996), where in our case, the E-step is assigning the data points to the closest cluster and the M-step is computing the centroid of each cluster. Specifically, we first randomly select $|J|$ data points $d_{i_1}, d_{i_2}, \ldots, d_{i_{|J|}}$ as the centroids where $i_j \in I$ for all $j \in J$. In the E-step, we fix $\mu_j = d_{i_j}$ for all $j \in J$ and solve problem (5), leading to the following optimal solution:

$$m^*_{ji} = \begin{cases} 1, \text{ if } j = \arg\min_{j' \in J} \| d_i - \mu_{j'} \|^2, \\ 0, \text{ otherwise.} \end{cases} \tag{6}$$

In the M-step, given the optimal assignment $m_{ji} = m^*_{ji}$ for all $i \in I$, $j \in J$, we optimize the objective function (5) over $\boldsymbol{\mu}$ to obtain an updated set of centroids:

$$\mu^*_j = \frac{\sum_{i=1}^{|I|} m_{ji} d_i}{\sum_{i=1}^{|I|} m_{ji}}, \quad \forall j \in J. \tag{7}$$

We then fix $\mu_j = \mu^*_j$ for all $j \in J$ and keep iterating over these two steps until there is no change to the centroids, that is, the assignment of data points to clusters is not changing. We summarize the detailed step of the K-means clustering algorithm in Algorithm 2.

Algorithm 2 (Use K-Means to Cluster $|I|$ Customers into $|J|$ Groups)

1: Given O-D pairs of $|I|$ customers, we use Google API to extract a point-by-feature matrix $\{(O_i^{\text{lat}}, O_i^{\text{long}}, D_i^{\text{lat}}, D_i^{\text{long}})\}_{i=1}^{I}$.
2: Data standardization: rescale the data matrix along each column to get mean 0 and standard deviation 1.
3: Initialization: randomly select $|J|$ data points for the centroids without replacement.
4: **while** the assignment of data points to clusters is changing **do**
5: **Assignment step**: assign each data point to the cluster with the nearest centroid following Equation (6).
6: **Update step**: compute the centroids for the clusters by taking the average of the all data points that belong to each cluster following Equation (7).
7: **end while**

Given $|J|$ clusters of customers, we assign one vehicle to each cluster and then solve model (1) with fixed $(\boldsymbol{x}, \boldsymbol{y})$-values but without the second-stage cost to obtain a schedule and routing plan efficiently, which we denote as KM for short. One can also incorporate the second-stage cost as we discuss in the next heuristic.

4.2. Heuristic 2: K-Means with Swap

Algorithm 2 does not consider the information of time windows when performing clustering. Therefore, we may end up with a cluster of customers who have similar planned start time and short time windows, resulting in either infeasible solutions or extremely long waiting time for some customers in the scheduling phase. Moreover, as we cannot control the number of data points in each cluster, some vehicles may have extremely high demand volumes whereas others are idle in most of the operation time. In this heuristic, we propose a swapping method to distribute all customers more evenly to each vehicle, in terms of their time window distributions and the number of customers in each cluster.

Specifically, we first apply Algorithm 2 to obtain an initial clustering result $S = \{S_1, S_2, \ldots, S_{|J|}\}$. Let S_j^k be the kth element of cluster j. For each cluster $j \in J$, we evaluate the distance of time windows between any two adjacent customers S_j^k and S_j^{k+1}. If the distance is smaller than a given threshold $\underline{T}$, then we reassign either S_j^k or S_j^{k+1} to the cluster $j_{\min}$ that has the smallest number of customers currently, that is, $j_{\min} = \arg\min_{j \in J} \texttt{length}(S_j)$, where we use $\texttt{length}(S_j)$ to represent the number of customers in cluster S_j. The selection criteria is to make sure that after inserting one of the customers into cluster $j_{\min}$, the distances of time windows between the customer and the previous/next ones are no less than the threshold $\underline{T}$. As a result, we can ensure that there is enough time for each vehicle to transit between customers, and the numbers of customers in all the clusters are similar. We summarize the algorithmic details in Algorithm 3.

Algorithm 3 (K-Means with Swap to Improve the Clustering Results by K-Means in Algorithm 2)

1: Perform Algorithm 2 to get an initial clustering result $S = \{S_1, S_2, \ldots, S_{|J|}\}$.
2: Set the iteration number $\ell = 1$ and the maximum iterations to $\ell_{\max}$.
3: **while** not converged and $\ell < \ell_{\max}$ **do**
4: Calculate the cluster index with the smallest length, that is, $j_{\min} = \arg\min_{j \in J} \texttt{length}(S_j)$.
5: **for** $j = 1, \ldots, |J|$ and $j = j_{\min}$ **do**
6: Sort customers in S_j based on their time windows.
7: Set $k = 0$.
8: **while** $k < \texttt{length}(S_j) - 1$ **do**
9: Denote prev_k and next_k as the previous and next customer index of customer S_j^k that belong to cluster $j_{\min}$.
10: **if** $|\bar{a}_{S_j^k} - \bar{a}_{S_j^{k+1}}| < \underline{T}$ **then**
11: **if** $|\bar{a}_{S_j^{k+1}} - \bar{a}_{\text{prev}_{k+1}}| \geq \underline{T}$ and $|\bar{a}_{S_j^{k+1}} - \bar{a}_{\text{next}_{k+1}}| \geq \underline{T}$ **then**
12: Reassign customer S_j^{k+1} to cluster $j_{\min}$.
13: **else if** $|\bar{a}_{S_j^k} - \bar{a}_{\text{prev}_k}| \geq \underline{T}$ and $|\bar{a}_{S_j^k} - \bar{a}_{\text{next}_k}| \geq \underline{T}$ **then**

14: Reassign customer S_j^k to cluster $j_{\min}$.
15: **else**
16: $k = k + 1$.
17: **end if**
18: **else**
19: $k = k + 1$.
20: **end if**
21: **end while**
22: **end for**
23: Check for convergence: if in every cluster, the time windows of any two adjacent customers are no less than the threshold $\underline{T}$, then set converged to True; otherwise, set converged to False.
24: $\ell = \ell + 1$.
25: **end while**

Similarly, after obtaining the assignment decisions from Algorithm 3, one can solve model (1) either with the second-stage cost $Q(\boldsymbol{u}, \boldsymbol{r}, \boldsymbol{z}, \omega)$ to account for the randomness (denoted as KMSS), or without it to obtain solutions in a quick fashion (denoted as KMS). We will compare these two approaches in Section 5.

