Dynamic Courier Routing for a Food Delivery Service

Abstract: Services like Grubhub and UberEats have revolutionized the way that diners can find and order from restaurants. The standard business model for such services, however, allows diners to order from only one restaurant at a time. Inspired by a food delivery service in the southeastern United States, this paper proposes the framework for a more flexible business model in which multiple restaurants may be included in a single customer's order. We formally define this new problem, the virtual food court delivery problem (VFCDP), and provide a mixed integer linear programming formulation. For implementation in a dynamic setting, an auction-based heuristic has also been developed. This so-called "proactive" heuristic anticipates future system states, and seeks solutions which are effective at both serving customers in the present and preparing couriers to handle future demand. This is facilitated through the calculation of metrics describing equity and dispersion. Furthermore, this heuristic is capable of handling both split and non-split delivery policies. An extensive numerical study is conducted in a simulation environment to examine characteristics of this new business model. This study reveals that the proactive heuristic is effective at improving system performance (over an entirely myopic heuristic) according to several customer experience-based metrics (e.g. freshness, earliness, etc.). Furthermore, a non-split delivery policy is shown to deliver all of a customer's items no later than the last item would have arrived in the split-delivery case, on average. It does this while also avoiding any waiting time for the customer between deliveries, and while reducing the number of miles traveled by a courier fleet throughout the day. Additional managerial policies, such as the type of delivery window offered to customers, are also discussed.

Keywords: Dynamic vehicle routing; Pickups and deliveries; Non-split deliveries; Split deliveries; Time windows; Heuristic algorithms

1 Introduction

Food delivery services, in which a courier delivers items to a customer from one of multiple partnering restaurants, have become increasingly popular. Nationwide examples include Grubhub and UberEats, although numerous other regional and local services also exist. In such a system, customers place orders with the food delivery service, choosing items from participating restaurants. The delivery service communicates the orders to the appropriate restaurants. Each restaurant then provides the delivery service with an expected ready time. When the customer's order is available, a courier with the delivery service picks up the food order from the appropriate restaurant and delivers it to the customer. There may be multiple modes of transportation available to couriers employed by the delivery service, including driving, biking, or walking. Each courier is available for service during a pre-determined working shift.

When an order is placed, the customer is given an expected delivery time. Some delivery services advertise general delivery durations (e.g., 30-minutes or less), while others provide a guaranteed delivery time (e.g., your order will be delivered by 6:30 p.m). As new orders are placed, the delivery service is faced with the problem of routing (or re-routing) couriers. The underlying problem is to assign couriers to sequences of pickups and deliveries that maximize customer satisfaction (measures of which are explored) without violating committed delivery times. We refer to this problem as the *virtual food court delivery problem* (VFCDP). This paper further contributes a mathematical model and a heuristic which are capable of handling both split delivery (i.e. items ordered by one customer from different restaurants can be picked up and delivered by separate couriers) and non-split delivery (i.e. all of a customer's items must be picked up by one courier and delivered simultaneously) policies for such a system. Novel to this heuristic is the inclusion of two future-looking metrics, referred to as equity and dispersion, which measure a system's preparedness for unknown future demand. More generally, this paper contributes a thorough analysis of various managerial policies, providing guidance to those who operate a food delivery service.

There are multiple sources of dynamism in the VFCDP. For example, customer orders are not known in advance; they are revealed over time. Traffic conditions can affect travel times, and car-

based couriers in particular may spend extra time looking for a parking space. Although restaurants provide an estimated time that an order will be ready, these estimates may be inaccurate. Additionally, depending on staffing levels at the restaurant, the time required by the courier to take possession of the food order may vary.

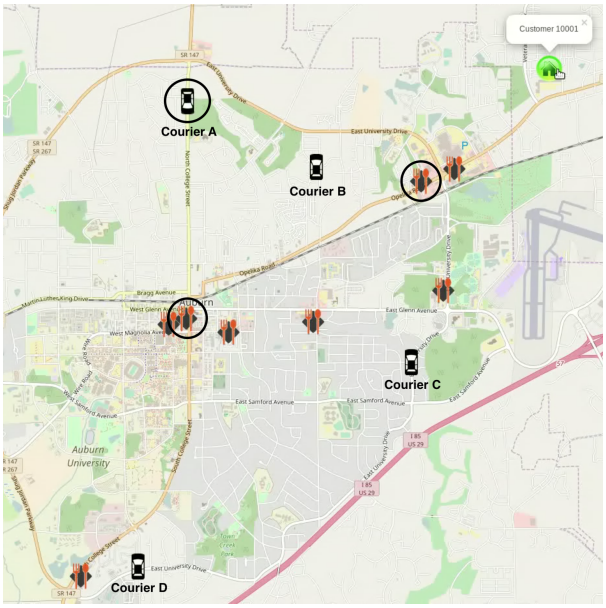
It should be noted that the VFCDP as a problem is dynamic-stochastic. The proposed integer programming formulation for the VFCDP, however, is static-deterministic. For example, it is assumed that dispatchers are only aware of mean values for truly stochastic values (e.g. the mean food preparation time at each restaurant in the network), and are aware of only information which has been revealed up to their decision time. The heuristic which was developed for the VFCDP iteratively reoptimizes a sequence of sub-problems (containing some subset of the known customer information) using our proposed integer programming model. Our integer programming formulation, then, is not so much a model of the VFCDP problem itself, but rather a piece of the tool which we have developed to solve that problem. This solution approach is well established in the dynamic vehicle routing literature (c.f. Psaraftis (1980), Savelsbergh and Sol (1998), Montemanni et al. (2005), Chen and Xu (2006)). However, stochasticity is reintroduced in the heuristic through the use of two future-looking metrics which are computed rapidly, and which factor into assignment decisions. These metrics help to select routes which may be more responsive to future customer orders. Numerical testing reveals that such a future-looking approach outperforms a version of the heuristic that ignores this probabilistic information.

To help illustrate the types of decisions made by the VFCDP, Figure 1 depicts the status of a food delivery system at four different points throughout a particular simulation. These images were captured from the visualization component of the simulation environment which has been developed.

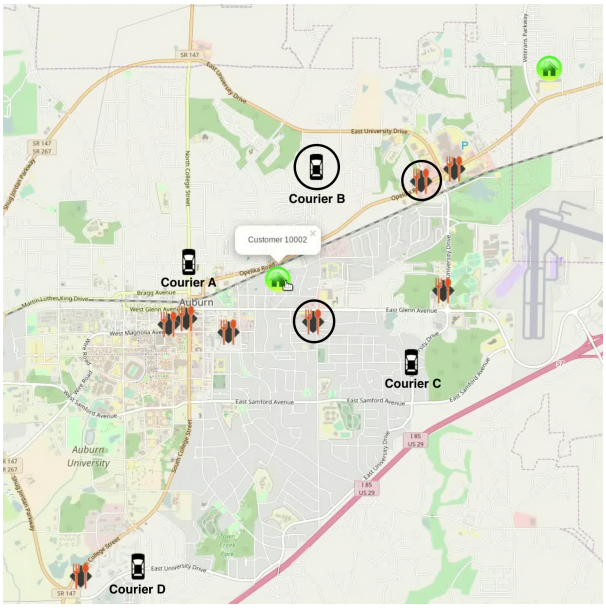
Traditional food delivery services limit each customer to placing an order from only one restaurant, and this is reflected in the existing literature. A key contribution of this paper is the introduction of a food delivery model which allows individual customers to order from more than one restaurant. In addition to solving the dynamic courier re-routing problem, this paper addresses the following managerial questions:

1. How is system efficiency affected by allowing individual customers to order from more than one restaurant?
2. In a system like the one described in question number 1, what are the impacts of split versus non-split deliveries?
3. What is the value of accounting for potential future orders while handling known orders in the present? What information can be leveraged to prepare for future demand?
4. How does the chosen promised delivery window affect solution quality (customer service)? Note that the delivery service will remain committed to this promised delivery time, even as new orders arrive (or traffic conditions change, food pickup is delayed by the restaurant, etc.).
5. What is an appropriate metric (similarly, objective function) with which to measure customer service?

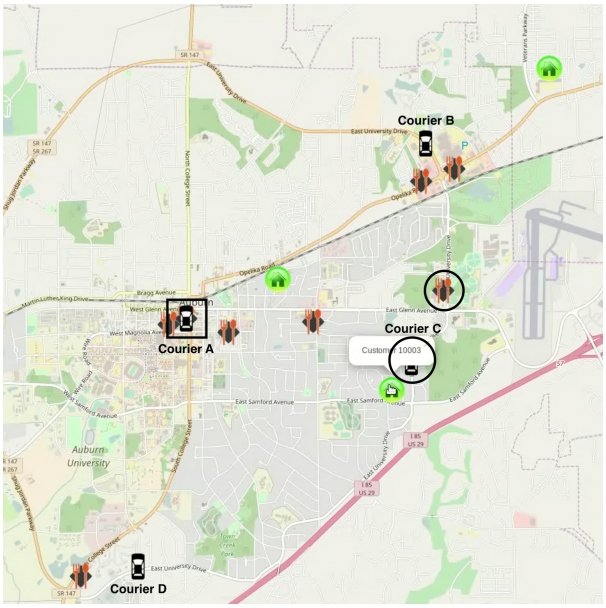
The remainder of this paper is organized as follows. A review of related literature is provided in Section 2. Section 3 introduces a formal problem definition of the VFCDP and a mathematical programming formulation of the problem. Due to the complexity of this integer programming formulation, a heuristic solution approach is required and is proposed in Section 4. Section 5 details the system architecture, which encompasses optimization, simulation, and visualization components. Experimental results and analyses are then discussed. Finally, a summary and suggestions for future research are provided in Section 6.



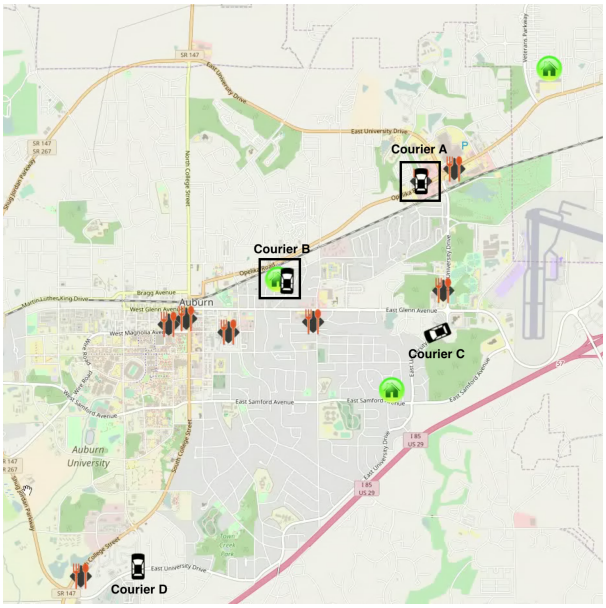
(a) The first customer of the day has just placed an order from the two restaurants in circles, and has been assigned to Courier A.



(b) Courier A is en route to a restaurant, at which point a second customer has arrived. Once again, they ordered from the circled restaurants and are assigned to Courier B.



(c) A third customer has arrived, ordered from a single restaurant, and has been assigned to Courier C. At the same time, Courier A is picking up food from a restaurant.



(d) Courier B is delivering food items to the customer from 1b, while Courier A is picking up food for the customer from 1a.

Figure 1: Visualizing system states in the VFCDP. Car symbols represent current courier locations, fork-and-spoon icons represent restaurant locations, and green houses represent active customers.

2 Related Literature

As no problem in the literature considers all of the problem features present in the VFCDP, we review several related problems. Most generally, the well-studied traveling salesman problem (TSP) and the vehicle routing problem (VRP), along with their extensions, seek to generate vehicle routes that cover a given set of locations. A detailed discussion on the TSP can be found in surveys by Golden et al. (2008) and Eksioglu et al. (2009). Dumitrescu et al. (2010) discuss a TSP variant with pickups and deliveries (TSPPD), where each delivery request corresponds to exactly one pickup. A further extension of the TSP known as the multiple traveling salesman problem (mTSP) considers scenarios with multiple vehicles (c.f. Bektas (2006) and Kergosien et al. (2009)). Similarly, both Gelareh et al. (2013) and Paolo and Vigo (2014) provide an overview of the VRP and its many extensions.

Dynamic VRP A survey of the dynamic VRP (DVRP) is provided by Psaraftis (1995). A model for event-driven optimization of a DVRP is presented by Pillac et al. (2012), in which a problem is re-optimized upon the arrival of a new request. More specific considerations such as capacitated vehicles, waiting strategies in dynamic environments, and solution approaches for dynamic problems can be found in Pillac et al. (2013). Stochastic DVRP models and models which implement rolling horizons can be found in Bektas et al. (2014). Additional problem features such as heterogeneous vehicle fleets, time windows, and traffic networks have also been explored (c.f. Fleischmann et al. (2004), van Woensel et al. (2008), Schyns (2015)).

