Eco-Friendly Crowdsourced Meal Delivery: A Dynamic On-Demand Meal Delivery System with a Mixed Fleet of Electric and Gasoline Vehicles

Haishan Liu, *Graduate Student Member, IEEE*, Peng Hao, *Member, IEEE*, Yejia Liao, *Graduate Student Member, IEEE*, Shams Tanvir, *Member, IEEE*, Kanok Boriboonsomsin, *Member, IEEE*, and Matthew J. Barth, *Fellow, IEEE*

***Abstract*— The emerging prevalence of electric vehicles (EVs) in shared mobility services has led to a groundbreaking trend for decarbonizing the shared mobility sector. However, it is still unclear how to maximize the efficiency of EVs to reduce greenhouse gas (GHG) emissions while maintaining high service quality, particularly considering the ongoing transition towards a fully electrified service fleet. In this paper, focusing on meal delivery, we proposed an eco-friendly on-demand meal delivery (ODMD) system to maximize the utilities of EVs to mitigate GHG emissions and maintain low operational cost and delay cost. The main feature of our system is that its fleet consists of electric and gasoline vehicles mirroring the evolving electrification trend in the shared delivery sector. A rolling horizon framework integrated with the adaptive large neighborhood search (RH-ALNS) algorithm was proposed to efficiently solve the meal order dispatching and routing problem with the mixed fleet. Three delivery policies were explored in the numerical study. Experiment results demonstrated that it is necessary for online meal delivery platforms to actively collect information of electric vehicles and take initiative to employ an eco-friendly delivery policy.**

***Index Terms*— Dynamic on-demand meal delivery, mixed fleet dispatching, eco-friendly meal delivery, adaptive large neighborhood search (ALNS), eco-friendly delivery.**

I. Introduction

CATALYZED by the prevalence of information and communication technology and boosted by the unexpected COVID-19 pandemic, on-demand meal delivery (ODMD) has achieved explosive growth [1]. In the U.S., food delivery now comprises 14% of the total restaurant market [2]. The revenue from ODMD is projected to reach 63.02 billion dollars by 2022, growing annually at 8.9% [3]. Crowdsourced delivery emerged to meet the demand for faster and lower-cost deliveries by outsourcing meal delivery tasks to on-demand, non-professional drivers. The operation of crowdsourced meal delivery is supported by online platforms (e.g., UberEats, Meituan, Deliveroo, etc.), which handles everything from offering various food options, facilitating food ordering and digital payment, to scheduling delivery tasks and driver routing [4].

Meanwhile, electric vehicles (EVs) are gaining fast adoption due to their zero tailpipe emissions. In ODMD services, DoorDash reported that over 56 million orders were fulfilled using low or no-emissions vehicles [5]. The foreseen increasing EVs in ODMD service can be attributed to several factors. Firstly, as EVs continue to penetrate the car market, it is becoming increasingly common for delivery drivers to use their own EVs to complete meal delivery tasks. Secondly, shared delivery companies have launched pilot programs to encourage EV owners to participate in ride-hailing and on-demand delivery services. For instance, Uber provides an additional one-dollar subsidy for trips completed by fully electric vehicles [6]. Thirdly, local and state governments are also implementing regulations to guide the electrification of shared mobility services. The Clean Miles Standard proposed in California, for example, requires transportation network companies such as Uber and Lyft to achieve 90% electric vehicle miles traveled (eVMT) by 2030 [7]. Both Uber and Lyft have committed to transitioning to a 100% EV fleet [8], [9]. Despite these initiatives, challenges such as limited range, high upfront purchase price, unevenly distributed charging stations, and long charging time have delayed the transition to a purely EV delivery fleet. Most on-demand mobility and delivery companies in the U.S. currently operate a mixed delivery fleet consisting of conventional gasoline-fueled vehicles and electric vehicles, which offers opportunities to curtail GHG emissions during the delivery process if EVs are utilized properly.

In this research, we aim to explore an effective meal delivery strategy that can efficiently utilize EVs in the mixed fleet for meal order deliveries to reduce greenhouse gas (GHG) emissions while minimizing operational cost and delay cost. To achieve this goal, we first proposed an eco-friendly crowdsourced meal system with the following key characteristics: (1) dynamism: order demand and driver resources are continuously arriving; (2) a mixed fleet: crowdsourced drivers

Manuscript received 14 May 2023; revised 2 November 2023; accepted 22 January 2024. This work was supported by the National Center for Sustainable Transportation (NCST). The Associate Editor for this article was X. Li. *(Corresponding author: Peng Hao.)*

Haishan Liu, Yejia Liao, and Matthew J. Barth are with the Department of Electrical and Computer Engineering, University of California at Riverside, Riverside, CA 92507 USA.

Shams Tanvir is with the Department of Civil Engineering & Construction Engineering Management, California State University Long Beach, Long Beach, CA 9084 USA.

Peng Hao and Kanok Boriboonsomsin are with CE-CERT, University of California at Riverside, Riverside, CA 92507 USA (e-mail: haop@cert.ucr.edu).

Digital Object Identifier 10.1109/TITS.2024.3367621

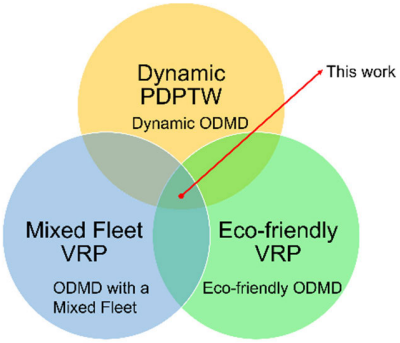

Fig. 1. Review scope and application in ODMD.

utilize both GVs and EVs; and (3) eco-friendliness: explicitly minimize GHG emissions and other costs in ODMD. Next, an efficient and effective optimization approach, RH-ALNS, was developed to solve the dynamic ODMD problem. This approach includes a rolling horizon (RH) framework to handle dynamic meal orders and driver information and an adaptive large neighborhood search (ALNS) algorithm to obtain the optimized plan for order dispatching and routing.

To validate the proposed algorithm and demonstrate the benefits of the eco-friendly crowdsourced meal system, we conducted real-world numerical simulations in the city of Riverside, California. Three delivery policies were tested in the numerical experiments to demonstrate the importance of actively utilizing EVs in the mixed fleet to curtail GHG emissions. In summary, the main contributions of this paper could be summarized as follows.

- We modeled the mixed fleet scenario in the dynamic ODMD business, which is an important scenario that could significantly reduce GHG emissions by effectively utilizing the EVs in the delivery fleet. To the best of our knowledge, this is the first work investigating mixed fleet dispatching policy to improve the sustainability of ODMD.
- By fully considering the feature of dynamic ODMD and the charging needs of EVs, the proposed RH-ALNS algorithm was tailored by specifying the requirements for the removal and repair operators to achieve the optimized dispatching and routing plan. The performance of RH-ALNS was compared to two other state-of-the-art heuristic algorithms.
- A real-world meal delivery simulation was constructed in the City of Riverside, California. Simulation results consolidated the importance for ODMD platforms to fully utilize the EVs in the mixed fleet to reduce GHG emissions.

II. Literature Review

This section first briefly reviewed the literature on pickup and delivery problem with time windows (PDPTW) and vehicle routing problem (VRP) related to the proposed ODMD system characteristics: dynamic, mixed fleet, and eco-friendly. Next, in terms of the application scenario, we focused on the recent work about ODMD and classified them based on the three characteristics. The reviewed scope is shown in Figure 1.

A. Dynamic PDPTW

The proposed meal delivery problem can be considered as an extension of the general pickup and delivery problem with time window (PDPTW) [10], [11], as delivery drivers need to pick up the orders first then deliver to the corresponding location with time window constraints. Unlike the static PDPTW where delivery demand and fleet supply were deterministic and known as a priori of the model, our studied problem falls into the dynamic type [12], where the system information including meal orders and crowdsourced drivers are dynamically revealed. A popular approach to solve dynamic PDPTW is the rolling horizon method which aims to re-optimize the solution according to the newly revealed information. The time interval between each re-optimization is critical. Some studies proposed a reactive time interval, indicating the system re-optimizes upon each new request arrival [13], [14]. This approach allows real-time updates but is computationally expensive if the requests arrive continuously in every time unit. Instead, other studies applied the periodic rolling horizon which divides the operating horizon into a sequences of time intervals and performs a re-optimization step of the solution. For example, Karami et al. proposed a periodic optimization approach with buffering strategy to solve a dynamic logistics scheduling problem. Their results show that the performance of scheduling plan is positively affected by the system dynamism while negatively affected by the request urgency [15]. Mitrovic-Minic et al. considered the impact of long-term and short-term time interval setting and proposed a double-horizon based heuristics to leverage the slack time in the future to help reduce routing cost [16]. Jia et al. utilized a periodic and event-driven rolling horizon procedure to minimize the total travelling time with dynamic events of traffic congestion, customer delivery time change, and new requests arrival [17]. In this study, we implemented a periodic rolling horizon framwork to update the whole system status and monitor the task completeness.

B. Mixed Fleet Vehicle Routing Problem

There are many inspiring works regarding the dispatching and management of a mixed energy fleet. Different with the traditional VRP problem, the mixed fleet VRP has to explicitly enforce the fleet composition constraint which guarantees that dispatching of vehicles is consistent with the composition of the fleet. Additionally, given the range limit of EVs and longer charging times, the model should consider the fleet capacity changes due to the EV charging needs. Goeke and Schneider developed a comprehensive mixed fleet routing problem for EVs and internal combustion vehicles (ICVs), integrating a realistic energy consumption model for EVs to determine driving range and charging times at stations. The study's results indicate that the objective function has a substantial impact on EV usage [18]. Hiermann et al. further studied a more complicated fleet which combining with ICVs, EVS and plug-in hybrid vehicles (PHEV) and constructed a genetic algorithm with local and large neighborhood search to efficiently solve the routing problem. Their results show that effective usage of a mixed fleet is beneficial in saving

operational cost compared to the single type fleet case [19]. Maasmoudi et al. investigated the mixed fleet routing problem in the dial-a-ride scenario, which is most related to our studied problem. But this research only considered a static order information and was with a focus on analyzing the algorithm performance instead of investigating the dispatching policy of a mixed fleet [20]. Besides, bi-objective optimization was explored in the mixed fleet VRP in order to explore the trade-off between service efficiency and environmental impacts [21], [22], [23], [24]. Other studies further extended the mixed fleet VRP by integrating the charging behaviors of EVs, such as partial battery recharging [25], charging station selection considering charging cost [26] and charging power [23], battery swapping [27], etc.