5. Computational Results

We compare different approaches for the static and dynamic vehicle routing and service scheduling problem using a diverse set of instances generated based on features of real data collected from operating Ford Motor Company's NEMT service in 2019. We conduct in-sample and out-of-sample tests of model (1) and the heuristic-based Algorithms 2 and 3 for handling uncertain service duration and cancellations. In Section 5.1, we describe detailed parameter settings in the baseline case and our experimental setup. We vary parameter choices to conduct sensitivity studies and report in-sample results in Section 5.2, and compare out-of-sample results of the optimization model and clustering-based heuristics in Section 5.3. In Section 5.4, we present the SAA analysis results, and in Sections 5.5 and 5.6, we present the results of applying the rolling horizon approach on small- and large-scale instances, respectively. We use Gurobi 9.0.3 coded in Python 3.6.8 for solving all mixed-integer programming models. Our numerical tests are conducted on a Windows 2012 server with 128 gigabytes (GB) RAM and an Intel 2.2 GHz processor.

5.1. Experimental Design and Setup

We generate customer time windows according to the temporal demand density reported by Ford GoRide Health team, shown in Figure 5, where the x-axis represents the requested pick-up time in hours and the y-axis indicates the kernel density. From Figure 5, vehicles start to operate at 4 a.m. and all services end at 7 p.m., yielding a total operational time of 15 hours. Therefore, $L_j = 900$ minutes for all $j \in J$. For each customer $i \in I$, we sample the earliest pick-up time $\underline{a}_i \in [0, 900]$ following the given density function, and then set $\bar{a}_i = \underline{a}_i + 30$, meaning that each customer has a 30-minute time window.

During the year 2019, the NEMT service was operated in Sterling Heights, Wayne, Southfield, Dearborn, Taylor, and Ann Arbor in Southeast Michigan, mainly for transporting patients who are elderly, disabled, or have chronic disease from and to their medical requests. We display the population estimate, percentage of people who are either over 65 years old or disabled in the six cities in the first three columns of Table 2, based on the most updated information posted by the U.S. Census Bureau (2010). In Table 2, we also calculate the number of target customers and the corresponding demand ratios in each city in the last two columns. In our baseline case, the total

Figure 4. Illustration of Rolling Horizon Timings

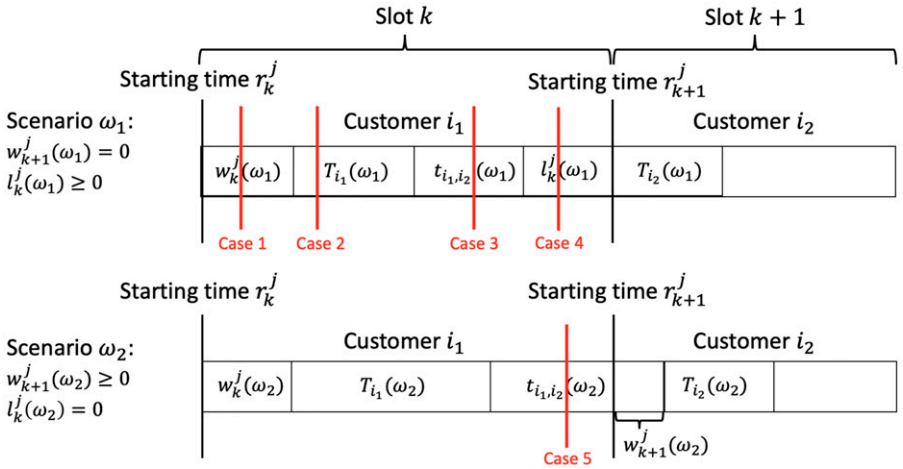
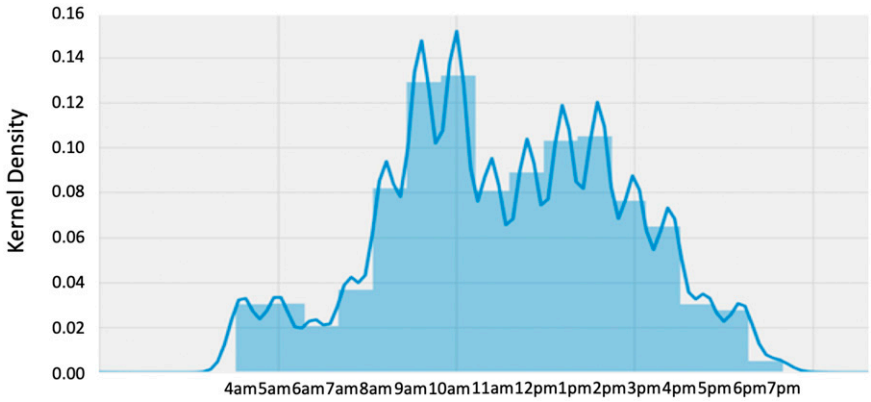


Figure 5. Density of Requested Pick-Up Time During 4 a.m. to 7 p.m. in a Daily Base



number of customers and vehicles in Southeast Michigan are $|I|=100$, $|J|=20$, respectively, and we also decompose the whole service area into six service regions (cities) while distributing all the customers/vehicles to each city according to their demand ratios, with one depot in each city.

We select representative hospitals and senior housing locations in the six cities and mark them in red and blue, respectively, in Figure 6, with a total of 23 hospitals and 48 senior housing locations, from which we can sample O-D pairs of all service requests received. Then we use Google API to calculate the average travel time between each pair of the sampled locations, serving as the empirical mean of the random travel time. For example, we use $\tilde{\tau}_{O_i,D_i}$ to denote the empirical mean of the random travel time τ_{O_i,D_i} from O_i to D_i, which follows a normal distribution $\mathcal{N}(\tilde{\tau}_{O_i,D_i}, \tilde{\tau}_{O_i,D_i} \times \sigma)$ with the standard deviation σ being 0.2 in the baseline case.

