Supply and Demand Steering in On-demand Meal Delivery

A.Arslan [a], S. Sharif Azadeh[b] and M. Savelsbergh[c],Y.Maknoon [a*]

[a] Faculty of Technology, Policy, and Management, Delft University of Technology
M.Y.Maknoon@tudelft.nl, A.M.Arslan@tudelft.nl
[b] Department of Transport & Planning, Delft University of Technology
S.SharifAzadeh@tudelft.nl
[c] H. Milton Stewart School of Industrial and Systems Engineering, Georgia Institute of Technology martin.savelsbergh@isye.gatech.edu

January 13, 2022

Keywords: on-demand meal delivery, sequential decisions, courier guidance, dynamic delivery promises

1 INTRODUCTION

On-demand logistics has been evolving rapidly due to the ever-increasing consumer desire for faster services. Online restaurant aggregators, or meal delivery platforms, are the pioneers of this development. Meal delivery platforms have disrupted restaurant-operated delivery systems by providing access to a number of restaurants in a region and offering high customer service, e.g., fast delivery, user-friendliness, availability, etc. The skyrocketing growth of the sector reflects the appeal of the business model (Singh, 2019).

Managing an on-demand meal delivery service involves multiple interrelated tasks, e.g., recruiting couriers (strategic) and coordinating courier activities (tactical and operational). Many of the meal delivery platforms rely on ad-hoc crowdsourced delivery capacity, i.e., rely on individuals willing to make deliveries in their free time (Arslan *et al.*, 2019, Ulmer & Savelsbergh, 2020, Dayarian & Savelsbergh, 2020). Some platforms manage temporal and spatial imbalances between delivery capacity (supply) and order placements (demand) by adjusting compensation to attract or deter couriers (e.g., UberEats and Deliveroo). However, because customers expect to pay only a small fee for delivery, adjustments in courier compensation cannot easily be passed on the customers. Furthermore, regulatory changes related to crowdsourced delivery capacity, e.g., minimum payment guarantees, provision of benefits, have led platforms to switch to the use of advance scheduled couriers (e.g., JustEat and Takeaway.com). We focus on the latter model, in which the couriers available to make deliveries during an operating period are scheduled in advanced.

A major challenge for platforms is to effectively handle the spatial and temporal demand variations that naturally occur from day to day and from hour to hour. Assuming that frequently making dispatch decisions (i.e., repeatedly solving courier-order assignments) will achieve desired on-time performance and courier utilization is unrealistic; a "wild goose chase" phenomenon is commonly observed (Castillo *et al.*, 2017). When a platform cannot dynamically adjust delivery capacity and cannot dynamically adjust delivery fees (and influence demand), other tools are necessary to effectively manage the highly stochastic meal delivery demand. For that purpose, we introduce *demand steering*, i.e., adjusting delivery time promises, and *supply steering*, i.e., adjusting courier-order assignments, to increase customers' service quality, measured by delivery time promise delays and delivery time promise.

Supply and demand steering mitigates the shortcomings of myopic dispatching polices by (i) proactively directing couriers towards areas requiring additional delivery capacity and (ii) temporally lowering service by displaying increased delivery time promises in certain areas.

In this paper, we investigate and answer the following questions: (i) Under what circumstances does fleet steering provide advantages? (ii) Under what circumstances does demand steering provide advantages? (iii) Under what circumstances is it beneficial to use both supply and demand steering? To explore these questions, we introduce the meal-delivery problem with advanced scheduled couriers (MDP-ASC) and two related time-expanded networks that differ in granularity in both time and space. The finer network represents orders and couriers using actual location and time information, and is used to determine dispatching decisions. The coarser network represents orders and couriers at an aggregated level, across both time and space, and is used to determine steering actions.

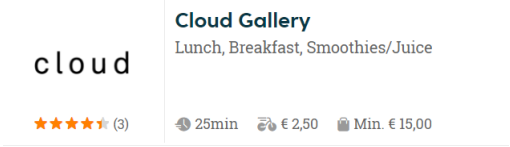
As in many other meal delivery settings, in the MDP-ASC a set of couriers with known working periods, also referred to as blocks, serve unknown orders that arrive during the operating period. A customer expects to receive his meal at or before the delivery time promise displayed at the time the order is placed. The platform assigns orders, more specifically the delivery tasks associated with orders, to couriers. A delivery task involves the pick up of a meal at a restaurant and the delivery of that meal at a customer's location. We refer to decisions related to the assigning couriers to orders as *dispatching decisions*. Typically, a meal delivery platform relies only on their dispatching decisions to maximize service (e.g., on-time performance) and courier utilization.