Pickup and Delivery Problems for Goods A pickup and delivery problem (PDP) utilizes a fleet of vehicles to satisfy a set of both pickup and delivery requests. Overviews of the static-deterministic PDP are provided by Berbeglia et al. (2007) and Parragh et al. (2008). Crainic et al. (2016) consider various classifications of both travel and demand types. In the dynamic PDP (DPDP), pickup and delivery requests are handled in real time (c.f. Berbeglia et al. (2010)). Savelsbergh and Sol (1995) allow for multiple pickups per delivery request, while Liu et al. (2013) consider various types and combinations of pickups and deliveries. In Savelsbergh and Sol (1998), time windows constraints are applied to pickups, deliveries, and vehicle availability. Like the DVRP, the DPDP has been widely studied with respect to various problem features (Goel and Gruhn (2008), Lin (2011)) and solution techniques (Benavent et al. (2015), Cherkesly et al. (2015), Cherkesly et al. (2016)).

As described in Battarra et al. (2014), PDPs are traditionally classified as either one-to-one, one-to-many-to-one, or many-to-many. The VFCDP shares the most in common with the many-to-many class of PDPs, given that it involves multiple customers placing orders from a set of restaurants. The same authors discuss the notion of "mixed" PDPs, which allow for alternating sequences of pickup and delivery operations. For a single customer in the VFCDP, all pickups must precede a delivery, but across multiple customers the VFCDP is mixed.

Pickup and Delivery Problems for Passengers In the dial-a-ride problem (DARP), a driving service dynamically receives requests from customers to be transported from one location to another. Both pickups and drop-offs must occur within defined time windows. Vehicles are typically considered to be capacitated, and an upper limit is placed on the time between pickup and drop-off (the ride length) for any one person. For a deeper discussion on the classical DARP, the interested reader is referred to the review papers by Cordeau and Laporte (2007) and Berbeglia et al. (2010).

The traditional DARP is provider-centric in that it seeks to minimize total cost (route length). Parragh et al. (2009) provide a multi-objective model, which additionally seeks to minimize the av-

erage ride time for a customer. The modeling of customer-experience considerations is also discussed in Doerner and Salazar González (2014). Once again, heterogeneous vehicle fleets and stochastic system conditions have been studied in relation to the DARP (c.f. Jain and Van Hentenryck (2011) and Xiang et al. (2008), respectively). Various heuristic techniques have also been applied to the dynamic DARP, including parallel tabu search, fuzzy logic, and hyperheuristics (Attanasio et al. 2004, Maalouf et al. 2014, Urra et al. 2015).

The school bus routing problem (SBRP), first defined in Newton and Thomas (1969), shares strong ties to the DARP. Like the VFCDP, the SBRP is a many-to-many problem, which seeks to transport children from various bus stops to their respective schools. In Doerner and Salazar González (2014), a version of the SBRP which forces clusters of students to be picked up within specific time windows is considered. Bögl et al. (2015) provides a further extension of the SBRP which allows students to transfer buses en route to their final destination. Another extension of the SBRP which considers stochastic demand can be found in Caceres et al. (2017). Unlike the VFCDP, the SBRP is a static problem, and typically requires buses to make only one trip to each bus stop.

Meal Delivery Problems Research related specifically to dynamic vehicle routing for meal delivery services has recently received increased attention. An exact solution approach to the meal delivery routing problem (MDRP) is presented in Yildiz and Savelsbergh (2017). In Reyes et al. (2018), the authors introduce a dynamic-deterministic model for the MDRP, which they solve using a rolling horizon approach. Pickups that are to be made at the same restaurant may be grouped into "bundles" and treated as a single pickup operation. Future system states are also accounted for, both through the assignment of dummy "follow-on" pickup and delivery chains, and through pre-positioning techniques. Of note, couriers cannot receive updated route instructions which would cause them to be diverted while en route to a restaurant (a restriction which is not enforced in the VFCDP heuristic).

Similar to the MDRP, the restaurant meal delivery problem (RMDP) is defined by Ulmer et al. (2017). A dynamic-stochastic model is presented, along with an insertion heuristic. Assignment decisions which are considered "not-yet-important" may be delayed, allowing for additional information about the system to be revealed before routing decisions are made. Furthermore, the system accounts for unknown future events by penalizing solutions in which arrivals occur close to deadlines, thus favoring scenarios which leave some buffer around scheduled pickup and delivery operations. In all of the work on meal delivery problems referenced here, it is assumed that each customer places an order from exactly one restaurant.

3 Problem Definition and Formulation

Because orders inherently link customers to restaurants, this problem lends itself to a network representation. The network is characterized by four types of nodes: initial courier locations, customer locations, pickup points (customer/restaurant pairs), and final courier locations. For the first node type, each courier in the set of available couriers, V, is assigned a unique node representing their location at the beginning of the planning horizon (i.e., when the problem must be re-solved), such that courier $v \in V$ is currently located at node Δ_v^0. The set of active customers, C, will change upon the arrival of new orders or the completion of a delivery. In each of these two instances, a node directly correlates to a single, static location. A required final courier node, Δ^*, represents a static location as well. However, this location is virtual, with the travel time from any courier's penultimate location to node Δ^* having a value of zero. Upon the completion of a

courier's final assigned task, it is not required to return to a physical depot. This policy is drawn from the assumption that there is no central depot in the VFCDP, and also that couriers will likely receive new assignments shortly after completing old ones in this dynamic system.

One or more couriers may travel to the same restaurant location for different purposes. As such, a single node must represent one unique customer/restaurant pair. Each restaurant node is copied multiple times for each pickup task to avoid visiting nodes more than once. Let I_c represent the collection of restaurant nodes from which customer $c \in C$ has placed an order. More specifically, each element $i \in I_c$ represents customer c's order at a single restaurant. Each $i \in I_c$ is therefore a node, the location of which is identical to the location of the corresponding restaurant. Only nodes which have not yet been visited are included in I_c. An example of the relation among customers, restaurants, and nodes is shown in Table 1.

Table 1: Each customer-restaurant pair is mapped to one node.
$C = \{1, 2\}$, $I_1 = \{1, 2, 3\}$, $I_2 = \{4, 5\}$

Customer	**Restaurant**	**Node**
1	1	1
	3	2
	4	3
2	2	4
	4	5

Having defined the four node classes, let $N = \{\cup_{v \in V} \Delta_v^0 \cup C \cup_{c \in C} I_c \cup \Delta*\}$ represent the set of all nodes in the network. Each arc of the network represents a traversable path for a given courier $v \in V$, and connects an origin node to a destination node. For formulation purposes, it is beneficial to define the set of possible destination nodes for each courier, Δ_v^+, first. Each $j \in \Delta_v^+$ represents a node to which courier $v \in V$ may travel at some point. For example, imagine a network with customer nodes $\{1, 2\}$, restaurant nodes $\{3, 4\}$, and the dummy final node $\{\Delta^*\}$. If courier 1 can travel to any node in the network, then $\Delta_1^+ = \{1, 2, 3, 4, \Delta^*\}$. Now imagine that courier 2 is a bicyclist who cannot serve customers outside the city limits, and that customer 2 is one such customer. In this case, $\Delta_2^+ = \{1, 3, 4, \Delta^*\}$. Subsequently, the set of destination–specific origin nodes, Δ_{vj}^-, is defined. Here, each element $i \in \Delta_{vj}^-$ represents an origin node from which courier $v \in V$ could have traveled to destination node $j \in \Delta_v^+$. As an example, consider the fact that couriers cannot reach node Δ^* (their final node) if they still have food items on board. As such, a courier's final node cannot be reached directly from any restaurant node (where they would be picking up food items). In the above example, then, $\Delta_{v\Delta^*}^- = \{\Delta_v^0, 1, 2\}\ \forall\ v \in V$, where node Δ_v^0 is their starting node. Traveling directly from node Δ_v^0 to node Δ^* simply implies that courier v was assigned no tasks. Formally, we may define $\Delta_v^+ = \{C_v\} \cup \{I_c : c \in C_v\} \cup \Delta^*$. Similarly, we may define

$$\Delta_{vj}^- = \begin{cases} \{C_v\} \cup \Delta_v^0 & \text{if } j = \Delta^* \\ \{C_v \setminus j\} \cup \{I_c : c \in C_v\} & \text{if } j \in C_v \\ \{C_v \setminus c : j \in I_c\} \cup \{I_c : c \in C_v\} \cup \Delta_v^0 & \text{if } j \in R, \end{cases}$$

where C_v is the set of all customers which are currently assigned to courier v (regardless of whether or not courier v has picked up any of their items yet).

The definition of sets Δ_v^+ and Δ_{vj}^- further allow for a logical means of tracking travel time. At this point, τ_{vij} can be completely defined as the travel time for courier $v \in V$ to travel to

destination node $j \in \Delta_v^+$ from origin node $i \in \Delta_{vj}^-$. In our system, travel times are queried directly from the MapQuest Developer Network API (MapQuest, 2018) each time a solve is triggered, and thus accurately reflect the road networks over which problems were tested. Because real-time data from a true road network is used to populate the travel time matrix, factors such as heterogeneous speed limits and traffic congestion will be reflected in routing decisions. Along with the travel time τ_{ij} between any pair of nodes (i, j), the MapQuest API also returns the sequence of waypoints which define the path from i to j along the true road network (this information is used in our simulation environment to ensure that vehicles which have been routed from i to j follow real roads, obey one-ways, etc. along the way). Furthermore, these travel times are dependent on that courier's travel mode (e.g., by car, by bike, or walking). The travel time for any courier to their final node, $\tau_{vi\Delta^*}$, is zero from any node in the network.

Each courier may have a time window denoting the start and end of the period over which it is available for service (e.g., working hours). Let t'_v (t''_v) represent the start (end) time of the work shift of courier $v \in V$. Each courier may also have a different carrying capacity, denoted by W_v. This capacity could be in terms of volume, weight, or quantity. For the sake of simplicity, this model handles capacity in terms of the quantity of items in the possession of a courier. Furthermore, $\hat{w}_j$ represents the quantity of items picked up from location $j \in I_c$ ($\hat{w}_j > 0$). Similarly, $\hat{w}_c$ represents the quantity of items dropped off at location $c \in C$ ($\hat{w}_c < 0$).

When a customer order is placed, each restaurant from which the customer has ordered provides an "earliest available pickup time" for each customer/restaurant pair. This is represented by $t_i^{\min}$ for $i \in I_c$ $c \in C$. A courier cannot pick up an item from node i prior to $t_i^{\min}$. A service time of s_j is associated with all $j \in N$ (where a service time of zero is assigned to the couriers' initial and final nodes). When the courier arrives at a restaurant, then, an expected service time of s_j time units is assumed. The same is true when a courier makes a delivery to a customer.

Three customer-focused objectives are explored. The first looks to maximize the total earliness to all customers (in relation to the promised delivery times). An objective function penalty of p units is incurred per unit time that a delivery is made beyond $t_c^{\max}$ for any customer $c \in C$. No penalty is incurred if the delivery is made within $t_c^{\max}$ of the order being placed, where $t_c^{\max}$ is the pre-defined period within which delivery has been guaranteed. The length of $t_c^{\max}$ may be defined according to any preferred company policy (e.g., $t_c^{\max} = 1800(s)$ in a "30 minute or less" policy). The use of a penalty is necessary to allow couriers to make late deliveries when on–time delivery is not possible in order to retain mathematical feasibility. To ensure reasonable solutions, however, a finite length of time beyond $t_c^{\max}$, defined as $\delta^{\max}$, is established within which a customer must receive their order.

The second objective minimizes the total time between order items being *ready* for pickup and when those items are delivered to the customer. The third objective minimizes the total time between the pickup and delivery of order items.

3.1 Mathematical Programming Formulation

As part of our approach to solving the VFCDP, we introduce the following mixed integer linear program (MILP). This formulation is used to assign both customers and routes to the fleet of couriers, according to the information which has been revealed up to the decision point in our system (Section 4 will further clarify the role that this MILP plays in our heuristic solution procedure). Tables 2 and 3 provide summaries of the necessary parameter and decision variable notations, respectively.

