Designing and Optimizing an Integrated Car-and-ride Sharing System for Mobilizing Underserved Populations

Miao Yu
Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, MI miaoyu@umich.edu,

Siqian Shen
Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, MI siqian@umich.edu

The fast-growing carsharing and ridesharing businesses are generating economic benefits and societal impacts in the modern society. However, they possess limitations to cover diverse populations and areas. In this paper, we consider an integrated car-and-ride sharing system and optimize its operations to improve the mobility of underserved populations under transportation scarcity. We consider two types of demands: Type 1 drivers who rent shared cars and Type 2 users who need rides from Type 1 drivers. We propose a two-phase model to maximize demand coverage. In Phase I, we match Type 1 drivers and Type 2 users based on a spatial-temporal network; in Phase II, we optimize pick-up and delivery routes and schedules for matched Type 1 and Type 2 users under random driving time via solving a two-stage stochastic mixed-integer programming model. We minimize the total travel cost plus expected penalty cost of users’ waiting time and system overtime, and develop a decomposition algorithm for improving computational efficiency. We conduct computational studies on various instances using census data of underserved populations in Washtenaw County, Michigan. Our results show high demand fulfillment rates and effective matching and scheduling with low risk of waiting and overtime. The integrated system also achieves better performance if we allow vehicle relocation.

Key words: Carsharing and ridesharing, public service for underserved populations, two-stage stochastic integer programming, integer L-shaped method

History:

1. Introduction

Underserved communities, whose residents are primarily low-income and African American, are experiencing a significant health disparity in the United States (Betancourt et al. 2003). Unequal
distribution of public service resources is a main cause, leading to unfairness of access to health care and transit (see Social Determinants of Health 2008, Giang et al. 2008). One major factor of social inequality and health-related disparities is lack of transportation, which creates barriers to health-enhancing resources such as employment (Lichtenwalter et al. 2006), healthy food (Walker et al. 2010), and health care (Syed et al. 2013). Underserved populations typically have high volumes of transportation needs, hindered by lower rates of vehicle ownership, financial barriers to traditional taxi service, and limited public transportation resource. For example, in Metropolitan Detroit, Michigan, 40% of residents do not own cars, and another 40% do not have access to vehicles (Firth 2016). Even worse, the city has reduced and eliminated much of its public bus service due to financial crisis (McLaughlin 2015).

In recent years, shared-mobility has shown its flexibility and strength in providing convenience to personal travel and reducing congestions on public roads by reducing car ownership (Shaheen et al. 2015, Martin 2016). With increasing concerns about climate change, congestions, and fossil fuel dependency, shared-mobility attracts more attention and is undergoing a fast rise in popularity and industrial growth (Shaheen et al. 2015, Chan and Shaheen 2012, Millard-Ball 2005). Companies with shared-mobility business have established their success with fast-growing users. For example, as Year 2016, Zipcar has exceeded its 1 million paid membership milestone and eliminated the need for personally owned vehicles by over 400,000 worldwide (Zipcar 2016). Via efficiently resource pooling, shared-mobility provides practical solutions to people’s daily living needs.

Two main shared-mobility forms are carsharing and ridesharing. The former is a service that provides users with access to car rental service on hourly basis (Millard-Ball 2005). Two examples are Zipcar and Car2Go, which implement reservation-based and free-float-based carsharing services, respectively. Meanwhile, ridesharing, a service initially aiming to group travelers with common itineraries (Chan and Shaheen 2012), has evolved significantly as the development of smartphone and GPS technologies. Ridesharing companies, such as Uber, Lyft, and Didi Chuxing, are offering both economic and high-quality ridesharing services worldwide (Dias et al. 2017).
Although both carsharing and ridesharing provide useful alternatives to owning personal cars, they consider different user groups. Carsharing customers usually book the service for longer trips for running errands with multiple short stops. To use carsharing service, customers are required to meet driving eligibility requirements. Differently, ridesharing service targets on customers with short, and often one-time ride need. It especially benefits customers who are not able to drive, unfamiliar with city transit system, and/or commuting with poor transit access.

Despite that each service provider has different markets to cover, they both target on areas with high population densities and moderate-income populations. Opening a service area with less population density may increase vehicle relocation cost in carsharing business and increase ridesharing driver’s idle time between two successive rides. However, this means that people who are not living in such areas would be left behind by these convenient services. Indeed, current models of shared-mobility service are not optimized for the needs of underserved populations. Residents from underserved communities are experiencing financial, technical, skill-based, informational and social barriers to the use of shared-mobility service (Ge et al. 2016).

1.1. Problem Description

In this paper, we propose a car-and-ride sharing (CRS) system, in which carsharing and ridesharing services are integrated and co-provided to heterogeneous users, to boost the mobility of underserved populations. Moreover, the system is financially and operationally self-sustained such that customers with ridesharing demands are served by drivers with carsharing demands.

We consider a service region that mainly consists of underserved populations, such as jobless, elderly, and disabled, to whom transportation is a scarce resource and there exists vast wealth gap across racial groups. We partition the service region into zones, and each zone represents a community. Shared vehicles are located a priori in some designated parking spaces in each zone. We classify two types of customers: Type 1 customers who want shared cars for private use and Type 2 customers who have ridesharing demand but cannot drive. For example, Type 1 customers could be jobless but healthy adults and Type 2 customers could be the disabled and/or elderly.
We aim to build a self-sustained CRS system to encourage Type 1 drivers to “serve” Type 2 users at their capability to gain the discount for car rental cost. To implement the system, we develop a reservation system online that enables Type 1 customers to reserve cars by providing their car rental information along with available time windows for serving others, and Type 2 customers to input their ridesharing requests with origin/destination of their trips and time windows for pick-up. We assume that demands arise at discrete time periods over a finite horizon. In each period, the unfulfilled demand is immediately lost. Our goal is to maximize the fulfillment of two types of demands while self-covering the operational cost using revenue from both car rental and ridesharing.

There are two primary carsharing forms: One-way and round-trip. For one-way service, a customer can rent a car from one location and return the car at a different location; however, for round-trip service, customers are required to return cars to their pick-up locations. In practice, one-way service is harder to operate considering the necessity to re-balance car volumes at different locations. We consider both one-way and round-trip carsharing services to Type 1 customers in our later proposed models. Similarly, ridesharing has two primary forms: Reservation and on-demand. From customers’ perspective, on-demand ridesharing service is more appealing since it provides an instant solution for customer’s travel needs. In this paper, due to special properties of underserved populations such that they are more time-wise flexible, we consider a system using reservation-based ridesharing. We have all customers input their carsharing and ridesharing demand from smartphone Apps before planning vehicle assignment and routes. (See an example development available at http://communityride.us/, which is about to be implemented in Detroit communities to conduct community-based participatory research by the authors and their collaborators.)

1.2. Solution Approach Overview

Here we propose a two-phase approach for the overall design and operations of the CRS system: In Phase I, we plan for carsharing operations and maximize satisfied demand by accepting enough Type 1 drivers to serve Type 2 users, while maintaining the cost self-sustainability of the system; in
Phase II, we plan for ridesharing by optimizing routes and pick-up schedules. Moreover, we consider stochastic travel time and service time, and propose a two-stage stochastic programming variant of Phase II model with an expected penalty cost from the waiting of customers and overtime of the system. The workflow of proposed CRS system is demonstrated in Figure 1.

<table>
<thead>
<tr>
<th>Collect Reservations:</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Type 1 users:</td>
</tr>
<tr>
<td>- Service Zone</td>
</tr>
<tr>
<td>- Available work time</td>
</tr>
<tr>
<td>• Type 2 users:</td>
</tr>
<tr>
<td>- Origin-Destination</td>
</tr>
<tr>
<td>- Time windows for pick-up</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>System Optimization:</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Phase I Model</td>
</tr>
<tr>
<td>- Carshare approval</td>
</tr>
<tr>
<td>• Phase II Model</td>
</tr>
<tr>
<td>- Rideshare approval</td>
</tr>
<tr>
<td>- Routing and ride-pooling</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Result Notification:</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Type 1 users:</td>
</tr>
<tr>
<td>- Shared car approval</td>
</tr>
<tr>
<td>- Which Type 2 users to serve</td>
</tr>
<tr>
<td>• Type 2 users:</td>
</tr>
<tr>
<td>- Shared ride approval</td>
</tr>
<tr>
<td>- Scheduled pickup</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>System Operations and Evaluation:</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Demand fulfillment rate</td>
</tr>
<tr>
<td>• Type 2 user wait time</td>
</tr>
<tr>
<td>• Total system overtime</td>
</tr>
</tbody>
</table>

Figure 1  Workflow of the proposed CRS system

1.3. Contributions and Main Results

This paper focuses on developing a new CRS system for underserved communities. We aim to optimize resource pooling, supply-demand matching and shared service scheduling for multiple underserved populations. The contributions of this paper are three-fold.