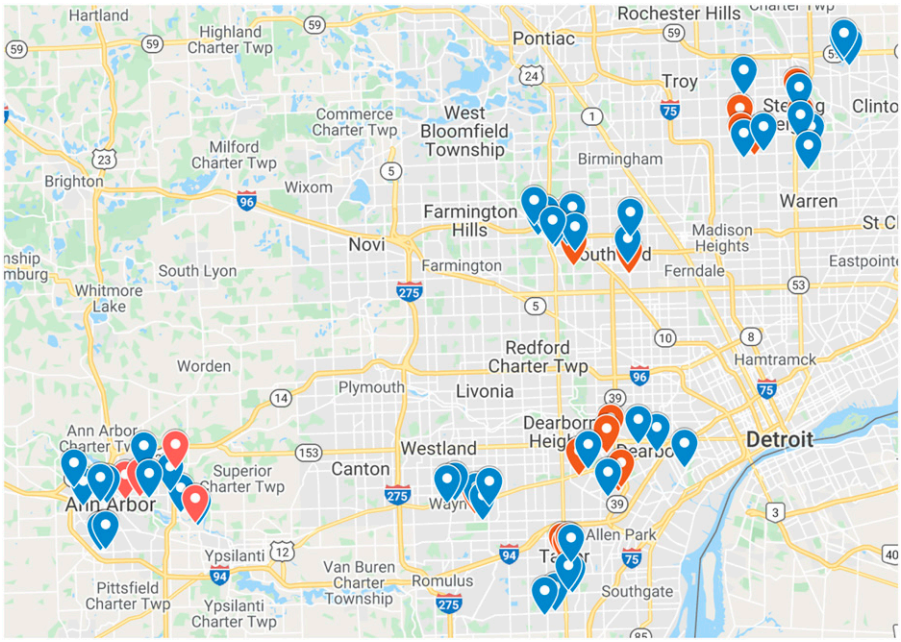
The service in the customers' origins/destinations mainly includes loading/unloading them to/from the vehicle, which takes 20/5 minutes on average in the Ford's NEMT system. Therefore, we let the service duration $\hat{T}_{O_i}$ and $\hat{T}_{D_i}$ follow normal distributions $\mathcal{N}(20, 20 \times \sigma)$ and $\mathcal{N}(5, 5 \times \sigma)$, respectively. Recall that we use $q_i(\omega)$ to denote the cancellation status of customer i in each scenario $\omega \in \Omega$, which equals 1 if the customer shows-up, and 0 otherwise. We sample all the $q_i(\omega)$-values following a Bernoulli distribution with showing-up probability $= 0.89$ as according to Ford, the cancellation rate of all trips is 11% in 2019. According to our discussions with the Ford GoRide Health team, we set the daily operational cost of a vehicle as $c_j = \$240$ for all vehicles $j \in J$ and set the per-minute penalty cost of vehicle being idle, customer waiting, and overtime as $c^l = \$1$, $c^w = \$2$, $c^o = \$10$, respectively. As Ford hires vans and drivers on a daily basis, it does not have significant cost associated with the total travel time and, accordingly, we set the per-minute penalty cost of vehicles' travel time as $c^d = \$0$, but will present the total travel time results for comparing different approaches.

We focus on the operations of NEMT in Ann Arbor, Michigan, in Sections 5.2–5.5, which has $|I| = 100 \times 16\% = 16$ customers and $|J| = 20 \times 16\% = 3$ vehicles in the baseline case according to Table 2. For the in-sample tests of TS-MILP, the number of scenarios is set to $|\Omega| = 10$, and we evaluate solutions given by TS-MILP and clustering-based heuristics on 1,000 independently generated out-of-sample scenarios. Note that in the objective function (3a), we do not penalize the first idle time of each vehicle (i.e., l_0^j), as we can

Table 2. Distributions of Elderly (over 65 Years Old) and Populations with Disability Based on Census Data in Six Cities in Southeast Michigan

City	Population	Percent of customers	Number of customers	Proportion
Sterling Heights	132,964	26%	34,571	29%
Wayne	16,896	28.5%	4,815	4%
Southfield	73,158	31.1%	22,752	19%
Dearborn	94,333	21.2%	19,999	17%
Taylor	61,148	29.4%	17,978	15%
Ann Arbor	121,890	15.6%	19,015	16%

Figure 6. Map of Southeast Michigan with 48 Senior Housing Locations Marked by Blue and 23 Hospitals Marked by Red



always inform the vehicles to start at the requested pick-up time of their first customer. Therefore, we exclude l_0^j in our results when calculating average idle time per vehicle. We compute instances based on the whole Southeast Michigan service zone in Section 5.6 for demonstrating the scalability results of TS-MILP and service clustering heuristics.

5.2. In-Sample Results and Sensitivity Analysis of TS-MILP

Using the baseline setting, we first vary the standard deviation σ from 0.2 to 0.8 to see the effects of the variances of travel time and service duration on in-sample solutions, reported in Table 3. We then vary the showing-up probability for each customer from 0.2 to 0.8 while keeping $\sigma = 0.2$ to illustrate the impacts of service cancellation, presented in Table 4.

In Tables 3 and 4, ID, OT, TT, and WT denote the average idle/overtime/total travel time (in minutes) per vehicle per scenario and average waiting time (in minutes) per customer per scenario across all in-sample scenarios. The last two columns display the overall objective value of model (1) in dollars and the computational time in seconds, respectively. From Tables 3 and 4, when increasing the variance of travel time and service duration, the overall objective values and the average waiting time both increase; when fewer customers cancel their reservations, the overall objective values, average idle time, and waiting time are better. Therefore, to obtain satisfactory quality of service, one key component is to maintain low variance of travel time and service duration and low cancellation rates.

To illustrate how the vehicles' routes and schedules change in response to different customers' show-up probability, we present each vehicle's operational status when $\hat{q}_i = 0.2, 0.8$ in Figure 7, where the x-axis denotes the time during operation with 4 a.m. being 0 and 7 p.m. being 900 minutes, and the y-axis denotes the three vehicle statuses: working, waiting, and idle. From Figure 7, (a)–(c), when customers all show up with a lower probability, the three vehicles start to work at different times of day (i.e., around 7 a.m., 12 p.m., and 5 p.m., respectively), so as to minimize the average idle time. On the contrary, when $\hat{q}_i = 0.8$, the