C. Eco-Friendly Vehicle Routing Problem

From the eco-friendly perspective, some VRPs focused on minimizing the energy consumption or pollutant emissions. A comprehensive survey of eco-friendly VRP can be found in [28]. Fuel consumption is an important component when constructing the energy minimizing VRP model. Kara et al. first proposed the Energy Minimizing Vehicle Routing Problem (EMVRP) which considering the travel distance and load of vehicles [29]. But the energy consumption model is simplistic which fail to consider the vehicle speed during the routing process. Further, a more realistic energy consumption model was proposed by Goeke et al with the consideration of vehicle mass, travel speed an road gradient [18]. An overview of electric vehicle energy consumption models can be found in [30]. Regarding the emission reduction for vehicle routing, Bektas and Laporte proposed the pollutant routing problem (PRP) which minimizes the weighted sum of GHG emissions, operational cost, and fuel consumption [31]. Sawik et al. presented multi-criteria optimization models to study the problem of maximization of truck capacity meanwhile minimization of fuel consumption, carbon mission and noise in the logistics production [32]. Further, a Comprehensive Modal Emission Model (CMEM) [33] was proposed by Barth et al. that can predict the high-resolution vehicle emissions given the vehicle trajectory (speed, acceleration, road grade, etc.), which has been widely adopted in the eco-friendly vehicle routing research, such as [20] and [31]. In this research, we used the EMFAC model to estimate the GHG emission from passenger GV, which only requires the vehicle link-level speed to obtain the emission rate and is well maintained by California Air Resources Board [34].

D. On-Demand Meal Delivery (ODMD)

On-demand meal delivery service is an emerging area, as most research results were published during the past five year. A portion of studies assumed perfect information of meal orders to sidestep the system dynamism. Liu et al. proposed to leverage the taxi resources to deliver food order either in opportunistic manner or in dedicated manner with the goal to minimize taxi number and distance cost [35]. Tu et al. developed an online dynamic optimization framework which includes order collection, solution generation on and sequential delivery [36]. Wang et al. presented an insertion-based heuristic to solve a single driver food delivery routing problem along with the geographic information to accelerate the insertion process [37]. Yildiz and Savelsbergh assumed perfect information of meal orders and introduced the concept of work-package which integrates both space and time consistency constraints. The problem was solved by a column and row generation method [38]. Liao et al. studied a green meal delivery routing problem with multi-objectives to maximize the customer satisfaction, rider utilization and minimize carbon footprint [39].

On the other hand, the dynamic meal delivery problem also receives attentions from many researchers. Reyes et al. studied the meal-delivery routing problem (MDRP) and proposed a rolling-horizon algorithm to solve the dynamic vehicle routing and capacity management problem [40]. Zhou et al. formulated an online order dispatched system with new orders arrival, but this research only solved the problem in one time interval without considering the platform update [41]. Huang et al. proposed a UAV-based ODMD system, however, UAV is largely constrained by the load limits [42]. Steever et al. developed a 'pro-active' heuristic to predict the future delivery needs [43].

The most proposed objectives in the ODMD problems are from the operational efficiency perspective, including minimizing driver delivery distance [35], [44], [45], driver waiting time [45], total driver compensation [38], order delivery delay [43], [46], order cancellation rate [41], etc. Only Liao et al explicitly minimize the carbon footprint of meal delivery drivers [39]. But this paper only considered the static ODMD scenario with the generated datasets to verify the algorithm performance without noticing the fleet composition. Based on the research from Liu et al. [47] and Allen et al. [48] about model-based and comparative evaluation of emissions impacts from the ODMD deliveries, results have emphasized that it is necessary to encourage the use of zero emission vehicles for these journeys. However, it is still unclear about how to efficiently utilize the green vehicles in the mixed fleet. On the other hand, some studies and industry companies have proposed the UAV for meal deliveries [42], [49], which still has a long way to go before mass adoption.

III. Problem Description and Formulation

A. Problem Description

We considered a decision maker represented by an online platform that performs on-demand meal delivery with a mixed fleet of electric and gasoline vehicles. Specifically, G-driver and E-driver were used to distinguish the delivery driver with a gasoline vehicle and an electric vehicle respectively. Meal orders are arriving dynamically, and the system only optimizes the solution based on the revealed order information. Each meal order has the information about order placing time, customer location, and restaurant location. The crowdsourced drivers can log on and log off the system freely during the operating time depending on drivers' working schedules. The platform needs to dispatch meal orders to either E-drivers or G-drivers according to the objectives and the dispatching policy.

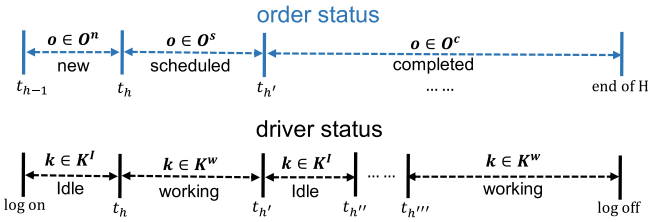

Fig. 2. Meal order and driver status definition.

Given the dynamic nature of meal orders and crowdsourced drivers, the platform should update the dispatching result periodically, where the rolling horizon approach was widely used [40], [46]. Generally, the whole operational horizon H was divided into $\lceil \frac{H}{\tau} \rceil$ time intervals with a length of τ. Suppose the platform begins at time t_0. Then at every time step t_h $(where\ t_h = t_0 + h \times \tau\ ,\ h = (1, 2, \ldots, \lceil \frac{H}{\tau} \rceil)$, the system re-optimizes the order dispatching decision regarding the new meal order demand and the updated driver information during $[t_{h-1}, t_h)$.

As shown in Figure 2, we defined three types of order status. At time t_h, if an order o is placed in the range of $[t_{h-1}, t_h)$, then it falls in the new order set O^n. The platform dispatches the new orders to drivers at the re-optimization time, after which the order status changes to "scheduled" in O^s. The order status is switched to completed (defined as O^c) when the order is finally delivered. Each crowdsourced driver has two states: working (in setK^w) or idle (inset K^I). Driver k first logs on to the platform and is at idle status waiting for the order dispatching notification. Suppose at time t_h, the driver receives delivery tasks and starts working. In addition, some drivers may have multiple delivery trips and their statuses keep switching between "working" and "idle" until they leave the platform.

The system overview is shown in Figure 3. First, at re-optimization time t_h the online operation platform collects new order information, generates pick-up and drop-off tasks for each order, and checks current active drivers. Given this information as input, the ODMD system optimizes the order dispatching and driver routing decisions. The goal is to minimize greenhouse gas (GHG) emissions, operational cost, and meal delivery delays. Note that only the GHG emissions from G-drivers during the meal delivery process are considered since the GHG emissions from E-drivers during the delivery process are negligible. The life cycle GHG emissions in electricity generation for EVs are not considered in this study. The order dispatching is subject to constraints of order time windows, number of available drivers, range limit of EVs, etc. With the optimized solution, drivers will receive updated routing information and finish the delivery tasks sequentially. At the next time step t_{h+1}, the system will repeat the above steps to construct a new sub-problem with respect to the new delivery demand.

B. Problem Assumption

To formulate the dynamic ODMD problem, we have the following assumptions:

- All orders must be delivered. The order rejection and cancellation are not covered in this study. Drivers cannot reject the assigned order in the system.
- After the assignment, order cannot be transferred between drivers. A driver should finish both pick-up and drop-off tasks of a meal order.
- New drivers should wait at the initial location before the first order is assigned. After one delivery trip, drivers should wait at the last drop-off location until the next order dispatching.
- Drivers should complete the ongoing task and then adjust the following delivery plan with respect to the new command. They cannot change their immediate destination when heading to a restaurant/customer.
- To ensure delivery speed, we restricted E-drivers from getting battery charged during the working hour. If EVs reach the charging threshold, drivers should log off the system to reach a charging station. E-drivers should get fully charged by themselves and will be treated as new drivers if they return to the ODMD system. Thus, the EV charging problem is not considered in the platform but is delegated to the E-drivers.

C. Problem Formulation

An optimization model can be formulated at each reoptimization time step t_h. The notation definitions are summarized in Table I. Let us consider the online platform with a set P^h of pick-up tasks, a set D^h of drop-off tasks and a set K^h of crowdsourced drivers. Current delivery fleet K^h includes m_E^h E-drivers and m_G^h G-drivers. Each E-driver has a range limit of RL_k. Given the meal delivery demand and the mixed fleet, the system needs to determine the matching between drivers and meal orders and the routing plan for drivers. A directed graph $G = (V, E)$ can be defined. Each node in V $\left(V = P^h \cup D^h \cup K^h\right)$ represents the location of a customer, a restaurant, or a driver. Each arc in $E(E = V \times V)$ represents movement from one node to another.

