Ride-Sharing Matching Under Travel Time Uncertainty Through Data-Driven Robust Optimization

XIAOMING LI\(^1\), JIE GAO\(^2\), CHUN WANG\(^3\), (Member, IEEE), XIAO HUANG\(^3\), AND YIMIN NIE\(^4\)

\(^1\)Concordia Institute for Information Systems Engineering (CIISE), Concordia University, Montréal, QC H3G 1M8, Canada
\(^2\)HEC Montréal, Université de Montréal, Montréal, QC H3T 2A7, Canada
\(^3\)Concordia John Molson School of Business (JMSB), Concordia University, Montréal, QC H3G 1M8, Canada
\(^4\)Global Artificial Intelligence Accelerator (GAIA) Innovation Hub, Ericsson Inc., Montréal, QC H4R 2A4, Canada

Corresponding author: Xiaoming Li (xiaoming.li@mail.concordia.ca)

ABSTRACT

In ride-sharing services, travel time uncertainty significantly impacts the quality of matching solutions for both the drivers and the riders. This paper studies a one-to-many ride-sharing matching problem where travel time between locations is uncertain. The goal is to generate robust ride-sharing matching solutions that minimize the total driver detour cost and the number of unmatched riders. To this end, we formulate the ride-sharing matching problem as a robust vehicle routing problem with time window (RVRPTW). To effectively capture the travel time uncertainty, we propose a deep learning-based data-driven approach that can dynamically estimate the uncertainty sets of travel times. Given the NP-hard nature of the optimization problem, we design a hybrid meta-heuristic algorithm that can handle large-scale instances in a time-efficient manner. To evaluate the performance of the proposed method, we conduct a set of numeric experiments based on real traffic data. The results confirm that the proposed approach outperforms the non-data-driven one in several important performance metrics, including a proper balance between robustness and inclusiveness of the matching solution. Specifically, by applying the proposed data-driven approach, the matching solution violation rate can be reduced up to 85.8%, and the valid serving rate can be increased up to 42.3% compared to the non-data-driven benchmark.

INDEX TERMS

Data-driven robust optimization, gated recurrent units, mobility-on-demand, ride-sharing matching, time-series prediction.

I. INTRODUCTION

Coupled with networking technology development and the advances in intelligent transportation systems (ITS), large-scale ride-sharing platforms such as Uber, Lyft, and Didi have substantially reshaped the transportation pattern. These ride-sharing platforms match drivers and riders with similar itineraries and time schedules, which may provide significant societal and environmental benefits by reducing the number of vehicles used for personal travel [1]. A key question for ride-sharing platforms is how to effectively match drivers and riders in a dynamic environment where travel time between locations could be subject to uncertainty [2]. If only expected travel time is considered, a deterministic optimization-based solution could be infeasible in the presence of travel time uncertainty. Conversely, if we consider all possible travel time delays, a robust solution could be too conservative at the expense of a low matching rate. Therefore, in the context of ride-sharing services, it is crucial to balance the robustness and inclusiveness of the matching solution. As reported in [3], quantifying the connection between the driver’s travel time and the ride-sharing matching solution remains unresolved.

In this paper, we develop a learning-based data-driven framework that integrates deep learning with robust optimization in matching drivers and riders under travel time uncertainty. Specifically, we formulate the ride-sharing problem...
as a Robust Vehicle Routing Problem with Time Window (RVRPTW) [4] to compute a near-optimal matching solution that remains feasible with high probability under the maximum travel time delay (worst-case scenario). Our work fills this research gap by proposing a learning-based data-driven robust optimization framework for the ride-sharing matching problem with travel time uncertainty. In particular, the framework allows us to dynamically estimate the uncertainty set using historical travel time data based on the deep learning-based time-series forecast approach Gated Recurrent Units (GRUs) [15].

Our major contributions can be summarized as follows:

- We develop an innovative, learning-based data-driven robust optimization framework to address the ride-sharing matching problem under travel time uncertainty.
- We propose a novel, data-driven approach to dynamically estimate the uncertainty set from historical travel time data for the robust optimization model.
- We design a hybrid meta-heuristic algorithm that integrates local search and simulated annealing to solve large-scale instances.
- The proposed framework delivers superior performance and a desirable balance between robustness and inclusiveness for the one-to-many ride-sharing matching problem.

The remainder of the paper is structured as follows. We classify and summarize related articles as well as our contribution to the literature in Section II. The problem setting and base formulations are discussed in Section III. Data-driven robust optimization framework and hybrid meta-heuristic algorithm are discussed in Section IV. The performance of our proposed approach is evaluated in Section V. Section VI summarizes our work with future research directions.

## II. LITERATURE REVIEW

Matching is essential for a successful design of a ride-sharing system. Recent surveys [2], [3], [5], [6] have classified ride-sharing matching approaches into deterministic and non-deterministic optimization models, where the parameters are subject to uncertainty.

In deterministic models, the problem environment does not involve any uncertainty, and all model parameters are assumed to be known. For example, a ride-sharing transfer problem under a disaster evacuation scenario is studied in [7]. To minimize the total routing cost, the problem is formulated as a two-phase optimization model that involves trip planning and the ride-sharing process. In addition, the Lagrangian relaxation technique is introduced to solve the model in real-world network scenarios. A ride-sharing optimization framework is proposed in [8] where the drivers and customers are formulated as supply and demand from the perspective of a two-sided market. The framework can be applied in both online and offline ride-sharing environments to maximize the driver’s profit. In addition, two heuristic algorithms based on drivers’ locations and marginal value and one approximate algorithm based on a greedy algorithm are designed to solve large-scale instances. Guo et al. [9] develop a ride-sharing framework that aims to maximize the overall shared route percentage and mitigate the unmatched travelers considering dynamic timeshare and anticipation-based migration. Based on the historical matching data, a multi-strategy local search heuristic algorithm is designed to generate near-optimal solutions in a short computational time. With a similar objective, a binary integer programming model to match drivers and riders in a ride-sharing system is designed in [10]. The ride-sharing system operates on a rolling horizon to extend the static ride-sharing service to a dynamic counterpart.