Fleet steering seeks to incorporate knowledge about the future state of the system (although uncertain) into the dispatching decisions. That is, it seeks to influence courier - order assignments (including preemptive repositioning decisions) to ensure that delivery capacity is always in the right place at the right time. Demand steering supports the management of delivery capacity by dynamically and temporarily increasing the displayed delivery time promises, which eases the pressure on dispatching decisions as more time is available to deliver future orders. Increasing the delivery time promise allows more on-time deliveries, but it can negatively affect customers' experience (and even customers' choice). The premise underlying demand steering is that customers favor upfront knowledge of delay (an increased delivery time promise) over unexpected delay (a missed delivery time promise). To capture this concept, we introduce a metric called *quality loss*, which accounts for any increase in delivery time promise as well as any lateness of delivery (relative to the delivery time promise).

2 Problem statement

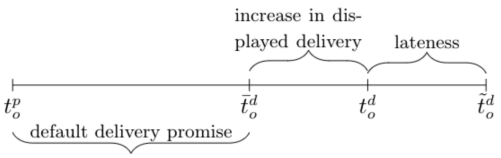
In the MDP-ASC, a set of couriers K with working periods that have been scheduled in advance have to perform delivery tasks associated with unknown set of meal delivery orders O that arrive during a finite operating period $[0, T]$. The service region has a set of restaurants, R, where the meals are prepared. Each arriving order o is characterized by a placement time, $t_o^p \in [0, T]$, a ready time, $t_o^r \in [0, T]$ $(t_o^r \geq t_o^p)$, a pickup location (a restaurant), ℓ_o^p, and a delivery location, ℓ_o^d.

Figure 1: Screenshot of the platform interface. The displayed delivery time promise is 25 minutes.



Each order o also has a delivery time promise t_o^d, which is derived from the displayed delivery time promise at the time of order placement. The displayed delivery time promise for an order is typically a default delivery time promise $\hat{t}_o^d$, which is strategically determined by the platform

Figure 2: Quality loss for order o



and depends on the restaurant, the delivery location, and the time of day, and accounts for meal preparation time, travel time from pickup to delivery location, and market competition. However, the platform can increase the displayed delivery time promise to manage any demand and supply imbalances.

Given an order's delivery time promise of t_o^d, the service quality depends on (i) any **increase** in displayed delivery time promise, i.e., $t_o^d - \hat{t}_o^d$ and (ii) any **lateness** of the delivery (relative to the delivery time promise), i.e., $\max(0, \tilde{t}_o^d - t_o^d)$, where $\tilde{t}_o^d$ is the realized delivery time of order o. We define the *quality loss* $Q(\tilde{t}_o^d, t_o^d)$ of order o to be

$$Q(\tilde{t}_o^d, t_o^d) = \beta_1(t_o^d - \hat{t}_o^d) + \beta_2(\tilde{t}_o^d - t_o^d \mid t_o^d = \hat{t}_o^d) + \alpha\beta_2(\tilde{t}_o^d - t_o^d \mid t_o^d > \hat{t}_o^d), \tag{1}$$

where $\beta_1 : \mathbb{R} \mapsto \mathbb{R}^+$ is a function that computes the quality loss due to an increase of a delivery time promise, $\beta_2 : \mathbb{R} \mapsto \mathbb{R}^+$ is a function that computes the quality loss due to lateness of a delivery, and $\alpha \geq 1$ captures the fact that a late delivery is perceived as worse when the delivery time promise has already been increased.

Each courier k has a start time $\underline{t}_k$, a start location $\underline{\ell}_k$, an end time $\bar{t}_k$, and an end location $\bar{\ell}_k$. As courier k is compensated for the period $\bar{t}_k - \underline{t}_k$, we assume that the courier performs all tasks assigned to the courier during that period. (The first delivery task cannot start earlier than $\underline{t}_k$ and later than $\bar{t}_k$.) . We consider three possible tasks: **delivery**, the courier travels to the pickup location to pick up a meal and travels to the delivery location to drop off a meal, **reposition**, the courier travels from the current location to a specified location, and **wait**, the courier remains idle in the current location for a specified period. The platform seeks to minimize the expected total quality loss over the orders arriving during the operating period:

$$\min \ \mathbb{E}\left[\sum_{o \in \mathcal{O}} Q(\hat{t}_o^d, t_o^d)\right]. \tag{2}$$

To do so, the platform determines (i) which courier to assign to each order, (ii) whether or not to reposition a courier and, if so, how to reposition the courier, and (iii) whether or not to temporarily increase the displayed delivery time promise for an order (increasing the display delivery time promise for an order to infinity is allowed and indicates that no meal can be ordered from the restaurant).