- We propose and testify a new design concept of mobility sharing that integrates both carsharing and ridesharing services. We focus on underserved populations, who are currently being left behind by the shared-mobility industry. Taking into account their special transportation needs, the system will provide prominent solutions for practical use and step towards improving social equity and closing health gap.

- The classification of two types of customers leads to self-sustained operations of the system via supply-demand matching within each service region. However, combining two types of services results in substantial operational challenges. We decompose the problem into two phases, and successfully reduce operational complexity while obtaining good-quality results shown by our computational studies. Our numerical results show promising computational performance and high
quality of service. Furthermore, we show the importance of vehicle relocation in CRS system, as we can achieve higher demand fulfillment rates even we only allow round trips of shared cars.

- We extend the basic model to a stochastic programming model to capture the randomness of vehicle travel time and service time. We develop efficient decomposition algorithms for optimizing the stochastic program with finite samples of the uncertain parameter. When applied to synthetic data, our proposed model achieves significant operations efficiency (measured by waiting and overtime of the system), compared with the deterministic model using the expected values.

1.4. Organization of the Paper

The remainder of this paper is organized as follows. In Section 2, we review the literature related to carsharing, ridesharing, and their optimization models and algorithms. In Section 3, we describe the problem and formulate a two-phase approach to optimize the integrated CRS design and operations. We further present a stochastic programming variant of the Phase II model by considering random vehicle travel time and time of serving Type 2 customers. In Section 4, we develop an efficient way to decompose the proposed models and derive valid cuts based on Benders decomposition and integer L-shaped method. In Section 5, we demonstrate the effectiveness of the CRS system and present computational results using instances generated based on demographics of underserved populations in Washtenaw County, Michigan. We conclude the paper and present future research directions in Section 6.

2. Literature Review

Different mathematical optimization approaches have been applied to existing problems related to carsharing service at strategic, tactical and operational levels. Laporte et al. (2015) classify literature in shared-mobility under five main topics: Station location, fleet dimensioning, station inventory, rebalancing incentives, and vehicle repositioning. Nourinejad and Roorda (2014) propose a dynamic optimization-simulation model to study the relationship between fleet size and reservation time (i.e., the time between reservation and picking up vehicles). Febbraro et al. (2012) propose an optimal relocation policy to address vehicle relocation problem. Boyaci et al. (2015)
determine the optimal fleet size and charging station location for electric vehicle carsharing. Nair
and Miller-Hooks (2014) use a bi-level mixed-integer linear program to optimize station location
and capacity, as well as vehicle inventories. To handle vehicle imbalance in the one-way carsharing
system, Kek et al. (2009) introduce a spatial-temporal network to model the movement of the
vehicle and determine the workforce needed for relocation. Similar approaches have also been used
by de Almeida Correia and Antunes (2012) to optimize parking locations, and by Fan (2014) to
optimize the allocation and relocation in carsharing systems that allow one-way car rental.

For ridesharing service, most literature focus on fleet management. Agatz et al. (2012) provide
a comprehensive survey of optimization approaches for dynamic ridesharing. Pavone et al. (2012)
focus on fluid approximation method and Zhang and Pavone (2016) propose queuing-based methods
to model and analyze ridesharing systems.

To match ridesharing requests to available cars, as well as to route shared vehicles, most papers
consider the problem as variants of vehicle routing problem (VRP), where a fleet of vehicles are
dispatched from depots to visit all the nodes in an underlying network with time windows con-
straints. Toth and Vigo (2014) summarize the formulations and solution approaches for VRP
variants, including multi-depot VRP, VRP with time windows, and VRP with simultaneous pickup
and delivery, etc. Existing exact solution approaches for VRP are mostly based on branch-and-cut
(B&C) (Lysgaard et al. 2004, Achuthan et al. 2003, Ralphs et al. 2003) and branch-cut-and-price
(BCP) (Fukasawa et al. 2006, Pecin et al. 2014). Yu et al. (2017) consider a VRP variant with com-
patibility constraints that minimize the system makespan and propose an $O(\log n)$—approximation
algorithm to solve the problem efficiently.

For problems in shared-mobility under uncertainty, He et al. (2016) employ a distributionally
robust optimization approach to optimize service zone selection, by assuming uncertain carsharing
demand and fuel price with ambiguously known distribution. Lu et al. (2018) study a carsharing
fleet allocation problem with stochastic one-way and round-trip demands. They propose a two-stage
stochastic program to minimize the total cost of parking lots/permits and car allocation, as well as
penalty cost from unfulfilled demand. Taş et al. (2013) develop heuristic approaches for VRP with soft time windows and stochastic travel time based on tabu search method. In addition to routing cost, they also consider the cost of late and early arrivals. To the best of our knowledge, we are the first to propose optimization-based models and approaches for popularizing shared-mobility service in underserved communities.

3. Problem Formulations

For the proposed CRS system, we first formulate a deterministic two-phase model for operating the proposed reservation-based CRS system, and then derive a stochastic variant of Phase II model to address the uncertainties of travel time and service durations.

Consider a fleet of $K$ vehicles for serving a set $I$ of zones. Let $L = \{1, 2, \ldots, |L|\}$ be a set of reservations received from Type 1 drivers. Each $l \in L$ is associated with a tuple, $(o_l, d_l, [s_l, t_l], [g_l, h_l])$, which includes the pick-up zone $o_l \in I$, return zone $d_l \in I$ of a rental car, time window $[s_l, t_l]$ during which a car is needed, and the time window $[g_l, h_l]$ during which the Type 1 driver is able to provide ridesharing service to some Type 2 users. Let $J$ be a set of ridesharing demand from Type 2 (non-driver) users. Each $j \in J$ is associated with a tuple, $(o'_j, d'_j, e_j, g'_j, h'_j)$, where $o'_j$ and $d'_j$ represent the trip’s origin and destination, respectively, $e_j$ is the total service time needed (including driving time from $o_j$ to $d_j$, time of loading passengers or goods, etc.), and $[g'_j, h'_j]$ is the available time window for picking up the corresponding Type 2 user at $o'_j$. We define a binary parameter vector $w = (w_{jl}, j \in J, l \in L)^T$, where $w_{jl} = 1$ if ridesharing $j \in J$ can be served by carsharing trip $l \in L$ and 0 otherwise. We set $w_{jl} = 1$ if $|[g'_j, h'_j] \cap [g_l, h_l]| \geq e_j$, meaning that Type 2 user $j \in J$ can be served within the time window specified by Type 1 driver $l \in L$.

For each Type 1 demand $l \in L$, we charge $r^\text{car}_l$ per period of car use (which depend on the pick-up location and drop-off location). We also pay $r^\text{drive}_l$ for each time period a driver serves Type 2 users. For each Type 2 demand $j \in J$, we charge $r^\text{ride}_j$ per period dependent on the origin, destination, and time window needing the service. A car in use will incur a service cost $c^\text{ser}$ (including maintenance, insurance, and other types of cost) per period to the system operation and will incur an idle cost $c^\text{idle}_l$ (including $c^\text{ser}$ and parking cost) if it sits idle at zone $i \in I$. 
3.1. Phase I: Carsharing Planning and Operations

Given service requests from Type 1 drivers and Type 2 users, we first implement Phase I to decide which requests to fulfill and pass them to Phase II. Specifically, we determine where to allocate \( K \) cars and decide which Type 1 and Type 2 users’ requests to be passed to Phase II for further routing and scheduling. We construct a spatial-temporal network, \( G = (N, A) \), to capture carsharing reservations from Type 1 drivers over \( T \) periods, where each node \( n_{it} \in N \) represents a zone \( i \in I \) at period \( t \in \{0, 1, \ldots, T\} \). We partition \( A \) into three types of arcs, \( A = A^{\text{travel}} \cup A^{\text{idle}} \cup A^{\text{rel}} \), as follows.

- **Travel arcs** \((n_{it}, n_{i't'}) \in A^{\text{travel}}\) are created for each Type 1 demand \( l \in L \) where \( i = o_l, i' = d_l, t = s_l \) and \( t' = t_l \). The amount of flows on arc \( a_l \in A^{\text{travel}} \) indicates a vehicle being rented by Type 1 driver \( l \in L \). We set \( f_{al} = (t_l - s_l)(r_{i'\text{car}} - e^{\text{set}}) - r_{i\text{drive}}(h_l - g_l) \) as the unit flow revenue and \( u_{al} = 1 \) as the capacity of arc \( a_l \in A^{\text{travel}} \), for all \( l \in L \).