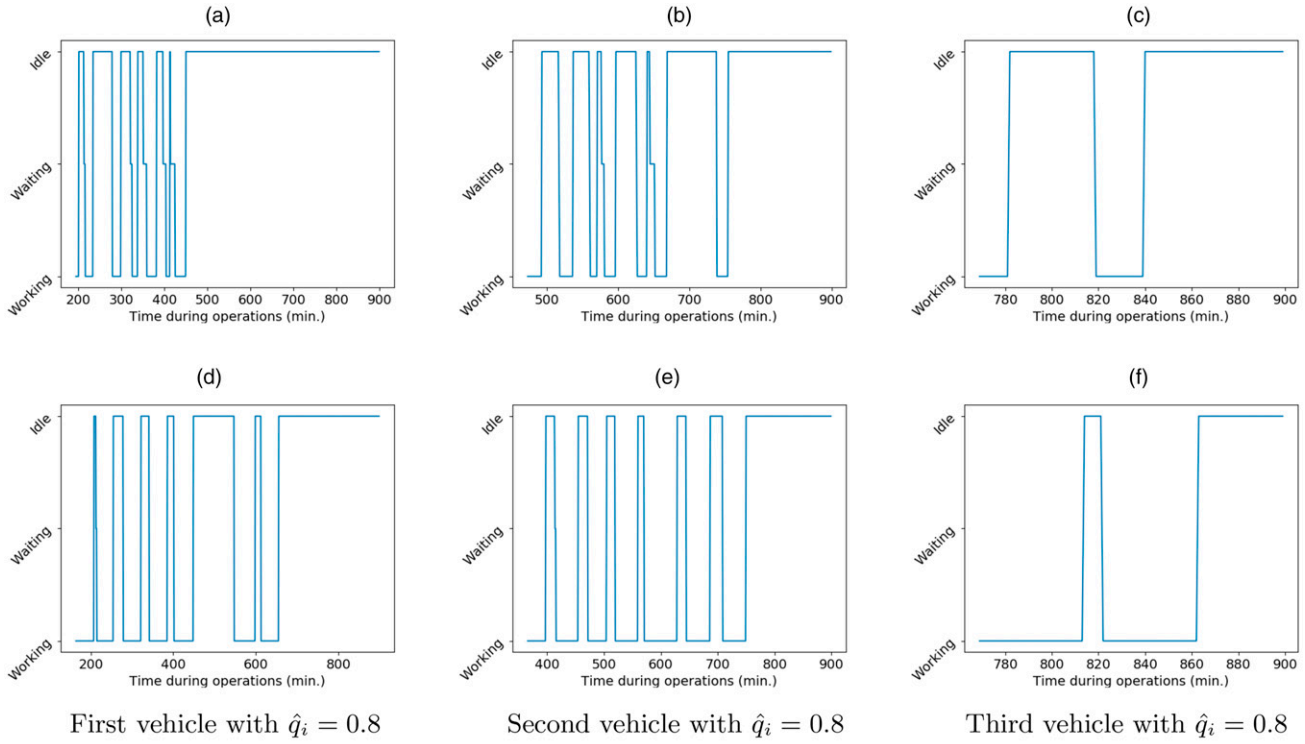
Table 3. In-Sample Results of TS-MILP with Varying σ

σ	ID (min.)	WT (min.)	OT (min.)	TT (min.)	Obj. ($)	Time (sec.)
0.2	218.08	0.41	0.00	217.89	1,387.34	103.00
0.4	218.80	1.13	0.00	222.28	1,412.71	138.71
0.6	222.68	1.96	0.00	221.69	1,450.77	186.22
0.8	229.40	2.69	0.09	220.16	1,496.81	221.67

Table 4. In-Sample Results of TS-MILP with Varying Show-Up Probability $\hat{q}_i$

$\hat{q}_i$	ID (min.)	WT (min.)	OT (min.)	TT (min.)	Obj. ($)	Time (sec.)
0.2	325.04	2.21	0.00	71.50	1,765.91	95.15
0.4	311.35	1.02	0.00	116.87	1,686.78	111.71
0.6	276.51	0.50	0.00	158.02	1,565.64	134.08
0.8	237.51	0.35	0.00	198.84	1,443.62	132.05

Figure 7. Illustration of Vehicle Operational Status with Different $\hat{q}_i$



Notes. (a) First vehicle with $\hat{q}_i = 0.2$; (b) second vehicle with $\hat{q}_i = 0.2$; (c) third vehicle with $\hat{q}_i = 0.2$; (d) first vehicle with $\hat{q}_i = 0.8$; (e) second vehicle with $\hat{q}_i = 0.8$; and (f) third vehicle with $\hat{q}_i = 0.8$.

tasks are distributed to vehicles more evenly, as depicted in Figure 7, (d)–(f).

Next, we fix the baseline setting and vary $|J|$ from 3 to 5 and $|I|$ from 16 to 32 in Table 5, where we mark the optimality gaps of the instances that cannot be solved within 7,200 seconds in the bracket. From Table 5, the waiting time per customer is almost negligible compared with the idle time per vehicle. Moreover, the total travel time per vehicle decreases when we have more vehicles and it almost doubles when we increase the number of customers from 16 to 32. It is also noteworthy that TS-MILP cannot be solved to optimality within two hours when we have 32 customers, which brings the need to use heuristics to derive suboptimal solutions in a quick fashion.

5.3. Out-of-Sample Tests and Results of Different Approaches

We first compare the out-of-sample results of solving model (1) using off-the-shelf solvers directly and using the classical Benders' decomposition algorithm presented in online Appendix B (Benders 1962) in Table 6, where we record the relative gaps between the upper bound (UB) and lower bound (LB) provided by Benders' decomposition and the gaps of its upper bound and the optimal objective value (OPT) in the last two columns, respectively. We terminate the Benders decomposition algorithm in 25 and 50 iterations, and denote them by Benders-25 and Benders-50, correspondingly. From Table 6, the Benders decomposition fails to solve the problem to optimality within

Table 5. In-Sample Results of TS-MILP with Varying $|J|$ and $|I|$

$\|J\|$	$\|I\|$	ID (min.)	WT (min.)	OT (min.)	TT (min.)	Obj. ($)	Time (sec.)
3	16	218.08	0.41	0.00	217.89	1,387.34	103.00
4		235.30	0.58	0.00	168.95	1,919.62	105.56
5		188.30	0.57	0.00	132.70	2,159.62	722.88
3	32	218.13	0.92	0.00	417.87	1,433.20 (24.9%)	7,200.00
4		246.30	1.21	0.00	321.70	2,022.40 (17.98%)	7,200.00
5		220.80	1.75	0.00	257.70	2,416.20 (14.15%)	7,200.00

Table 6. Out-of-Sample Results of Solving TS-MILP via Gurobi and Benders' Decomposition with $|J|=3$ and $|I|=16$

$\|J\|$	$\|I\|$	Method	ID (min.)	WT (min.)	OT (min.)	TT (min.)	Obj. ($)	Time (sec.)	(UB-LB)/LB	(UB-OPT)/OPT
3	16	TS-MILP	221.04	0.53	0.02	215.23	1,400.40	103.00	N.A.	N.A.
		Benders-25	347.60	8.04	0.00	219.24	2,019.91	8,152.35	90.49%	45.07%
		Benders-50	220.60	0.72	0.00	226.08	1,404.94	89,606.44	5.78%	1.25%

24 hours (or 50 iterations), which performs much worse than directly solving it using a state-of-the-art solver. This observation was also revealed in Zhan et al. (2021, p. 103), where the authors pointed out that "the lower and upper bounds are improved quite slowly (in our experiments, the lower bounds hardly increase within several hours, even after hundreds of iterations)." Because of that, we will compare the results of solving model (1) using Gurobi with heuristic approaches in the remaining of the paper.