3 Results

We design our numerical experiments with the set of instances derived from the public GrubHub instances, see (Reyes *et al.*, 2018) for detail explanation. We use ten public Grub-hub instances varying in the service area, number of orders, restaurants and couriers. We use the location of restaurants and the orders destination as they are, and the arrival time of the orders.

The proposed steering actions provide a joint supply and demand steering guidance to the dispatching operations in the meal delivery service. We compare this joint dynamic supply and demand steering policy to a series of benchmarks. In total we consider following benchmark policies: *No steering (NS).* The platform carries out courier-order assignments solely based on solving dispatching problem every minute *Fleet steering (FS). Fleet and demand steering (FDS).*

Figure 3: Relative Quality Loss to No Steering, $\frac{\beta_2(\cdot)}{\beta_1(\cdot)} = 4,\ \alpha = 3$

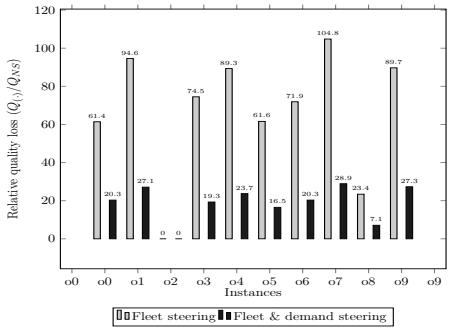


Figure 3 presents the change of the total quality loss when the platform uses the steering actions. The figure shows the relative quality loss to the no steering case *i.e.,* $\frac{Q_{FS}}{Q_{NS}}$ gives the relative quality loss when only fleet steering in use. The result shows that the total quality loss drops drastically with steering actions.

When we look at the average of ten experiments, 51.5% of the total quality loss could be avoidable. It is possible to see the bigger impact in the reduction of late deliveries. With the steering actions, 96.1% of the late deliveries would be delivered on time. On the other side, 31.5% of arriving order will see extended displayed delivery promise.

Table 1: Late deliveries

	Late deliveries %			Ave Lateness (min)		Maximum lateness (min)	
Instances	NS	FS	FDS	NS	FS	NS	FS
0o100t100s1p100	2.0	1.4	0	11.9	11.5	30.8	20.8
1o100t100s1p100	8.9	16.9	0	21.3	11.9	76.3	43.5
2o100t100s1p100	0.3	0.0	0	5.1	0	9.9	0
3o100t100s1p100	0.4	0.3	0	6.3	6.1	15.3	12.4
4o100t100s1p100	13.6	24.3	0	20.9	11.9	53.8	31.5
5o100t100s1p100	18.9	39.0	0	29.2	24.4	133.8	93.9
6o100t100s1p100	19.6	26.0	0	24.3	15.6	91.7	54.9
7o100t100s1p100	7.3	7.3	0	12.4	12.6	39.4	52.5
8o100t100s1p100	1.3	3.9	0	31.6	6.7	148.2	55.6
9o100t100s1p100	2.5	3.4	0	17.6	10.3	59.9	49.2

4 REFERENCES

References

Arslan, Alp M., Agatz, Niels, Kroon, Leo, & Zuidwijk, Rob. 2019. Crowdsourced delivery—a dynamic pickup and delivery problem with ad hoc drivers. *Transportation Science*, **53**(1), 222–235.

Castillo, Juan Camilo, Knoepfle, Dan, & Weyl, Glen. 2017. Surge pricing solves the wild goose chase. *Pages 241–242 of: EC 2017 - Proceedings of the 2017 ACM Conference on Economics and Computation*, vol. 1. New York, NY, USA: Association for Computing Machinery, Inc.

Dayarian, Iman, & Savelsbergh, Martin. 2020. Crowdshipping and same-day delivery: Employing in-store customers to deliver online orders. *Production and Operations Management*, **29**(9), 2153–2174.

Reyes, Damián, Erera, A., Savelsbergh, M., Sahasrabudhe, Sagar, & O'Neil, Ryan J. 2018. The Meal Delivery Routing Problem.

Singh, Sarwant. 2019 (9). *The Soon To Be $200B Online Food Delivery Is Rapidly Changing The Global Food Industry.*

Ulmer, Marlin, & Savelsbergh, Martin. 2020. Workforce Scheduling in the Era of Crowdsourced Delivery. *Transportation Science*, **54**(4), 1113–1133.