Although deterministic models are straightforward to consider, the solution may be infeasible when the environment is subject to uncertainty. In the ride-sharing context, this may involve random demand, travel time, and location. To this effect, a stream of work has examined ride-sharing optimization problems under various sources of uncertainty. For instance, Lin et al. [11] study a demand-aware ride-sharing routing problem considering travel demand uncertainty. The issue is formulated as a two-stage stochastic optimization problem by exploring the new design space enabled by the demand-aware approach. The objective is to maximize the platform revenue that is subject to the rider’s travel time delay. In [12], the researchers study stochastic travel time and stochastic delivery location as two stochastic variant ride-sharing problems. A heuristic algorithm is designed based on an adaptive large neighborhood search to find the solution to maximize the expected profit of serving passengers and parcels. Another stream of optimization under the uncertainty modeling technique is robust optimization, which optimizes the objective function under the worst-case scenario. Further, a static ride-sharing matching problem considering travel time uncertainty is studied in [13]. The problem is formulated as a one-stage robust optimization model to minimize the overall system cost under maximum travel time delay (under travel time worst-case scenario). In addition, a heuristic algorithm based on Tabu search is designed to solve large-scale instances. Considering the driver’s travel time uncertainty, a stochastic optimization approach is designed in [14].
TABLE 2. Notation table for the mathematical models.

| Sets | Description |
|------|-------------|
| $\mathcal{D}$ | A set of drivers, indexed by $d$ |
| $\mathcal{R}$ | A set of riders, indexed by $r$ |
| $\mathcal{K}$ | A set of ride-sharing travellers, index by $k$, $K = \mathcal{D} \cup \mathcal{R}$ |
| $\mathcal{V}^{\mathcal{K}}$ | A set of origin and destination locations of all riders, $\mathcal{V}^{\mathcal{K}} = \mathcal{V}^\mathcal{R} \cup \mathcal{V}^\mathcal{D}$ |
| $\mathcal{V}^\mathcal{D}$ | A set of origin and destination locations of all drivers, $\mathcal{V}^\mathcal{D} = \mathcal{V}^\mathcal{R} \cup \mathcal{V}^\mathcal{D}$ |
| $\mathcal{V}$ | A set of travellers’ locations, $\mathcal{V} = \mathcal{V}^\mathcal{R} \cup \mathcal{V}^\mathcal{D}$ |
| $A_{\mathcal{I}, \mathcal{J}}$ | A set of arcs that connect locations in sets $\mathcal{I}$ and $\mathcal{J}$, $\mathcal{I}, \mathcal{J} \subseteq \mathcal{V}$ |

| Parameters | Description |
|------------|-------------|
| $\bar{t}_{u,v}$ | The nominal travel time from location $u$ to location $v$ |
| $\hat{t}_{u,v}$ | The maximum travel time delay from location $u$ to location $v$ |
| $\varphi$ | The degree of uncertainty of driver $d$ for polyhedral uncertainty set |
| $o(k)$, $w(k)$ | The origin and destination of traveller $k$ |
| $ed(k)$, $la(k)$ | The earliest departure and latest arrival times of traveller $k$ |
| $\alpha$ | The coefficient to convert travel time to travel cost |
| $\beta$ | The penalty coefficient for unmatched riders |
| $C_d$ | The capacity (number of available seats) of driver $d$ |
| $H$ | A large number to linearize the “if” constraint |

| Variables | Description |
|-----------|-------------|
| $z_{u,v}^d$ | Status that is equal to 1 if driver $d$ traverses from location $u$ to location $v$, equals 0 otherwise |
| $y_r$ | Status that is equal to 1 if rider $r$ is not served by any drivers |
| $dt_u$ | Driver’s departure time at location $u$ |

maximize both the total generalized trip cost-saving and the number of matches.

For the ride-sharing applications under the data-driven environment, the parameters in the optimization models, such as rider demand and travel time, are subject to uncertainty. Although [11], [13], and [14] have considered relevant uncertainties, parameters of the uncertainty set and probability distribution are typically assumed to be known and static, which may not be the best reflection of dynamic environments for ride-sharing.

III. ONE-TO-MANY RIDE-SHARING MATCHING PROBLEM

Consider a ride-sharing platform with a service operator and a pool of drivers and riders. The service operator matches the drivers and the riders such that the total travel cost of the travelers is minimized. Each driver or rider (hereafter referred to as a “traveler”) carries a trip type that involves a travel time window, an origin, and a destination.

The notations, including sets, parameters, and decision variables in the optimization model, are summarized in Table 2. Let $D$ denote the set of drivers, $R$ the set of riders, and $P = D \cup R$ the set of travelers. The trip type of a driver $d \in D$ can be represented by a quintuple $(o(d), w(d), ed(d), la(d), C_d)$, where $o(d)$ and $w(d)$ are the trip origin and destination of driver $d$, $ed(d)$ is the earliest time that driver $d$ can depart from its origin and $la(d)$ is the latest time it must arrive at the destination. $C_d > 0$ is the capacity (or the number of available seats) of driver $d$. Similarly, the trip type of a rider $r \in R$ can be represented by a quadruple $(o(r), w(r), ed(r), la(r))$, where $o(r)$ and $w(r)$ are the trip origin and destination of rider $r$, $ed(r)$ and $la(r)$ are the earliest departure time and latest arrival time of rider $r$.

Let $\mathcal{V}^\mathcal{R} = \{o(r), w(r) \mid r \in R\}$ be the set of origin and destination of the riders, $\mathcal{V}^\mathcal{D} = \{o(d), w(d) \mid d \in D\}$ be the set of origin and destination of the drivers, and $\mathcal{V} = \mathcal{V}^\mathcal{R} \cup \mathcal{V}^\mathcal{D}$ that for all travellers. The route of driver $d$ can be represented by $\{v_1, v_2, \ldots, v_z\}$ where $v_i \in \mathcal{V}$ for $i \in \{1, 2, \ldots, z\}$, a series of locations in the sequence to be visited. In particular, the first and the last locations of the route, $v_1$ and $v_z$, are the origin $o(d)$ and destination $w(d)$ of driver $d$, respectively. Other locations are according to the riders’ origins and destinations. That is, if rider $r$ is matched to driver $d$, there must exist $1 \leq i < j \leq z$ such that $v_i = o(r)$ and $v_j = w(r)$. Further, for modeling simplicity, we assume that for each matched driver, no riders are allowed to drop off before all the riders are picked up. Meanwhile, the matched driver cannot pick up extra riders once the drop-off starts. Lastly, we denote $t_{u,v}$ as the travel time from location $u$ to location $v$ and $A_{\mathcal{I}, \mathcal{J}}$ the set of arcs that connect two sets of locations $\mathcal{I}$ and $\mathcal{J}$ where $\mathcal{I}, \mathcal{J} \subseteq \mathcal{V}$.

The objective is to identify the optimal matching solution among the drivers and the riders and the optimal route for each driver such that travelers’ total travel cost, i.e., the sum of the drivers’ travel cost (proportional to total travel time) and unmatched riders’ cost (modeled as a penalty), is minimized.