Before proceeding to other heuristics, we examine the out-of-sample performance of solving model (1) with different in-sample scenarios $|\Omega|$. Table 7 presents the results where we vary $|\Omega|$ from 10 to 100, and the last column indicates the computational time for the in-sample tests. From Table 7, there are no significant result improvements when we increase the sample size, whereas the computational time increases drastically. As a result, we continue using $|\Omega| = 10$ in our subsequent tests.

In Table 8, we compare the out-of-sample results between TS-MILP in Section 3 and the three heuristics proposed in Section 4, namely K-means, K-means with swap, and K-means with swap and the second-stage cost (KM, KMS, and KMSS for short). We also set the maximum number of swapping steps $\ell_{\max} = 5$ and the threshold $\underline{T} = 50$ at default in Algorithm 3. We vary the value of $\underline{T}$ in Table 9.

From Table 8, TS-MILP always obtains the best out-of-sample performance in terms of the idle time and

Table 7. Out-of-Sample Results of TS-MILP with Varying In-Sample Scenario Size $|\Omega|$

$\|J\|$	$\|I\|$	$\|\Omega\|$	ID (min.)	WT (min.)	OT (min.)	TT (min.)	Obj. ($)	Time (sec.)
3	16	10	221.04	0.53	0.02	215.23	1,400.40	103.00
		50	220.97	0.47	0.00	218.15	1,397.97	376.10
		100	221.52	0.40	0.00	220.14	1,397.53	816.26

Table 8. Out-of-Sample Tests and Results of Different Approaches with Varying $|J|$ and $|I|$

$\|J\|$	$\|I\|$	Method	ID (min.)	WT (min.)	OT (min.)	TT (min.)	Obj. ($)	Time (sec.)
3	16	TS-MILP	221.04	0.53	0.02	215.23	1,400.40	103.00
		KM	435.40	2.22	0.24	203.52	2,104.54	0.13
		KMS	434.65	0.12	0.00	207.14	2,027.80	0.13
		KMSS	404.92	0.10	0.00	207.14	1,937.87	5.19
4	16	TS-MILP	237.34	0.68	0.04	167.19	1,932.62	105.56
		KM	425.00	0.59	0.00	153.24	2,678.99	0.15
		KMS	420.29	0.12	0.00	152.01	2,644.99	0.18
		KMSS	390.44	0.11	0.00	152.00	2,525.36	6.66
5	16	TS-MILP	189.93	0.68	0.03	131.06	2,172.79	722.88
		KM	441.20	0.09	0.00	123.15	3,408.86	0.22
		KMS	446.32	0.10	0.00	122.74	3,434.83	0.21
		KMSS	411.29	0.09	0.00	123.16	3,259.55	8.16
3	32	TS-MILP	218.07	1.13	0.00	416.53	1,446.53	7,200.00
		KM	331.07	25.48	8.47	408.24	3,598.21	0.95
		KMS	327.74	14.02	1.55	410.15	2,646.75	0.37
		KMSS	297.54	11.55	1.07	409.87	2,383.79	33.96
4	32	TS-MILP	243.93	1.79	0.00	320.98	2,050.40	7,200.00
		KM	396.71	7.38	0.00	309.88	3,019.21	0.81
		KMS	416.40	2.74	0.00	312.52	2,800.92	0.38
		KMSS	386.27	0.92	0.00	312.52	2,563.91	38.02
5	32	TS-MILP	220.63	2.15	0.00	258.56	2,440.83	7,200.00
		KM	387.86	4.05	0.00	240.26	3,398.61	0.59
		KMS	457.65	1.44	0.00	245.49	3,580.42	0.35
		KMSS	421.92	0.81	0.00	244.91	3,361.18	39.85

Table 9. Out-of-Sample Tests and Results of K-Means with Swap with Varying Threshold $\underline{T}$

$\|J\|$	$\|I\|$	$\underline{T}$ (min.)	ID (min.)	WT (min.)	OT (min.)	TT (min.)	Obj. ($)	Time (sec.)
3	16	30	435.40	2.22	0.24	203.52	2,104.54	0.13
		40	434.80	0.12	0.00	207.62	2,028.24	0.16
		50	434.65	0.12	0.00	207.14	2,027.80	0.13

overall objective values, as it is designed to optimize the expected objectives under uncertainty. All the heuristics perform significantly worse in idle time whereas they slightly improve the waiting time and total travel time. On the other hand, they reduce the computational time from hundreds of seconds to less than 10 seconds for the small-scale instances when $|I| = 16$ based on Ann Arbor. When $|I| = 32$, the heuristics can still solve the problems within 40 seconds, whereas TS-MILP cannot be optimized within two hours. Moreover, KMS improves the overall objective values of KM, which is further reduced by KMSS while maintaining the computational efficiency.

As can be seen from Algorithm 3, the threshold $\underline{T}$ plays an important role in determining the swap between customers and the termination of the algorithm. In Table 9, we fix $|J| = 3$, $|I| = 16$ and test K-means with swap where we vary the threshold $\underline{T}$ from 30 to 50 minutes. With more transit time intentionally left for adjacent customers, all performance metrics are improved, where the improvements from $\underline{T} = 30$ to 40 are much more significant than the ones from 40 to 50. As a result, in the following tests, we continue to fix $\underline{T} = 50$. We also present the results of K-means-based heuristics with different swapping steps $\ell_{\max}$ and different input feature matrix $\{d_i\}_{i=1}^{|I|}$ in online Appendix C.

5.4. Comparison Between Stochastic and Deterministic Approaches

We compare the in-sample and out-of-sample performance of TS-MILP and its deterministic counterpart, where we generate $|\Omega|$ in-sample scenarios to obtain an optimal TS-MILP solution and use the empirical mean of these $|\Omega|$ in-sample scenarios to obtain a deterministic optimal solution. Then we evaluate these two solutions on the same 1,000 out-of-sample scenarios based on the overall objective values.

We generate 10 independent sets of scenario samples, each of size $|\Omega|$, and conduct the tests independently for each set of scenarios. We calculate the average objective values of in-sample and out-of-sample tests in Table 10, and display the box plot in Figure 8.

Table 10. Average Objective Values of In-Sample and Out-of-Sample Tests and Their Gaps (Across 10 Replications) Using TS-MILP and Its Deterministic Counterpart

	TS-MILP IS	TS-MILP OS	Gap 1	DT IS	DT OS	Gap 2
Average	1,408.21	1,420.111	0.85%	1,337.96	1,442.07	7.78%

In both Table 10 and Figure 8, TS-MILP IS, TS-MILP OS, and gap 1 denote the average objective values of in-sample and out-of-sample tests and the gap between the two, whereas DT IS, DT OS, and gap 2 represent those of the deterministic counterpart, respectively.