The goal of the Eco-friendly ODMD system is to balance the operational efficiency and the environmental impact. The objective function (equation (1)) consists of three components: delivery distance cost, meal delay cost and GHG emission cost, which in essence is a multi-objective optimization problem [50]. In this paper, the weighted sum method is employed to iteratively search for a Pareto-efficient solution in the dynamic ODMD scenario. Due to the linearity of the cost function, it is also guaranteed that a solution obtained from the weight sum method is a Pareto-efficient solution [51].

$$\begin{aligned} \min F = \ & \alpha \sum\nolimits_{k \in K^h} \sum\nolimits_{(i \in V, j \in V, i \neq j)} c_d d_{ij} x_{ij}^k \\ & + \beta \sum\nolimits_{i \in D^h} c_t p_c \max\left(0,\ t_{do}^i - t_{do}^{ie}\right) \\ & + \gamma \sum\nolimits_{k \in K^h} \sum\nolimits_{(i \in V, j \in V, i \neq j)} c_e E_{ij} x_{ij}^k y^k \end{aligned} \tag{1}$$

The first term sums up the total delivery distance, multiplied by an operational cost factor c_d. The second term is the delay cost, where the delivery delay is defined with the difference between the order's actual drop-off time t_{do}^i and the expected drop-off time t_{do}^{ie} provided by the system. A linear delay

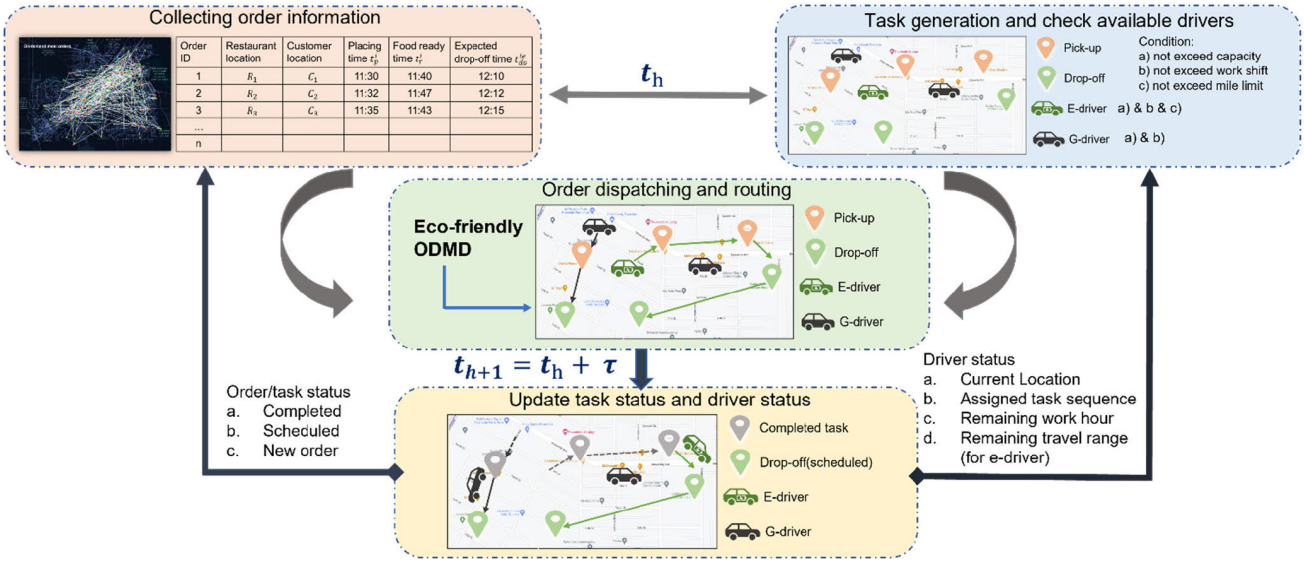

Fig. 3. The overview of the dynamic ODMD system with a mixed fleet.

penalty p_c is enforced to penalize long delay time. The delay term is multiplied by the delay cost factor c_t to represent the delay cost. The last term is the total GHG emission cost from G-drivers. The GHG emission E_{ij} is multiplied by the emission cost factor c_e. The weighted factors of the three terms are α, β, γ, which can be tuned to achieve the balance among the objectives.

The operational constraints of the ODMD system are grouped into the following five categories.

1) Route Construction Constraints:

$$\sum_{k\in K^h}\sum_{j\in V} x_{ij}^k = 1 \quad \forall i \in P^h \cup D^h \tag{2}$$

$$\sum_{j\in V} x_{k_0,\,j}^k = 1 \quad \forall\, k \in K^h \tag{3}$$

$$\sum_{i\ \in V} x_{i,k_0}^k = 1 \quad \forall k \in K^h \tag{4}$$

$$\sum_{i\ \in V} x_{ij}^k - \sum_{i\in V} x_{ji}^k = 0 \quad \forall j\ \in P^h \cup D^h, \quad \forall k \in K^h \tag{5}$$

$$\sum_{j'\ \in V} x_{ij'}^k - \sum_{j'\ \in V} x_{j'j}^k = 0 \quad \forall (i,j)\ \in R^h,\ \forall\, k \in K^h \tag{6}$$

$$T_i^k \le T_j^k \quad \forall k \in K^h,\ \forall (i,j)\ \in R^h \tag{7}$$

2) Mixed Fleet Constraints:

$$\sum_{j\in V} x_{k_0 j}^k y^k \le m_G^h \quad \forall k \in K^h \tag{8}$$

$$\sum_{j\in V} x_{k_0 j}^k (1-y^k) \le m_E^h \quad \forall k \in K^h \tag{9}$$

$$\sum_{(i\in V, j\in V, i\ \neq j)} (1-y^k)(d_{ij}x_{ij}^k + \ \delta * RL_k) \le RL_k^h \quad \forall k \in K^h \tag{10}$$

3) Capacity Constraints:

$$Q_j^k \ \ge\ Q_i^k + q_j \quad \forall\ i, j \in \mathrm{V},\ \forall k \in K^h \tag{11}$$

$$Q_i^k \le Q^k \quad \forall i\ \in V,\ \forall k \in K^h \tag{12}$$

4) Time Window Constraints:

$$t_{pu}^i = \max\left(t_r^i,\ T_i^k\right) \quad \forall i \in P^h,\ \forall k \in K^h \tag{13}$$

$$T_j^k \ \ge\ t_{pu}^i + t_{ij} + s_i \quad \forall\ i \in P^h,\ \forall j \in V,\ \forall k \in K^h \tag{14}$$

$$T_j^k \ \ge\ t_{do}^i + t_{ij} + s_i \quad \forall\ i \in D^h,\ \forall j \in V,\ \forall k \in K^h \tag{15}$$

$$T_j^k \ \ge\ T_i^k + t_{ij} + CT \quad \forall\ i \in F',\ \forall j \in P^h, \forall k \in K^h \tag{16}$$

5) Variable Constraints:

$$x_{ij}^k \in \{0,\ 1\} \quad \forall (i,j) \in E,\ \forall k \in K^h \tag{17}$$

$$y^k \in \{0,\ 1\} \quad \forall k \in K^h \tag{18}$$

$$Q_i^k \ge 0 \quad \forall i\ \in V,\ \forall k \in K^h \tag{19}$$

$$T_i^k \ge 0 \quad \forall i\ \in V,\ \forall k \in K^h \tag{20}$$

Route construction constraints ensure the fundamental requirements of a route. Constraint (2) requires all meal orders to be completed. Constrain (3)-(5) indicates each driver should log on from the initial location, return to it after logging off the system, and the route should satisfy the flow conservation. Constraint (6) requires the pair of pick-up and drop-off tasks from one order should be fulfilled by the same driver. Constraint (7) guarantees the driver picks up the meal order first, and then delivers it to the customer.

The second category specifies the mixed fleet constraints. Constraint (8) and (9) ensure that the maximum number of utilized drivers adhere to the delivery fleet composition. Constraint (10) states that E-drivers' remaining travel range should cover the travel distance of the next assigned tasks and the predefined charging threshold $\delta * RL_k$. The charging threshold setting is to ensure that EVs have enough battery to reach an available charging station.

Constraint (11) and (12) are the capacity constraints, which states each driver delivered order number change along the route and limits the maximum orders that can be assigned at any given time.

TABLE I
VARIABLES DEFINITION AND DESCRIPTION

Horizon	
τ	Time interval length
t_h	System update and re-optimized time, $t_h = t_0 + h \times \tau, h \in (1, 2, \dots, \left\lfloor \frac{H}{\tau} \right\rfloor)$
indices	
i/j	Pick-up/ drop-off location
k/k_0	Driver k, k_0 represents the initial location of driver k
Sets	
R^h	Set of all meal order request. Each request consists of a pair of pick-up and drop-off tasks (i, j), where $R^h = O^n \cup O^s$ during time interval $[t_{h-1}, t_h)$
P^h	Set of all pick-up tasks from R^h
D^h	Set of all drop-off tasks from R^h
K^h	Set of delivery drivers, where $K^h = K^l \cup K^w$ during time interval $[t_{h-1}, t_h)$
F'	Set of charging stations
Parameters and constants	
m_E^h	Number of Electric vehicles
m_G^h	Number of internal combustion vehicles
RL_k	Range limit of an EV based on battery maximum capacity.
RL_k^h	Range limit of an EV at time t_{h-1}.
Q^k	Capacity of driver k
y^k	Binary vehicle type indicator. y^k=0, if driver k is an E-driver. Otherwise, y^k=1, driver k is a G-driver.
E_{ij}	Emissions of a vehicle travels from location i to location j
q_i	Order number to be served at location i. Positive when i is a pick-up location; negative when i is a drop-off location.
s_i	Service time of task i (load/unload)
t_p^i	The meal order place time, $i \in D$
t_r^i	The meal order ready time, $i \in P$
t_{do}^{ie}	The meal order expected drop-off time, $i \in D$
t_{ij}	Travel time of OD pair (i, j)
d_{ij}	Travel distance of OD pair (i, j)
CT^k	E-driver k charging time
δ	A ratio to define charging threshold
Intermediate variables	
t_{pu}^i	The actual visit time at the pick-up location i, $i \in P$
t_{do}^i	The actual visit time at the drop-off location i, $i \in D$
Q_i^k	The number of tasks when driver k leaves location i
T_i^k	Time when diver k arrives location i
Decision Variable	
x_{ij}^k	$x_{ij}^k = 1$ if OD pair (i, j) traveled by driver k ; otherwise, $x_{ij}^k = 0$.

The fourth group includes all the time constraints. Each pick-up task has a ready time t_r^i provided by the restaurant. Constraint (13) states that one driver can arrive earlier at the restaurant but should wait until the t_r^i to pick up the meal order. If the driver arrives later than t_r^i, then he/she can directly pick up the order. Constraint (14) states that if the previous visit location i is a restaurant, then the time when the driver arrives at the next consecutive location j is no earlier than the order pick-up time t_{pu}^i plus the sum of service time s_i and travel time from i to j. Constraint (15) implies that if the previous visit location i is a customer, then the time when the driver arrives at consecutive location j is no earlier than the order drop-off time t_{do}^i plus the sum of service time s_i and travel time from i to j. Constraint (16) covers the E-driver charging case, which indicates that the E-driver can only log on to the system again after charging time CT. The last group defines the decision variables. x_{ij}^k is a binary variable indicating if driver k travels from i to j. y^k is a binary variable as well. When $y^k = 0$, indicating driver k is an E-driver. Otherwise, $y^k = 1$, indicating driver k is a G-driver.