- **Idle arcs** \((n_{it}, n_{it+1}) \in A^{\text{idle}}\) are created for \( i \in I \) and \( t \in \{0, 1, \ldots, T - 1\} \). The amount of flows on arc \( (n_{it}, n_{it+1}) \in A^{\text{idle}} \) indicates the number of vehicles being idle in zone \( i \) from \( t \) to \( t + 1 \). We set unit flow revenue and capacity of arc \( a \in A^{\text{idle}} \) as \( f_a = -c_i^{\text{idle}} \) and \( u_a = K \), respectively.

- **Relocation arcs** \((n_{it}, n_{i't'}) \in A^{\text{rel}}\) are created for \( i \neq i' \in I \) and \( t' > t \in \{0, 1, \ldots, T - 1\} \). The amount of flows on \((n_{it}, n_{i't'}) \in A^{\text{rel}}\) indicates the number of vehicles being relocated from zone \( i \) at \( t \) to zone \( i' \) at \( t' \). We set unit flow revenue and capacity of arc \( a \in A^{\text{rel}} \) as \( f_a = -c_i^{\text{rel}}(t' - t) \) and \( u_a = K \), respectively.

We define integer decision vector \( x = (x_i, \ i \in I)^T \) where \( x_i \) is the number of cars initially located in zone \( i \in I \), integer decision vector \( y = (y_a, \ a \in A)^T \) where \( y_a \) indicates the amount of flows on arc \( a \in A \), and binary decision vector \( z = (z_{jl}, j \in J, \ l \in L)^T \) where \( z_{jl} = 1 \) indicates that Type 2 user \( j \in J \) is served by Type 1 driver \( l \in L \), and \( z_{jl} = 0 \) otherwise. We let \( \delta^+(n_{it}), \delta^-(n_{it}) \) be the sets of arcs to which node \( n_{it} \) is their tail and head nodes, respectively, and formulate an integer program \( P1 \) as follows.

\[
\begin{align}
\text{[P1]} & \quad \text{maximize} \, \sum_{a \in A} f_a y_a + \sum_{j \in J} x_j^{\text{idle}} \sum_{l \in L} c_j z_{jl} \\
& \quad \text{subject to} \, \sum_{i \in I} x_i \leq K
\end{align}
\]
The objective function (1a) maximizes the potential total profit from both carsharing and ridesharing operations, which is also equivalent to maximize the total number of Type 1 and Type 2 requests served in the system. Constraint (1b) allows the total number of vehicles used in the CRS system to be no more than $K$. Constraints (1c) are flow balance constraints formulated for the spatial-temporal network. Constraints (1d) are arc capacity constraints. Constraints (1e) ensure that the accepted carsharing requests from all Type 1 drivers can potentially provide sufficient total time to serve accepted ridesharing requests from all Type 2 users. Constraints (1f) ensure that $z_{jl} = 1$ only if a Type 1 driver $l \in L$ can serve a Type 2 user $j \in J$. Constraints (1g) ensure that each ridesharing request will be served at most once.

3.2. Phase II: Ridesharing Routing and Scheduling

After solving P1, we obtain solutions as (i) the set of Type 1 drivers to accept and (ii) their matched Type 2 users. We then construct a Phase II model to find the routing and scheduling decisions for both Type 1 and Type 2 requests that can be accepted. Note that our final service acceptance decisions will be made after Phase II and then both types of users will be notified whether we can provide carsharing or ridesharing service to them.

We define a network based on the CRS service region as $G' = (V, E)$ where $V$ is the node set and $E = \{(u,v) : u, v \in V, \ u \neq v\}$ is the edge set. Let $V = V_0 \cup V_1 \cup V_2$ where $V_0 = \{o_l : l \in L\}$ and $V_1 = \{d_l : l \in L\}$ are the sets of pickup and return locations for approved Type 1 carsharing requests.
from solving P1, respectively. Set $V_2 = \{v_j : j \in J\}$ contains each Type 2 ridesharing request. We assume that all the arcs can be traveled along both directions but can have different travel time, and therefore $G'$ is directed. We use $c_{uv}$ to denote the estimated travel time between two nodes such that $(u, v) \in E$. For each $v_j \in V_2$, $j \in J$, we calculate the estimated travel time in the following way: We set $c_{u,v_j} = c_{u,o_j}$ and $c_{v_j,u} = c_{d_j,u}$ for all $u \in V$, where $o_j$, $d_j$ represent origin and destination of Type 2 user $j \in J$.

We define binary decision vector $\alpha = (\alpha^l_{uv}, (u, v) \in E, l \in L)^T$ such that $\alpha^l_{uv} = 1$ indicates that Type 1 driver $l \in L$ travels along arc $(u, v)$, and 0 otherwise. We define binary vector $\beta = (\beta_j, j \in J)^T$ such that $\beta_j = 1$ indicates that Type 2 user $j \in J$ is served, and 0 otherwise. We define continuous decision vector $\gamma = (\gamma_v, v \in V)^T$ such that $\gamma_v$ is the planned time of a vehicle arriving at location $v \in V$. We can formulate the Phase II problem as a variant of the VRP with time windows and multiple depots as follows.

\[
\begin{align*}
\text{[P2]} \quad & \text{maximize} \sum_{j \in J} \gamma^{\text{ride}}_j e_j \beta_j \tag{2a} \\
\text{subject to} \quad & \sum_{l \in L} \sum_{u \in V} \alpha^l_{uv_j} = \beta_j \quad \forall v_j \in V_2, \tag{2b} \\
& \sum_{v : (u,v) \in E} \alpha^l_{o_jv} - \sum_{v : (v,o_j) \in E} \alpha^l_{vo_j} = 1 \quad \forall o_j \in V_0, l \in L \tag{2c} \\
& \sum_{u : (u,d_j) \in E} \alpha^l_{u_dj} - \sum_{d_j : (d_j,u) \in E} \alpha^l_{d_ju} = 1 \quad \forall d_j \in V_1, l \in L \tag{2d} \\
& \sum_{u : (u,v) \in E} \alpha^l_{uv} - \sum_{v : (v,u) \in E} \alpha^l_{vu} = 0 \quad \forall v \in V_2, l \in L \tag{2e} \\
& \alpha^l_{uv} = 0 \quad \forall u \in V_0, v \in V, u \neq o_l, l \in L \tag{2f} \\
& \gamma_{o_l} + c_{o_lv} - T \left(1 - \alpha^l_{o_lv}\right) \leq \gamma_v \quad \forall o_l \in V_0, (o_l,v) \in E, \tag{2g} \\
& \gamma_{v_j} + e_j + c_{v_ju} - T \left(1 - \sum_{l \in L} \alpha^l_{v_ju}\right) \leq \gamma_u \quad \forall v_j \in V_2, (v_j,u) \in E, \tag{2h} \\
& g_l \leq \gamma_{o_l} \leq \gamma_{d_j} \leq h_l \quad \forall l \in L, \tag{2i} \\
& g'_j \leq \gamma_{v_j} \leq h'_j \quad \forall v_j \in V_2, \tag{2j} \\
& \alpha \in \{0,1\}^{|E| \times |L|}, \beta \in \{0,1\}^{|J|}, \tag{2k}
\end{align*}
\]
where the objective function (2a) maximizes the total profit from providing ridesharing services. Constraints (2b) ensure that \( \beta_j = 1 \) if the origin \( o_j' \) for Type 2 user \( j \in J \) has been visited. Constraints (2c)–(2e) balance vehicle flows for each approved Type 1 driver \( l \in L \). Constraints (2f) forbid flows for approved Type 1 driver \( l \in L \) to visit nodes representing pickup/return location for Type 1 driver \( l' \in L \), for all \( l' \neq l \). Constraints (2g)–(2h) formulate the arrival time at each origin \( v \) of Type 2 user to be greater than the arrival time at node \( u \) (which is either pickup location of a Type 1 driver or destination location of another Type 2 user) plus travel and service time, if a trip travels from \( u \) to \( v \) directly. Constraints (2i)–(2j) ensure that departure and return time for Type 1 driver, as well as departure time for each Type 2 user fall into the corresponding time windows.

3.3. Phase II Variant Under Stochastic Travel and Service Time

In practice, various types of uncertainties exist and affect the operations of the proposed CRS system. For example, travel time and service time can be random due to varying road conditions, weather, and traffic. Therefore, the primary task in this section is to propose a two-stage stochastic programming formulation that incorporates random travel and service time for Phase II.