Table 2: Sets and Parameters

V	Set of couriers (delivery drivers).
C	Set of customers.
C_v	Set of active customers which are assigned to courier v
C'_v	Set of active customers for which $v \in V$ has already picked up at least one item for this customer.
R	Set of restaurants from which any customer may place an order.
I_c	Set of restaurants included in the order placed by customer $c \in C$.
$t_c^{\max}$	Latest allowable unpenalized delivery time to customer $c \in C$.
p	Objective function penalty per unit time that a customer receives their order to beyond $t_c^{\max}$ for any customer $c \in C$.
$\delta^{\max}$	Maximum allowable penalized time beyond $t_c^{\max}$ to deliver to any customer $c \in C$.
$t_i^{\min}$	Earliest time that order $i \in I_c$ may be picked up.
s_j	Service time at node $j \in N$.
t'_v (t''_v)	Start (end) time of courier $v \in V$'s shift.
$\hat{w}_j$	Quantity of items picked up from location $j \in I_c$ ($\hat{w}_j > 0$) or dropped off at location $c \in C$ ($\hat{w}_j < 0$).
$\bar{w}_v$	Number of items already being carried by courier $v \in V$ at the start of the planning horizon.
W_v	Maximum capacity of courier $v \in V$.
Δ_v^0	Initial node for courier $v \in V$.
Δ^*	Dummy (virtual) final node for all couriers; $\Delta^* = 0$.
Δ_v^+	Set of all nodes to which courier $v \in V$ may travel; $\Delta_v^+ \subseteq \{C \cup P \cup \Delta^*\}$.
Δ_{vj}^-	Set of all nodes from which courier $v \in V$ may travel to node $j \in \Delta_v^+$; $\Delta_{vj}^- \subseteq \{\Delta_v^0 \cup \{\cup_{c \in C} I_c \setminus j\} \cup \{C \setminus j\}\}$.
N	Set of all nodes in the network; $N = \{\cup_{v \in V} \Delta_v^0 \cup C \cup_{c \in C} I_c \cup \Delta*\}$.
τ_{vij}	Time required by courier $v \in V$ to travel to node $j \in \Delta_v^+$ from node $i \in \Delta_{vj}^-$.

Table 3: Decision Variables

$x_{vij} \in \{0, 1\}$	Binary decision variable equals one if courier $v \in V$ travels to node $j \in \Delta_v^+$ from node $i \in \Delta_{vj}^-$; zero otherwise.
$y_{vi} \in \{0, 1\}$	Binary decision variable equals one if courier $v \in V$ picks up an item from node $i \in I_c$; zero otherwise.
$t_{vi} \geq 0$	Continuous decision variable representing the time at which courier $v \in V$ visits node $i \in \Delta_v^+$.
$w_{vij} \geq 0$	Continuous decision variable representing the quantity of items in possession of courier $v \in V$ while en route to node $j \in \Delta^+$ after visiting node $i \in \Delta_{vj}^-$.
$0 \leq \delta_c \leq \delta^{\max}$	Delay in delivering to customer $c \in C$ after time $t_c^{\max}$ (i.e. lateness to customer $c \in C$). A unity penalty of p is applied to δ_c in the objective function.

$$\text{Max} \quad \sum_{c \in C} \sum_{v \in V} \sum_{i \in \Delta_{vc}^-} (t_c^{\max} x_{vic} - t_{vc}) - p \sum_{c \in C} \delta_c \tag{1}$$

$$\text{s.t.} \quad \sum_{v \in V} \sum_{i \in \Delta^-_{vc}} x_{vic} = 1 \quad \forall\, c \in C, \tag{2}$$

$$\sum_{i \in \Delta^-_{vc}} x_{vic} = 1 \quad \forall\, v \in V, c \in C'_v, \tag{3}$$

$$\sum_{i \in I_c} y_{vi} = |I_c| \sum_{i \in \Delta^-_{vc}} x_{vic} \quad \forall\, c \in C, v \in V, \tag{4}$$

$$t_{vc} \le t_c^{\max} + \delta_c \quad \forall\, c \in C, v \in V, \tag{5}$$

$$t_{vj} \le \left(\max_{c \in C} \{t_c^{\max}\} + \delta^{\max} \right) \sum_{i \in \Delta^-_{vj}} x_{vij} \quad \forall\, v \in V, j \in \Delta^+_v, \tag{6}$$

$$t_{vi} \le t_{vc} - (s_i + \tau_{v,i,c})\, y_{vi} \quad \forall\, v \in V, c \in C, i \in I_c, \tag{7}$$

$$t_{vi} \ge t_i^{\min} y_{vi} \quad \forall\, c \in C, i \in I_c, v \in V, \tag{8}$$

$$t_{vj} \ge t_{vi} + s_i + \tau_{v,i,j} - \left(\max_{c \in C} \{t_c^{\max}\} + \delta^{\max} + s_i + \tau_{v,i,j} \right) (1 - x_{vij})$$
$$\forall\, v \in V, j \in \Delta^+_v, i \in \{\Delta^-_{vj} \setminus \Delta^0_v\}, \tag{9}$$

$$w_{vij} \le W_v \quad \forall\, v \in V, j \in \Delta^+_v, i \in \Delta^-_{vj}, \tag{10}$$

$$w_{v,\Delta^0_v,j} = (\bar{w}_v + \hat{w}_j)\, x_{v,\Delta^0,j} \quad \forall\, v \in V, j \in \Delta^+_v, \tag{11}$$

$$w_{vjk} \ge \sum_{i \in \{\Delta^-_{vj} : i \ne k\}} w_{vij} + \hat{w}_k x_{vjk} - W_v(1 - x_{vjk}) \quad \forall\, v \in V, k \in \Delta^+_v, j \in \{\Delta^-_{vk} \setminus \Delta^0_v\}, \tag{12}$$

$$y_{vk} = \sum_{i \in \Delta^-_{vk}} x_{vik} \quad \forall\, v \in V, c \in C, k \in I_c, \tag{13}$$

$$y_{vi} = 1 \quad \forall\, v \in V, c \in C'_v, i \in I_c, \tag{14}$$

$$\sum_{j \in \Delta^+_v} x_{v,\Delta^0_v,j} = 1 \quad \forall\, v \in V, \tag{15}$$

$$\sum_{i \in \Delta^-_{v,\Delta^*}} x_{v,i,\Delta^*} = 1 \quad \forall\, v \in V, \tag{16}$$

$$\sum_{i \in \Delta^-_{vj}} x_{vij} = \sum_{k \in \{\Delta^+_v : j \in \Delta^-_{vk}\}} x_{vjk} \quad \forall\, v \in V, j \in \{\Delta^+_v \setminus \Delta^*\}, \tag{17}$$

$$t_{vj} \ge \left(t'_v + \tau_{v,\Delta^0_v,j}\right) x_{v,\Delta^0_v,j} \quad \forall\, v \in V, j \in \Delta^+_v, \tag{18}$$

$$t_{v,\Delta^*} \le t''_v \quad \forall\, v \in V, \tag{19}$$

$$0 \le \delta_c \le \delta^{\max} \quad \forall\, c \in C, \tag{20}$$

$$t_{vi} \ge 0 \quad \forall\, v \in V, i \in \Delta^+_v, \tag{21}$$

$$w_{vij} \ge 0 \quad \forall\, v \in V, j \in \Delta^+_v, i \in \Delta^-_{vj}, \tag{22}$$

$$x_{vij} \in \{0,1\} \quad \forall\, v \in V, j \in \Delta^+_v, i \in \Delta^-_{vj}, \tag{23}$$

$$y_{vi} \in \{0,1\} \quad \forall\, v \in V, c \in C, i \in I_c. \tag{24}$$

The objective function (1) seeks to maximize the total earliness to all customers, while also penalizing lateness to customers. This function is piecewise linear about point t_c^{max} for each customer c. Given a positive penalty value $p > 0$, making two deliveries 5 minutes early each is preferable to making one delivery 15 minutes early and another delivery 5 minutes late (of course, if a user prefers

not to penalize lateness, they could set $p = 0$; in such a case, both of the aforementioned scenarios would have the same objective value). If a particular courier $v \in V$ does not serve customer $c \in C$ then $x_{vic} = t_{vc} = 0$. Otherwise, a reward is earned by delivering in advance of the committed delivery time, $t_c^{\max}$ (which is set upon customer c's arrival). If the delivery occurs between time $t_c^{\max}$ and $t_c^{\max} + \delta^{\max}$, a per-unit penalty of $p \geq 0$ is assessed. The value of p may be determined by the user, depending upon the user's willingness to allow late deliveries. If $\delta^{\max} = 0$, no late deliveries will be allowed. Since this may lead to infeasible problems, it is advised to allow $\delta^{\max}$ to be non-zero but with a large value of p.

Constraints (2) – (4) satisfy delivery requirements (in terms of quantity). Constraints (2) state that each customer must be served exactly once by a single courier. If a courier has already picked up some items for a customer, that courier must complete the corresponding delivery to that customer, as in Constraints (3). Constraints (4) ensure that all customer items are picked up as part of the order.

Constraints (5) – (9) govern the timing relationships. Specifically, Constraints (5) state that delivery must occur before the committed-to delivery time (plus a penalized late time). If courier $v \in V$ does not deliver to customer $c \in C$, then t_{vc} must take a value of zero per Constraints (6); otherwise, this constraint will be non-binding ($\max_{c \in C}\{t_c^{\max}\} + \delta^{\max}$ is a sufficiently large value). Constraints (7) state that items must be collected from a restaurant before they are delivered to the customer, including service time at the restaurant and travel time from that restaurant to the customer. If courier $v \in V$ does not deliver to customer $c \in C$, then such a constraint is non-binding ($t_{vc} = t_{vi} = y_{vi} = 0$ in this case). In Constraints (8), if courier v collects an order for customer $c \in C$ from node $i \in I_c$, then the collection time must not be before the order is expected to be ready ($t_i^{\min}$). Finally, Constraints (9) state that if v travels from i to j, the earliest arrival at j equals the arrival time at i plus the service time at i plus the travel time from i to j. In the event that courier v does not travel directly from i to j, this constraint will be non-binding. Note that these constraints ignore departures from each courier's initial location; this condition is addressed in Constraints (18).

Payload capacities are addressed in Constraints (10), (11), and (12). Specifically, Constraints (10) establish the maximum payload capacity for each courier. Constraints (11) define a courier's load after their first stop in the new/updated route to be their initial load when the dynamic re-tasking procedure is executed plus the quantity picked up (or dropped off) by the courier during that first stop. Similarly, Constraints (12) determine the net payload in the possession of courier v as they travel from node j to node k in the network (where that payload equals zero if courier v does not actually travel from j to k).

Constraints (13) establish the relationship between decision variables y and x. Constraints (14) capture existing assignments. Specifically, if any item for a particular customer has already been picked up, then this customer must now be tied to this courier. In other words, the integer programming model does not allow for split deliveries (a property which may be relaxed in the heuristic by treating each customer as m unique customers, where m is the number of restaurants included in their order). C'_v contains customers for whom at least one item has been picked up. For all $c \in C'_v$, then, the set I_c will only contain the subset of the entire list of restaurants from which they ordered. The nodes corresponding to items which have been picked up for a customer will have been removed from the set I_c for that customer.

The next block of constraints addresses courier routing. Constraints (15) state that each courier must leave their current/initial location, while Constraints (16) indicate that each courier must terminate their route at the dummy (virtual) ending node. Constraints (17) guarantee that each courier will visit and depart a node the same number of times (either once or never). Constraints

(18) state that courier v cannot begin a route before time t'_v (the beginning of their shift), while Constraints (19) ensure that each courier v will finish their route before the end of their shift (i.e., by time t''_v). Finally, Constraints (20) – (24) are decision variable definitions.

3.1.1 Alternate Objective Functions

Two alternative objective functions which consider the freshness of food are also considered. The objective function defined in (25) minimizes the total "ready-to-delivery" time, or the gap between the anticipated time that food will be ready for pickup (t_i^{min}) and the time at which that food is delivered to the customer (t_{vi}). If courier v does not visit customer c, then $t_{vc} = 0$. Therefore, the constant term t_i^{min} is multiplied by y_{vi}, the binary decision variable representing whether or not courier v picks up item $i \in I_c$. The second freshness-focused objective function, (26), minimizes the total "pickup-to-delivery" time. This term represents the amount of time between when food is picked up at a restaurant and when it is delivered to the customer, or equivalently, the amount of time each food item spends in a courier's possession.

$$\text{Min} \quad \sum_{c \in C} \sum_{v \in V} \sum_{i \in I_c} \left(t_{vc} - t_i^{min} y_{vi} \right) \tag{25}$$

Objective function (26) further allows us to demonstrate the value of a system in which orders are not immediately revealed to restaurants after being received by the delivery service. For example, imagine that an item's expected preparation time is 30 minutes, but no courier can arrive at the corresponding restaurant until 40 minutes after the order is placed. By waiting 10 minutes before revealing that order to the restaurant (and given that the food indeed takes 30 minutes to prepare), that food will spend 10 less minutes waiting to be picked up. While such a system is not explicitly modeled, the difference between objective values obtained with (25) and (26) could allow a system administrator to estimate the room for improvement with respect to freshness.

$$\text{Min} \quad \sum_{c \in C} \sum_{v \in V} \sum_{i \in I_c} \left(t_{vc} - t_{vi} \right) \tag{26}$$

3.2 Triggering a Re-solve

Because the set of customers and overall system state change dynamically, this model must be re-solved periodically. Specifically, the model above may be re-solved each time a new order is placed, if a courier is "very" early or late to a node in the network, or if the time at which an order is ready to be picked up differs "significantly" from what was expected. Of course, here the meanings of "very" and "significantly" are subjective, and exact values in these cases may be defined by the system user.