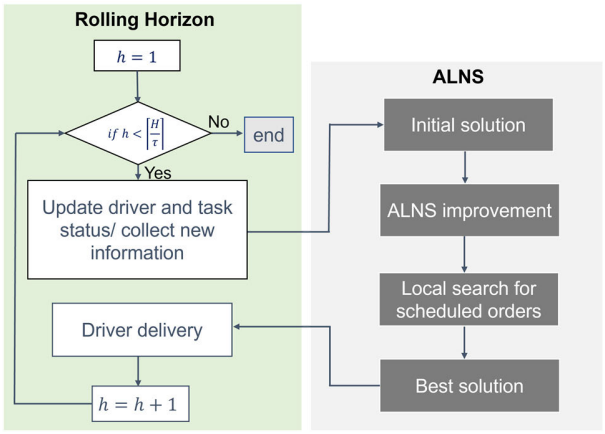

Fig. 4. RH-ALNS algorithm flowchart.

During the dynamic order dispatching and routing process, $|P^h| \leq |D^h|$ holds, as some meal orders might be picked up at the restaurants but haven't been delivered at the customer location before time t_h. The system input and model formulation will be updated at every re-optimization time t_h according to the new arrival orders and the status of previous tasks, which will be presented in the next section.

IV. PROPOSED RH-ALNS ALGORITHM

The meal delivery problem in an ODMD system can be considered as a dynamic pickup and delivery problem with time window problem which is famous for its NP-hardness. To accommodate the newly received order demand and the updated delivery fleet, we proposed a rolling horizon based adaptive large neighborhood search (RH-ALNS) algorithm to solve the order dispatching problem. The main components of RH-ALNS are shown in Figure 4. The RH component is designed to update the status of previous tasks and collect new information during the new time interval. A feasible initial solution is first generated in the new optimization scenario, then the ALNS algorithm is used to optimize the order dispatching and routing solution. Besides, to further intensify the searching space, we implemented a local search algorithm to reschedule the previous tasks. The delivery drivers will be notified with the updated task sequence based on the solution to execute pick-up/drop-off accordingly. The details of each component are introduced in the following sections.

A. Rolling Horizon Framework

The rolling horizon framework is a time-based periodic approach that divides the total operational horizon H into a number of time intervals of length τ. At the re-optimized time step t_h, from the task respective (since each order has a pick-up and a drop-off task), the system will first remove any completed pick-up/drop-off task i if $t_{pu}^i \leq t_h$ or $t_{do}^i \leq t_h$. Other

previous tasks are the scheduled tasks that will be tied with the assigned driver. Then the system will check new orders and generate new pick-up and drop-off tasks accordingly.

Meanwhile, the current location of each driver k should be updated. Previous active drivers who haven't been assigned any tasks and the new active drivers will be idle at the origin initial location k_0. For drivers that have departed to execute tasks, the current location is updated to the on-going task's location since we assume drivers to finish the on-going task before changing the routing. In the case of drivers who have completed a sequence of tasks, the driver's location is updated to the last visited location. The driver task sequence will be updated according to the task status, such as deleting the completed tasks. In particular, E-drivers should also update the remaining travel range according to battery life. E-drivers will log off the system either when their total working hours are consumed, or when the remaining travel range reaches the predefined threshold. G-drivers will log off the system according to the drivers' working shift.

B. Initial Solution

We need to insert the new orders to obtain a feasible initial solution. Algorithm 1 illustrates the proposed nearest driver with priority strategy to dispatch new orders in each time interval. New orders are sorted in ascending order according to the expected drop-off time t_{do}^{ie}(line 1). Then each order will be assigned to one active driver to construct an initial solution. The driver matching strategy is the nearest driver with priority where the nearest idle driver is searched (line 5). If there is no idle driver, we will start searching for the nearest working driver for the order (line 6). With this strategy, when the system has more drivers than orders, we intend to encourage a single bundle. This helps the system to warm up quickly and reduce the delay penalty as much as possible. After finding the nearest driver k, order o will be inserted into the best position that causes minimum objective value increase (line 9). If an idle driver has been assigned an order, then we should update it into the working driver set K^W (line 10).

Algorithm 1 Nearest Driver with Priority

Input: New orders O^n in time interval h, driver set $K^I \cup K^W$
Output: The initial assignment of each order to a specific driver
1: Priority Queue L ← Sort Order by expected drop-off time (O^n)
2: Pop out order sequentially
3: **for** each order $o \in O^n$ **do**
4: **if** $|K^I| > 0$:
5: find nearest driver k in K^I
6: **else:**
7: find nearest driver k in K^W
8: **end if**
9: find best position in driver k to insert order o
10: update driver k to K^W if k is chosen from K^i
11: **end for**

C. ALNS Improvement for Dynamic ODMD

The Adaptive Large Neighborhood Search (ALNS) algorithm was developed to further improve the initial solution. The advantage of ALNS compared to other heuristics is that it can diversify and intensify the initial solution to get the optimized solution by leveraging multiple removal and repair operators. It also has an adaptive scoring mechanism to explore large neighborhoods in a structured manner and a simulated annealing acceptance criterion to allow solution exploration, thus having the potentiality to escape the local minimum. In this research, to ensure the ODMD solution satisfies our assumptions in section II, we specified requirements for the ALNS removal and repair process.

First, transferring the previously assigned orders to another driver is not allowed, so in the ALNS removal process, only the new orders can be removed. Second, we only allow drivers to change route until finish the on-going task, either pick-up or drop-off. Thus, in the ALNS repair process, no tasks can be inserted earlier than the on-going task. Meanwhile, after inserting the new orders, an intra-route search after the on-going task will be executed between all orders assigned to the driver. With the above specific setup in the removal and repair process, then we present the ALNS algorithm for dynamic on-demand meal delivery as follows.

1) Removal Operators: We implemented five removal operators to perturb the solution and remove the undesired order assignments. These removal operators are: 1) random removal, 2) worst removal, 3) Shaw removal, 4) Distance-based path removal, and 5) Delay-based path removal. In our meal delivery problem, all removal operators only perform within the new orders.

Given a feasible solution, random removal selects N orders randomly to remove in order to perturb the solution space. Worst removal is to remove the order with the highest insertion cost. The cost is defined in equation (1). We rank the insertion cost of every order in increasing order and introduce a random number $y \in (0, 1)$ and a parameter p, then remove the order located at $y^p |N|$. This randomization is implemented to avoid removing the same task repeatedly. Shaw removal is based on the similarity of orders [11], we calculate the similarity in (21), where $d_{p(i),p(j)}$ is the distance between the restaurants of order i and order j, $d_{d(i),d(j)}$ is the distance between the two drop-off locations, $t_{pi} - t_{pj}$ is the time difference of the pick-up time, $t_{di} - t_{dj}$ is the time difference of drop-off time. A similar randomization trick is also introduced as in worst removal. For distance-based and delay-based path removal, we aim to remove the longest travel distance cost path and worst delay penalty cost path respectively. Since we only perform removal between new orders, before the path removal, we need to check if the path is constructed with only new orders.

$$R_{i,j} = f_1\left(d_{p(i),p(j)} + d_{d(i),d(j)}\right) + f_2\left(\left|t_{pi} - t_{pj}\right| + \left|t_{di} - t_{dj}\right|\right) \quad (21)$$

2) Repair Operators: After the removal, we need to insert back the orders to construct a new solution. In this paper, the parallel insertion heuristic is chosen where multiple routes are built simultaneously. We first sort the removed orders with

the target drop-off time then insert each order to construct a complete solution. Four repair operators are constructed: 1) random repair, 2) greedy repair, 3) regret-2 repair, 4) regret-3 repair. The repair operator should ensure that the insertion position cannot be earlier than the on-going task.

Random repair selects the inserting positions randomly to insert the removed orders. Greedy repair aims to find the best position such that the increase of the objective value is minimal. Regret-q repair leverages the look-ahead information to avoid the situation that the most "expensive" order is left at the last iteration where we lack the flexibility to deal with. Let Δc_i^j indicate the objective change when inserting order i into its j_{th} cheapest position. Then we need to find the order i that maximizes the regret value in (22). In each iteration, we select the most regretted order i and insert it into its best position. In this paper, we construct Regret-*2* and Regret-*3* insertion operators.

$$\max_i \left\{ \sum_{j=1}^{q} \left(\Delta c_i^j - \Delta c_i^1 \right) \right\} \tag{22}$$

3) Adaptive Scoring and Selection: Alternating between different operators could balance with diversification and intensification of the final solution and could deliver robust results overall. The Roulette Wheel Selection Principle is implemented to select the removal and repair operator independently in every iteration.

The total search approach of N iterations is divided into K segments. Each segment has $\lceil \frac{N}{K} \rceil$ iterations. At the beginning of a segment k, each operator is equally weighted and the accumulated score μ is initialized to zero. In each iteration, a score γ will be assigned to the selected operators based on the performance of the complete solution S^+. The adaptive score is shown in (23). The accumulative score μ_i of operator i is updated according to γ in (24), where π_i is the total number of times the operator i has been chosen in this segment.

$$\gamma = \begin{cases} \gamma^1 & \text{if a new best solution is obtained} \\ \gamma^2 & \text{if the solution is better than the current solution} \\ \gamma^3 & \text{if the solution is worse than the current one} \\ & \text{but accepted} \end{cases} \tag{23}$$

$$\mu_i = \sum\nolimits_{j=1}^{\pi_i} \gamma_j \tag{24}$$

Based on the score, the adaptive weight adjustment method (see (25)) is utilized to update the operator weight.

$$w_i^k = (1-\rho)\, w_i^{k-1} + \rho \frac{\mu_i}{\pi_i} \tag{25}$$

The weight of operator i at segment k (w_i^k) is based on the weight at segment $k-1 (w_i^{k-1})$ and the performance in the current segment. ρ is a reaction factor controlling how quickly the algorithm reacts to the effectiveness of the operators. If $\rho = 0$, then the weight is constant as at the last segment. If $\rho = 1$, then it only considers the current segment to decide the weight. Then given n operators, the probability of choosing operator i during the segment k is defined as:

$$P_i^k = \frac{w_i^k}{\sum_{j=1}^{n} w_j^k} \tag{26}$$

4) Acceptance and Termination Criteria: A simulated annealing strategy is used in the ALNS framework to avoid trapping in a local minimum. We accept a worsen solution $S^{'}$ with the probability of $e^{-\frac{f\left(S^{'}\right)-f(S)}{T}}$, where f is the objective function and T> 0 is the temperature. T decreases with a cooling rate δ : $T = \delta T$ $(0 < \delta < 1)$. Considering the practicality, we prefer good results in short time rather than getting the optimal solution in long computational time. The algorithm terminates under two conditions: 1) The maximum number of iterations φ_{max} is reached; 2) φ iterations have been executed without any improvements.