Let \( \tilde{c}_{uv} \) be the random travel time along arc \((u,v) \in E\) and \( \tilde{e}_j \) be the random service time for ride \( j \in J \). Under uncertainty, one or both of the following scenarios could happen: (i) Type 1 driver may arrive late to pick-up the scheduled Type 2 user and (ii) Type 1 driver may return the shared vehicle later than the scheduled returning time. Therefore, our goal is to optimize the start time of each ride to maximize the revenue from ridesharing operation minus the expected penalty cost due to Type 2 users’ waiting and system overtime. (We define the overtime as the sum of late time of returning vehicles by all Type 1 drivers.) We denote \( p^w \) and \( p^o \) as the penalty cost of per unit waiting time and overtime, respectively.

Let \( \bar{e}_j \) be an estimated service time for Type 2 user \( j \in J \), which can be taken as the mean value of \( \tilde{e}_j \). We revise \( P2 \) and present its stochastic variant as:

\[
[\text{Stoch-P2}] \quad \text{maximize} \quad \sum_{j \in J} r_j^{\text{ride}} \bar{e}_j \beta_j - E(Q(\alpha, \gamma, \bar{c}, \bar{e}))
\]

subject to: \((2b)–(2f), (2i)–(2k),\)
where $Q(\alpha, \gamma, \hat{c}, \hat{e})$ is the total penalty cost of random waiting time and overtime given solution $(\alpha, \gamma)$ and uncertainty $(\hat{c}, \hat{e})$, and $\mathbb{E}(\cdot)$ denotes the expectation of random variable $\cdot$. To approximate $\mathbb{E}(Q(\alpha, \gamma, \hat{c}, \hat{e}))$, we apply the Sample Average Approximation (SAA) method (Kleywegt et al. 2002). The idea is to generate a finite set of samples following the Monte Carlo sampling approach and approximate the expectation by its sample average function.

Let $\Omega$ denote the set of all sampled scenarios, and therefore the probability of realizing each scenario is $1/|\Omega|$ when applying the SAA approach, i.e., $\mathbb{E}(Q(\alpha, \gamma, \hat{c}, \hat{e})) = \frac{1}{|\Omega|}\sum_{\omega \in \Omega} Q(\alpha, \gamma, \hat{c}(\omega), \hat{e}(\omega))$, where $\hat{c}(\omega)$, $\hat{e}(\omega)$ are realizations of $\hat{c}$, $\hat{e}$ in scenario $\omega$, respectively. We define auxiliary decision variables $W_j(\omega)$ as waiting time for Type 2 ride request $j \in J$, and $O_l(\omega)$ as overtime for Type 1 driver service $l \in L$ for each scenario $\omega \in \Omega$. Given a $(\alpha, \gamma)$-solution and realized value $(\hat{c}(\omega), \hat{e}(\omega))$ of $(\hat{c}, \hat{e})$ in scenario $\omega \in \Omega$, we specify the sample-based linear program for computing the value of $Q(\alpha, \gamma, \hat{c}(\omega), \hat{e}(\omega))$ as

\begin{align*}
\text{minimize} & \quad \sum_{j \in J} p^w W_j(\omega) + \sum_{l \in L} p^o O_l(\omega) \\
\text{subject to} & \quad \gamma_v + \hat{c}(\omega)_v - T \left( 1 - \alpha_v \right) \leq \gamma_v + W_j(\omega) \quad \forall (u, v) = (o_l, o'_l) \in E, \quad (4a) \\
& \quad \gamma_{o'_l} + W_j(\omega) + \hat{e}(\omega) + \hat{c}(\omega)_{d_j} - T \left( 1 - \sum_{l \in L} \alpha_{d_j} \right) \leq \gamma_{d_l} + W_j(\omega) \quad \forall (d'_l, d_l) \in E, \quad (4b) \\
& \quad \gamma_{d'_l} + W_j(\omega) + \hat{e}(\omega) + \hat{c}(\omega)_{d_j} - T \left( 1 - \alpha_{d_j} \right) \leq \gamma_{d_l} + O_l(\omega) \quad \forall (d'_l, d_l) \in E, \quad (4c) \\
& \quad W_j(\omega) \geq 0 \quad \forall j \in J, \quad (4d) \\
& \quad O_l(\omega) \geq 0 \quad \forall l \in L. \quad (4f)
\end{align*}

Here the objective function (4a) minimizes the total penalty cost of all Type 2 users’ waiting time and all Type 1 drivers’ overtime of returning vehicles. Let $S_j(\omega)$ be the actual service starting time for Type 2 user $j \in J$ and $S'_l(\omega)$ be the actual vehicle return time for Type 1 driver $l \in L$ for each sampled scenario $\omega \in \Omega$, respectively. We have

- $S_j(\omega) = \gamma_v + W_j(\omega)$ for $v \in V_2$ representing Type 2 user $j \in J$, and sampled scenario $\omega \in \Omega$;
- $S'_l(\omega) = \gamma_v + O_l(\omega)$ for $v \in V_1$ representing Type 1 driver $l \in L$, and sampled scenario $\omega \in \Omega$.

Note that $S_j(\omega)$ and $S'_l(\omega)$ correspond to the right-hand sides of constraints (4b)–(4c) and (4d), respectively. Therefore, for each Type 1 driver $l \in L$, the corresponding constraint (4b) calculates...
the actual service start time of its firstly served Type 2 user \( j \) and ensures that it is no earlier than the planned time that driver \( l \) departs origin \( o_l \) plus the travel time from \( o_l \) to \( o'_l \). Similarly, each constraint in (4c) propagates this time relationship for all the subsequent Type 2 users who will be served by Type 1 driver \( l \), and ensures that their actual service start time will be no earlier than the actual service start time of their predecessors plus the realized service time and travel time before driver \( l \) arrives. Lastly, constraint (4d) calculates the actual time of returning vehicle by Type 1 driver \( l \in L \) and ensures that it is no earlier than the time of completing service in the last Type 2 user’s location plus the travel time to the return location.

4. Solution Approaches

To solve Stoch-P2 efficiently, we propose an algorithm based on the Benders decomposition and the integer L-shaped method (Birge and Louveaux 2011, Laporte and Louveaux 1993). We decompose the problem into a relaxed master problem, which contains variables of all the decisions made before realizing the values of \((\bar{c}, \bar{c})\), and a series of subproblems, which involve recourse decisions associated with each sampled scenario. The master problem matches and assigns sequences of Type 2 users to each Type 1 driver. Subproblems are then formulated for each Type 1 driver, to find an optimal schedule to pick up assigned Type 2 users, and each subproblem aims to minimize the expected penalty cost of the assigned Type 2 users’ total waiting time and the corresponding Type 1 driver’s overtime use of a vehicle. We initially set the lower bound of the expected penalty cost for each Type 1 driver as zero. Then we iteratively generate cuts from each subproblem and add them to the master problem to improve the lower bound.

We define decision variables \( \pi = (\pi_u, u \in V_2) \), such that \( \pi_u \) indicates the rank of arrival at node \( u \in V_2 \) of a sequence, and variables \( \theta = (\theta_l, l \in L) \), such that \( \theta_l \) is the optimal objective value (i.e., the optimal expected penalty cost of waiting time and overtime) of the subproblem formulated for Type 1 driver \( l \), for each \( l \in L \). We formulate a relaxed master problem in the current iteration as:

\[
\begin{align*}
\text{[MP]} \quad \max_{\alpha, \beta, \pi, \theta} & \sum_{j \in J} \alpha_j \beta_j \sum_{l \in L} \theta_l
\end{align*}
\]

(5a)
subject to: (2b)–(2f), (2k)

\[
\pi_u - \pi_v + |V_2|\alpha_{uv}^l \leq |V_2| - 1 \quad \forall u, v \in V_2, \ l \in L
\]  

(5b)

\[
L(\alpha, \theta_l) \geq 0 \quad \forall l \in L
\]  

(5c)

\[
\pi_u \in \mathbb{Z}_{\geq 0} \quad \forall u \in V_2
\]  

(5d)

where in (5c), \(L(\alpha, \theta_l) \geq 0\) denotes the set of cuts for improving the lower bound of \(\theta_l\), generated from solving subproblems (described later) in previous iterations. In the above model, constraints (2b)–(2f) enforce sufficient vehicles to cover matched Type 2 users as explained in the previous section. Constraints (5b) and (5d) eliminate every possible subtour (i.e., a cycle only visits nodes in \(V_2\)) since a flow on arc \((u, v)\) increments the rank of node \(v\) from the rank of node \(u\) by at least one.