Once a re-solve has been triggered, several pre-processing activities are required before the new solution procedure begins. First, for both bookkeeping purposes and to remove unneeded complexities, any customer whose orders have been delivered by the time of the re-solve should be removed from C. Similarly, any node $i \in I_c$ corresponding to an order which has already been picked up should be removed from I_c (even if that customer's order has not yet been delivered). The sets Δ_v^+ and Δ_{vj}^- should be updated to reflect only unvisited nodes, and to incorporate the current location of each courier. A courier's current location at the time of a re-solve is treated as their new "initial" location node, Δ_v^0. As such, $\tau_{v,\Delta_v^0,j}$ must be recalculated for each $v \in V$ and for

each $j \in \Delta_v^+$ each time a re-solve is triggered.

If a node $i \in I_c$ is removed from I_c, but the corresponding customer $c \in C$ is not removed from C, then some portion of that customer's order still needs to be picked up. Recall that C'_v represent the set of customers whose orders have not yet been delivered, but for whom at least one item has been picked up by courier $v \in V$ at the time of a re-solve. Therefore, any courier-customer relationship formed by set C'_v must remain during the re-solve. The sets Δ_v^+ and Δ_{vj}^- should subsequently be updated once more to prohibit any other courier from visiting nodes which must be served by such a courier. The number of items in the possession of a courier at the time of the re-solve ($\bar{w}_v$) should then be updated.

Finally, any τ_{vij} which is affected by traffic, weather, or other conditions may be updated. In our system, travel times are updated in real time by querying the MapQuest API prior to each solve. Some $t_i^{\min}$ values might also need to be updated if an order is delayed by the restaurant, or if the order is ready earlier than expected.

4 Auction-based Heuristic

As described in Section 3.2, the VFCDP model presented in Section 3 must be re-solved upon each new arrival. Unfortunately, the computation time required to solve the VFCDP model is prohibitively large for this dynamic setting. As such, an auction-based heuristic has been developed.

Rather than considering the entire fleet of couriers simultaneously, the heuristic considers one courier at a time. The goal is to find the optimal route for courier $v \in V$ if they were to be assigned the newly arrived customer. That route, and the corresponding objective function value, are generated by solving the VFCDP model over a restricted set of inputs - courier v, all of the customers currently being served by courier v, and the newly arrived customer. This process is then repeated for each of the active couriers ($\forall\ v \in V$). The VFCDP instance corresponding to the scenario in which courier v is assigned the new customer is referred to courier v's sub-problem. Because the inputs to these sub-problems are restricted to a single courier and only a subset of the active customers (usually no more than two or three), they solve rapidly (each one in under a second - the total auction time, then, is the time required to solve one sub-problem for each $v \in V$). Each courier's objective value is then normalized by the number of customers which they would be serving if the new customer was added to their list of assigned tasks (i.e. the cardinality of the set of customers which was input to their sub-problem). Each courier's normalized objective value is referred to as their bid value, b_v, and reflects the average service experience of the customers which they would be handling.

4.1 Myopic Variant

In the so-called "myopic" version of the heuristic, the courier with the maximum bid value ($v^* = \max_{v \in V}(b_v)$) wins the auction, and is assigned the newly arrived customer. In this case, their current route is replaced by the route which was returned by their sub-problem (and all other courier's routes remain unchanged). Let Γ represent the set of all courier routes, $\gamma_v \in \Gamma$ represent courier v's current route, and γ'_v represent the route returned by courier v's sub-problem. In the myopic case, then, $\gamma_{v^*} \leftarrow \gamma'_{v^*}$ ($\Gamma = \Gamma \cup \gamma_v \setminus \gamma_{v^*}$).

4.2 Proactive Variant

An alternate version of this heuristic (referred to as the "proactive" heuristic) has also been developed which further considers two future-looking metrics. The notion of measuring a dynamic vehicle routing system's preparedness for future demand is well established in the literature. In their work on the DARP, Diana and Dessouky (2004) define a measure of decentralization, or dispersion, which ensures that vehicles are spread out. Similarly, Ritzinger et al. (2016) define spatial and temporal utilization measurements for the stochastic VRP. Batta et al. (2014) define a dispersion metric for a facility location problem which requires at least a certain distance exists between each pair of facilities. Similarly, an equity metric pulls facilities toward a set of known population centers. This concept is adapted for use in the VFCDP heuristic by exploiting the fact that customer deliveries are bottlenecked by a set of known restaurant locations. Specifically, this heuristic employs measures of equity and dispersion, each of which relates to couriers' potential locations at future points in time. Assignment decisions in the proactive heuristic are therefore made on the basis of bid, equity, and dispersion values, as opposed to bid values alone (as is the case in the myopic heuristic).

Consider the scenario in which courier v is assigned a newly arrived customer. We can consider what the set of courier routes, Γ, would look like in this scenario by temporarily replacing γ_v with $\gamma_v^{'}$ ($\Gamma_v^{'} = \Gamma \cup \gamma_v^{'} \setminus \gamma_v$). Because these routes are defined along a true road network (with known waypoints, speed limits, etc.), the position of any courier along its route at some future time can be estimated. As such, the set of courier locations along route set $\Gamma_v^{'}$ at various future points in time are projected. Let the set of future times to be considered be represented by T^f. For each future time $\hat{t}_i \in T^f$, the couriers' projected locations are used to calculate time-specific equity and dispersion values ($\bar{e}_{\hat{t}_i v}$ and $\bar{d}_{\hat{t}_i v}$, respectively). Afterward, weighted averages are taken across the sets of time-specific equity and dispersion values, resulting in a single equity (e_v) and dispersion (d_v) value for that scenario. This entire process is then repeated for each potential assignment scenario (over $\Gamma_v^{'}$ $\forall$ $v \in V$). Algorithm 1 summarizes the process of calculating the equity and dispersion metrics. Relevant equations are defined in Sections 4.2.1 and 4.2.2.

Algorithm 1 Calculating the Equity and Dispersion Metrics

```
for v ∈ V do
    Γ'_v = Γ ∪ γ'_v \ γ_v
    for t̂_i ∈ T^f do
        for z ∈ V do
            Project courier z's location at t̂_i along γ_z ∈ Γ'_v
        end for
        Calculate ē_{t̂_i v} and d̄_{t̂_i v}
    end for
    Calculate e_v and d_v
end for
```

4.2.1 Equity Metric

The equity metric considers how far away the nearest courier(s) will be to each restaurant $r \in R$ at some future time $\hat{t}_i$. In other words, we anticipate that another customer will arrive in the near future. Shortly thereafter, that customer's food will be ready for pickup. Call this conceptual time $\bar{t}$. Unfortunately, we do not yet know which restaurant(s) the next customer will order from. The

equity metric therefore considers how "accessible" each restaurant would be at $\bar{t}$, in case they need to be reached.

In order to measure equity, we must introduce some new terminology. First, the future time $\bar{t}$ must be estimated. Rather than estimating a single value, however, T^f is populated with a set of possible values $\hat{t}_i$, which are calculated as follows:

$$\hat{t}_i = \left[t_{now} + \mu_a + \frac{\sum_{r \in R}(p_r * l_r)}{|R|} \right] + (i)(\sigma_a) \qquad \forall\ i = -2, ..., 2, \tag{27}$$

where μ_a is the mean customer interarrival time, σ_a is the standard deviation associated with μ_a, p_r is the mean food preparation time associated with restaurant r, t_{now} is the time in the day at which $\hat{t}_i$ is being calculated, and i is simply an integer multiplier.

Next, we define a weight for each restaurant $r \in R$. We refer to a restaurant's weight as its likelihood value, l_r, as this value roughly represents the likelihood that a random future customer will place an order from that restaurant. These values are not probabilities, and need not sum to one. It is assumed that likelihood values can be generated based on order volume data for each restaurant, which would obviously be in the possession of a food delivery service that handles such orders. Restaurants with higher order volumes receive higher likelihood values.

Each future time is also assigned a weight, $\omega_{\hat{t}_i}$, which is sampled from a normal distribution. As $|i|$ increases, a time $\hat{t}_i$ receives a lower weight. This calculation is expressed formally as

$$\omega_{\hat{t}_i} = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}i^2}. \tag{28}$$

Finally, we define the set of n-nearest couriers to restaurant r (at time $\hat{t}_i$, given route set $\Gamma_v^{'}$). Let this set be denoted by $V_{r\hat{t}_i v}$. A number of factors could prevent a courier from being assigned to pick up food from a nearby restaurant (i.e. causing lateness to another active customer). Therefore, as the likelihood that a restaurant will be ordered from increases, more couriers are considered when examining that restaurant's equity value. The set $V_{r\hat{t}_i v}$ is composed of the n active couriers (z) with the smallest straight-line distance between their projected location (at time $\hat{t}_i$ according to route set $\Gamma_v^{'}$) and restaurant r's location. Let that distance be denoted by $\beta_{zr\hat{t}_i v}$. Straight-line distance is used in this metric simply for the sake of speed (as opposed to the real road network data which is used during the routing procedure, but takes longer to query).

The actual equity calculation is fairly straightforward. First, at each future time, calculate the time-specific equity value ($\bar{e}_{\hat{t}_i v}$) using Equation (29). This term is simply a weighted average of the straight-line distances ($\beta_{zr\hat{t}_i v}$) between restaurants and their nearest couriers. Notice that the straight line distance terms appear in a denominator, implying that large equity values are good. Then, using Equation (30), calculate the time-weighted average across the $\bar{e}_{\hat{t}_i v}$ values in order to obtain a single equity value (e_v) for the current scenario.

$$\bar{e}_{\hat{t}_i v} = \frac{\sum_{r \in R} l_r \sum_{z \in V_r} \frac{1}{\beta_{zr\hat{t}_i v}}}{|R|} \tag{29}$$

$$e_v = \sum_{\hat{t}_i \in T^f} \omega_{\hat{t}_i} \bar{e}_{\hat{t}_i v} \tag{30}$$

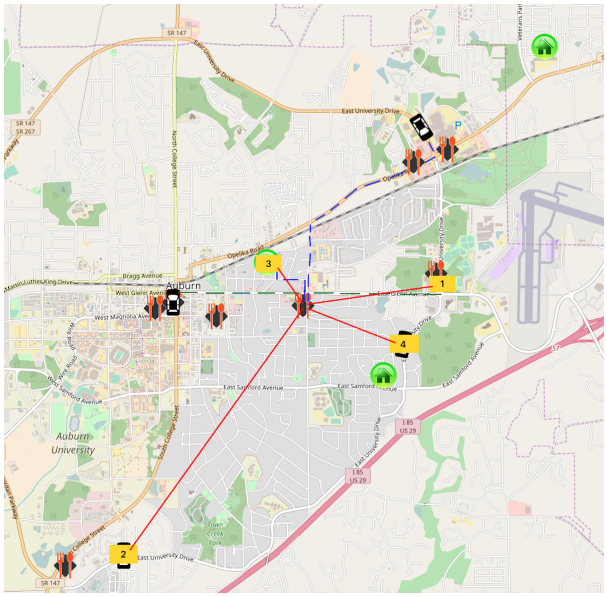
4.2.2 Dispersion Metric

The dispersion metric employed here measures the "spread" of couriers over the city at hand. Like the equity metric, dispersion is measured according to couriers' positions at some future time $\hat{t}_i$

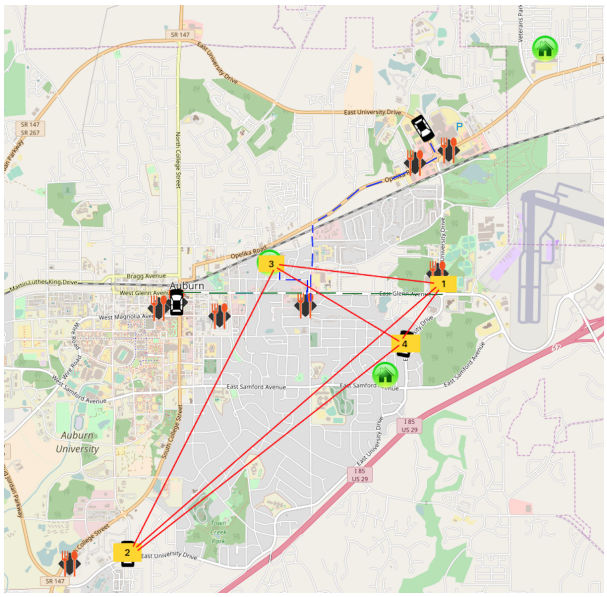
along a given route set, $\Gamma_v^{'}$. At that point, the average straight-line distances between each possible pair of couriers are calculated using Equation (31). This average is not weighted, as no one courier is considered any more important that the others. After this has been done for each future time $\hat{t}_i \in T^f$, the time-weighted average of all $\bar{d}_{\hat{t}_i v}$ values (denoted by d_v) is computed using Equation (32). Similar to the equity metric, large dispersion values are considered to be good, as this implies that couriers are not clustered together. Figure 2 has been provided to help visualize the difference between equity and dispersion.

$$\bar{d}_{\hat{t}_i v} = \left(\frac{2!(|V|-2)!}{|V|!} \right) \sum_{i \in V} \sum_{j \in \{V : j \neq i\}} \beta_{ij\hat{t}_i v} \tag{31}$$

$$d_v = \sum_{t \in T'} \omega_{\hat{t}_i} \bar{d}_{\hat{t}_i v} \tag{32}$$



(a) Equity - examining the distances from one restaurant to four active couriers in order to estimate how easily the restaurant could be reached.