D. Local Search for Scheduled Orders

In the ALNS improvement process, we only focused on optimizing the new orders dispatching result since previously scheduled orders cannot be transferred to another driver. We implemented the intra-route local search to rearrange the scheduled tasks. Two operators: forward movement & back-forward movement [52], are performed within each route to intensify the solution.

V. Evaluation and Results

In this section, we presented the experimental studies and results. First, the performance of the proposed RH-ALNS algorithm is evaluated compared to two baseline algorithms. Secondly, a comprehensive analysis is conducted to investigate the difference of three delivery policies and highlight the importance of efficiently utilizing EVs in the mixed fleet. Thirdly, sensitivity analysis regarding the EV ratio in the mixed fleet is presented. All algorithms are coded in python 3.8 and run with ThinkPadX1 Carbon 2021 with 16GB of RAM and an Intel Core I7- 1165G7 processor.

A. Setup of Experiment

Instead of randomly generating meal data, we used the CEMDAP model [53] to generate the meal orders in the City of Riverside, California. Given various land-use, sociodemographic, activity, and transportation level-of-service attributes, CEMDAP provides comprehensive daily activity-trip patterns for individuals. Among all activities, we specifically focused on eat-out trips and extracted the meal delivery orders. In this research, we sampled 100, 300, 500, 700 meal orders out of 1523 eat-out trips during the lunchtime window from 11:30 am to 12:30 pm. These instances were denoted as C100, C300, C500, C700, accordingly. Each order contains information such as order place time, restaurant location, and customer location. To illustrate, Figure 5 depicts the meal orders in C500 instance between 12:00pm and 12:10 pm.

The traffic network in the city of Riverside was extracted from BEAM [54]. Once all locations were defined, including drivers' initial location, customer location and restaurant location based on the sampled meal orders and drivers starting

Fig. 5. Meal orders map in a 10-minute period. Green dots represent the customer location, yellow dots represent the restaurant location. The white line indicates meal order. The background is the Riverside network extracted from BEAM.

locations, we computed the distance matrix and travel time matrix among the locations based on the fastest path with the NetworkX module [55]. These matrices were then saved into a table for quick reference. The link-level emissions were calculated with the link-level distance and speed information. First, we integrated the EMFAC model [45] to obtain the emission rate ($ER_{i,j}$), which is contingent on the link speed. Subsequently, the link emission $E_{i,j}$ was obtained as $ER_{i,j} \times d_{i,j}$, where $d_{i,j}$ is the link distance. The total emissions along the route were computed by summing the emissions from all links traveled by drivers. The emission matrix containing GHG emissions from one location to another was pre-calculated to save the computational time. If an order is matched with an E-driver, we simply set the emission term to zero.

The parameter settings were summarized in Table II. In the ODMD system, each order has a place time t_p^i. The meal preparation time is randomly generated from 5 to 20 minutes, determining the order ready time: $t_r^i = t_p^i + T_{pre}$. The platform expects to deliver all meal orders within 40 minutes after the order place time, termed as expected click to door time T_{ctd}^e. Then the expected drop-off time t_{do}^{ie} is defined as $t_{do}^{ie} = t_p^i + T_{ctd}^e$. If the actual drop-off time t_{do}^i is later than t_{do}^{ie}, then a delay occurs. Additionally, each pick-up/drop-off task needs one minute of service time for miscellaneous tasks, i.e., walking to the apartment.

The crowdsourced drivers arrive at a ratio of 5 orders per driver within each time interval. To prevent driver shortages, the system has additional drivers who start working at time t_0. Considering the order volume, the number of additional drivers is set as 10, 20, 30, 30 for C100, C300 C500, C700 instance respectively. Among all drivers, we assume E-drivers account for 40% of delivery drivers and G-drivers make up the remaining 60%. The range limit of E-drivers is set as 400 km. When the remaining distance is less than 40 km, then E-drivers will log off the system to get charged. Further, we assumed each driver could be assigned with no more than 10 orders at any given time. The system update time window is set at 10 minutes for all instances.

TABLE II
MEAL ORDER PARAMETERS AND COST FACTORS

Variable	Value	unit
Food preparation time T_{pre}	(5,20)	min
Expected click to door time T_{ctd}^e	40	min
Service time s_i	1	min
Order/driver ratio	5	-
E-drivers ratio	40	%
E-drivers range limits RL	400	km
Remaining range threshold	0.1	-
Driver maximum order Q_k	10	order
System update time τ	10	min
Operational cost factor c_d	0.26	$/km
delay cost factor c_t	0.28	$/min
GHG cost factor c_e	50	$/metric ton

In the objective function, the operational cost is calculated based on the total delivery distance, with an assumed cost factor of 0.26 $/km. The delay cost factor is set as 0.28 $/min. The GHG emission cost factor is 50 $/metric ton, obtained from [56], representing the long-term social cost.

For parameter setting in ALNS algorithm, we mainly took reference from [57]. For the number of orders to be removed in each iteration, a random number p is generated from $4,0.2n$, where n is the total number of new orders in the current solution. The operator score $(\gamma^1, \gamma^2, \gamma^3)$ is set to (33,15,9). Each segment includes 50 iterations, and we executed 100 segments in total. The number of non-improvement iterations was set as 500.

B. Discission on Weight Value in the Objective Function

The objective function is a weighted sum of three components: emission cost, distance-based operational cost, and delay cost. Given the monetary cost with factors (c_d, c_t, c_e), the operational cost and delay cost would significantly outweigh the emission cost, leading to insufficient EV utilization in the delivery fleet. In order to achieve a dispatching plan with a better balance between operational performance and emission impact, we conducted preliminary tests to fine-tune the factors α, β and γ.

To find the factor γ that best represents the research goal of this paper, we conducted experiments in C100 instance to calibrate with the RH-ALNS algorithm by setting the weight factor α and β both equal to 1, then steadily increasing the weight factor γ. The preliminary results showed that a small γ led to the neglect of GHG emission costs within the delivery process. If we steadily increase γ from 1 to 10000, the system can use EV to replace GV without significantly increasing distance and delay cost. If γ keeps increasing, the system will assign higher priority to EVs to deliver orders while significantly sacrificing the distance and delay cost. With the preliminary test result, we thus set the weighted factor in the objective function (α, β, γ) equal to (1,1,10000) in the following numerical studies to balance both emission cost and operation cost. However, this is just a preliminary discussion regarding the weight parameter. One can calibrate own weight factors according to the specific study objectives in practice.

C. Benchmark Policies and Evaluation Metrics

With more EVs participating in the delivery process, it is important for ODMD platforms to efficiently utilize the mixed fleet. In this study, we considered three benchmark delivery policies.

B1: Eco-friendly delivery policy. The platform integrates the emission cost term in the objective function, aiming to maximize the EV's capability to reduce GHG emissions while upholding meal delivery quality.

B2: Cost-effective delivery policy. The platform treats all vehicles homogenously and focuses on minimizing operational costs and delay costs. In this case, EVs have no priority over other vehicles.

B3: Time-first delivery policy. This policy only minimizes the delay cost with the goal of delivering meal orders in the shortest time.

After obtaining the dispatching results based on the delivery policies, the performance and impacts of ODMD were evaluated from three perspectives, including monetary cost, delivery efficiency, and environmental impact.

Monetary cost: calculated with equation (27), consisting of three parts: operational cost, delay cost, and emission cost.

$$Total\ cost = c_d \times total\ distance + c_t \times total\ delay + c_e \times total\ emission \tag{27}$$

Delivery Efficiency: measured by Click to Door (CtD) time, which is the time difference between the meal order place time and actual drop-off time.

Environmental impact: evaluated with eVMT share and GHG emissions. The eVMT share is the travel distance contributed by E-drivers divided by total travel distance, providing a measure of electric vehicle utilization.

D. Performance of the Proposed RH-ALNS Algorithm

To validate the performance of RH-ALNS algorithm, we conducted the experiments with two baseline methods. The first baseline method, named as RH-IG, replace the ALNS component with the iterated greedy (IG) algorithm proposed in [52]. The IG method only uses a random removal operator to disturb the solution space and a greedy insertion operator to repair the solution. The second baseline method, termed as RH-ALNS-e, inspired by the idea of reducing running time in the searching process of ALNS, we implemented a driver forward slack time evaluation [58]. If the driver has no slack time to insert a new order, then the driver needs to finish all the tasks before receiving new assignment.

We solved the C100, C300, C500, and C700 instance with the system update time window of 10 minutes combining all three algorithms and three delivery policies. To fairly compare the algorithm performance and avoid the randomness, the C100 and C300 instances were repeated for 10 runs, and C500 and C700 instances were repeated for 5 runs. Total 270 experiments were conducted. Then the best result for each scenario was reported and summarized in Table III. The objective cost was calculated with the objective function according to the three delivery polices. The gap presents the deviation from our proposed algorithm, calculated with Gap = (ALNS_obj – V_obj)/V_obj), where ALNS_obj is the objective cost from RH-ALNS, V_obj is the objective value from the two baseline methods. A negative gap indicates our proposed algorithm is better than the baseline method. The run time is the average running time per time interval.

The results show the superiority of RH-ALNS in obtaining high-quality solutions from medium to large-scale instances. Under the B1 (Eco-friendly) and B3 (Time-first) delivery policies, the RH-ALNS-e is significantly worse than RH-ALNS, although it can slightly reduce the running time. This is because the slack time evaluation will reduce the driver selection pool and prevent the potential better neighborhood solutions. Under the B2 (cost-effective) policy, the solution quality from RH-ALNS-e quality is comparable to our method, only 3%-5% worse, with the running time reduced by 100 seconds in the C500 and C700 cases. Thus, RH-ALNS-e is preferred in large-scale cases under the cost-effective delivery policy.