After solving MP, we obtain a tentative optimal solution \((\bar{\alpha}, \bar{\pi})\) that can be used to recover the routing sequence for each driver \(l \in L\) as follows. We first use the values of \(\bar{\alpha}_{uv}^l\) for \(v \in V_2\) to obtain the set of Type 2 users assigned to driver \(l \in L\). Then we obtain the sequence of visiting them by sorting the set according to an increasing order of elements in solution \(\bar{\pi}\). For each \(l \in L\), let \((\sigma^l(1), \sigma^l(2), \ldots, \sigma^l(n_l))\) be such a sequence with \(n_l\) denoting the total number of Type 2 users assigned to driver \(l\). Let \(\sigma^l(0)\) and \(\sigma^l(n_l + 1)\) be the depots where Type 1 driver \(l \in L\) pick-up and return the vehicle. Recall that \(S_i(\omega)\) be the actual service start time at node \(i\) if \(i \in V_2\) or actual vehicle return time at node \(i\) if \(i \in V_1\). We formulate the subproblem for each Type 1 driver \(l \in L\) as follows.

\[
[S_{P_l}] \quad \text{minimize} \quad \frac{1}{|\Omega|} \left( \sum_{i=1}^{n_l} p^w W_{\sigma^l(i)}(\omega) + p^o O_i(\omega) \right)
\]  

(6a)

subject to (2i) and (2j)

\[
S_{\sigma^l(1)}(\omega) \geq \gamma_{\sigma^l(0)} + c_{\sigma^l(0), \sigma^l(1)}(\omega) \quad \omega \in \Omega,
\]  

(6b)

\[
S_{\sigma^l(i+1)}(\omega) \geq S_{\sigma^l(i)}(\omega) + c_{\sigma^l(i), \sigma^l(i+1)}(\omega) \quad \omega \in \Omega, \ i = 1, \ldots, n_l,
\]  

(6c)

\[
S_{\sigma^l(i)}(\omega) = \gamma_{\sigma^l(i)} + W_{\sigma^l(i)}(\omega) \quad i = 1, \ldots, n_l, \ \omega \in \Omega,
\]  

(6d)

\[
S_{\sigma^l(n_l+1)}(\omega) = \gamma_{\sigma^l(n_l+1)} + O_l(\omega) \quad \omega \in \Omega,
\]  

(6e)
where constraint (6b) ensure that the actual service start time at the first assigned Type 2 user is no earlier than the actual time Type 1 driver leaves depot plus the travel time from depot to the Type 2 user’s pick-up location. Constraints (6c) ensure that the actual service start time at the \((i + 1)\)th node (or actual vehicle return time if such a node represents a depot) of Type 1 driver \(l\) is no earlier than the actual service start time at the \(i\)th Type 2 user plus the time for completing the service at the \(i\)th Type 2 user and the travel time from the \(i\)th user to the \((i + 1)\)th node. For all scenarios \(\omega \in \Omega\), constraints (6d) let the actual time of serving each assigned Type 2 user be the planned arrival time plus the waiting time of the user and constraints (6e) let the actual time for returning the vehicle be the planned time plus the overtime. Both waiting time and overtime auxiliary decision variables are nonnegative according to (6f) and (6g). These constraints are analogous to (4b)–(4d) given in Stoch-P2. Each subproblem \(\text{SP}_l\) will output an optimal schedule for Type 1 driver \(l \in L\) to serve the assigned Type 2 users (given by \(\text{MP}\)), and also the minimum expected penalty cost given the current visiting sequence.

After solving \(\text{MP}\), we obtain an integer solution \(\tilde{\alpha}\) as well as solutions \(\tilde{\theta}_l\) for all \(l \in L\). Let \(\hat{\theta}_l\) be the optimal objective obtained by solving \(\text{SP}_l\) for each \(l \in L\). If \(\tilde{\theta}_l < \hat{\theta}_l\), to enforce the true expected penalty cost for each visiting sequence given the current \(\text{MP}\) solution, following the integer L-shaped method (see, e.g., Laporte et al. 2002), we propose to add the following cut to the \(\text{MP}\) and re-solve it:

\[
\theta_l \geq \hat{\theta}_l \left( \sum_{u,v;\tilde{\alpha}_{uv} = 1} \alpha_{uv}^l - \sum_{u,v} \alpha_{uv}^l + 1 \right). \tag{7}
\]

\text{Theorem 1.} Cut (7) is a valid inequality for \(\text{MP}\) and enforces that no same values of \(\tilde{\alpha}, \tilde{\theta}_l\) can be obtained in future iterations.

\text{Proof.} Inequality (7) only takes effect when \(\sum_{u,v;\tilde{\alpha}_{uv} = 1} \alpha_{uv}^l - \sum_{u,v} \alpha_{uv}^l \geq 0\), which is equivalent to \(\alpha_{uv}^l = 1\) for all \(u,v\) with \(\tilde{\alpha}_{uv} = 1\). Each subproblem \(\text{SP}_l\) outputs the optimal scheduling for Type 1
driver \( l \) and calculates the minimum expected penalty cost, \( \theta_l \). The variable representing recourse penalty cost for the current visiting sequence, should be bounded below by \( \hat{\theta}_l \). Therefore, (7) is valid for any optimal \((\alpha, \theta_l)\), and the same value of \((\bar{\alpha}, \bar{\theta}_l)\) will not repeat if it is not optimal yet. □

Algorithm 1 An integer L-shaped method for solving Stoch-P2.

1: Initialize MP as (5a)–(5d) and set \( L(\alpha, \theta_l) \geq 0 = \emptyset \) for all \( l \in L \).

2: Solve MP by branch-and-bound to obtain a tentative solution \((\bar{\alpha}, \bar{\pi}, \bar{\theta})\).

3: Recover route sequence \((\sigma^l(1), \sigma^l(2), \ldots, \sigma^l(n_l))\) for each \( l \in L \) from \((\bar{\alpha}, \bar{\pi})\).

4: Solve SP\(_l\) for each route sequence of driver \( l \in L \), and let \( \hat{\theta}_l \) be the optimal objective for SP\(_l\).

5: if \( \sum_{l \in L} \bar{\theta}_l < \sum_{l \in L} \hat{\theta}_l \) then

6: for \( l \in L \) do

7: Add Cut (7) to the cut set \( L(\alpha, \theta_l) \geq 0 \) in MP.

8: goto Step 2.

9: else

10: Store the solution \( \bar{\gamma} \) from each SP\(_l\) as planned service start time for each assigned Type 2 user to driver \( l \).

11: Return solution \((\bar{\alpha}, \bar{\gamma}, \bar{\theta})\) as an optimal solution to the overall Stoch-P2 problem.

In Algorithm 1, we present the algorithmic details of the foregoing decomposition-based cutting-plane algorithm.

Theorem 2. Algorithm 1 converges in finite steps.

Proof. In each iteration, we have \( \bar{\theta}_l < \hat{\theta}_l \) only if current visiting sequence for driver \( l \in L \) has not been explored, otherwise cut (7) enforces \( \bar{\theta}_l \geq \hat{\theta}_l \). Since we have a finite number of visiting sequences for a given Type 2 to Type 1 assignment, the algorithm converges in finite steps.

Remark 1. To solve Stoch-P2, we can also decompose the problem by scenario (instead of decomposing the model by driver \( l \) proposed in Algorithm 1) and construct a relaxed master problem that matches Type 1 drivers to Type 2 users and determine pick-up routes and schedules, and
then formulate subproblems to compute the penalty cost of Type 2 users’ waiting time and Type 1 drivers’ overtime in each scenario. However, this traditional way of decomposing the problem cannot produce sufficiently good routes and schedules in the master problem under uncertain travel time and service time, which will only be realized in the subproblems. Furthermore, comparing to constraints in Stoch-P2 that determine feasible routes and schedules, the driver-based decomposition algorithm as Algorithm 1 avoids “big M” constraints in both MP and $SP_l$, for all $l \in L$. Due to these two reasons, our decomposition approach significantly improves the computation of Stoch-P2, which we will demonstrate later in numerical studies.

We may also further decompose $SP_l$ by scenario, which reformulates it as a two-stage stochastic programming model. Since $SP_l$ only contains continuous decision variables and is a linear program that can be quickly solved to obtain cut (7), we directly solve it without further decomposition in our numerical studies.

5. Numerical Results

We conduct numerical studies and demonstrate the performance of proposed models using randomly generated instances based on statistical and census data of underserved populations in Washtenaw County, Michigan.