A. DETERMINISTIC MODEL FOR RIDE-SHARING MATCHING

Without travel time uncertainty, the one-to-many ride-sharing matching problem can be formulated as a Mixed Integer Linear Programming (MILP) model with the following decision variables:

$$x_{u,v}^d \in \{0, 1\} \quad \forall (u,v) \in A_{\mathcal{V}, \mathcal{V}}, \forall d \in D, u \neq v,$$  \hspace{1cm}  (1)

$$y_r \in \{0, 1\} \quad \forall r \in R,$$ \hspace{1cm}  (2)

$$dt_u \in \mathbb{R}_+, \forall u \in \mathcal{V}, d \in D.$$ \hspace{1cm}  (3)

Specifically, $x_{u,v}^d$ is a binary variable corresponding to driver’s route, i.e., $x_{u,v}^d = 1$ if driver $d$ traverses directly from location $u$ to $v$, and $x_{u,v}^d = 0$ otherwise. $y_r$ is a binary variable representing riders’ matching status, i.e., $y_r = 1$ if rider $r$ is not served by any driver and $y_r = 0$ otherwise. Lastly, $dt_u$ denotes the departure time of driver $d$ at location $u$.

To minimize the travelers’ total travel cost under the worst-case scenario, the decision-making problem can be formulated as a one-stage robust optimization model in what follows. The objective function (4) aims at minimizing the total travel cost of all travelers under the worst-case travel time scenario. The first term converts drivers’ total travel time to travel costs. The second term represents the penalty for the ride-sharing system due to unmatched riders.

$$\min \alpha \sum_{d \in D} \sum_{(u,v) \in A_{\mathcal{V}, \mathcal{V}}} \tilde{t}_{u,v} x_{u,v}^d + \beta \sum_{r \in R} y_r$$ \hspace{1cm}  (4)

$\tilde{t}_{u,v}$ is a parameter representing the travel time from location $u$ to location $v$.

$^1$The similar problem setting can be found in [13].
The problem needs to satisfy a set of constraints including the routing constraints, time constraints, and capacity constraints [6].

1) ROUTING CONSTRAINTS

A rider can be matched to at most one driver, in which case they must travel from origin to destination. This leads to the following constraints:

$$\sum_{d \in D} \sum_{v \in V_R \cup \{o(r)\}} x_{o(r),v}^d \leq 1 \quad \forall r \in R$$

In addition, a driver must start from his origin to pick up a rider or travel to his destination directly, which leads to the following constraints:

$$\sum_{v \in V_R \cup \{w(d)\}} x_{o(d),v}^d = 1 \quad \forall d \in D$$

Moreover, each driver reaches his destination either directly from his origin or another rider’s destination. The relevant constraints follow:

$$\sum_{u \in V_R \cup \{o(d)\}} x_{u,w(d)}^d = 1 \quad \forall d \in D$$

As commonly assumed in ride-sharing matching problems, the ride-sharing routes are non-recurring, which is captured by the following constraints:

$$x_{u,v}^d + x_{v,u}^d \leq 1 \quad \forall d \in D, \forall u, v \in V, u \neq v$$

Further, for the transitive locations $V_R$, the conservation of flow constraints [16] must be satisfied:

$$\sum_{u \in V_R \cup \{o(d)\}} x_{u,v}^d = \sum_{v' \in V_R \cup \{w(d)\}} x_{v',v}^d \quad \forall v \in V_R, \forall d \in D$$

For each non-destination location, there is at most one successive location, which is captured by the following constraints.

$$x_{u,v}^d + x_{v,u}^d \leq 1, \quad \forall u \in \{o(d)\} \cup V_R, v, v' \in \{w(d)\} \cup V_R, \forall d \in D, \quad v \neq v'$$

In addition, each rider, if assigned, must travel from its origin to its destination, which leads to the following constraints:

$$\sum_{v \in V_R \cup \{o(r)\}} x_{o(r),v}^d = \sum_{v' \in V_R \cup \{w(r)\}} x_{v',v}^d \quad \forall r \in R, \forall d \in D$$

Finally, each rider is served by one driver at most, which enforces the following constraints.

$$y_r + \sum_{d \in D} \sum_{v \in \{o(r'),w(r')\}} x_{o(r'),v}^d = 1 \quad \forall r \in R$$

2) TIME CONSTRAINTS

Time constraints ensure that the travel time windows, i.e., the earliest departure time and the latest arrival time of each traveler, are respected in the solution.

Firstly, a driver’s departure time at a location $v$ must be later than the departure time at the precedent location $u$ plus travel time $t_{u,v}$ between location $u$ and $v$ if the driver traverses the arc $(u, v)$. These can be captured by the following constraints.

$$d_{t_v}^d \geq d_{t_u}^d + t_{u,v} + H(x_{u,v}^d - 1), \quad \forall d \in D, \forall u \in \{o(d)\} \cup V_R, \forall v \in \{w(d)\} \cup V_R, u \neq v$$

where $H$ is a large positive number to linearize the “if” constraint.

In addition, both departure times must be within the serving time window, namely, between the rider’s earliest departure time and the latest arrival time. This is captured by the following two constraints:

$$e_d(r) \leq d_{t_{o(r)}}^d \leq d_{t_{w(r)}}^d \leq a_l(r) \quad \forall r \in R, \forall d \in D$$

Similarly, a driver’s departure and arrival time should be within its own travel time window:

$$e_d(d) \leq d_{t_{o(d)}}^d \leq d_{t_{w(d)}}^d \leq a_l(d) \quad \forall d \in D$$

3) CAPACITY CONSTRAINTS

For each driver, the maximum number of served riders must be less than or equal to the capacity $C_d$, which leads to the following constraints. Notice that if the driver travels alone, the number of arcs the driver visits is 1 (from the driver’s origin to destination). Otherwise, the driver serves the riders by visiting their pickup and drop-off locations. Namely, the driver visits $2 \times C_d$ locations. Therefore, the maximum arcs that the driver visits is $2 \times C_d + 1$.

$$\sum_{(u,v) \in A_{(o(d))} \cup A_{w(d)}} x_{u,v}^d = 2 \times C_d + 1, \quad \forall d \in D$$

B. ROBUST OPTIMIZATION MODEL FOR RIDE-SHARING MATCHING WITH TRAVEL TIME UNCERTAINTY

Travel time uncertainty can play a critical role in our ride-sharing problem, as the optimal solution may be infeasible when actual travel time is significantly different than its nominal value, thus violating some time window constraints. In this section, we formulate a robust optimization model to find a robust solution accounting for travel time uncertainty. Let $\tilde{t}_{u,v}$ and $\check{t}_{u,v}$ be the nominal travel time and maximum travel time delay, respectively. Additionally, it is reasonable to assume that $S^d$ is the set of arcs for driver $d$ that is subject to uncertainty controlled by $\Gamma^d$, where $\Gamma^d$ represents the degree of conservatism (also known as the “degree of uncertainty” in literature). The objective of the robust optimization model is to minimize the
overall cost under maximum travel time delay (also known as the worst-case scenario in literature).