(b) Dispersion - examining the pairwise distances between active couriers in order to estimate how well distributed the couriers are.

Figure 2: Visualizing equity vs. dispersion. Numbered labels highlight anticipated courier locations ten minutes into the future, given the set of routes currently being considered.

4.2.3 Bid Selection

At this point, we have considered the scenario of each possible courier being assigned a new customer. The bid selection procedure begins by identifying the scenario with the maximum bid value and setting that scenario equal to the incumbent auction winner. In other words, $v^* = max_{v \in V}(b_v)$. Should this v^* win the auction, the assignment decision would be no different than in the myopic case. However, we are allowed to swap the incumbent auction winner out for an alternate scenario ($v^a \neq v^*$) if the following conditions hold:

1. The percent loss in bid value from b_{v^*} to b_{v^a} is *below* an acceptable threshold, θ, and

2. Any one of the following:

   (a) The percent improvement in equity from e_{v^*} to e_{v^a} is *above* a required threshold, ϕ, or

   (b) The percent improvement in dispersion from d_{v^*} to d_{v^a} is *above* a required threshold, ψ, or

   (c) The percent improvements in equity and dispersion from (e_{v^*}, d_{v^*}) to (e_{v^a}, d_{v^a}) are both above a required threshold, χ (where $\chi < \phi$)

If no alternate scenarios v^a satisfy the above conditions, the original incumbent v^* is accepted as the auction winner. This implies that courier v^* is assigned the new customer, and the current route set is updated ($\Gamma \leftarrow \Gamma'_{v^*}$). If one or more alternate scenarios satisfy the swapping criteria, we will select an alternate scenario v^a to be the auction winner. This is because alternate scenarios which satisfy the swapping criteria exemplify an acceptable amount of loss in short term performance, while providing a meaningful improvement to the system's preparedness for future demand. In the case that exactly one alternate scenario v^a meets the swapping criteria, we declare scenario v^a the auction winner ($v^* \leftarrow v^a$, $\Gamma \leftarrow \Gamma'_{v^*}$). Otherwise, we have a list of multiple scenarios which satisfy the swapping criteria. We call this list the candidate scenario list (CSL). From among the scenarios on the CSL, we select the scenario with the maximum bid value to win the auction ($v^* \leftarrow max_{v^a \in CSL}(b_v)$, $\Gamma \leftarrow \Gamma'_{v^*}$). This is done because the bid values are derived from the VFCDP's objective function, and are therefore still the primary focus.

5 System Analysis

In order to test the efficacy of the VFCDP heuristic, a dynamic simulation environment was developed in Python. This system is outlined in Section 5.1. Subsequently, this simulation environment was used to conduct a number of experiments. Section 5.2 presents the full-factorial experimental design which was constructed, along with several supplemental experiments which were run. Numerical analyses of experimental results are then provided in Section 5.3.

5.1 System Implementation

Outside of the simulation environment itself, a series of test problems were first generated. One test problem is represented by a pre-defined customer stream (where each customer has an associated location, arrival time, and order set), as well as all relevant restaurant and courier information (e.g. the start and end times for each courier's shift). Customer locations are generated according to a uniform random distribution, whereas information (e.g. the set of restaurants from which a customer places an order, expected food preparation times, etc.) are sampled from distributions based on publicly available market data.

Throughout the course of a simulation, customers are revealed to the system at their pre-defined arrival times. Each customer arrival triggers a re-solve. Once an assignment decision has been made and courier routes have been updated, couriers' movements and pickup and delivery operations are simulated up to the arrival of the next customer. Clearly, then, the system tracks which couriers are working, where they are located, which tasks they are handling, and so on. The system additionally introduces elements of stochasticity. For example, routing decisions are made based on expected food preparation times. When a courier arrives at a restaurant, however, they may find that a particular item is not yet ready for pickup. A visualization component has also been developed through the online mapping service, Leaflet (the images shown in Figure 1, save the circles and rectangles, are screenshots taken from that visualization component).

5.2 Experimental Design

Table 4 presents the factors (and their corresponding levels) which were included in the full-factorial experimental design. In order to examine the VFCDP's scalability, simulations were run for systems operating in Auburn, AL (a small city), and Buffalo, NY (a significantly larger city). Policies which allow customers to order from a maximum of one, two, and three restaurants at a time were also tested. Next, two delivery window policies were tested. The first policy offers a fixed delivery window to each customer (45-minutes or less in Auburn, and 60-minutes or less in Buffalo). The second policy offers tailored delivery windows which are related to the maximum expected preparation time among the food items included in a customer's order set. Each of the three objective functions defined in Section 3 were tested in order to measure the system's performance when driven by various measures of customer satisfaction. Finally, three versions of the heuristic were tested: the myopic and proactive versions defined in Section 4, as well as a mixed version (which considers equity and dispersion only during peak demand hours). Two replicates were tested for each of these 108 treatment combinations, for a total of 216 experimental trials. The entire full-factorial experimental sequence was conducted using a non-split delivery policy.

Table 4: All factors (and their levels) which were included in the full factorial experimental design. All combinations of levels across the five experimental factors were tested, for a total of 108 unique treatment combinations. Two replicates were tested for each treatment combination.

Factor	Levels
City	Auburn, Buffalo
Delivery Window Type	Fixed, Tailored
Objective Function	(1), (25), (26)
Heuristic Version	Proactive, Mixed, Myopic
Max Restaurants/Customer	1, 2, 3

Of the 108 test problems in the first replication of full-factorial experimental design, 72 consider a policy which allows customers to order from more than one restaurant. These trials were run an additional time, but with a split-delivery policy. Such a policy was modeled by simply treating each customer as m separate customers, where m is the number of restaurants that customer ordered from. Each solve, then, introduces only the items ordered by a new customer from a single restaurant. Note that this does not require any change to the VFCDP model or heuristic (just the data structure used to represent customer information). The corresponding 144 trials are used to compare the split and non-split delivery policies.

Finally, several test problems were run for which the VFCDP model was re-solved over the entire sets of active couriers and customers upon each customer arrival. While the computation time required for such a strategy is prohibitively long for implementation in a dynamic setting, this provides a benchmark for the VFCDP heuristic's performance. For each problem, an arbitrary four hour window was simulated. All testing was conducted on a Dell OptiPlex 3046 with 16 GB of RAM, 8 CPUs (with a clock speed of 3.40 GHz), and an Intel Core I7-6700 processor.

5.3 Numerical Analysis

Experiments in Auburn and Buffalo included total courier fleets of size 15 and 40, respectively. However, heterogeneous courier shifts were designed such that only a subset of all the couriers were working at any given time, with the largest number of couriers working during peak demand hours. While this strategy results in a more strained system than if the entire fleet was always available,

it also reduces the size of the solution space each time the heuristic is called. During peak demand hours, Auburn and Buffalo experienced customer arrival rates of 10 and 24 customers per hour, respectively. These rates were estimated based on order volume data from several food delivery services' investor reports and filings for the U.S. Securities and Exchange Comission (SEC). Figure 3 depicts the customer arrival rate in both cities throughout the four hour simulation period as a percentage of that city's peak demand rate.

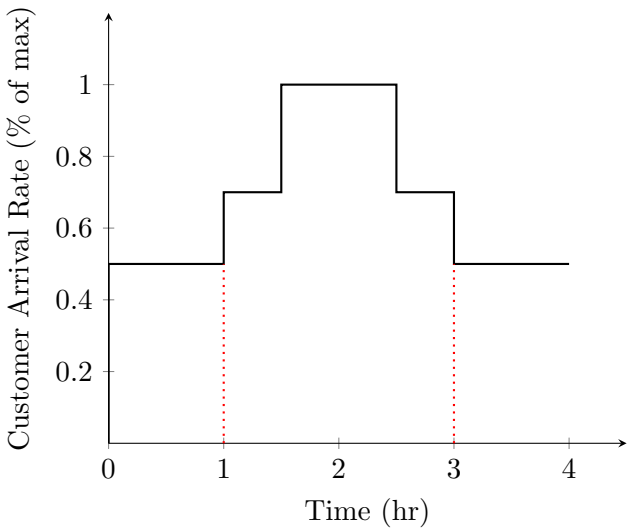


Figure 3: Defining time zones between which different auction parameter values are used, based on changing customer arrival rates throughout the day.

Test problems in both Auburn and Buffalo considered sets of real restaurants which are actively participating in online food delivery services (14 and 34 for the two cities, respectively). For each restaurant, their mean food preparation times were also extracted from one delivery service's website. Based on these values, as well as the size of each city, we used values of 45 minutes and 60 minutes as the fixed delivery windows in the two cities, respectively. Statistics on the tailored delivery windows which were eventually offered to customers across all test problems are presented in Table 5 (in an hours:minutes:seconds format).

Table 5: Dynamic Delivery Window Statistics

	Auburn	**Buffalo**
mean	0:47:38	1:03:20
stdev	0:4:42	0:23:14
min	0:34:27	0:33:52
max	1:02:55	1:41:47

Prior to formal experimentation, preliminary tests were conducted to determine appropriate values for a number of parameters. For example, the results in Table 6 are average values taken over a set of five problems (a subset of problems generated for our full factorial experimental sequence), which were each run using all three heuristic versions. As the value assigned to the penalty term was increased, the average earliness to customers increased as well (note that negative earliness simply implies lateness). However, the minimum earliness (a.k.a. maximum lateness) for a food

item was lowest when $p = 1$. Since this case demonstrated the strongest worst-case performance, and it's average earliness was still positive, the setting $p = 1$ was used for all subsequent testing. This particular value has the effect of causing each unit second of lateness to be double counted in the objective function. Note, however, that a system operator is free to use any penalty value they prefer.

Table 6: Average values for earliness to customers and the maximum pickup-to-delivery times from preliminary tests used to tune penalty parameter p in objective function (1).

Metric	$p = 0$	$p = 0.5$	$p = 1$	$p = 2$	$p = 5$
Earliness (s)	-81	-4	57	92	103
Min Earliness	-1,781	-1,792	-1,521	-1,623	-1,624

Similar preliminary tests were run to establish reasonably good values for the θ, ϕ, and χ thresholds discussed in Section 4.2.3. Note that myopic heuristic does not require any values to be set for these parameters, as it does not consider the equity or dispersion metric at any time.

For the proactive heuristic, values were assigned to these parameters such that equity and dispersion were more strongly considered during periods of high demand. Namely, between hours 1 and 3 in the simulation, the proactive heuristic used values $\theta = 25\%$, $\phi = 25\%$, and $\chi = 0\%$. During the slower periods at the beginning and end of each simulation, values $\theta = 15\%$, $\phi = 25\%$, and $\chi = 10\%$ were used by the proactive heuristic. These values make it less likely that an alternate scenario will be swapped in for the incumbent scenario based on equity and dispersion values during the bid selection process. In other words, the present receives more emphasis than the future during slower periods. A third heuristic version, referred to as the "mixed" heuristic, was also introduced. Like its name implies, this heuristic operates like the myopic heuristic at some times (during the periods of low demand), and like the proactive heuristic at others (during periods of high demand).

All of the values that were assigned for the θ, ϕ, and χ thresholds were tuned such that they generated sufficiently different solutions from the myopic heuristic (otherwise, a comparison between the different heuristics would not be fruitful). Formal testing on how to dynamically vary these values throughout the day is a key area for future research. More generally, it should be noted that the goal of our analysis is to provide insight into the tradeoffs between different managerial policies. We do not make a blanket recommendation for any one policy or parameter value over another.

5.3.1 Heuristic vs. Mathematical Model

In order to assess the performance of the VFCDP heuristic, several small problems were run for which the entire sets of active couriers and customers were passed as inputs to the VFCDP model each time a re-solve was triggered. Recall that this model is able to solve for the set of routes which are globally optimal in the present. Unlike the heuristic, however, the mathematical model by itself does not consider any future-looking metric.

For even the smallest Auburn test problem, there were several solve triggers for which the full mathematical model was unable to solve optimally within one hour. This did not allow for a fair comparison to the heuristic's performance. As such, several new (and smaller) test problems were generated over the Auburn network, with no more than 20 customers arriving throughout the 4 hour simulation period. Each problem was run with this full mathematical model solution strategy, as well as with each of the three heuristic versions. For each of these four solution strategies, each

problem was also run using all three objective functions.

Table 7 compares the performance of each heuristic to the optimal solutions acheived by solving the VFCDP model over the full set of known customers upon each new arrival. Specifically, Table 7 presents the optimality gaps for each heuristic strategy with respect to average earliness to customers. For these small test problems, both the mixed and proactive heuristics' average earliness falls within 20% of optimal.