5.1. Experiment Design

We generate test instances based on the most updated United States Census data for Washtenaw County in Michigan, collected in April 2010 (see United States Census Bureau 2010). In the dataset, Washtenaw County is divided into 100 different census tracts, and each contains the information of population and location (i.e., longitude and latitude of the geographical center). We assume that each census tract is represented by its geographical center and construct a corresponding network with 100 nodes. The travel time between any of two nodes can be obtained by accessing Google Map API\textsuperscript{1}. In the original data set, we can get the population for each census tract grouped by different ages. Under safety concerns, we enforce the age requirement for Type 1 drivers as 21 to

\textsuperscript{1}https://developers.google.com/maps/documentation/distance-matrix/
65 years old, which matches the minimum age requirement of current carsharing service providers (e.g., Zipcar requires drivers to be at least 21 years old\(^2\). To simulate Type 2 users in underserved populations, we consider the populations with age over 50 and disabled.

To pick the carsharing service zones in set \(I\), we sort the 100 nodes based on the population of Type 1 drivers in each node and select the first \(|I|\) nodes. In our numerical test, we set \(|I| = 10\). We assume that the operational hours of our system is from 7 am to 7 pm, and thus \(T = 12\) hours.

We simulate the set of Type 1 driver reservations, \(L\), as follows. We test instances with \(|L| \in \{5, 10, 15\}\). For each Type 1 driver reservation \(l \in L\), we simulate the pick-up and return locations, \(a_l\) and \(d_l\), from the node set \(I\) based on the population distribution of targeted users in the service area. To simulate car pick-up time \(s_l\) of each Type 1 driver \(l \in L\), considering the time flexibility of underserved populations, we sample \(s_l\) uniformly from 7 am to 4 pm with 1 hour time interval. We assume that each user will reserve a car for 1, 2, 3, 4 hours with probabilities of 0.1, 0.2, 0.3, and 0.4, respectively (if \(s_t = 4\) pm, we generate rental periods of service as 3 with a probability of 0.7). We uniformly sample the start time \(g_l\) of each time window from \([s_l, \ldots, t_l - 1]\) and set the end time \(h_l = t_l\) of each time window.

We consider instances with \(|J| \in \{10, 15, 20\}\) for Type 2 ridesharing demand. For each Type 2 user \(j \in J\), we sample the origin, \(o'_{j}\), based on the population density of target Type 2 users (age over 50 years old and disabled) and sample the destination, \(d'_{j}\), uniformly over the 100. At the same time, we avoid \(o'_{j} = d'_{j}\) in all our sampled instances. We generate three types of Type 2 demand: For instances with wide pick-up time windows, we set the time windows \([g'_j, h'_j] = [7\ \text{am}, 7\ \text{pm}]\) for all \(j \in J\); for instances with small pick-up time windows, we sample the time windows \([g'_j, h'_j]\) from \([7\ \text{am}, 1\ \text{pm}]\) and \([1\ \text{pm}, 7\ \text{pm}]\) with equal probabilities.

In all our test instances, for each Type 2 reservation \(j \in J\), we set \(e_j = c_{o'_{j}, d'_{j}}\) plus a constant loading/unloading time that is the same for all \(j \in J\) and thus can be omitted. Following the standard VRP literature such as Polus (1979), Laporte et al. (1992), we consider Gamma distribution for

\(^2\text{https://support.zipcar.com/hc/en-us/articles/220333808-Eligibility-Requirements} \)
sampling the random travel time between pairs of locations. We assume that $\tilde{c}_{ij} = c_{ij} \times \xi$ where $\xi$ follows a Gamma distribution with shape parameter $\alpha$ and scale parameter $\lambda$, and $c_{ij}$ is the deterministic travel time obtained from Google Map API for arc $(i, j) \in E$. Then follow the definition of Gamma distribution, we have

$$E(\tilde{c}_{ij}) = \alpha \lambda c_{ij},$$

$$Var(\tilde{c}_{ij}) = \alpha \lambda^2 c_{ij}$$

We choose parameters $\alpha = 1, \lambda = 1$ for the base case. In out-of-sample tests, we generate 1000 i.i.d. samples (scenarios) for each test instance.

For the rest of parameters, we set $r_{\text{car}} = $8 per hour and $r_{\text{ride}} = $20 per hour to match the price of current carsharing and ridesharing service, e.g. Zipcar and Uber, in Washtenaw County. Note that the pricing strategy for ridesharing company is composed by two parts, mileage ($0.95$/mile) and service time ($0.15$/minute) in Washtenaw County\(^3\). Here we combine them together to obtain reasonable price settings. We set $r_{\text{drive}} = $16 per hour to reasonably incent Type 1 drivers to provide ridesharing service. We let $c_{\text{serv}} = $1 per hour and $c_{\text{idle}} = $1 per hour. The penalty cost parameters are set as $p^w = 0.1$ per minute per user and $p^o = 0.1$ per minute per car for the stochastic programming model for Phase II. Except for the tests in Section 5.2.1, we use 500 in-sample scenarios for model $\text{Stoch-P2}$.

To better demonstrate the necessity of our proposed algorithm, we also benchmark the results of our approaches with the one of simple heuristic methods. The heuristic approach is designed as follows. We first obtain the assignment of Type 2 users to Type 1 users from the solution of model $P1$ (which is very fast to compute) and then form routing decisions for each picked Type 1 driver. We run a 2-opt heuristic (Bräysy and Gendreau 2005) on each route to obtain the routing with minimum travel time and get the scheduled pick-up time for each Type 2 user from such a route.

All instances are solved using Java 8. We call the solver Gurobi 7.5 to optimize all mixed-integer linear programming models. All programs are run on a desktop computer with Microsoft Windows

\(^3\)https://www.uber.com/fare-estimate/
8.1 64-bit operating system, an Intel Core i7-3770 central processing unit (CPU) with 3.50 GHz, and 8.0 GB RAM.

5.2. Computational Studies

For our numerical experiments, we analyze our proposed model from the aspects of computational time, quality of service, revenue and cost, and out-of-sample tests.

5.2.1. Computational Time We present the computational results of the proposed model for various test instances. For each instance, we run ten replications and report the maximum (max), minimum (min), and average (avg) CPU time. We choose parameters as described in Section 5.1 and use 50 to 500 scenarios for the stochastic programming model variant Stoch-P2. We set the CPU time limit as 30 minutes for computing each instance.

In Tables 1 and 2, we report the CPU time in seconds for each test instance with different number of available vehicles ($K$), number of Type 1 reservations ($|L|$), and number of Type 2 reservations ($|J|$). In Table 1, we report CPU time for both P1 and P2, and optimality gap for P2 when reaching the time limit.

Table 2 demonstrates the effectiveness of Algorithm 1 for Stoch-P2. We test Stoch-P2 with different in-sample scenario sizes, and for each sample size, we test three training data sets. We report the CPU seconds if the problem can be solved to optimality, and the final optimality gap otherwise. We report results for both commercial optimization solver (see Rows “Gurobi”) and our proposed solution approach (see Rows “L-Shaped”).

According to our numerical results, P1 model can be solved very fast whereas P2 is relatively difficult to optimize for both deterministic and stochastic cases. In addition, Stoch-P2 is more challenging to solve compared to P2, as only 1 out of 5 test instances can be solved to optimality when the travel time and service time are both random.

In Table 1, we observe that the CPU time for P2 drastically increases as the number of Type 1 and Type 2 reservations increases. For example, given $|L| = 10$, the average CPU time increases from 1.27 seconds to 51.46 seconds as $|J|$ grows; given $|J| = 15$, the average CPU time increases
Table 1  CPU time (in seconds) for model P1 and model P2

| K  | |L| | |J| | |Phases I| Phases II| # of solved instances| Phases II Gap |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
|   | |   |   |   | max | min | avg | max | min | avg | max | min | avg |
| 3  | 5  | 5  | 0.01 | 0.01 | 0.01 | 0.28 | 0.01 | 0.05 | 10 | 0% | 0% | 0% |
| 3  | 5  | 10 | 0.01 | 0.01 | 0.01 | LIMIT | 0.09 | 14.42 | 6 | 93% | 0% | 21% |
| 7  | 10 | 10 | 0.03 | 0.01 | 0.02 | 5.89 | 0.42 | 1.27 | 10 | 0% | 0% | 0% |
| 7  | 10 | 15 | 0.02 | 0.01 | 0.02 | LIMIT | 4.08 | 38.26 | 5 | 10% | 0% | 4% |
| 7  | 10 | 20 | 0.02 | 0.01 | 0.02 | LIMIT | 28.84 | 51.46 | 6 | 24% | 0% | 5% |
| 11 | 15 | 10 | 0.02 | 0.01 | 0.02 | 1.52 | 0.10 | 0.78 | 10 | 0% | 0% | 0% |
| 11 | 15 | 15 | 0.02 | 0.02 | 0.04 | LIMIT | 1233.12 | 2.79 | 253.52 | 10 | 0% | 0% | 0% |
| 11 | 15 | 20 | 0.22 | 0.02 | 0.02 | LIMIT | 52.73 | 101.55 | 7 | 14% | 0% | 3% |