The IG component may get stuck in the local minima, terminate prematurely and yield solutions even worse than that from RH-ALNS-e.

Meanwhile, in the C100 instance, RH-ALNS takes approximately 10 seconds longer than the other two methods but achieves superior solution improvements. In practical applications, deploying a clustering step first to divide total orders into clusters of approximately 100 orders each and then applying RH-ALNS simultaneously within each cluster can generate optimized solutions within 30 seconds. This strategic approach ensures both efficiency and solution quality in real-life scenarios.

E. Analysis of Different Delivery Policy

In this section, we compared the delivery efficiency of each scenario with the optimized results from RH-ALNS. Figure 6 presents four subgraphs: subgraph (a) reports the total monetary cost, while subgraphs (b)-(d) detail the three cost components. We observed that the total cost with policy B1 is slightly higher than that with B2, as shown in subgraph (a). This difference can be attributed to the fact that B1, which prioritizes eco-friendly delivery methods, tends to rely on E-drivers more frequently than B2. In this case, the system bundles multiple orders for E-drivers, causing detours and subsequently higher delay costs. However, this policy results in lower emission costs due to the use of E-drivers.

Figure 7 provides the analysis of delivery efficiency by presenting the CtD distribution of all four instances across the three different delivery policies. Results indicated that even with the implementation of an eco-friendly delivery policy, meal delivery efficiency is still well maintained. Among all four instances, 75% of meal orders are delivered on time, and the maximum CtD time does not exceed 50 minutes, which is considered satisfactory in real-world practice. These findings demonstrated that eco-friendly delivery policies can maintain delivery efficiency without sacrificing environmental benefits, making them a viable and promising option for ODMD systems.

TABLE III
COMPARISON OF OBJECTIVE COST AND COMPUTATION TIME OF SOLUTIONS OBTAINED BY THREE ALGORITHMS

	Policy	B1: Eco-friendly			B2: Cost-effective			B3: Time-first		
Instance	Algorithm	Obj cost (×1000)	Obj Gap (%)	Run time(s)	Obj cost	Obj Gap	Run time(s)	Obj cost	Obj Gap	Run time(s)
C100	RH-IG	41.25	-21.56%	**17.53**	269.01	-5.26%	**17.55**	18.71	-100%	**13.59**
	RH-ALNS-e	51.04	-36.64%	21.09	270.29	-5.71%	29.41	**0.00**	**0.00%**	15.71
	RH-ALNS	**32.33**	**0.00%**	28.55	**254.86**	**0.00%**	27.96	**0.00**	**0.00%**	25.48
C300	RH-IG	138.98	-63.38%	**59.29**	1053.51	-43.02%	**61.15**	483.45	-99.99%	**59.83**
	RH-ALNS-e	96.76	-47.40%	85.59	623.30	-3.68%	117.41	0.63	-94.26%	94.71
	RH-ALNS	**50.90**	**0.00%**	114.53	**600.29**	**0.00%**	126.32	**0.036**	**0.00%**	79.97
C500	RH-IG	243.25	-61.70%	**125.61**	3112.61	-68.35%	**143.53**	1507.93	-98.87%	**145.47**
	RH-ALNS-e	155.49	-40.09%	300.48	1026.58	-4.03%	329.10	42.33	-59.78%	300.49
	RH-ALNS	**93.16**	**0.00%**	397.38	**985.22**	**0.00%**	419.95	**17.02**	**0.00%**	466.70
C700	RH-IG	345.65	-48.83%	**248.17**	5287.81	-74.69%	**279.06**	3300.32	-97.44%	**278.25**
	RH-ALNS-e	250.82	-29.47%	716.15	1399.85	-4.37%	798.04	517.85	-83.71%	485.19
	RH-ALNS	**176.88**	**0.00%**	933.80	**1338.63**	**0.00%**	943.36	**84.37**	**0.00%**	757.41

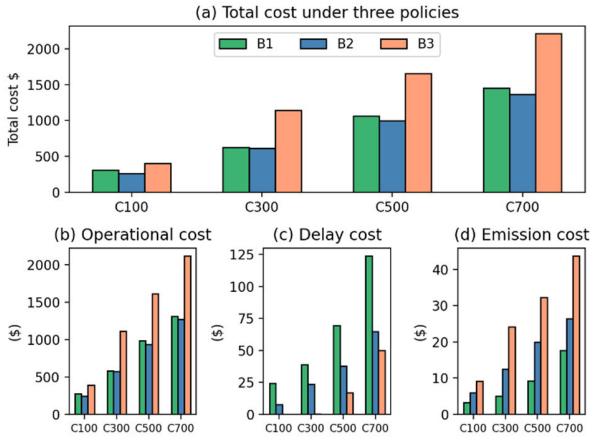

Fig. 6. Total cost under three delivery policies. B1 (eco-friendly delivery), B2 (cost-effective delivery), B3 (time-first delivery).

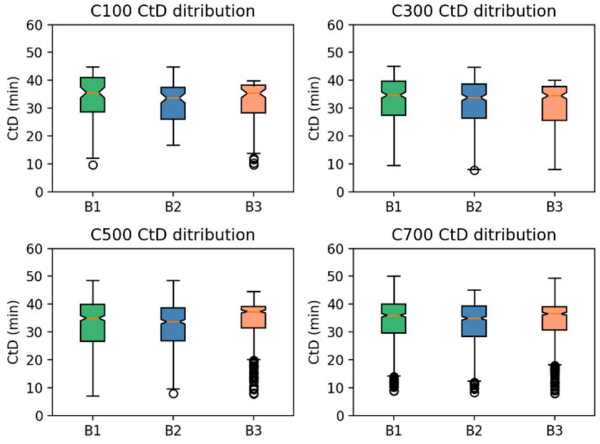

Fig. 7. Click-to-door time (CtD) distribution of all instances under three delivery policies. B1 (eco-friendly delivery), B2 (cost-effective delivery), B3 (time-first delivery).

TABLE IV
eVMT SHARE UNDER DIFFERENT EV RAIO WITH TWO DELIVERY POLICIES

	C100		C300	
EV ratio	B1	B2	B1	B2
0.2	22.02%	20.20%	46.45%	24.44%
0.4	71.07%	39.97%	72.88%	42.30%
0.6	76.22%	63.04%	87.73%	61.61%
0.8	85.15%	82.87%	98.56%	83.25%
	C500		C700	
EV ratio	B1	B2	B1	B2
0.2	38.19%	19.26%	35.88%	18.92%
0.4	74.71%	48.93%	63.63%	42.43%
0.6	94.94%	64.31%	83.25%	68.09%
0.8	99.97%	77.41%	96.08%	83.44%

In Figure 8, we presented the environmental impact of different delivery policies by showing the eVMT share and the GHG emission resulting from all three policies. If we treat E-drivers the same as G-drivers, disregarding their ability to reduce emissions, their eVMT share would be around 40% as expected, aligning with their proportion in the delivery fleet. However, with the eco-friendly policy, E-drivers contribute to a significantly higher eVMT share due to their zero GHG emissions during delivery. With the B1 policy, the eVMT share ranges from 65% to 78% across all instances, making a substantial increase compared to the approximate 40% eVMT share observed with policies B2 and B3, which focus more on cost and delay minimization.

To quantify the GHG emissions savings achieved with policy B1, we compare it to the GHG emissions change with other policies. The right bar of each column in Figure 7 shows that B1 results in substantial GHG emission savings compared to B2 and B3. By actively utilizing E-drivers, we can achieve at least a 30% reduction in GHG emissions. These results emphasize the potential of eco-friendly delivery policies in

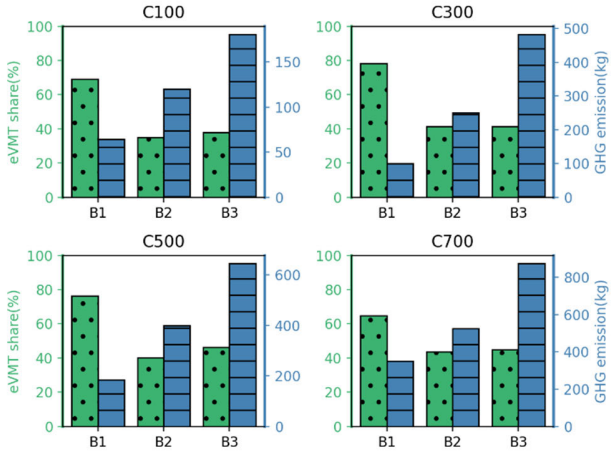

Fig. 8. eVMT share and GHG emissions of B1 (eco-friendly delivery), B2 (cost-effective delivery), and B3 (time-first delivery).

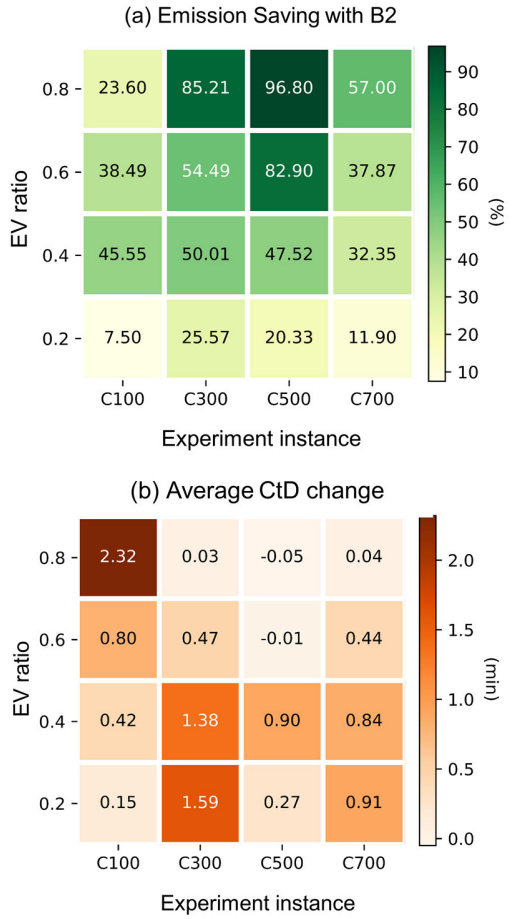

Fig. 9. (a)GHG emission saving and (b)average CtD change compared B1 with B2 in multiple cases.

reducing the environmental impact of ODMD systems while maintaining delivery efficiency.