$$\min \alpha \sum_{d \in D} \left( \sum_{(u,v) \in A_{U,V}} \bar{t}_{u,v} x_{u,v}^d + \max_{\{S^d|S^d \subseteq A_{U,V}, |S^d| \leq \Gamma^d\}} \sum_{(u,v) \in S^d} \tilde{t}_{u,v} x_{u,v}^d \right) + \beta \sum_{r \in \mathcal{R}} y_r$$

(17)

s.t. \hspace{1cm} (5) - (12), \hspace{1cm} (14) - (16)

$$dt_u^d \geq d_t^d + \tilde{t}_{u,v} + \bar{t}_{u,v} + H(x_{u,v}^d - 1), \forall d \in D, \forall u \in \{o(d)\} \cup \mathcal{V}^\mathcal{R}, v \in \{w(d)\} \cup \mathcal{V}^\mathcal{R}, u \neq v$$

(17b)

To efficiently solve the above optimization problem, we introduce a robust counterpart (RC) technique [17] to reformulate the original robust model, where the duality theory is applied to eliminate the inner maximization problem, as will be discussed in the next section.

C. THE ROBUST COUNTERPARTS REFORMULATION

In this paper, we consider the polyhedral (also known as 1-norm type) uncertainty set that has been intensively used in the domain of robust optimization. For the RC using a polyhedral uncertainty set, we have the following proposition:

**Proposition 1:** The RC of the ride-sharing matching robust optimization model using a polyhedral uncertainty set can be reformulated to the following MILP model:

$$\min \alpha \sum_{d \in D} \left( \sum_{(u,v) \in A_{U,V}} \bar{t}_{u,v} x_{u,v}^d + \Gamma^d g_d + \sum_{(u,v) \in A_{U,V}} h_{u,v}^d \right)$$

$$+ \beta \sum_{r \in \mathcal{R}} y_r$$

(18)

s.t. \hspace{1cm} (5) - (12), \hspace{1cm} (14) - (16)

$$g_d + h_{u,v}^d \geq \tilde{t}_{u,v} x_{u,v}^d, \forall d \in D, \forall (u,v) \in A_{U,V}.$$ \hspace{1cm} (18a)

$$g_d \in \mathbb{R}_+, \forall d \in D$$ \hspace{1cm} (18b)

$$h_{u,v}^d \in \mathbb{R}_+, \forall d \in D, \forall (u,v) \in A_{U,V}.$$ \hspace{1cm} (18c)

**Proof:** The min term in the objective function (17) is equivalent to the following optimization problem (see the proof in [18)).

$$\max \tilde{t}_{u,v} x_{u,v}^d + \Gamma^d$$

(19)

s.t. \hspace{1cm} \sum_{(u,v) \in A_{U,V}} z_{u,v}^d \leq \Gamma^d, \forall d \in D$$

(19a)

$$z_{u,v}^d \leq 1, \forall d \in D, \forall (u,v) \in A_{U,V}.$$ \hspace{1cm} (19b)

By strong duality, The dual problem of (19) can be written as the following optimization problem:

$$\min \Gamma^d g_d + \sum_{(u,v) \in A_{U,V}} h_{u,v}^d$$

(20)

s.t. \hspace{1cm} $$g_d + h_{u,v}^d \geq \tilde{t}_{u,v} x_{u,v}^d, \forall d \in D, \forall (u,v) \in A_{U,V}.$$ \hspace{1cm} (20a)

$$g_d \in \mathbb{R}_+, \forall d \in D$$ \hspace{1cm} (20b)

$$h_{u,v}^d \in \mathbb{R}_+, \forall d \in D, \forall (u,v) \in A_{U,V}.$$ \hspace{1cm} (20c)

where $$g_d$$ and $$h_{u,v}^d$$ are the dual variables of constraints (19a) and (19b), respectively.

IV. DEEP LEARNING-BASED DATA-DRIVEN ROBUST OPTIMIZATION FRAMEWORK

In this section, we introduce our learning-based data-driven robust optimization framework. First, we apply a deep learning-based strategy to construct the uncertainty set of the robust optimization model. We then design a meta-heuristic algorithm that integrates local search and simulated annealing to solve the large-scale ride-sharing matching problem.

A. THE CONSTRUCTION OF UNCERTAINTY SET FOR TRAVEL TIME

Travel time can be typically modelled as time-series data [19], [20], [21]. For travel time uncertainty, existing work [13] has assumed a static uncertainty set where the nominal travel time and the travel time delay do not evolve throughout time. This clearly falls short in dynamic environments such as ITS [22].

To model the sequential relation of travel time among ride-sharing locations, we apply Gated Recurrent Units (GRUs) in training Recurrent Neural Networks (RNN) for travel time forecasting. GRUs can encode useful information from the past in single or multiple layers. The input of each layer is the output of the previous layer concatenated with the network input. Each GRUs layer predicts its output based on its current input and internal state.

We then use the GRUs technique to construct a dynamic uncertainty set for travel time that evolves along the time window. Given the input as an s-sequence time-series data $$(t_i^{(1)}, t_i^{(2)}, \ldots, t_i^{(s)})$$ where $$t_i^{(s)}$$ is the actual travel time between location $$u$$ and $$v$$ at time $$i$$, GRUs will derive $$t_i^{(s+1)}$$ and $$\tilde{t}_i^{(s+1)}$$, the predicted nominal travel time and travel time delay at time $$(s+1)$$, respectively. Subsequently, the uncertainty set for the travel time between location $$u$$ and $$v$$ at time $$i$$, $$\mathcal{U}$$, is updated to $$(\tilde{t}_i^{(s+1)} - t_i^{(s+1)}, \tilde{t}_i^{(s+1)} + t_i^{(s+1)})$$. Note that it is possible for the actual travel time $$t_i^{(s+1)}$$ to fall out of range of the aforementioned uncertainty set, leading to time constraint violation. In this sense, an accurate uncertainty set can precisely capture the inherent travel time uncertainty and ensure a robust ride-sharing matching solution.