From Table 10 and Figure 8, the gap between the average in-sample and out-of-sample objective values in TS-MILP is 0.85%, indicating that the optimal solutions computed by $|\Omega|$ in-sample scenarios also perform well in the 1,000 out-of-sample performance. However, in the deterministic counterpart, the gap between the average out-of-sample and in-sample objective values is 7.78%. Although solving the deterministic model with empirical means can return a better in-sample result, the out-of-sample performance could be much worse than the out-of-sample performance of TS-MILP.

5.5. Results of the Rolling Horizon Method on Ann Arbor Instances

In this section, we present results from the rolling horizon approach in Section 3.3 for optimizing real-time vehicle routing and service scheduling. In Algorithm 1, first, model (1) is solved to obtain an initial vehicle-customer assignment. Then, in each period, demand is realized and fulfilled on a first-come, first-served

Figure 8. Box Plot of In-Sample and Out-of-Sample Results and Gaps Across 10 Replications Given by TS-MILP and Its Deterministic Counterpart

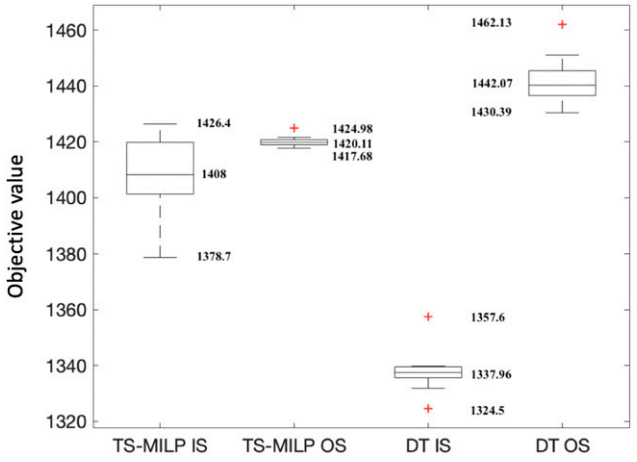


Table 11. Results of Rolling Horizon Algorithm for Different Approaches

$\|J\|$	$\|I\|$	Method	ID (min.)	WT (min.)	OT (min.)	TT (min.)	Obj. ($)	Time (sec.)
3	24	TS-MILP	179.67	2.38	0.00	487.33	1,373.24	55.23
		KM	217.33	4.71	0.00	480.00	1,598.00	0.16
		KMS	224.00	6.08	0.00	473.33	1,684.00	0.15
		KMSS	240.00	0.33	0.00	420.33	1,456.00	3.05
4	24	TS-MILP	250.75	0.25	0.00	383.50	1,975.00	45.05
		KM	220.25	3.63	0.00	450.50	2,015.00	0.14
		KMS	262.75	0.67	0.00	408.00	2,043.00	0.11
		KMSS	259.75	0.54	0.00	379.25	2,025.00	2.63
5	24	TS-MILP	214.00	1.92	0.00	473.40	2,362.00	235.41
		KM	243.80	1.33	0.00	404.00	2,483.00	0.14
		KMS	247.40	0.38	0.00	400.40	2,455.00	0.16
		KMSS	258.20	0.92	0.00	353.60	2,535.00	3.86

basis, and we adaptively change the assignment and schedule plan. Alternatively, one can use the two heuristics to match vehicle-customer pairs in each period, which gives us a combination of Algorithm 1 with Algorithms 2 and 3. We still focus on operations in Ann Arbor, which has 16 initial customers with reservations one-day ahead, and then we set another eight real-time customers who request at least 30 minutes in advance, and their time windows are also drawn from the density function presented in Figure 5. Therefore, we have a total of $|I| = 16 + 8 = 24$ customers, and in Table 11, we present the results of the rolling horizon method combined with different algorithms, where the last column displays the average computational time across all periods.

From Table 11, when we increase the number of vehicles $|J|$ from three to four, the average waiting time decreases while the average idle time increases. The differences between TS-MILP and the heuristic approaches in terms of the overall objective values are much smaller than those in the static setting, and the heuristics even improve the waiting/travel time in some instances. Moreover, the heuristic approaches maintain the computational efficiency by solving all the instances within four seconds, which sheds light on the applicability of these heuristics in dynamic settings.

Next, we present the performance of the rolling horizon method combined with TS-MILP in Table 12, where we vary the sample size $|\Omega|$ from 10 to 100. From Table 12, with more in-sample scenarios, although the average waiting and travel time decrease, vehicles are idle for a longer period of time and the overall objective cost also increases. This is because we do not take into account the uncertainty of future customers when making decisions, and enlarging the sample size for the current stage's uncertainty would not necessarily help in the dynamic environment.

5.6. Results of Large-Scale Operations Using Clustering-Based Heuristics

Having witnessed the efficiency of clustering-based heuristics, we now present the performance of them on large instances based on Southeast Michigan with six operating cities, and vary the number of vehicles from 20 to 40 and the number of customers from 100 to 300 in Table 13. In these instances, we set the maximum number of swapping steps $\ell_{\max} = 5$ in Algorithm 3 for implementing KMS and KMSS.

From Table 13, when increasing the number of customers, the idle time per vehicle decreases while the waiting time per customer and the overtime per vehicle increase drastically. The waiting time and overtime also drop significantly with doubled vehicles. Moreover, KMSS improves the performance drastically by shortening waiting time per customer and overtime per vehicle.

Next, we present the results of using the rolling horizon method with KM, KMS, and KMSS for operating Southeast Michigan in Table 14 and set the number of dynamically arrived customers as half of the number of customers with reservations, such that the total $|I|$ ranges from 150 to 450. Comparing Table 14 with Table 13, the idle time per vehicle decreases, while the travel time per vehicle almost doubles as we include dynamically arriving customers. The computational time also decreases as we average among all periods

Table 12. Results of Rolling Horizon Algorithm Solved by TS-MILP with Varying In-Sample Scenario Size $|\Omega|$

$\|J\|$	$\|I\|$	$\|\Omega\|$	ID (min.)	WT (min.)	OT (min.)	TT (min.)	Obj.	Time (sec.)
3	24	10	179.67	2.38	0.00	487.33	1,373.24	55.23
		50	221.67	1.21	0.00	443.00	1,443.08	92.15
		100	263.33	0.50	0.00	403.33	1,534.00	162.32

Table 13. Performance of Clustering-Based Heuristics on Large Instances of Southeast Michigan