Table 7: Optimality gap for each of the heuristic versions, with respect to average earliness to customers.

Solution Strategy	**Optimality Gap**
Myopic	21.13%
Mixed	16.58%
Proactive	19.15%

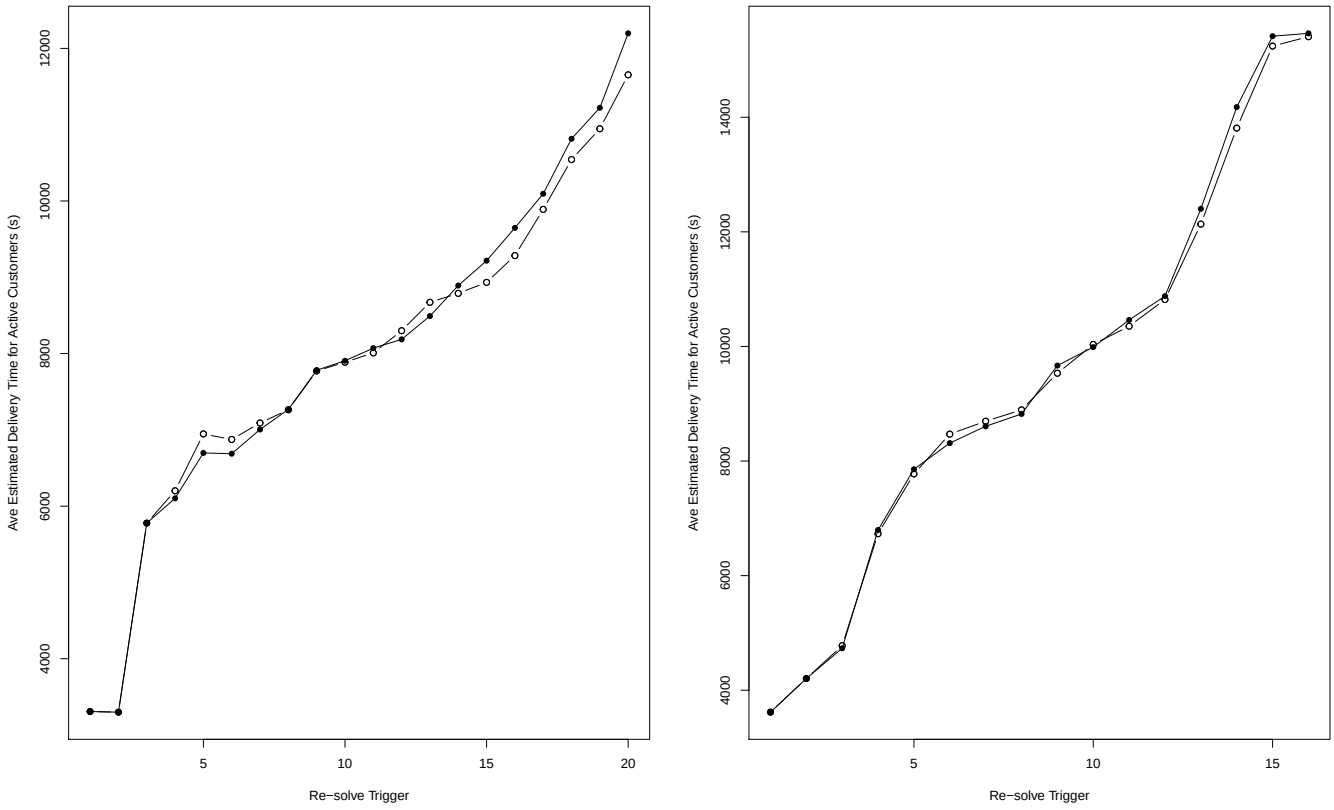
While the average performance is indeed best when using the full mathematical model, Figure 4 tells a more detailed story. After each re-solve, the average expected delivery time among all active customers was calculated. This allows the system's performance to be examined throughout the day. In Figure 4, these values are plotted for two different test problems. Solid lines represent the values from the full mathematical model, while dotted lines represent the proactive heuristic.

In general, the proactive heuristic and the full mathematical model perform similarly throughout the entire day. Upon closer inspection, one can see that the full mathematical model slightly outperforms the heuristic (it has lower expected delivery times) early on in the day. In the latter half of the simulations, however, the heuristic outperforms the full mathematical model. This is made possible by the heuristic's use of the equity and dispersion metrics. In this way, Figure 4 highlights the value of accepting sub-optimal solutions in a dynamic setting in order to prepare for future demand.

5.3.2 Full Factorial Analysis

Results from the full factorial experimental design were analyzed with respect to objective-heuristic pairs. Each of the nine possible pairs is represented by a number-letter abbreviation. The number represents the objective function which was used, while the letter represents the heuristic version which was used ("Y" for the myopic heuristic, "P" for the proactive heuristic, and "M" for the mixed version). As an example, results for Buffalo under the abbreviation "(26)Y" correspond to the set of 12 test problems which were run in Buffalo using objective function (26) and the myopic heuristic.

Figure 5 presents boxplots for the earliness to customers under each of the nine objective-heuristic combinations. This figure demonstrates that the earliness-based objective function, (1), is actually outperformed with respect to earliness by the two objective functions which relate to item freshness, (25) and (26). This fact can be attributed to several principles. First, objective function (1) includes an additional penalty on lateness, and is therefore not a strict measure of earliness. Furthermore, we have seen from preliminary test results that the performance of objective function (1) is clearly tied to the value assigned to penalty parameter, p (recall that tradeoffs exist with respect to average earliness and maximum lateness, and that a fairly conservative approach was taken when setting $p = 1$). Finally, because the system is dynamic, a solution which is optimal under one objective function in the present may prove to be an inferior choice in retrospect. That is, a solution which maximizes average earliness right now may leave couriers out of position to serve the next arrival. In this way, the item-based objective functions (25) and (26) may provide



(a) Problem A

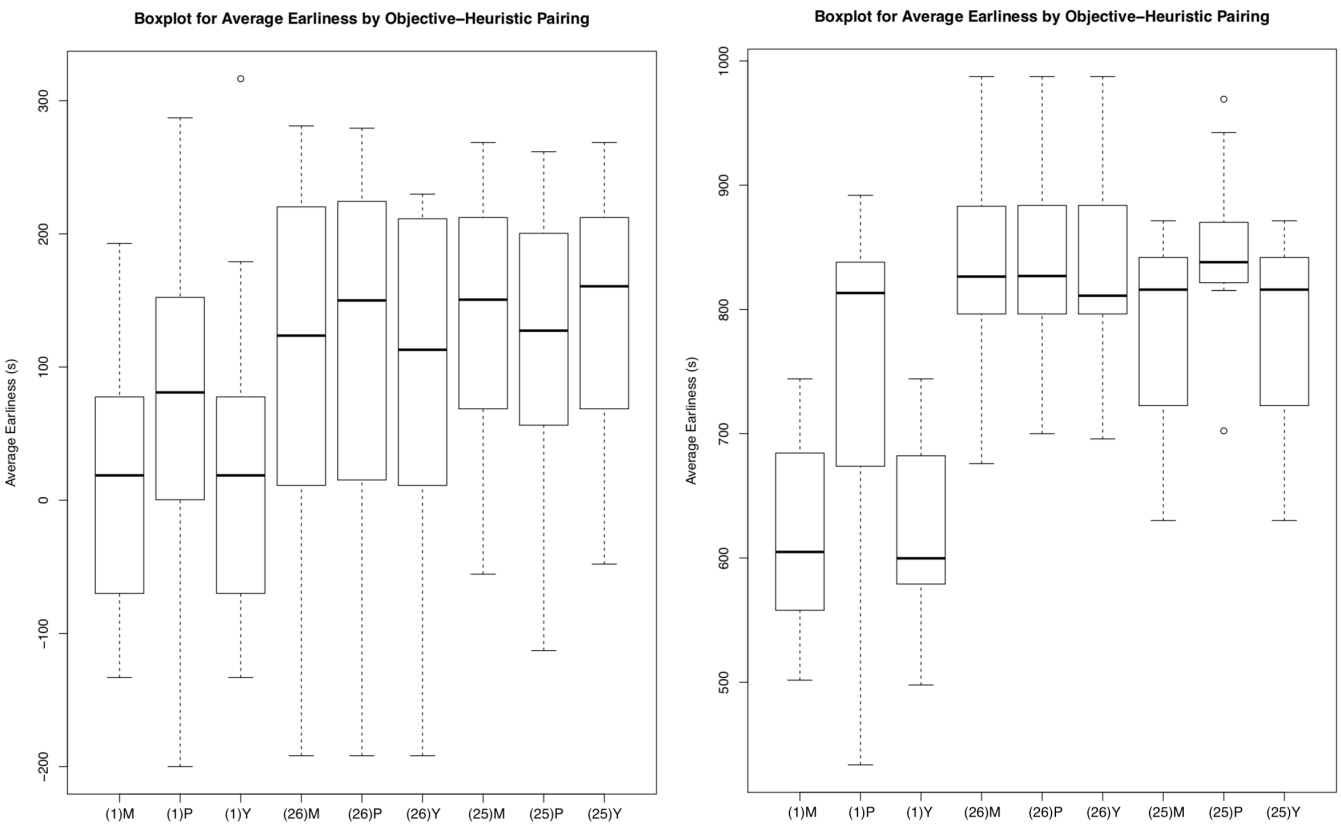
(b) Problem B

Figure 4: Average expected delivery time among active customers after each re-solve, for two separate test problems. Solid lines represent results from the full mathematical model, while dotted lines represent results from the proactive heuristic.

more flexibility than the customer-based objective function (1) for a system which is constantly evolving.

Strong performances by objective functions (25) and (26) can be seen on a number of other metrics as well. For example, the maximum ready-to-delivery which was recorded under objective function (1) was 54 minutes 42 seconds, as compared to 46 minutes 4 seconds with function (25). Similarly, the maximum pickup-to-delivery time recorded under function (1) was and 49 minutes 1 second, as compared to 40 minutes 43 seconds with function (26). The poor showing by function (1) on these time-based metrics may be due to the fact that, by adding a penalty term for lateness, it is the only objective function which does not purely measure time.

Figure 5 further reveals that, given an objective function, the proactive heuristic will generally result in the best mean performance with respect to earliness. In several cases, however, the proactive heuristic also results in the largest variance. The same trend appears in Figure 6, which provides boxplots for the number of late deliveries which are made under the various objective-heuristic combinations. In general, then, the equity and dispersion metrics are effective at preparing the system to handle future demand, and have been shown to improve the mean customer expe-



(a) Auburn

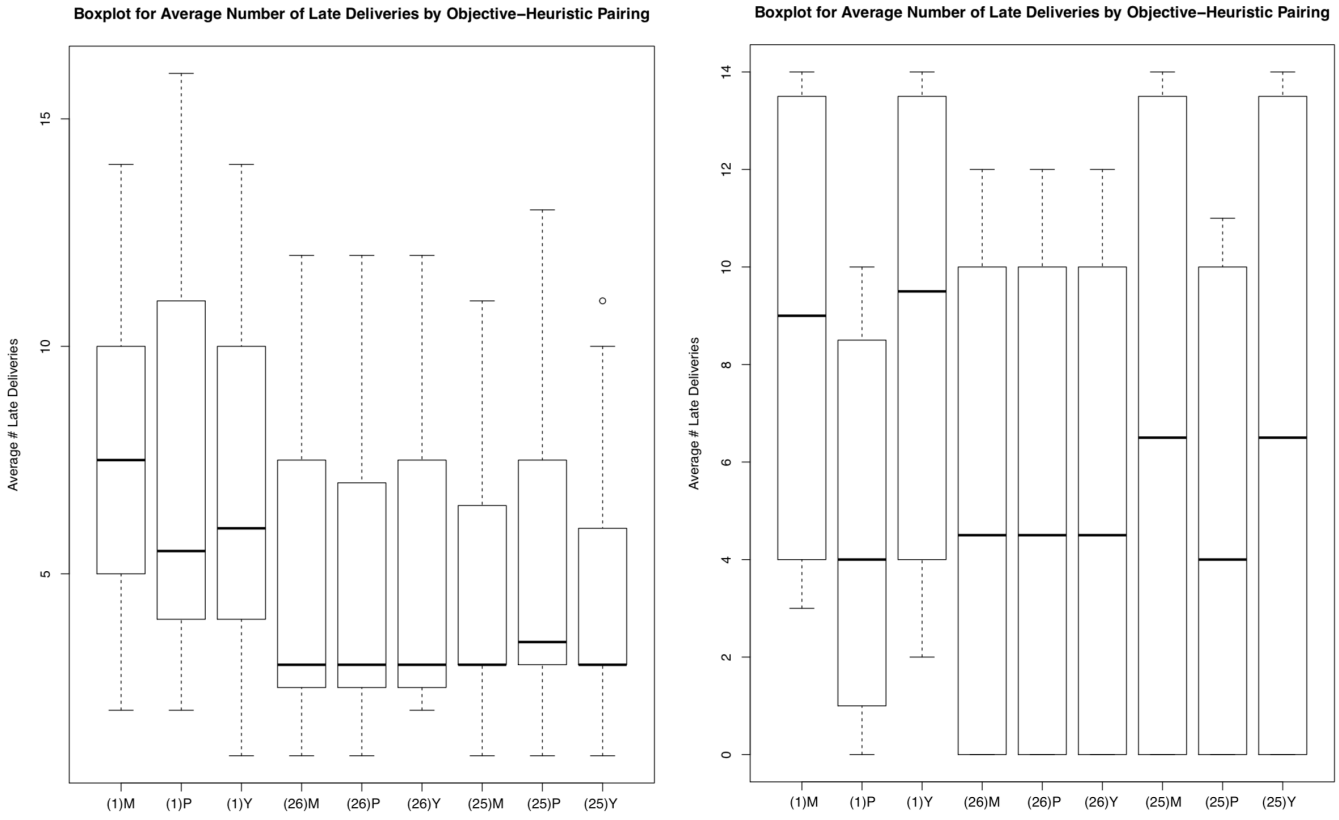
(b) Buffalo

Figure 5: Boxplots for the earliness to customers, by objective-heuristic pair

rience. However, the high-variance nature of this approach may be less appealing to business operators who favor a more conservative approach to customer service.