B. SOLUTIONS BASED ON HYBRID LOCAL SEARCH AND SIMULATED ANNEALING ALGORITHM

We underscore that the one-to-many ride-sharing matching problem is a variant of VRPTW, which is known to be NP-hard. To solve the large-scale instances, we design and implement a hybrid meta-heuristic algorithm called LS-SA that combines local search (LS) and simulated annealing (SA) [23]. The components of LS-SA are introduced as follows.
1) SOLUTION REPRESENTATION
The solution is designed as a group of sequences with traveler locations. Each sequence begins with the driver’s origin location, followed by a set of rider’s origin locations and a set of rider’s destination locations, the driver’s destination location is used as the last element in the sequence. Notice that if the driver travels alone (no riders are matched with this driver), the sequence will only contain two elements: the driver’s origin and destination.

2) SOLUTION INITIALIZATION
A high-quality initial solution can accelerate algorithm convergence. We design the initial solution using the following mechanism. For each driver, the riders are filtered by the driver’s departure time and the driver’s latest arrival time. Namely, for the valid riders matched for the given driver, their latest departure time and latest arrival time must be between the driver’s departure time and the driver’s latest arrival time. Next, randomly select a few riders (less than or equal to capacity) with the sequence that origins in the first half of the sequence and destinations in the second half of the sequence (see Line 1, Algorithm 2). After the solution construction above, all the constraints in the ride-sharing matching model except the time constraint (10) are satisfied.

3) LOCAL SEARCH STRATEGY
We adopt the following two types of local search strategies. The first strategy is to randomly swap two riders’ origin (destination) locations within one route (see line 3, Algorithm 1). That is, if a rider’s departure (arrival) time at its origin (destination) violates the time constraints, swap the origin (destination) with the previous origin (destination) in the route. The second strategy is to move the violated riders’ origins (destinations) to another route (see line 7, Algorithm 1). Specifically, we select another route where the driver’s departure time is earlier than the latest departure time at the rider’s origin, and the driver’s latest arrival time is later than the latest arrival time at the rider’s destination. In addition, the number of pick-up riders must be less than the vehicle’s capacity. Notice that since multiple routes would satisfy the filtering conditions above, we define a ratio to measure the new route selection possibility as follows:

\[ \text{Rat}(d) = \frac{\text{lat}(d) - \text{ed}(d)}{|\mathcal{R}_d^\text{served}| + \delta} \]  

where \( \mathcal{R}_d^\text{served} \) denotes the set of served riders by driver \( d \), and \( \delta \) is a small number that is introduced to avoid zero-divided. Intuitively, a large ratio implies that the driver can potentially serve more riders. Then, we select the route that is traversed by driver \( d \) with the maximum ratio:

\[ d^* = \arg \max_{d \in \mathcal{D}} \text{Rat}(d) \]  

4) SOLUTION FITNESS COMPUTATION
Solution fitness results involve the following three parts: rider’s travel cost, unserved rider penalty cost, and rider matching time constraint violation cost. The first two parts can be obtained by computing the objective function value directly, and the last part is computed as the product of the number of violated riders and a large penalty coefficient \( \gamma \). The holistic description of LS-SA is summarized in Algorithm 2.

---

**Algorithm 1 Local Search**

**Input:** sol\text{incu}  
**Output:** sol\text{new}  
1: sol\text{new} ← sol\text{incu}  
2: \( u \) ← generate random number from range \([0, 1]\) uniformly  
3: if \( u \leq 0.5 \) then  
4: for each route with violated riders in the sol\text{new} do  
5: swap riders within the route  
6: end for  
7: else  
8: for each route with violated riders in the sol\text{new} do  
9: move the violated rider’s \((o, d)\) pair to another candidate route  
10: end for  
11: end if  
12: return sol\text{new}

**Algorithm 2 LS-SA**

**Input:** driver list (dl), rider list (rl), and the parameters of LS-SA (\( T_{\text{init}}, T_{\text{temp}}, \eta \))  
**Output:** sol\text{incu}  
1: sol\text{incu} ← get-initial-solution(dl, rl) \( \triangleright \) Initialization  
2: fit\text{incu} ← compute-fitness(sol\text{incu})  
3: \( T_{\text{curr}} ← T_{\text{init}} \)  
4: while \( T_{\text{curr}} \geq T_{\text{term}} \) \( \triangleright \) SA loop body  
5: sol\text{new} ← LocalSearch(sol\text{incu}) \( \triangleright \) Call Algorithm 1  
6: fit\text{new} ← compute-fitness(sol\text{new})  
7: if fit\text{new} ≤ fit\text{incu} then \( \triangleright \) Accept the better solution  
8: sol\text{incu} ← sol\text{new}  
9: fit\text{incu} ← fit\text{new}  
10: else \( \triangleright \) Accept the inferior solution with probability  
11: \( P_{\text{acc}} = \exp^{-1}\left(\frac{\text{fit}\text{new} - \text{fit}\text{incu}}{T_{\text{curr}}}\right) \)  
12: if random \( \leq P_{\text{acc}} \) then  
13: sol\text{incu} ← sol\text{new}  
14: fit\text{incu} ← fit\text{new}  
15: end if  
16: end if  
17: \( T_{\text{curr}} ← \eta * T_{\text{curr}} \)  
18: end while  
19: return sol\text{incu}
V. NUMERICAL STUDY

In this section, we conduct a set of numerical experiments to validate our proposed approach through metrics that are critical to the ride-sharing platform. We also evaluate the proposed approach by comparing its performance with the non-data-driven strategy.

A. EXPERIMENT SETUP

We use New York taxi trip records\(^3\) from January 2017 to June 2017 to validate our approach. The reformulated robust optimization models are solved by the commercial solver Gurobi 9.5.\(^4\) The experiments are run on a PC with Intel Core i7 CPU, 32GB RAM, Windows 10. In addition, the deep learning model of GRU is implemented by Python 3.7 and TensorFlow 2.1 under GeForce RTX 2080 GPU, 16 GB RAM, Ubuntu 18.04.

Since the duration of the rolling horizon is considered to be one hour, we discretize one day into 24 time slots and aggregate the riders’ information, e.g., origin, destination, earliest departure time, and latest arrival time, by each time slot. Since the drivers’ information is not recorded in the data set, we randomly generate their origins, destinations, earliest departure times, and latest arrival times for each time slot. Once the driver’s real information is available, it can be used as the input for the optimization model and LS-SA algorithm seamlessly.

\(\Gamma^d\) is introduced to control the degree of conservatism. The only uncertain factor is the travel time in this work. Hence, in our ride-sharing model, \(\Gamma^d\) denotes the number of arcs that are subject to uncertainty for driver \(d\). In this regard, the value of the bound \(B\) for the uncertainty set can be set to

\[
B = \frac{|D| + 2|R|}{|D|}
\]

by average. Then \(\Gamma^d\) varies in range \([0, B]\). Notice that if \(\Gamma^d\) is set to 0, the corresponding robust counterparts will become the deterministic models. The values of parameters in the robust optimization model and LS-SA algorithms are summarized in Table 3.