Table 2  CPU time (in seconds) or optimality gap for stochastic programming model Stoch-P2

<table>
<thead>
<tr>
<th>Sample Size</th>
<th>50</th>
<th>100</th>
<th>200</th>
<th>500</th>
</tr>
</thead>
<tbody>
<tr>
<td>K</td>
<td></td>
<td>L</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

|   |   |   |   |   | Gurobi | L-Shaped | Gurobi | L-Shaped | Gurobi | L-Shaped | Gurobi | L-Shaped | Gurobi | L-Shaped | Gurobi | L-Shaped |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 3  | 5  | 5  | 6.22* | 0.15* | 6.00* | 10.94* | 34.50* | 32.37* | 23.42* | 33.58* | 21.07* | 94.36* | 86.90* | 72.17* |
|   |   |   | 1.15* | 0.58* | 0.43* | 0.49* | 0.62* | 0.59* | 1.00* | 1.38* | 0.75* | 4.96* | 1.84* | 5.17* |
| 3  | 5  | 10 | 6.79% | 8.15% | 7.16% | 11.40% | 9.49% | 8.24% | 10.00% | 9.04% | 14.80% | 13.30% | 15.60% | 19.10% |
|   |   |   | 3.87% | 4.78% | 5.35% | 5.29% | 3.97% | 5.56% | 4.46% | 4.86% | 5.95% | 5.77% | 4.86% | 5.61% |
| 7  | 10 | 10 | 3.74% | 3.79% | 2.55% | 3.81% | 3.79% | 3.53% | 4.61% | 5.49% | 6.30% | 4.01% | 10.50% | 4.05% |
|   |   |   | 3.33% | 2.14% | 2.34% | 2.60% | 4.37% | 1530.19* | 0.73% | 1.76% | 3.04% | 5.75% | 0.18% | 774.93* |
| 7  | 10 | 15 | 28.20% | 63.70% | 26.80% | 96.10% | 30.30% | 65.90% | 86.90% | 85.50% | 80.20% | 187.00% | 317.00% | 139.00% |
|   |   |   | 11.54% | 11.15% | 11.71% | 11.24% | 10.57% | 10.04% | 11.50% | 12.26% | 12.74% | 11.50% | 11.95% | 11.98% |
| 7  | 10 | 20 | 17.30% | 14.70% | 12.60% | 23.00% | 25.30% | 29.60% | 50.50% | 106.00% | 236.00% | 433.00% | 310.00% |
|   |   |   | 9.14% | 7.12% | 6.84% | 8.43% | 8.24% | 7.19% | 9.74% | 9.50% | 8.72% | 8.59% | 8.71% | 9.76% |

*: problem solved to optimality and CPU time is reported, –: no solution found within given time limit.

from 38.26 seconds to 253.52 seconds as |L| grows from 10 to 15. Moreover, we observe that the CPU time and the number of solved instances depend on the ratio of the number of total Type 2 users over the number of total Type 1 drivers. When we have ten Type 2 users (|J| = 10), the instance with five Type 1 drivers can only solve 2 out of 10 replications, whereas replications of the instances with more Type 1 drivers can all be solved. A similar observation can be drawn for describing the results of optimality gap.
In Table 2, we observe the effectiveness of proposed solution approach, because both CPU time and optimality gap have been significantly improved from those given by general purpose solvers. Furthermore, we notice that increasing the size of in-sample training data set does not significantly affects our proposed solution approach, whereas it significantly impacts the solver’s performance, which is due to our driver-based instead of scenario-based decomposition scheme.

Although only a few instances can be solved to optimality when testing Stoch-P2, the optimality gap can be small when reaching the CPU time limit. Actually, the number of served trips output by the stochastic case is similar to that of the deterministic model, which will be shown in the next section. The difficulty of closing the optimality gap for Stoch-P2 is due to schedule adjustment to minimize the expected penalty cost. Therefore, although Stoch-P2 cannot be solved to optimality, we can still use the solutions of matched Type 1 drivers and Type 2 users, and the sequence of serving Type 2 users to schedule the operations in Phase II.

5.3. Quality of Service

In this section, we show the quality of service (QoS) of the proposed models regarding the percentage of acceptance demands and vehicle utilization.

5.3.1. Demand Service Rate Table 3 summarizes the demand service rate results for proposed models on various instances. The demand service rate is above 80% on average for both Type 1 carsharing and Type 2 ridesharing demand. Under current parameter settings, although the Phase II model is in general harder to optimize, demand fulfillment rates are still acceptable. Moreover, Stoch-P2 yields a similar average demand service rate as using P2 in Phase II. We also notice that the demand service rate of Type 2 users increases as the number of available Type 1 drivers increases, due to the effect that more drivers become available for ridesharing.

The demand service rate for carsharing is restricted by the availability of vehicles initially distributed in each zone. To address this, we further investigate the effect of vehicle relocation. Table 4 shows the demand service rate for P1 when vehicle relocation is enabled during carsharing and ridesharing operations. As shown in Table 4, demand service rate improves significantly in both
phases. For example, five out of eight instances show 100% demand service rate for Phase I model \textbf{P1}, and most of the instances in Phase II models, \textbf{P2} and \textbf{Stoch-P2}, yield more than 90% of demand service rate. This suggests that we should enable car relocation in carsharing operation to achieve high demand service rate.

<table>
<thead>
<tr>
<th>Table 3</th>
<th>Demand service rate for proposed models</th>
</tr>
</thead>
<tbody>
<tr>
<td>$K$</td>
<td>$</td>
</tr>
<tr>
<td></td>
<td>max</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>7</td>
<td>10</td>
</tr>
<tr>
<td>7</td>
<td>10</td>
</tr>
<tr>
<td>7</td>
<td>10</td>
</tr>
<tr>
<td>11</td>
<td>15</td>
</tr>
<tr>
<td>11</td>
<td>15</td>
</tr>
<tr>
<td>11</td>
<td>15</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 4</th>
<th>Demand service rate for proposed models with relocation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$K$</td>
<td>$</td>
</tr>
<tr>
<td></td>
<td>max</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>7</td>
<td>10</td>
</tr>
<tr>
<td>7</td>
<td>10</td>
</tr>
<tr>
<td>7</td>
<td>10</td>
</tr>
<tr>
<td>11</td>
<td>15</td>
</tr>
<tr>
<td>11</td>
<td>15</td>
</tr>
<tr>
<td>11</td>
<td>15</td>
</tr>
</tbody>
</table>

In Figure 2, we show an example plot of demand and vehicle utilization through twelve-hour operational periods with ten Type 1 drivers and seven vehicles: The black line shows the number
of Type 1 driver reservations and shaded area shows the amount of fulfilled demand. From the plot, the lost demand is resulted from the lack of vehicles (we have 8 demands at Period 4 but only 7 vehicles available in system for this instance). We observe similar results for other instances with unfulfilled demand and conclude that the main reason for the unfulfilled Type 1 demand is resulted from lacking of vehicles for our $\textbf{P1}$ model with relocation.

Figure 2     Example of demand and utilization plot for an instance with 10 Type 1 drivers

Figure 3     Average vehicle demand and relocation of instances with 10 Type 1 drivers
5.3.2. Relocation

We observe not only significant improvements in demand fulfillment rates after enabling vehicle relocation, but also a small number of relocation operations taken over operational time periods. Figure 3 shows an average demand and relocation plots through twelve-hour operational periods with 10 Type 1 drivers: The bars indicate the average number of vehicle demand whereas the line depicts the average number of relocation. We notice that the vehicle demands reach its peaks around periods 4 and 9 with values around 3.5, but the relocation is less than 0.5 from period 4 to 8. Recall the changes of demand fulfillment rate shown in Section 5.3.1. It implies that a small ratio of relocation is sufficient to improve the QoS of the system. Furthermore, the relocation operations concentrate on Periods 4 to 8, which shows that our P1 model yields a proper vehicle allocation at the initial operational periods. Also, as all demands diminishing at the end of operational periods, the CRS system requires less vehicle relocation later than earlier. Similar observations hold for other test instances and we omit the details here for brevity.