Although the focus of this paper is on improving the overall customer experience in a food delivery service, it is worth also examining these policies from the perspective of operational cost. Figure 7 presents boxplots for the number of miles traveled by the courier fleets under the various objective-heuristic combinations. It is difficult to claim that any one objective or heuristic has clearly separated itself from the rest in Auburn. Due to the increased problem size (from the road network to the number of customers), the system's performance varies much more widely across the different objective-heuristic combinations in Buffalo. More specifically, cases using the proactive heuristic result in both the smallest mean and the lowest variance with respect to the number of miles traveled by the courier fleet in Buffalo. Once again, then, the future-looking equity and dispersion metrics have yielded demonstrable improvements in the efficiency of a dynamic food delivery system.

In addition to the objective-heuristic combinations, we may examine the results with respect to the type of delivery window which is offered to customers. Table 8 presents the mean values for earliness, the number of late deliveries, and the number of miles traveled in Auburn and Buffalo by the delivery window type. The average customer perceives an earlier delivery in Auburn when using the tailored delivery windows, while the inverse is true in Buffalo. However, both delivery window policies perform reasonably well in each city. Similarly, the number of miles traveled under each



(a) Auburn

(b) Buffalo

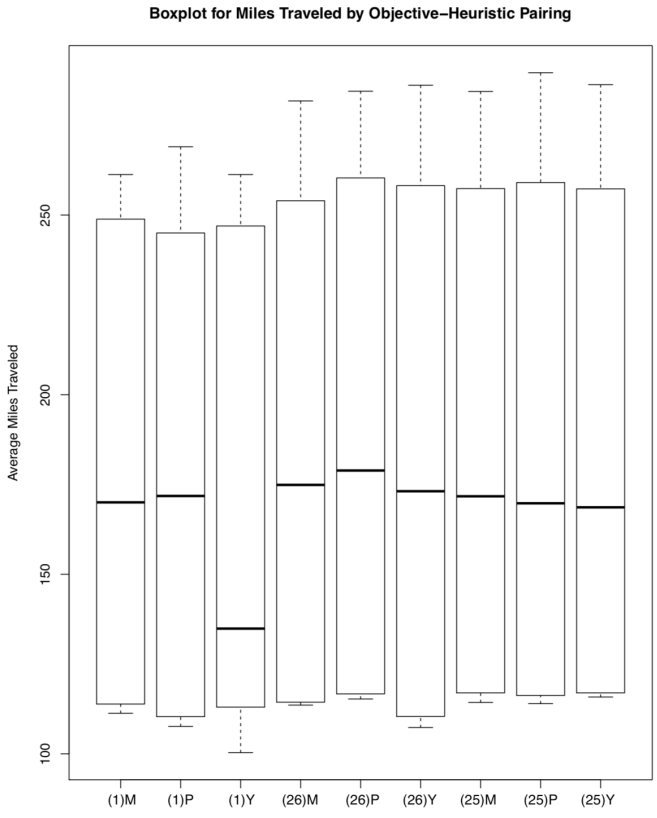
Figure 6: Boxplots for the number of late deliveries during a simulation, by objective-heuristic pair

policy are nearly identical in both cities. Meanwhile, the number of customers who experience late deliveries is lower when using tailored delivery windows in both Auburn and Buffalo, implying that the estimated delivery times were closer to the realized delivery times under this policy. The key takeaway, then, is that an operator can afford to offer customers accurate delivery time estimates without sacrificing system efficiency.

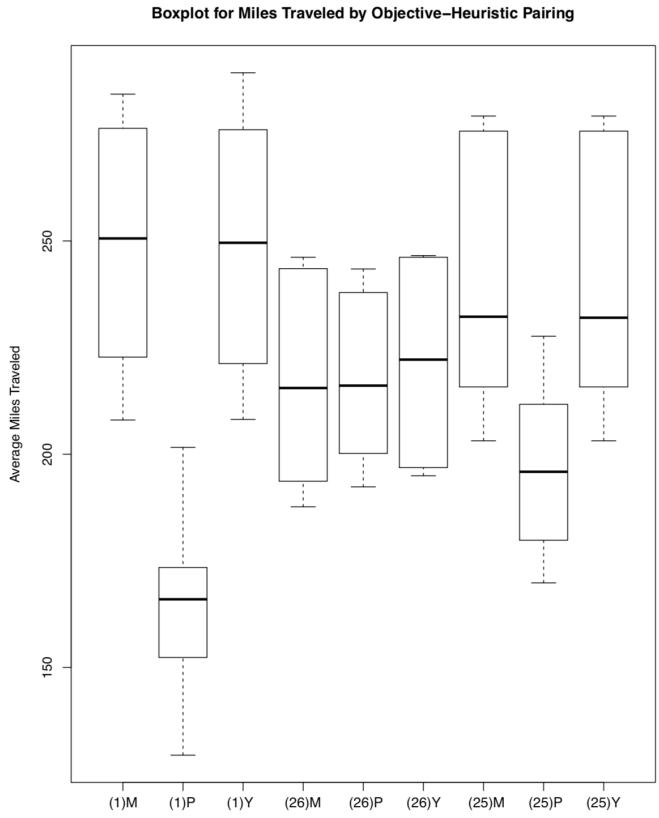
Table 8: Average values for the earliness, number of late deliveries, and number of miles traveled in Auburn and Buffalo, by delivery window type.

	Auburn		**Buffalo**	
Metric	**Fixed**	**Tailored**	**Fixed**	**Tailored**
Earliness (s)	16.41	158.50	789.09	742.97
Late Deliveries	6.96	4.67	11.39	1.04
Miles Traveled	184.76	179.17	222.17	221.15

Finally, the primary motivation for developing a heuristic was the ability to solve our proposed problem in a practical setting. This fact necessitates a discussion on solution times. Table 9 presents



(a) Auburn



(b) Buffalo

Figure 7: Boxplots for the number of miles traveled by the courier fleet during a simulation, by objective-heuristic pair

the average auction time in each city, where the term "auction time" refers to the total time between the arrival of a new customer and the assignment of updated courier routes. The Auburn problems demonstrate an average auction time of 2 seconds, while the much larger Buffalo problems yield an average auction time of only 15.70 seconds. In both cities, the average solution times are well below the mean customer interarrival times, and are sufficiently fast for implementation in a dynamic setting.

Table 9: Average auction times by city, where "auction time" refers to the total time between the arrival of a new customer and the assignment of updated routes to couriers.

City	Mean Auction Time (s)
Auburn	1.53
Buffalo	15.70

5.3.3 Split vs. Non-split Deliveries

One of this paper's contributions is the introduction of a model which allows customers of a food delivery service to order items from multiple restaurants, and to receive those items in a single delivery. An additional contribution is the fact that this same heuristic can also handle the case where customers receive split deliveries (as described in Section 5.2). The latter of these two facts has allowed us to compare the split and non-split cases directly.

In Table 10, average values for several metrics are presented for both the split and non-split delivery policies in Auburn and Buffalo. All of the results therein are intuitive. For example, when multiple items are picked up for a customer before making a single delivery operation, the earliest pickups will spend some amount of time with the courier en route to the later pickups. Similarly, the latest pickups will spend some amount of time waiting at a restaurant while a courier makes the preceding pickups. As such, both the average pickup- and ready-to-delivery times will be larger with non-split deliveries than with split deliveries (where all items in an order can be picked up and delivered by separate couriers). Finally, by making a sequence of pickups before a single delivery (as opposed to one delivery per every pickup), the number of miles traveled by the courier fleet is greatly reduced in both Auburn and Buffalo.

Table 10: Split vs. Nonsplit Deliveries

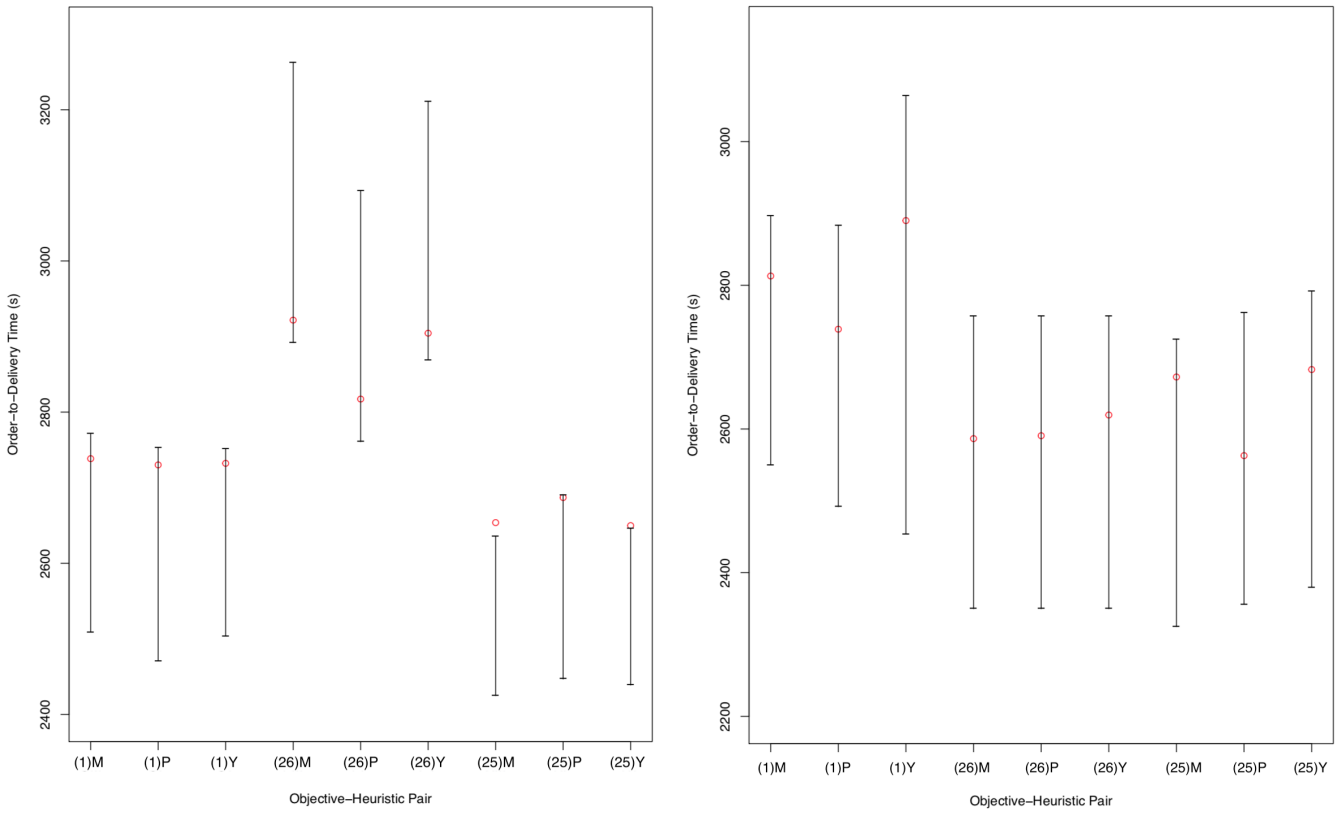
	Auburn		**Buffalo**	
Metric	**Split**	**Non-split**	**Split**	**Non-split**
R-to-D (s)	1,077	1,301	410	901
P-to-D (s)	818	890	196	566
Miles Traveled	371	263	240	201

When examining the split and non-split delivery results at a more granular level, a new insight is revealed. For each objective-heuristic combination in Figure 8, a bar is drawn between the average order-to-delivery times for the first item delivered to a customer and the last item delivered to a customer in the split delivery case. By contrast, the red circle marks the average order-to-delivery time with non-split deliveries (when all items arrive to a customer at the same time), under that same objective-heuristic pair. This figure yields an important result: A non-split delivery policy can, on average, deliver all of a customer's items no later than the time at which the latest item would have arrived in the split deliveries case. It does this while also avoiding any waiting time for the customer between deliveries, and while reducing the number of miles traveled by a courier fleet throughout the day.