In summary, eco-friendly delivery policy is promising in saving large extent of GHG emission while main high service quality of meal delivery.

F. Sensitivity Analysis

In this section, we performed a sensitivity analysis regarding the EV ratio in the delivery fleet. From the results from *Section F*, we observed that B1 and B2 are two similar strategies in terms of total cost, but B1 is more eco-friendly. In this part, to further investigate the eco-performance gap between the two policies, we varied the EV ratio including 0.2, 0.4, 0.6 and 0.8. Each instance was run with both policy B1 and policy B2,

Table IV summarizes the eVMT share of each specific case. With the B1 policy, the eVMT share exceeded the EV ratio in the delivery fleet, indicating the preference for E-drivers to deliver meal orders. Particularly, in the C300 and C500 instances with 80% of E-drivers in the fleet, the eVMT share increases to 98.56% and 99.97% respectively, indicating a near-exclusive use of E-drivers for meal deliveries. While with B2 policy, the eVMT share is roughly around the corresponding EV ratio.

Figure 9 depicts the GHG emission savings achieved by B1 compared to B2, as well as the average CtD time deviation between the two policies. When the EV ratio is 0.2, the emission gap ranges from 7.5% to 25%. This is because G-drivers dominate the delivery fleet, and the system only has a small portion of E-drivers to operate. The C300 instance produced the highest emission savings. As the EV ratio increases, the emission saving achieved by deploying B1 increases dramatically. In the C500 instance, with an EV ratio of 0.8, GHG emissions were reduced by 96.80% with the B1 policy compared to the B2 policy. This significant reduction is made possible by utilizing E-drivers to deliver meal orders in an eco-friendly delivery policy.

Additionally, we assessed the average CtD time deviation between the two policies. The B1 policy only resulted in a negligible increase in delivery time, and in most cases, the CtD time deviation was within 1 minute. In the C500 instance with an EV ratio of 0.6 and 0.8, the B1 policy even outperformed the B2 policy in terms of delivery efficiency. These results demonstrate that the eco-friendly policy not only leads to substantial GHG emission savings but also maintains high delivery efficiency, even with a higher proportion of E-drivers in the delivery fleet.

VI. Conclusion

In this paper, we studied a novel dynamic on-demand meal delivery problem with a mixed fleet of electric and gasoline vehicles. The objective is to minimize the operational cost, GHG emission cost and delay cost. The RH-ALNS algorithm, which consists of a rolling horizon framework and the ALNS algorithm, is proposed to efficiently deal with the meal order dynamism and large-scale order dispatching and routing problem. The numerical experiments demonstrate that the proposed algorithm performs better than two baseline methods in finding high-quality solutions. Additionally, a case study is conducted to investigate three delivery policies, including eco-friendly delivery (B1), cost-effective delivery (B2), and time-first delivery (B3), for medium to large-scale meal orders. The study results show that utilizing electric vehicles in the delivery fleet through the eco-friendly delivery policy can improve the sustainability of the delivery system. Experiment results demonstrate the necessity for online platforms to actively utilize EVs to delivery meal orders and take initiative to employ an eco-friendly delivery policy.

There are several promising directions for future research. Multi-objectives ODMD can be investigated to explore the trade-offs between delivery efficiency, environmental impacts, and various other interests in a more general way. Additionally, considering the computational efficiency, clustering techniques could be explored to achieve better solutions in a short time. Another interesting direction is to integrate charging scheduling considering charging cost and charging station recommendations on the ODMD platform to provide support for E-drivers during the delivery hours.

ACKNOWLEDGMENT

The contents of this article reflect only the views of the authors, who are responsible for the facts and the accuracy of the data presented.

REFERENCES

[1] A. Seghezzi, M. Winkenbach, and R. Mangiaracina, "On-demand food delivery: A systematic literature review," *Int. J. Logistics Manag.*, vol. 32, no. 4, pp. 1334–1355, Jan. 2021.

[2] Statista. (2013). *Proportion of Food Service Sales in the United States Which are Made Via Delivery From 2013 to 2022*. Accessed: Jul. 30, 2022. [Online]. Available: https://www.statista.com/statistics/1091419/food-service-sales-delivery-share-us/

[3] Statista. (2022). *Online Food Delivery—United States*. Accessed: Oct. 9, 2022. [Online]. Available: https://www.statista.com/outlook/dmo/eservices/online-food-delivery/united-states

[4] E. Pourrahmani and M. Jaller, "Crowdshipping in last mile deliveries: Operational challenges and research opportunities," *Socio-Economic Planning Sci.*, vol. 78, Dec. 2021, Art. no. 101063.

[5] (2022). *DoorDash 2022 ESG Update*. Accessed: Oct. 17, 2023. [Online]. Available: https://ir.doordash.com/governance/ESG-Resources/default.aspx

[6] Uber. (2023). *Together on the Road to Zero Emissions*. Accessed: Oct. 24, 2023. [Online]. Available: https://www.uber.com/us/en/drive/services/electric/

[7] (2022). *Clean Miles Standard*. Accessed: Sep. 6, 2022. [Online]. Available: https://ww2.arb.ca.gov/our-work/programs/clean-miles-standard/about

[8] (2022). *Electric Vehicles on the Lyft Platform*. Accessed: Sep. 6, 2022. [Online]. Available: https://www.lyft.com/impact/electric

[9] (2023). *Millions of Trips a Day, Zero Emissions and a Shift To Sustainable Packaging*. Accessed: Sep. 23, 2023. [Online]. Available: https://www.uber.com/us/en/about/sustainability/?uclick_id=d69e0be6-b9d7-46db-bd4f-d00fc8dc8825

[10] S. Ropke and J.-F. Cordeau, "Branch and cut and price for the pickup and delivery problem with time windows," *Transp. Sci.*, vol. 43, no. 3, pp. 267–286, Aug. 2009.

[11] S. N. Parragh, K. F. Doerner, and R. F. Hartl, "A survey on pickup and delivery problems," *J. Für Betriebswirtschaft*, vol. 58, no. 2, pp. 81–117, Jun. 2008.

[12] H. N. Psaraftis, M. Wen, and C. A. Kontovas, "Dynamic vehicle routing problems: Three decades and counting," *Networks*, vol. 67, no. 1, pp. 3–31, Jan. 2016.

[13] M. Schyns, "An ant colony system for responsive dynamic vehicle routing," *Eur. J. Oper. Res.*, vol. 245, no. 3, pp. 704–718, Sep. 2015.

[14] M. S. Chang, S. R. Chen, and C. Hsueh, "Real-time vehicle routing problem with time windows and simultaneous delivery/pickup demands," *J. Eastern Asia Soc. Transp. Stud.*, vol. 5, no. 30, pp. 2273–2286, 2003.

[15] F. Karami, W. Vancroonenburg, and G. Vanden Berghe, "A periodic optimization approach to dynamic pickup and delivery problems with time windows," *J. Scheduling*, vol. 23, no. 6, pp. 711–731, Dec. 2020.

[16] S. Mitrović-Minić, R. Krishnamurti, and G. Laporte, "Double-horizon based heuristics for the dynamic pickup and delivery problem with time windows," *Transp. Res. B, Methodol.*, vol. 38, no. 8, pp. 669–685, Sep. 2004.

[17] Y. Jia, C. Wang, and L. Wang, "A rolling horizon procedure for dynamic pickup and delivery problem with time windows," in *Proc. IEEE Int. Conf. Autom. Logistics*, Aug. 2009, pp. 2087–2091.

[18] D. Goeke and M. Schneider, "Routing a mixed fleet of electric and conventional vehicles," *Eur. J. Oper. Res.*, vol. 245, no. 1, pp. 81–99, Aug. 2015.

[19] G. Hiermann, R. F. Hartl, J. Puchinger, and T. Vidal, "Routing a mix of conventional, plug-in hybrid, and electric vehicles," *Eur. J. Oper. Res.*, vol. 272, no. 1, pp. 235–248, 2019.

[20] M. A. Masmoudi, M. Hosny, E. Demir, and E. Pesch, "Hybrid adaptive large neighborhood search algorithm for the mixed fleet heterogeneous dial-a-ride problem," *J. Heuristics*, vol. 26, no. 1, pp. 83–118, Feb. 2020.

[21] P. Zhao, F. Liu, Y. Guo, X. Duan, and Y. Zhang, "Bi-objective optimization for vehicle routing problems with a mixed fleet of conventional and electric vehicles and soft time windows," *J. Adv. Transp.*, vol. 2021, pp. 1–11, Sep. 2021, doi: 10.1155/2021/9086229.

[22] X. Ren, H. Huang, S. Feng, and G. Liang, "An improved variable neighborhood search for bi-objective mixed-energy fleet vehicle routing problem," *J. Cleaner Prod.*, vol. 275, Dec. 2020, Art. no. 124155.

[23] A. Amiri, S. H. Amin, and H. Zolfagharinia, "A bi-objective green vehicle routing problem with a mixed fleet of conventional and electric trucks: Considering charging power and density of stations," *Exp. Syst. Appl.*, vol. 213, Mar. 2023, Art. no. 119228.

[24] N. Mouhrim, A. El Hilali Alaoui, and J. Boukachour, "Vehicle routing problem with mixed fleet of electric and conventional vehicles under emissions allowances," in *Proc. 4th Int. Conf. Logistics Operations Manag. (GOL)*, Apr. 2018, pp. 1–5.

[25] V. F. Yu, P. Jodiawan, and A. Gunawan, "An adaptive large neighborhood search for the green mixed fleet vehicle routing problem with realistic energy consumption and partial recharges," *Appl. Soft Comput.*, vol. 105, Jul. 2021, Art. no. 107251.

[26] O. Sassi, W. R. Cherif, and A. Oulamara. (2014). *Vehicle Routing Problem With Mixed Fleet of Conventional and Heterogenous Electric Vehicles and Time Dependent Charging Costs*. Accessed: Oct. 24, 2023. [Online]. Available: https://hal.science/hal-01083966

[27] Y. Chen, D. Li, Z. Zhang, M. I. M. Wahab, and Y. Jiang, "Solving the battery swap station location-routing problem with a mixed fleet of electric and conventional vehicles using a heuristic branch-and-price algorithm with an adaptive selection scheme," *Exp. Syst. Appl.*, vol. 186, Dec. 2021, Art. no. 115683.