$\|J\|$	$\|I\|$	Method	ID (min.)	WT (min.)	OT (min.)	TT (min.)	Obj. ($)	Time (sec.)
20	100	KM	335.38	7.43	0.69	173.98	13,129.87	0.30
		KMS	412.41	0.30	0.38	174.73	13,185.72	0.22
		KMSS	384.06	0.19	0.38	174.73	12,595.56	12.35
20	200	KM	286.14	42.32	31.58	336.90	33,764.87	0.93
		KMS	284.15	12.58	14.13	350.38	18,343.01	0.55
		KMSS	257.90	7.33	11.47	350.46	15,184.04	85.86
20	300	KM	207.27	126.76	137.80	517.21	112,561.06	2.42
		KMS	168.20	85.28	88.44	535.63	77,017.12	1.29
		KMSS	148.42	69.61	85.04	534.65	66,542.21	335.05
40	100	KM	375.08	2.54	0.21	89.76	25,194.28	0.34
		KMS	411.96	0.04	0.19	90.24	26,163.24	0.28
		KMSS	382.86	0.02	0.19	90.24	24,995.25	14.72
40	200	KM	361.19	13.64	8.71	171.42	32,986.82	1.12
		KMS	417.96	0.89	4.58	172.40	28,508.52	0.72
		KMSS	388.05	0.39	4.58	172.40	27,111.77	129.63
40	300	KM	341.93	42.30	26.39	251.73	59,211.74	2.20
		KMS	365.13	3.17	5.16	253.41	28,168.64	1.15
		KMSS	335.96	0.98	4.44	253.41	25,404.40	293.83

while later periods having much smaller sizes can be solved relatively quickly.

6. Conclusion

In this paper, we modeled a TS-MILP for solving a static and dynamic vehicle routing and scheduling problem, where we applied the rolling horizon method to solve the dynamic variant. To speed up computation, we proposed K-means-based heuristics to cluster geographically similar customers and then separately decide a routing and scheduling plan in each cluster. We conducted various experiments based on data collected by Ford Motor Company's GoRide Health team. Results indicate that the clustering-based heuristics can solve large-scale instances efficiently and effectively.

For future research, one possibility is to design a branch-and-price algorithm for solving the TS-MILP that, unlike many VRP variants, involves nonnegligible service duration at each customer location and also multiple parameter uncertainties. The development of appropriate pricing subproblems to generate route-and-schedule-combined columns is challenging. Moreover, as the pick-up/drop-off locations are typically hospitals and senior apartments, a large portion of customers may share similar routes or the same origins/destinations. It would be beneficial to pool these customers, which may reduce the total operational cost but increase individuals' waiting time.

Table 14. Results of the Rolling Horizon Approach with Clustering-Based Heuristics on Large-Scale Instances

$\|J\|$	$\|I\|$	Method	ID (min.)	WT (min.)	OT (min.)	TT (min.)	Obj. ($)	Time (sec.)
20	150	KM	204.35	8.71	4.00	425.30	12,301.00	0.15
		KMS	211.65	2.47	3.95	448.45	10,563.00	0.13
		KMSS	213.80	1.16	0.15	414.70	9,454.00	5.37
20	300	KM	158.15	17.38	17.20	551.50	21,833.00	0.33
		KMS	163.80	14.41	15.20	556.40	19,762.00	0.31
		KMSS	129.90	6.68	9.45	557.40	13,296.00	38.40
20	450	KM	112.05	27.82	25.55	612.70	37,191.00	0.69
		KMS	121.40	25.62	20.80	606.80	34,442.00	0.72
		KMSS	87.45	14.90	34.40	641.10	26,839.00	135.94
40	150	KM	197.98	1.95	1.50	474.48	18,703.00	0.26
		KMS	200.68	0.52	1.63	447.88	18,433.00	0.23
		KMSS	210.08	0.13	0.08	385.18	18,073.00	8.17
40	300	KM	184.33	6.36	7.00	480.63	23,589.00	0.61
		KMS	202.33	1.69	7.70	455.58	21,789.00	0.51
		KMSS	205.98	0.51	3.45	409.10	19,525.00	54.16
40	450	KM	168.75	9.96	7.65	491.45	28,378.00	1.24
		KMS	189.33	2.30	9.28	485.08	22,955.00	0.88
		KMSS	169.30	1.71	4.45	472.93	19,688.00	134.12

Acknowledgments

The authors sincerely thank Yuhai Hu and Heng Liu at Ford Motor Company for helpful discussions.

References

Allaoua H, Borne S, Létocart L, Calvo RW (2013) A matheuristic approach for solving a home health care problem. *Electronic Notes Discrete Math.* 41:471–478.

Bard JF, Shao Y, Qi X, Jarrah AI (2014) The traveling therapist scheduling problem. *IIE Trans.* 46(7):683–706.

Benders JF (1962) Partitioning procedures for solving mixed-variables programming problems. *Numer. Math.* 4(1):238–252.

Bennett AR, Erera AL (2011) Dynamic periodic fixed appointment scheduling for home health. *IIE Trans. Healthcare Systems Engrg.* 1(1):6–19.

Bent RW, Van Hentenryck P (2004) Scenario-based planning for partially dynamic vehicle routing with stochastic customers. *Oper. Res.* 52(6):977–987.

Bent RW, Van Hentenryck P (2007) Waiting and relocation strategies in online stochastic vehicle routing. *Internat. Joint Conf. Artificial Intelligence* 7:1816–1821.

Berbeglia G, Cordeau J-F, Laporte G (2010) Dynamic pickup and delivery problems. *Eur. J. Oper. Res.* 202(1):8–15.

Berbeglia G, Cordeau J-F, Laporte G (2012) A hybrid tabu search and constraint programming algorithm for the dynamic dial-a-ride problem. *INFORMS J. Comput.* 24(3):343–355.

Berg BP, Denton BT, Erdogan SA, Rohleder T, Huschka T (2014) Optimal booking and scheduling in outpatient procedure centers. *Comput. Oper. Res.* 50:24–37.

Bertsimas DJ, Simchi-Levi D (1996) A new generation of vehicle routing research: Robust algorithms, addressing uncertainty. *Oper. Res.* 44(2):286–304.

Bertsimas DJ, Van Ryzin G (1991) A stochastic and dynamic vehicle routing problem in the Euclidean plane. *Oper. Res.* 39(4): 601–615.

Bertsimas DJ, Van Ryzin G (1993) Stochastic and dynamic vehicle routing in the Euclidean plane with multiple capacitated vehicles. *Oper. Res.* 41(1):60–76.

Bertsimas D, Jaillet P, Martin S (2019) Online vehicle routing: The edge of optimization in large-scale applications. *Oper. Res.* 67(1):143–162.