B. EVALUATION METRICS

- Matching Rate (MR). MR refers to the proportion of riders assigned to the drivers before the travel time uncertainty is realized.

\[
MR = \frac{|\hat{R}|}{|R|}
\]

where \(\hat{R}\) and \(R\) denote the set of matched riders and set of requested riders, respectively.

- Violation Rate (VR). VR is the metric that measures the robustness of the solution. This is a fairly critical metric because of the uncertain factor, the solutions that yield from the mathematical models (both deterministic and robust optimization models) may be infeasible (violate one or a few time constraints). In this sense, a more robust solution should be less sensitive to the travel time uncertainty that leads to the low violation rate.

\[
VR = \frac{|\bar{R}|}{|R|}
\]

where \(\bar{R}\) denotes the set of riders who violate the time constraint (origin departure and/or destination arrival time violations).

- Valid Serving Rate (VSR). VSR is introduced to measure the fraction of riders that do not violate the time constraint after uncertain travel time is realized. It can be computed via the following equation:

\[
VSR = MR \times (1 - VR)
\]

This metric can be used to measure the quality of service of the ride-sharing matching system. A higher VSR indicates that the ride-sharing system can provide better service to the riders in terms of timeline.

| Parameter | Value | Description |
|-----------|-------|-------------|
| \(\alpha\) | 1 | same as Table 2 |
| \(\beta\) | 5 | same as Table 2 |
| \(\gamma\) | 100 | time constraint violation penalty |
| \(C_d\) | 3 | same as Table 2 |
| \(H\) | 999,999 | same as Table 2 |
| \(T_{init}\) | 240 | initial temperature |
| \(T_{term}\) | 1 | terminated temperature |
| \(\eta\) | 0.999 | cooling factor |

| Parameter | Setting |
|-----------|---------|
| Number of neurons for input layer | 24 |
| Number of neurons for hidden layers | 256, 128 |
| Number of dropout rates for hidden layers | 20%, 20% |
| Activation function | ReLU |
| Optimization algorithm | Adam |
| Number of epoch | 50 |
| Batch size | 32 |
| Cross validation | 5 folds |
| Patience | 5 times |

---

\(^3\)https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page

\(^4\)https://www.gurobi.com/academia/academic-program-and-licenses/
• Computational Time. We compare the computational time of both exact solutions (from Gurobi solver) and approximate solutions (from heuristic algorithm LS-SA) on different sizes of instances, which can reflect the scalability of our algorithm.

C. TRAVEL TIME PREDICTION
We adopt GRU to predict nominal travel time and travel time delay. The data sets are split into the training set (from January 2017 to May 2017) and the testing set (June 2017). The number of units for the input layer (also known as travel time sequence window size in this problem setting) is set to 24 (24 time slots daily). The numbers of units for two hidden layers are set to 256 and 128 with dropout rates 20% and 20%, respectively. ReLU [24] and Adam [25] are used as the activation function and stochastic optimization algorithm, respectively. The number of epochs is set to 100, and the cross-validation is set to 5 to reduce the bias on the testing set. The parameters in the GRU deep learning model are summarized in Table 4. Finally, the travel time uncertainty set is dynamically constructed by the predicted outcomes as discussed in Section IV.

D. EXPERIMENTAL RESULTS
In this section, we consider two validation scenarios. The first one is a small-scale instance with five drivers and ten riders, the exact solution of which can be solved by the Gurobi solver in a reasonable time. The other is a large-scale instance that cannot be tackled by an exact method, and only a heuristic solution will be solved via LS-SA. In both scenarios, we compare the proposed approach with non-data-driven robust optimization using evaluation metrics developed in Sec V-B.

1) PERFORMANCE ON SMALL-SCALE INSTANCE
First, we compare our data-driven (DD) robust optimization model with the non-data-driven (NDD) one. For the NDD robust optimization model, the uncertainty set is simply constructed based on the historical average approach; for DD robust optimization model, the uncertainty set is constructed by the forecast results from GRUs. The performance of the two approaches in both exact solutions by Gurobi and heuristic solutions by LS-SA are compared and summarized in Table 5. The computational time using the Gurobi solver was approximate 198 seconds, while that for LS-SA was less than a second.

It can be observed that the objective value (OV) increases as the degree of conservatism ($\Gamma^d$) goes up, suggesting a higher total cost as conservatism weighs in. In particular, the RO model is reduced to a deterministic model when $\Gamma^d = 0$. As the degree of conservatism increases, Gurobi will likely land in a sub-optimal solution in avoiding time constraints violation, and this is accompanied by a higher OV. In addition,

| Testing Group | Description | Time Slot | $B$ |
|---------------|-------------|-----------|-----|
| A             | 100 drivers, 150 riders | 11-12 am, 10th | 4   |
| B             | 120 drivers, 198 riders  | 10-11 am, 1st  | 4.3 |
| C             | 150 drivers, 251 riders  | 6-7 pm, 30th   | 4.35|

For the exact and heuristic approaches, the results of the three metrics - MR, VR, and VSR are identical on this small-scale instance. The only difference between the two approaches is the overall system cost / objective value (OV). Therefore, we list the OV only.

The OV-Gap measures the solution gap between the approximate solution (from LS-SA) and the optimal solution (from Gurobi solver).

The data sets are split into the training set (from January 2017 to May 2017) and the testing set (June 2017). The number of units for the input layer (also known as travel time sequence window size in this problem setting) is set to 24 (24 time slots daily). The numbers of units for two hidden layers are set to 256 and 128 with dropout rates 20% and 20%, respectively. ReLU [24] and Adam [25] are used as the activation function and stochastic optimization algorithm, respectively. The number of epochs is set to 100, and the cross-validation is set to 5 to reduce the bias on the testing set. The parameters in the GRU deep learning model are summarized in Table 4. Finally, the travel time uncertainty set is dynamically constructed by the predicted outcomes as discussed in Section IV.

| Validating Set | Description | Time Slot | $B$ |
|----------------|-------------|-----------|-----|
| A              | 100 drivers, 150 riders | 11-12 am, 10th | 4   |
| B              | 120 drivers, 198 riders  | 10-11 am, 1st  | 4.3 |
| C              | 150 drivers, 251 riders  | 6-7 pm, 30th   | 4.35|

For the exact and heuristic approaches, the results of the three metrics - MR, VR, and VSR are identical on this small-scale instance. The only difference between the two approaches is the overall system cost / objective value (OV). Therefore, we list the OV only.