6 Summary and Future Research Directions

The recent expansion of some popular courier services into the food delivery business (e.g. UberEATS) reflects the rising popularity of food delivery services. Furthermore, as happened with e-commerce, consumers are likely to demand increasing amounts of immediacy and convenience as this industry continues to grow. The analyses provided in this paper therefore stand to inform a new generation of the food delivery industry.

In existing food delivery service models, any customer who wishes to order from more than one restaurant must place multiple separate orders. The VFCDP is capable of handling multiple



(a) Auburn

(b) Buffalo

Figure 8: Order-to-delivery times for the split and non-split delivery cases. Bars are drawn between the average order-to-delivery times for the first item delivered to a customer and the last item delivered to a customer in the split delivery case. Red circles mark the average delivery time with non-split deliveries case.

restaurants per customer with either a split or non-split delivery policy. By saving the amount of time that items spend either in transit with a courier or waiting to be picked up at a restaurant, the split delivery policy has proven to improve the average freshness of the food being delivered to customers.

The non-split policy has also proven valuable in its own right. Experimental results have demonstrated that such a policy can save a non-trivial percentage of operational costs. The non-split delivery policy exhibited approximately 29% and 16% reductions in the number of miles traveled by a courier fleet in Auburn AL and Buffalo, NY, respectively. Moreover, the non-split delivery policy is capable of delivering all of the items a customer has ordered no later than that same customer would have received the last of their items in the split delivery case. This may even allow a system operator to notify a customer how long they may be expected to wait between deliveries when that customer requests split-delivery service.

Test results also demonstrate that making use of the equity and dispersion metrics can improve the systems' preparedness for future demand. In both Auburn and Buffalo, the proactive auctioneer (which considers equity and dispersion during both peak and non-peak demand hours) has demonstrated superior performance with respect to the earliness of deliveries, the number of

customers which experience late deliveries, and the number of miles traveled by a courier fleet.

The value of the proactive heuristic is also apparent when comparing the heuristic against the full mathematical model for the VFCDP. Figure 4 has shown that, at the cost of accepting sub-optimal solutions early in the day, the heuristic successfully prepares the system to better handle future customers. Moreover, the average performance of all heuristic versions are reasonably good when compared to the performance of the full mathematical model. Given that the heuristic has also proven to be sufficiently fast for implementation in a dynamic setting, the validity of the VFCDP heuristic has been demonstrated.

The two freshness-based objective functions have proven effective at delivering food to customers which is both fresh and on time. In fact, this happens without either function taking promised delivery times into consideration. These two objective functions track values at the item level, as opposed to the customer level (like the earliness-based objective function). Additionally, they are pure measurements of time, without any penalty terms being added. All of this likely played a role in their strong showing, and may be taken into consideration by anyone wishing to operate a system like the one defined by the VFCDP. With respect to the delivery windows themselves, experimental results show that a system operator can afford to offer customers realistic delivery time estimates without sacrificing system efficiency.

Although many preliminary tests were run to establish the θ, ϕ, ψ, and χ thresholds used in the proactive (and mixed) heuristic, it is believed that these parameters present a major area for future research. A system which is capable of varying these parameter values in real time may yield improved system performance. On a broader scale, alternate methods for future consideration could also be tested. Additionally, the development of a route-based formulation using column generation (as opposed to the existing formulation, which is edge-based) for the VFCDP may improve the speed at which solutions can be generated by the mathematical model. Obtaining such a formulation may allow for our heuristic's performance to be compared to a theoretical "global" optimal solution. That is, a less computationally complex model may be able to solve an instance which encompasses an entire day's customer stream. Such a solution would represent the best attainable system performance (although no manager would ever have access to such information in a practical setting). Other future research directions may include the introduction of "handoffs" between couriers to swap items or customers altogether, and distance-based restrictions on the restaurants available to customers.

7 Acknowledgments

The authors are grateful to the editor and two anonymous referees. Their feedback and suggestions greatly helped to strengthen the paper.

References

A. Attanasio, J.-F. Cordeau, G. Ghiani, and G. Laporte. Parallel tabu search heuristics for the dynamic multi-vehicle dial-a-ride problem. *Parallel Computing*, 30(3):377–387, 2004.

R. Batta, M. Lejeune, and S. Prasad. Public facility location using dispersion, population, and equity criteria. *European Journal of Operational Research*, 234(3):819–829, 2014.

M. Battarra, J.-F. Cordeau, and M. Iori. Pickup-and-delivery problems for goods transportation. In Paolo Toth and Daniele Vigo, editors, *Vehicle routing: problems, methods, and applications*, chapter 6, pages 161–191. SIAM, 2014.

T. Bektas. The multiple traveling salesman problem: an overview of formulations and solution procedures. *Omega*, 34(3):209–219, 2006.

T. Bektas, P.P. Repoussis, and C.D. Tarantilis. *Vehicle Routing: Problems, Methods, and Applications, Second Edition*, chapter Dynamic Vehicle Routing Problems, pages 299–350. SIAM, 2 edition, 2014.

E. Benavent, M. Landete, E. Mota, and G. Tirado. The multiple vehicle pickup and delivery problem with LIFO constraints. *European Journal of Operational Research*, 243(3):752–762, 2015.

G. Berbeglia, J.-F. Cordeau, I. Gribkovskaia, and G. Laporte. Static pickup and delivery problems: a classification scheme and survey. *Top*, 15(1):1–31, 2007.

G. Berbeglia, J.-F. Cordeau, and G. Laporte. Dynamic pickup and delivery problems. *European Journal of Operational Research*, 202(1):8–15, 2010.

M. Bögl, K.F. Doerner, and S.N. Parragh. The school bus routing and scheduling problem with transfers. *Networks*, 65(2):180–203, 2015.

H. Caceres, R. Batta, and Q. He. School bus routing with stochastic demand and duration constraints. *Transportation science*, 51(4):1349–1364, 2017.

Z.L. Chen and H. Xu. Dynamic column generation for dynamic vehicle routing with time windows. *Transportation Science*, 40(1):74–88, 2006.

M. Cherkesly, G. Desaulniers, and G. Laporte. A population-based metaheuristic for the pickup and delivery problem with time windows and LIFO loading. *Computers & Operations Research*, 62:23–35, 2015.

M. Cherkesly, G. Desaulniers, S. Irnich, and G. Laporte. Branch-price-and-cut algorithms for the pickup and delivery problem with time windows and multiple stacks. *European Journal of Operational Research*, 250(3):782–793, 2016.

J.F. Cordeau and G. Laporte. The dial-a-ride problem: models and algorithms. *Annals of Operations Research*, 153(1):29–46, 2007.

T.G. Crainic, P.K. Nguyen, and M. Toulouse. Synchronized multi-trip multi-traffic pickup & delivery in city logistics. *Transportation Research Procedia*, 12:26–39, 2016.

M. Diana and M.M. Dessouky. A new regret insertion heuristic for solving large-scale dial-a-ride problems with time windows. *Transportation Research Part B: Methodological*, 38(6):539–557, 2004.

K.F. Doerner and J.J. Salazar González. Pickup-and-delivery problems for people transportation. In Paolo Toth and Daniele Vigo, editors, *Vehicle routing: problems, methods, and applications*, chapter 7, pages 161–191. SIAM, 2014.

I. Dumitrescu, S. Ropke, J.-F. Cordeau, and G. Laporte. The traveling salesman problem with pickup and delivery: polyhedral results and a branch-and-cut algorithm. *Mathematical Programming*, 121(2): 269–305, 2010.

B. Eksioglu, A.V. Vural, and A. Reisman. The vehicle routing problem: A taxonomic review. *Computers & Industrial Engineering*, 57(4):1472–1483, 2009.

B. Fleischmann, S. Gnutzmann, and E. Sandvoß. Dynamic vehicle routing based on online traffic information. *Transportation Science*, 38(4):420–433, 2004.

S. Gelareh, R. Merzouki, K. McGinley, and R. Murray. Scheduling of intelligent and autonomous vehicles under pairing/unpairing collaboration strategy in container terminals. *Transportation Research Part C: Emerging Technologies*, 33:1–21, 2013.

A. Goel and V. Gruhn. A general vehicle routing problem. *European Journal of Operational Research*, 191 (3):650–660, 2008.

B.L. Golden, S. Raghavan, and E.A. Wasil, editors. *The Vehicle Routing Problem: Latest Advances and New Challenges*, volume 43 of *Operations Research/Computer Science Interfaces Series*. Springer, 2008.

S. Jain and P. Van Hentenryck. Large neighborhood search for dial-a-ride problems. In *International Conference on Principles and Practice of Constraint Programming*, pages 400–413. Springer, 2011.

Y. Kergosien, C. Lenté, and J.-C. Billau. Home health care problem: An extended multiple traveling salesman problem. In *4th Multidisciplinary International Conference on Scheduling: Theory and Applications (MISTA '09), Dublin (Ireland)*, pages 10–12, 2009.

C.K.Y. Lin. A vehicle routing problem with pickup and delivery time windows, and coordination of transportable resources. *Computers & Operations Research*, 38(11):1596–1609, 2011.

R. Liu, X. Xie, V. Augusto, and C. Rodriguez. Heuristic algorithms for a vehicle routing problem with simultaneous delivery and pickup and time windows in home health care. *European Journal of Operational Research*, 230(3):475–486, 2013.

M. Maalouf, C.A. MacKenzie, S. Radakrishnan, and M. Court. A new fuzzy logic approach to capacitated dynamic dial-a-ride problem. *Fuzzy Sets and Systems*, 255:30–40, 2014.

MapQuest. Developer Network. URL `https://developer.mapquest.com/`.

R. Montemanni, L.M. Gambardella, A.E. Rizzoli, and A.V. Donati. Ant colony system for a dynamic vehicle routing problem. *Journal of Combinatorial Optimization*, 10(4):327–343, 2005.

R.M. Newton and W.H. Thomas. Design of school bus routes by computer. *Socio-Economic Planning Sciences*, 3(1):75–85, 1969.

P. Paolo and D. Vigo. *Vehicle routing: problems, methods, and applications*, volume 18. Siam, 2014.

S.N. Parragh, K.F. Doerner, and R.F. Hartl. A survey on pickup and delivery problems. *Journal für Betriebswirtschaft*, 58(1):21–51, 2008.

S.N. Parragh, K.F. Doerner, R.F. Hartl, and X. Gandibleux. A heuristic two-phase solution approach for the multi-objective dial-a-ride problem. *Networks*, 54(4):227–242, 2009.

V. Pillac, C. Guéret, and A.L. Medaglia. An event-driven optimization framework for dynamic vehicle routing. *Decision Support Systems*, 54(1):414–423, 2012.

V. Pillac, M. Gendreau, C. Guéret, and A.L. Medaglia. A review of dynamic vehicle routing problems. *European Journal of Operational Research*, 225(1):1–11, 2013.

H.N. Psaraftis. A dynamic programming solution to the single vehicle many-to-many immediate request dial-a-ride problem. *Transportation Science*, 14(2):130–154, 1980.

H.N. Psaraftis. Dynamic vehicle routing: Status and prospects. *Annals of Operations Research*, 61(1): 143–164, 1995.

D. Reyes, A. Erera, M. Savelsbergh, S. Sahasrabudhe, and R. O'Neil. The meal delivery routing problem. *Optimization Online*, 2018.

U. Ritzinger, J. Puchinger, and R.F. Hartl. A survey on dynamic and stochastic vehicle routing problems. *International Journal of Production Research*, 54(1):215–231, 2016.

M. Savelsbergh and M. Sol. DRIVE: Dynamic Routing of Independent Vehicles. *Operations Research*, 46 (4):474–490, 1998.

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

M. Schyns. An ant colony system for responsive dynamic vehicle routing. *European Journal of Operational Research*, 245(3):704–718, 2015.

M.W. Ulmer, B.W. Thomas, A.M. Campbell, and N. Woyak. The restaurant meal delivery problem: Dynamic pick-up and delivery with deadlines and random ready times. 2017.

E. Urra, C. Cubillos, and D. Cabrera-Paniagua. A hyperheuristic for the dial-a-ride problem with time windows. *Mathematical Problems in Engineering*, 2015, 2015.

T. van Woensel, L. Kerbache, H. Peremans, and N. Vandaele. Vehicle routing with dynamic travel times: A queueing approach. *European Journal of Operational Research*, 186(3):990–1007, 2008.

Z. Xiang, C. Chu, and H. Chen. The study of a dynamic dial-a-ride problem under time-dependent and stochastic environments. *European Journal of Operational Research*, 185(2):534–551, 2008.

B. Yildiz and M. Savelsbergh. Provably high-quality solutions for the meal delivery routing problem. *Georgia Institute of Technology*, 2017.