Bräysy O, Gendreau M (2005a) Vehicle routing problem with time windows, Part I: Route construction and local search algorithms. *Transportation Sci.* 39(1):104–118.

Bräysy O, Gendreau M (2005b) Vehicle routing problem with time windows, Part II: Metaheuristics. *Transportation Sci.* 39(1): 119–139.

Bryant M (2019) Ford enters NEMT space with national rollout of GoRide Health. HealthCareDrive (May 9), https://www.healthcaredive.com/news/ford-enters-nemt-space-with-national-rollout-of-goride-health/554413/.

Cappanera P, Scutellà MG (2015) Joint assignment, scheduling, and routing models to home care optimization: A pattern-based approach. *Transportation Sci.* 49(4):830–852.

Carello G, Lanzarone E (2014) A cardinality-constrained robust model for the assignment problem in home care services. *Eur. J. Oper. Res.* 236(2):748–762.

Centers for Disease Control and Prevention (2020) People who are at higher risk for severe illness. Accessed August 11, 2021, https://www.cdc.gov/coronavirus/2019-ncov/need-extra-precautions/people-with-medical-conditions.html.

Cömert SE, Yazgan HR, Sertvuran I, Şengül H (2017) A new approach for solution of vehicle routing problem with hard time window: An application in a supermarket chain. *Sadhana* 42(12):2067–2080.

Cordeau J-F, Laporte G (2007) The dial-a-ride problem: Models and algorithms. *Ann. Oper. Res.* 153(1):29–46.

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

Denton B, Gupta D (2003) A sequential bounding approach for optimal appointment scheduling. *IIE Trans.* 35(11):1003–1016.

Desrochers M, Desrosiers J, Solomon M (1992) A new optimization algorithm for the vehicle routing problem with time windows. *Oper. Res.* 40(2):342–354.

Dickey MR (2018). Ford launches on-demand medical transportation service. TechCrunch (April 18), https://techcrunch.com/2018/04/18/ford-launches-on-demand-medical-transportation-service/.

Dror M, Laporte G, Trudeau P (1989) Vehicle routing with stochastic demands: Properties and solution frameworks. *Transportation Sci.* 23(3):166–176.

Erdogan SA, Denton B (2013) Dynamic appointment scheduling of a stochastic server with uncertain demand. *INFORMS J. Comput.* 25(1):116–132.

Fikar C, Hirsch P (2017) Home healthcare routing and scheduling: A review. *Comput. Oper. Res.* 77:86–95.

Fukasawa R, Longo H, Lysgaard J, De Aragão MP, Reis M, Uchoa E, Werneck RF (2006) Robust branch-and-cut-and-price for the capacitated vehicle routing problem. *Math. Programming* 106(3): 491–511.

Gupta D, Denton B (2008) Appointment scheduling in healthcare: Challenges and opportunities. *IIE Trans.* 40(9):800–819.

Heching A, Hooker JN, Kimura R (2019) A logic-based Benders approach to home healthcare delivery. *Transportation Sci.* 53(2):510–522.

Jain AK (2010) Data clustering: 50 years beyond K-means. *Pattern Recognition Lett.* 31(8):651–666.

Jiang R, Shen S, Zhang Y (2017) Integer programming approaches for appointment scheduling with random no-shows and service durations. *Oper. Res.* 65(6):1638–1656.

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

Lanzarone E, Matta A (2014) Robust nurse-to-patient assignment in home care services to minimize overtimes under continuity of care. *Oper. Res. Health Care* 3(2):48–58.

Laporte G (1992) The vehicle routing problem: An overview of exact and approximate algorithms. *Eur. J. Oper. Res.* 59(3):345–358.

Laporte G (2007) What you should know about the vehicle routing problem. *Naval Res. Logist.* 54(8):811–819.

Moon TK (1996) The expectation-maximization algorithm. *IEEE Signal Processing Magazine* 13(6):47–60.

Nickel S, Schröder M, Steeg J (2012) Mid-term and short-term planning support for home healthcare services. *Eur. J. Oper. Res.* 219(3):574–587.

Parragh SN, Doerner KF, Hartl RF (2008) A survey on pickup and delivery problems. *J. für Betriebswirtschaft.* 58(1):21–51.

Pillac V, Gendreau M, Guéret C, Medaglia AL (2013) A review of dynamic vehicle routing problems. *Eur. J. Oper. Res.* 225(1):1–11.

Pinedo M (2012) *Scheduling: Theory, Algorithms, and Systems*, vol. 5 (Springer-Verlag, New York).

Powell WB (1996) A stochastic formulation of the dynamic assignment problem, with an application to truckload motor carriers. *Transportation Sci.* 30(3):195–219.

Powell WB, Towns MT, Marar A (2000) On the value of optimal myopic solutions for dynamic routing and scheduling problems in the presence of user noncompliance. *Transportation Sci.* 34(1):67–85.

Ralphs TK, Kopman L, Pulleyblank WR, Trotter LE (2003) On the capacitated vehicle routing problem. *Math. Programming* 94(2-3): 343–359.

Rasmussen MS, Justesen T, Dohn A, Larsen J (2012) The home care crew scheduling problem: Preference-based visit clustering and temporal dependencies. *Eur. J. Oper. Res.* 219(3):598–610.

Savelsbergh MW, Sol M (1995) The general pickup and delivery problem. *Transportation Sci.* 29(1):17–29.

Simao HP, Day J, George AP, Gifford T, Nienow J, Powell WB (2009) An approximate dynamic programming algorithm for large-scale fleet management: A case application. *Transportation Sci.* 43(2):178–197.

Toth P, Vigo D (2002) Models, relaxations and exact approaches for the capacitated vehicle routing problem. *Discrete Appl. Math.* 123(1-3):487–512.

U.S. Census Bureau (2010) QuickFacts data table on U.S. census website. Accessed August 11, 2021, https://www.census.gov/quickfacts/fact/table/MI/PST045219#.

Yuan B, Liu R, Jiang Z (2015) A branch-and-price algorithm for the home healthcare scheduling and routing problem with stochastic service times and skill requirements. *Internat. J. Production Res.* 53(24):7450–7464.

Zacharias C, Pinedo M (2014) Appointment scheduling with no-shows and overbooking. *Production Oper. Management* 23(5):788–801.

Zhan Y, Wan G (2018) Vehicle routing and appointment scheduling with team assignment for home services. *Comput. Oper. Res.* 100:1–11.

Zhan Y, Wang Z, Wan G (2021) Home service routing and appointment scheduling with stochastic service times. *Eur. J. Oper. Res.* 288(1):98–110.