The OV-Gap measures the solution gap between the approximate solution (from LS-SA) and the optimal solution (from Gurobi solver).
the proposed DD approach outperforms NDD in all dimensions from OV, MR, VR to VSR. Notice that some results (e.g., MR under $\Gamma^d = 0$) are not significant because of the small testing instance. This will become more evident with large-scale instances as will be shown later.

Second, we compare the performance of the exact solution by Gurobi and the heuristic solution by our proposed LS-SA meta-heuristic algorithm. It can be observed from the last two rows of Table 5 that LS-SA delivers a near-optimal solution with less than 10% gap in OV compared to Gurobi. This supports the application of LS-SA on large-scale instances for near-optimal matching solutions.

2) PERFORMANCE ON LARGE-SCALE INSTANCES

We select three large-scale instances with different numbers of drivers and riders as summarized in Table 6. The values of $\Gamma^d$ can be computed by Eq. (23). Since the LS-SA algorithm only provides an approximate solution, we average the validation results by running ten times for each instance. We then compare our proposed DD robust optimization approach with NDD in terms of MR, VR, and VSR for each large-scale instance. The results are summarized in Table 7, 8, and 9.

We first observe that the rider’s MR decreases with $\Gamma^d$ under both DD and NDD approaches, which is reasonable as the solution becomes more conservative when the number of allowed travel time delay increases. However, MR in the DD approach is normally higher than that in the NDD approach, with an average of 8.9% and for as much as 27.9%.

We also observe that VR drops under both DD and NDD approaches as the level of conservatism increases. This is because as the matching solution becomes more conservative, it can handle more realistic scenarios, which is also known as solution robustness. Specifically, compared to NDD approach, VR under the DD approach is reduced by 59.6% on average and up to 85.8%. This can be attributed to a more accurate travel time prediction in the DD approach. By leveraging the spatio-temporal travel time historical data, we can construct a more appropriate uncertainty set to describe the uncertainty where the violation happens with a low probability. As a result, riders can be assigned to drivers in a way that the time constraints are less likely to be violated compared to the NDD approach. Notice that if we are allowed to modify the time constraint type from hard to soft (i.e., the riders can tolerate a few minutes delay, which is quite common in real life), the VR could be reduced even further.

In connection with the above observations, the degree of conservatism has the same effect on MR and VR. Ideally, we wish to land on a more inclusive solution (high MR) without compromising robustness (low VR). However, a low degree of conservatism may sacrifice robustness (a high MR and a high VR), while a high degree of conservatism may reduce inclusiveness (a low MR and a low VR). It is therefore critical to reach a proper balance between MR and VR through the configuration of the degree of conservatism. For this purpose, the metric VSR can be a useful indicator. As can be observed from the numerical, VSR under DD is higher than under NDD with an average of 23.9% and may go as much as 42.3%. Therefore, our proposed DD approach is superior in balancing the trade-off between MR and VR. In addition, it can be observed across all instances that VSR
X. Li et al.: Ride-Sharing Matching Under Travel Time Uncertainty Through Data-Driven Robust Optimization

TABLE 9. Comparison of data-driven and non-data-driven approaches on large-scale instance, testing group C.

| Heuristic | 0%  | 10% | 20% | 30% | 40% | 50% | 60% | 70% | 80% | 90% | 100% |
|-----------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|------|
| MR        | NDD | 84.1% | 82.9% | 82.5% | 79.7% | 77.7% | 76.9% | 76.1% | 75.7% | 73.3% | 72.9% | 68.9% |
| DD        |     | 88.8% | 88.4% | 82.9% | 82.5% | 80.1% | 79.7% | 79.3% | 78.9% | 78.5% | 76.5% | 74.9% |
| VR        | NDD | 35.4% | 32.8% | 30.1% | 27.9% | 24.9% | 23.3% | 23% | 22.6% | 21.7% | 21.3% | 18.8% |
| DD        |     | 24.2% | 21.6% | 15.4% | 14.5% | 11.1% | 10% | 9% | 8.1% | 7.1% | 6.9% | 6.4% |
| VSR       | NDD | 54.3% | 55.7% | 57.7% | 57.8% | 58.4% | 59% | 58.6% | 58.6% | 57.4% | 57.3% | 55.9% |
| DD        |     | 67.3% | 69.3% | 70.1% | 70.5% | 71.2% | 71.7% | 72.2% | 72.5% | 72.9% | 71.2% | 70.1% |

FIGURE 1. Computational time of LS-SA for testing group A, B, and C.

peaks when $\Gamma^d$ is between 30% to 80% of $B$. This essentially recommends the range within which a decision-maker may adjust the degree of conservatism in obtaining the best robust matching solutions.

Finally, the computational time of the three instances is illustrated in Fig. 1. We highlight that the computational time for Testing Group C with 150 drivers and 251 riders only takes 20.7 seconds, which is quite satisfactory for a one-hour rolling horizon. In comparison, Gurobi cannot reach the optimum within 48 hours even for Testing Group A. Since the proposed algorithm is highly scalable, our proposed approach is considerably practical for large-scale applications. Also, the computational time can be further reduced through adjusting the cooling factor (i.e., reducing the number of iterations) or using GPUs as an acceleration strategy similar to [26].

VI. CONCLUSION AND FUTURE WORK

This paper proposes a data-driven robust optimization framework for addressing the one-to-many ride-sharing matching problem under travel time uncertainty. The approach integrates deep learning-based GRU technique with robust optimization modeling technique, which can be readily applied to many other relevant settings beyond ride-sharing. By exploiting historical travel time data, the approach can dynamically estimate travel time uncertainty throughout the entire rolling horizon. Numerical validations suggest that compared to a non-data-driven approach or conventional solver, our proposed method can significantly improve key performance metrics of the ride-sharing matching system in a time-efficient manner. In the proposed model, all the constraints in the robust optimization model are assumed to be hard, which is rather restrictive, given that some extent of flexibility does exist in ride-sharing systems. We can relax the time constraints to soft ones to increase the matching rate and reduce the violation rate.

This work can be extended in many different directions. One potential is to enhance the strategy in determining a more accurate uncertainty set given historical travel time data. Also, for data with complex patterns where the basic uncertainty set can no longer capture the intrinsic pattern, combining different types of uncertainty sets and relevant mechanisms behind them could be a meaningful future study.