[28] R. Moghdani, K. Salimifard, E. Demir, and A. Benyettou, "The green vehicle routing problem: A systematic literature review," *J. Cleaner Prod.*, vol. 279, Jan. 2021, Art. no. 123691.

[29] I. Kara, B. Y. Kara, and M. K. Yetis, "Energy minimizing vehicle routing problem," in *Combinatorial Optimization and Applications*. Berlin, Germany: Springer, 2007, pp. 62–71.

[30] Y. Xiao, Y. Zhang, I. Kaku, R. Kang, and X. Pan, "Electric vehicle routing problem: A systematic review and a new comprehensive model with nonlinear energy recharging and consumption," *Renew. Sustain. Energy Rev.*, vol. 151, Nov. 2021, Art. no. 111567.

[31] T. Bektaş and G. Laporte, "The pollution-routing problem," *Transp. Res. B, Methodol.*, vol. 45, no. 8, pp. 1232–1250, Sep. 2011.

[32] B. Sawik, J. Faulin, and E. Pérez-Bernabeu, "Multi-criteria optimization for fleet size with environmental aspects," *Transp. Res. Proc.*, vol. 27, pp. 61–68, Jan. 2017.

[33] M. Barth and K. Boriboonsomsin, "Energy and emissions impacts of a freeway-based dynamic eco-driving system," *Transp. Res. D, Transp. Environ.*, vol. 14, no. 6, pp. 400–410, Aug. 2009.

[34] EMFAC. (2021). *Emission Inventory*. Accessed: Jun. 7, 2022. [Online]. Available: https://arb.ca.gov/emfac/emissions-inventory/

[35] Y. Liu et al., "FooDNet: Toward an optimized food delivery network based on spatial crowdsourcing," *IEEE Trans. Mobile Comput.*, vol. 18, no. 6, pp. 1288–1301, Jun. 2019.

[36] W. Tu, T. Zhao, B. Zhou, J. Jiang, J. Xia, and Q. Li, "OCD: Online crowdsourced delivery for on-demand food," *IEEE Internet Things J.*, vol. 7, no. 8, pp. 6842–6854, Aug. 2020.

[37] X. Wang, L. Wang, S. Wang, J.-F. Chen, and C. Wu, "An XGBoost-enhanced fast constructive algorithm for food delivery route planning problem," *Comput. Ind. Eng.*, vol. 152, Feb. 2021, Art. no. 107029.

[38] B. Yildiz and M. Savelsbergh, "Provably high-quality solutions for the meal delivery routing problem," *Transp. Sci.*, vol. 53, no. 5, pp. 1372–1388, Sep. 2019.

[39] W. Liao, L. Zhang, and Z. Wei, "Multi-objective green meal delivery routing problem based on a two-stage solution strategy," *J. Cleaner Prod.*, vol. 258, Jun. 2020, Art. no. 120627.

[40] D. Reyes et al., "The meal delivery routing problem," *Optim. Online*, p. 6571, 2018. [Online]. Available: https://optimization-online.org/wp-content/uploads/2018/04/6571.pdf

[41] Q. Zhou et al., "Two fast heuristics for online order dispatching," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jul. 2020, pp. 1–8.

[42] H. Huang, C. Hu, J. Zhu, M. Wu, and R. Malekian, "Stochastic task scheduling in UAV-based intelligent on-demand meal delivery system," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 8, pp. 13040–13054, Aug. 2022.

[43] Z. Steever, M. Karwan, and C. Murray, "Dynamic courier routing for a food delivery service," *Comput. Oper. Res.*, vol. 107, pp. 173–188, Jul. 2019.

[44] S. Paul, S. Rathee, J. Matthew, and K. M. Adusumilli, "An optimization framework for on-demand meal delivery system," in *Proc. IEEE Int. Conf. Ind. Eng. Eng. Manag. (IEEM)*, Dec. 2020, pp. 822–826, doi: 10.1109/IEEM45057.2020.9309922.

[45] M. D. Simoni and M. Winkenbach, "Crowdsourced on-demand food delivery: An order batching and assignment algorithm," *Transp. Res. C, Emerg. Technol.*, vol. 149, Apr. 2023, Art. no. 104055.

[46] J.-F. Chen et al., "An imitation learning-enhanced iterated matching algorithm for on-demand food delivery," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 10, pp. 18603–18619, Oct. 2022.

[47] H. Liu, P. Hao, Y. Liao, K. Boriboonsomsin, and M. Barth, "Model-based vehicle-miles traveled and emission evaluation of on-demand food delivery considering the impact of COVID-19 pandemic," *Transp. Res. Rec., J. Transp. Res. Board*, 2023, Art. no. 036119812311692. [Online]. Available: https://journals.sagepub.com/doi/full/10.1177/03611981231169276

[48] J. Allen et al., "Understanding the transport and CO_2 impacts of on-demand meal deliveries: A London case study," *Cities*, vol. 108, Jan. 2021, Art. no. 102973, doi: 10.1016/j.cities.2020.102973.

[49] Y. Liu, "An optimization-driven dynamic vehicle routing algorithm for on-demand meal delivery using drones," *Comput. Oper. Res.*, vol. 111, pp. 1–20, Nov. 2019.

[50] K. Deb, "Multi-objective optimisation using evolutionary algorithms: An introduction," in *Multi-objective Evolutionary Optimisation for Product Design and Manufacturing*, L. Wang, A. H. C. Ng, and K. Deb, Eds. London, U.K.: Springer, 2011, pp. 3–34.

[51] T. Seo and Y. Asakura, "Multi-objective linear optimization problem for strategic planning of shared autonomous vehicle operation and infrastructure design," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 4, pp. 3816–3828, Apr. 2022.

[52] X. Wang et al., "An effective iterated greedy algorithm for online route planning problem," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jul. 2020, pp. 1–8.

[53] C. R. Bhat, J. Y. Guo, S. Srinivasan, and A. Sivakumar, "Comprehensive econometric microsimulator for daily activity-travel patterns," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 1894, no. 1, pp. 57–66, Jan. 2004.

[54] BEAM. (2020). *The Modeling Framework for Behavior, Energy, Autonomy, and Mobility*. Accessed: Jul. 1, 2022. [Online]. Available: https://transportation.lbl.gov/beam

[55] (2023). *NetworkX—NetworkX Documentation*. Accessed: Feb. 14, 2023. [Online]. Available: https://networkx.org/

[56] US Epa, OAR, OAP, and CCD. (2016). *The Social Cost of Carbon*. Accessed: Sep. 6, 2022. [Online]. Available: https://19january2017snapshot.epa.gov/climatechange/social-cost-carbon_.html

[57] S. Ropke and D. Pisinger, "An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows," *Transp. Sci.*, vol. 40, no. 4, pp. 455–472, Nov. 2006.

[58] J.-F. Cordeau and G. Laporte, "A Tabu search heuristic for the static multi-vehicle dial-a-ride problem," *Transp. Res. B, Methodol.*, vol. 37, no. 6, pp. 579–594, Jul. 2003.

Haishan Liu (Graduate Student Member, IEEE) received the bachelor's degree from Nanjing University of Aeronautics and Astronautics, China, in 2018, and the master's degree from Tongji University, China, in 2021. She is currently pursuing the Ph.D. degree in electrical engineering with the University of California at Riverside. Her current research interests include shared mobility, sustainable logistics, and large-scale vehicle routing problems.

Peng Hao (Member, IEEE) received the B.S. degree in civil engineering from Tsinghua University in 2008 and the Ph.D. degree in transportation engineering from Rensselaer Polytechnic Institute in 2013. Following graduation, he was a Post-Doctoral Scholar with CE-CERT. He is an Assistant Research Engineer with the Center for Environmental Research and Technology, University of California at Riverside. His research interests include connected vehicles, eco-approach and departure, sensor-aided modeling, signal control, and traffic operations.

Yejia Liao (Graduate Student Member, IEEE) received the B.S. degree in mathematics from Nankai University in 2018 and the M.S. degree in statistics from the University of Wisconsin, Madison. He is currently pursuing the Ph.D. degree with the Center for Environmental Research and Technology, University of California at Riverside. His research interests include intelligent transportation systems, transportation/emissions modeling, and truck routing strategies. His current research focuses on developing novel routing strategies based on the truck pollutant inhaled by communities and evaluating its performance using agent-based simulation at macroscopic levels.

Shams Tanvir (Member, IEEE) is an Assistant Professor with California State University Long Beach (CSULB). At CSULB, he directs the Sustainable Mobility Laboratory (SuMoLab). He served as a principal investigator in projects sponsored by agencies and industry totaling nearly $1M in research expenditures. He has published more than 50 peer-reviewed journals, book chapters, and conference papers. His research aims at the development and characterization of transportation technologies. He is a member of the National Academies TRB Committees on Highway Capacity and Quality of Service, and Transportation Air Quality and Greenhouse Gas Mitigation. He chairs the Sustainable Transportation Committee, American Society of Civil Engineers (ASCE).

Kanok Boriboonsomsin (Member, IEEE) is a Research Engineer (Research Faculty) with the College of Engineering-Center for Environmental Research and Technology (CE-CERT). With a solid background in transportation engineering, his research focuses on transportation planning, vehicle activity analysis, vehicle emissions modeling, traffic simulation, and sustainable freight transportation. He has served as the PI or Co-PI for a number of research projects funded by various transportation and air agencies. He is a current member of the Transportation Research Board's Transportation and Air Quality Committee and the Vice Chair of the Vehicle Activity and Technology Subcommittee.

Matthew J. Barth (Fellow, IEEE) received the M.S. and Ph.D. degrees in electrical and computer engineering from the University of California at Santa Barbara, in 1985 and 1990, respectively. He is currently the Yeager Families Professor with the College of Engineering, University of California at Riverside, USA. He is also the Director of the Center for Environmental Research and Technology. His current research interests include ITS and the environment, transportation/emissions modeling, vehicle activity analysis, advanced navigation techniques, electric vehicle technology, and advanced sensing and control. He has been active in the IEEE Intelligent Transportation System Society for many years. He served as the IEEE ITSS President in 2014 and 2015 and is currently the IEEE ITSS Vice President of Education. He is serving as a Senior Editor for IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS and IEEE TRANSACTIONS ON INTELLIGENT VEHICLES.