5.4. Revenue and Cost

In this section, we present the numerical results of operational cost and revenue for our proposed model. For the revenue, we report the revenue from approved Type 1 drivers (denoted by “Type 1”) and approved Type 2 users (denoted by “Type 2” and “Type 2-S” for P1 and Stoch-P2, respectively). For the cost, we report vehicle idle cost (idle) and the actual amount of payment to Type 1 drivers who are scheduled to serve Type 2 users (denoted by “Driver” and “Driver-S” for P1 and Stoch-P2, respectively). We also report the profit generated by operating the system, which is the total revenue minus cost in the deterministic and stochastic cases.

Table 5 summarizes the operational revenue and cost for all the proposed models. Both P2 and Stoch-P2 yield positive profit in most test instances. In general, as the number of both Type 1 and Type 2 reservations increases, the total profit increases.

Stoch-P2 yields less profit than the deterministic model as more drivers are hired to provide ridesharing services. This effect is because Stoch-P2 aims to design more reliable routes and schedules via penalizing late arrivals for picking up Type 2 users and overtime of the system. It
Table 5 Revenue and cost for proposed models

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>Revenue</th>
<th>Cost</th>
<th>Profit</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Type 1</td>
<td>Type 2</td>
<td>Type 2-S</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>88.00</td>
<td>47.00</td>
<td>47.00</td>
</tr>
<tr>
<td>5</td>
<td>10</td>
<td>48.00</td>
<td>65.50</td>
<td>59.50</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>176.00</td>
<td>152.00</td>
<td>152.00</td>
</tr>
<tr>
<td>10</td>
<td>20</td>
<td>176.00</td>
<td>144.50</td>
<td>148.50</td>
</tr>
<tr>
<td>15</td>
<td>10</td>
<td>304.00</td>
<td>288.00</td>
<td>100.00</td>
</tr>
<tr>
<td>15</td>
<td>15</td>
<td>352.00</td>
<td>288.00</td>
<td>100.00</td>
</tr>
<tr>
<td>15</td>
<td>20</td>
<td>320.00</td>
<td>203.50</td>
<td>171.00</td>
</tr>
</tbody>
</table>

indicates that we have to sacrifice some profit to provide reliable services. However, the difference in profit between the two models is not significant. We will demonstrate how Stoch-P2 yields better scheduling in the next section. Overall, we recommend using stochastic programming model in Phase II, which can achieve more reliable routes and schedules.

5.5. Out-of-Sample Tests

We conduct out-of-sample tests to evaluate proposed models by measuring Type 2 users’ waiting and Type 1 drivers’ overtime. Define Ω′ as the set of all out-of-sample test scenarios. Let s(l) be the scheduled time to return the car by Type 1 driver l ∈ L and s′(j) be the scheduled starting service time for Type 2 user j ∈ J. Values s(l), l ∈ L and s′(j), j ∈ J can be obtained from the optimal values of α in P2 and Stoch-P2. Let t(l, ω) be the actual time of returning the car by Type 1 driver l ∈ L in ω ∈ Ω′ and t′(j, ω) be the actual service starting time for Type 2 user j ∈ J in ω ∈ Ω′, which will be computed based on the routing sequence and random travel and service time in scenario ω ∈ Ω′. Then for each test scenario ω, we calculate and report:

- \( \text{wait}(j, \omega) = \max \{0, t(j, \omega) - s(j)\} \) for \( j \in J \).
- \( \text{overtime}(l, \omega) = \max \{0, t′(l, \omega) - s′(l)\} \) for \( l \in L \).

Table 6 shows the summary statistics of out-of-sample tests for Stoch-P2, in which we report the quantile statistics (i.e., 0, 5%, 25%, 50%, 75%, 95%, 100%) and average for waiting time per
Table 6  Statistics of waiting time and overtime per user in Stoch-P2 out-of-sample test (in minutes)

<table>
<thead>
<tr>
<th>Instance</th>
<th>Statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 10</td>
<td>Waiting: 0 0 0 0 0 33 181 5.07</td>
</tr>
<tr>
<td>10 10</td>
<td>Waiting: 0 0 0 0 0 30 263 4.24</td>
</tr>
<tr>
<td>10 15</td>
<td>Waiting: 0 0 0 0 8 58 247 9.95</td>
</tr>
<tr>
<td>15 10</td>
<td>Waiting: 0 0 0 0 0 5 151 1.17</td>
</tr>
<tr>
<td>15 15</td>
<td>Waiting: 0 0 0 0 0 37 272 5.51</td>
</tr>
<tr>
<td>15 20</td>
<td>Waiting: 0 0 0 0 0 44 324 6.90</td>
</tr>
</tbody>
</table>

Type 2 user and overtime per Type 1 driver in minutes, respectively. In Table 6, waiting time for each Type 2 is less than 10 minutes on average, and 95% quantile is around 30 minutes. Notice that in our numerical tests, because we set the stochastic travel time to follow the Gamma distribution with the variance being proportional to the mean value, the variation results in the much worse waiting time than the average in higher quantiles. The average system overtime varies from 2.34 to 25.57 minutes, and the 95% quantile varies from 12 to 110 minutes, due to the travel time variation.

Meanwhile, a higher ratio of the number of Type 2 users and the number of Type 1 drivers can lead to a higher average waiting time and overtime: Comparing the results for instance with 15 Type 1 drivers and 10 Type 2 users and the instance with 10 Type 1 drivers and 15 Type 2 users, the former has much lower average and 95% quantile in both waiting time per Type 2 user and overtime per Type 1 driver.

Figures 4 and 5 show the density plots of waiting time for each Type 2 user and overtime for each Type 1 driver one instance with $|L| = 10$ and $|J| = 10$, respectively. We report the results for three different models, P1, Stoch-P2, and heuristic method (H) that is introduced in Section 5.1. We produce the kernel density figures to reflect the distribution, with vertical line representing
the average. Both figures show a significant long-tail effect of the distributions of waiting time and overtime.
In Figure 4, the waiting time for each Type 2 user has peak density around its average value for all three models. However, by penalizing waiting time for each late arrival, model Stoch-P2 yields low average waiting time for each Type 2 user. For example, the average waiting time for Type 2 User No. 1 is about 0 minutes for model Stoch-P2, whereas it is over 20 minutes for model P2. Furthermore, the heuristic method performs very poorly: It yields average 50 minutes (out of the plotting range) of waiting time for Type 2 User No. 2 in this example. This shows the importance of using stochastic programming models to make reliable schedules in Phase II.

Similar effects also appear in Figure 5. Model Stoch-P2 yields schedules with low probabilities of overtime for returning vehicles. However, for model P2 and heuristic approach, we are at risk of having long overtime. For example, the average overtime for Type 1 driver No. 10 is about 15 minutes by model P2 and about 60 minutes (out of plotting range) by the heuristic approach, compared with around 0 minutes by model Stoch-P2.

We observe a universal effect, which we demonstrated in the previous example, appears in all test instances. Despite that model P2 can be solved relatively fast, it is necessary to adopt stochastic programming model to improve the reliability of service operations.

6. Conclusions

In this paper, we designed a new shared-mobility system to serve transportation needs by underserved populations that integrates both carsharing and ridesharing (called the CRS system), and developed a two-phase approach to design and operate such a system. The models were applied to various instances based on a simulated data set in Washtenaw County, Michigan, where a large number of populations are jobless, elderly, and disabled. Numerical results indicated the computational tractability of our proposed solution approaches. Furthermore, the quality of service of the system is maintained at high levels, despite the fact that few instances cannot be solved to optimality. Besides, we demonstrated the importance of vehicle relocation in the proposed model: When vehicle relocation allowed, car utilization rate and Type 1 demand fulfillment improved significantly, and they consequently increased available drivers in the CRS system to have higher Type 2 demand fulfillment.
We further extended the basic model to a two-stage stochastic programming model to capture the randomness of vehicle travel time and service time. Numerical comparisons with a deterministic counterpart using expected values of the random parameters and with a heuristic approach showed the advantages of our models. Both in-sample and out-of-sample test results demonstrated the effectiveness of matching and scheduling using our approach, where the risk of waiting and overtime both decreased. Meanwhile, we only sacrificed a small fraction of profit for running the system.

For future research, our models and results can be extended as follows. First, in this paper we minimize the expected cost of overtime and waiting time. Instead, a robust optimization model that focuses on the worst-case analysis, can be used for ensuring reliable CRS operations. Second, we used census data to test our models. We are now working with nonprofit organizations and city governments and have participated in the design of a website (http://communityride.us) to be used for real-world deployment of the CRS system. We aim to make more transportation data of underserved communities available to the public through further investigation of this topic.

Acknowledgments
The authors are grateful for the support from the National Science Foundation under grant CMMI-1636876 and CMMI-1727618.

References