REFERENCES

[1] N. Alisoltani, L. Leclercq, and M. Zargayouna, “Can dynamic ride-sharing reduce traffic congestion?” Transp. Res. B, Methodol., vol. 145, pp. 212–246, Mar. 2021.
[2] L. D. C. Martins, R. de la Torre, C. G. Corlu, A. A. Juan, and M. A. Masmoudi, “Optimizing ride-sharing operations in smart sustainable cities: Challenges and the need for agile algorithms,” Comput. Ind. Eng., vol. 153, Mar. 2021, Art. no. 107080.
[3] H. Wang and H. Yang, “Ridesourcing systems: A framework and review,” Transp. Res. B, Methodol., vol. 129, pp. 122–155, Nov. 2019.
[4] P. Munari, A. Moreno, J. De La Vega, D. Alem, J. Gondzio, and R. Morabito, “The robust vehicle routing problem with time windows: Compact formulation and branch-price-and-cut method,” Transp. Sci., vol. 53, no. 4, pp. 1043–1066, Jul. 2019.
[5] A. Tafreshian, N. Masoud, and Y. Yin, “Frontiers in service science: Ride matching for peer-to-peer ride sharing: A review and future directions,” Service Sci., vol. 12, nos. 2–3, pp. 44–60, Jun. 2020.
[6] A. Mourad, J. Puchinger, and C. Chu, “A survey of models and algorithms for optimizing shared mobility,” Transp. Res. B, Methodol., vol. 123, pp. 323–346, May 2019.
[7] W. Lu, L. Liu, F. Wang, X. Zhou, and G. Hu, “Two-phase optimization model for ride-sharing with transfers in short-notice evacuations,” Transp. Res. C, Emerg. Technol., vol. 114, pp. 272–296, May 2020.
[8] Y. Jia, W. Xu, and J. Liu, “An optimization framework for online ride-sharing markets,” in Proc. IEEE 37th Int. Conf. Distrib. Comput. Syst. (ICDCS), Jun. 2017, pp. 826–835.
[9] Y. Guo, Y. Zhang, and Y. Boulaksil, “Real-time ride-sharing framework and mechanism behind them could be a meaningful future study.
X. Li et al.: Ride-Sharing Matching Under Travel Time Uncertainty Through Data-Driven Robust Optimization

[13] Y. Li and S. H. Chung, “Ride-sharing under travel time uncertainty: Robust optimization and clustering approaches,” Comput. Ind. Eng., vol. 149, Nov. 2020, Art. no. 106601.

[14] J. Long, W. Tan, W. Y. Szeto, and Y. Li, “Ride-sharing with travel time uncertainty,” Transp. Res. B. Methodol., vol. 118, pp. 143–171, Dec. 2018.

[15] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, “Empirical evaluation of gated recurrent neural networks on sequence modeling,” 2014, arXiv:1412.3555.

[16] D. Bertsekas, Network Optimization: Continuous and Discrete Models. Nashua, NH, USA: Athena Scientific, 1998.

[17] Z. K. Li, R. Ding, and C. A. Floudas, “A comparative theoretical and computational study on robust counterpart optimization: I. Robust linear optimization and robust mixed integer linear optimization,” Ind. Eng. Chem. Res., vol. 50, no. 18, pp. 10567–10603, 2011.

[18] D. Bertsimas and M. Sim, “The price of robustness,” Oper. Res., vol. 52, no. 1, pp. 35–53, Jan. 2004.

[19] J.-S. Yang, “A study of travel time modeling via time series analysis,” in Proc. IEEE Conf. Control Appl., Aug. 2005, pp. 855–860.

[20] A. Guin, “Travel time prediction using a seasonal autoregressive integrated moving average time series model,” in Proc. IEEE Intell. Transp. Syst. Conf., Sep. 2006, pp. 493–498.

[21] G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, Time Series Analysis: Forecasting and Control. Hoboken, NJ, USA: Wiley, 2015.

[22] Y. Wang, D. Zhang, Y. Liu, B. Dai, and L. H. Lee, “Enhancing transportation systems via deep learning: A survey,” Transp. Res. C, Emerg. Technol., vol. 99, pp. 144–163, Jan. 2019.

[23] P. J. Van Laarhoven and E. H. Aarts, “Simulated annealing,” in Simulated Annealing: Theory and Applications. New York, NY, USA: Springer, 1987, pp. 7–15.

[24] A. L. Maas, A. Y. Hannun, and A. Y. Ng, “Rectifier nonlinearities improve neural network acoustic models,” in Proc. ICML. Princeton, NJ, USA, Citeseer, vol. 30, no. 1, 2013, p. 3.

[25] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” 2014, arXiv:1412.6980.

[26] J. Gao, T. Wong, and C. Wang, “Social welfare maximizing fleet charging scheduling through voting-based negotiation,” Transp. Res. C, Emerg. Technol., vol. 130, Sep. 2021, Art. no. 103304.

---

JIE GAO received the M.A.Sc. degree in information systems engineering from Concordia University, Montreal, Canada, in 2016, and the Ph.D. degree in information systems engineering from Concordia, in 2021. She is currently a Postdoctoral Research Fellow with HEC Montréal. Her research interests include data-driven optimization, game theory, mechanism design, and machine learning with applications in intelligent transportation systems, smart cities, and community healthcare.

CHUN WANG (Member, IEEE) is currently a Professor with the Concordia Institute for Information Systems Engineering, Concordia University, Canada. His research interests include the interface between economic models, operations research, and artificial intelligence. He is actively conducting research in multiagent systems, data-driven optimization, and economic model-based resource allocation with applications to healthcare management, smart grid, and smart city environments.

XIAO HUANG received the B.E. degree in electronic engineering from Tsinghua University, the M.S. degree in mathematical finance from the University of Southern California (USC), and the Ph.D. degree from the Marshall School of Business, USC. She is currently an Associate Professor and the Concordia University Research Chair in supply chain management with the John Molson School of Business, Concordia University. Her research interests include competition and cooperation in supply chains, product and pricing strategies, and data-driven decision-making.

YIMIN NIE received the B.S. and M.S. degrees in theoretical physics from Peking University, China, in 2002 and 2005, respectively, and the Ph.D. degree in computational neuroscience from the Canadian Center of Behavior Neuroscience, University of Calgary, Canada, in 2014. He is currently working as a Senior Data Scientist and an AI Researcher with Global AI Accelerator (GAIA), Ericsson Inc. He worked as a Senior Data Scientist in multiple business fields including e-commerce, finance, and telecommunication. His research interests include machine learning, computer vision, and natural language processing.

---

XIAOMING LI received the M.S. degree in computer software and theory from Northeastern University, China, in 2009, and the Ph.D. degree in information and systems engineering from Concordia University, Canada, in 2022. He is currently a Research Associate with Concordia University. His research interests include optimization under uncertainty, large-scale optimization, network optimization, machine learning with applications in intelligent transportation systems, supply chain optimization, and vehicular network design.

---

VOLUME 10, 2022