End-to-End Trip to Driver Dispatchment Timing of Scheduled Delivery: A General Decision-Making System for Crowdsourced Delivery Platforms

Tao Cao[1*], **Soumita Saha**[2], **Ankit Dhokariya**[1], **Weihang Ren**[1], **Minghui Liu**[1], **Yuan Wang**[1], **Sherry Lijie Wan**[1], **Jing Huang**[1], **Mingang Fu**[1], **Prakhar Mehrotra**[1], **Chittaranjan Tripathy**[1]

[1]Walmart Global Tech, Sunnyvale, CA, USA
[2]Walmart Global Tech India, Bangalore, India
Tao.Cao@walmart.com, Soumita.Saha@walmart.com, Ankit.Dhokariya@walmart.com, Weihang.Ren@walmart.com, minghui.liu@walmart.com, Yuan.Wang@walmart.com, lijie.wan@walmart.com, jing.huang@walmart.com, Mingang.Fu@walmart.com, Prakhar.Mehrotra@walmart.com, ctripathy@walmart.com

Abstract

In the early years, crowdsourcing delivery platforms used to face most of the demand from real-time on-demand requests. Recently, with increasing adoption by ecommerce retailers, we observed a shift of demand pattern from real-time on-demand to scheduled delivery requests. The increasing reliance on crowdsourced drivers for scheduled deliveries presents a new challenge. That is, the timings through the process of dispatching trips to drivers set the foundation of the delivery journey. Despite the critical role of end-to-end dispatch timing decisions, there has been a paucity of holistic approaches to this decision-making process that ensure cost-efficient, on-time deliveries. We present a novel system designed to optimize the end-to-end driver dispatch timing decisions. The system begins by predicting time duration of segments throughout delivery journeys. It then utilizes simulation, stochastic programming, and survival regression modeling to optimally determine arrival time of drivers, start time of driver dispatch, and initialization time of surge pay. The objectives are maximizing dispatch efficiency while meeting promised delivery time for customers. Our novel decision-making system is adaptable to dynamic market conditions through configurable parameters, which are learned automatically via market segmentation and operational constraints. We have implemented and deployed this system on Walmart's proprietary crowdsourced last-mile delivery platform. To evaluate its effectiveness, we conducted comprehensive simulation and experimentation studies. The results indicate significant improvements in dispatch efficiency, by reducing idle time (55% compared to previous solutions) in dispatch process, all while maintaining high rates of on-time deliveries in line with customer promises. Our work demonstrates that the general framework underlying the system has the potential to enhance delivery experiences and efficiency across other crowdsourced delivery platforms, where on-time delivery promise for customers are foundational.

Introduction

With the growing prevalence of online ordering across various sectors, such as retail and food, there is an escalating demand for a flexible and cost-efficient method to meet these needs. Crowdsourced deliveries, which leverage independent drivers to fulfill deliveries whenever and wherever they choose to be online, have emerged as the most widely adopted solution. As consumer preferences continue to evolve and online transactions become increasingly prevalent, the trajectory of crowdsourced last-mile deliveries is set for further expansion, reshaping the landscape of modern commerce (Skiver and Godfrey 2017). The e-commerce retail sector has seen a significant shift in consumer demand patterns. This change, driven by increased adoption of e-commerce platforms, has seen a move away from real-time, on-demand delivery requests towards scheduled deliveries. Scheduled deliveries, which allow customers to pre-define their preferred delivery windows, offer greater convenience and predictability. This trend not only reflects the evolving consumer preferences but also underscores the impact of technological advancements on online retail dynamics.

The primary objective for delivery platforms supporting scheduled deliveries is to ensure timely and cost-effective order fulfillment. The driver dispatch system is the cornerstone of the platform's objective. The system aims to engage the right drivers at the right time and with the right compensation. Timing is a critical aspect of this process. An end-to-end view of the business flow reveals multiple decisions related to timing that need to be made. The first decision is when to initiate the driver dispatch process (referred to as *dispatch start time*). Additionally, dynamic or surge pricing is often used as a real-time incentive to encourage drivers to accept deliveries that are deemed unattractive (Silva and Pedroso 2022), such as those that need to be fulfilled in bad weather. Therefore, a decision also needs to be made regarding when such dynamic pricing should be implemented (referred to as *surge time*). The last decision is when drivers should collect orders at pick-up locations (referred to as *arrival time*).

It's crucial to highlight the complex interconnections of these timing decisions, as they constitute an essential end-to-end decision-making process for scheduled deliveries. Viewing each decision separately, without considering its impacts on or from others, can lead to sub-optimal or even infeasible solutions. For instance, the time when a driver should arrive sets the reference for when to initiate driver dispatch. The

[*]Corresponding Author

goal of the latter is to start dispatch such that drivers can arrive at the suggested arrival time.

Studies in the timing domains for crowdsourced deliveries have been primarily focusing on projections rather than decision-making. Majority of studies predominantly involve on-demand than scheduled deliveries. These studies mainly aim to predict when certain events or milestones will occur, given historical and current information. The underlying problem to solve is usually the prediction of time durations. Numerous techniques have been proposed, implemented, and tested to improve prediction accuracy, ranging from decision tree ensemble models, deep learning models (Araujo and Etemad 2021) (Hu et al. 2022), Markov decision process (Zehtabian, Larsen, and Wøhlk 2022). However, these studies are conducted without any links to driver dispatch timing decisions, which are prerequisites for downstream timing predictions. Some studies (Wang et al. 2018) (Liu, He, and Shen 2020) (Ding et al. 2021) evaluated the importance of estimation in guiding upstream decisioning, but only on driver assignment (how to match drivers with deliveries) than timing decisioning. It is naturally expected that different dispatch timing decisions lead to different downstream time predictions. Moreover, time estimations can and should influence dispatch timing decisions. For instance, knowing the estimated delivery time in advance can guide when drivers should arrive at pickup locations to avoid late deliveries.

While there are numerous studies (Barbosa, Pedroso, and Viana 2023) (Miao et al. 2023) on the dynamic pricing mechanism in crowdsourced deliveries, a significant gap exists in the temporal dimension of dynamic pricing. Specifically, when surge pricing should be implemented. Most studies focus on structuring the pricing mechanism, i.e., determining the price amount and structure. A few studies (Lei et al. 2020) (Tong et al. 2018) (Wang, Wang, and Shuaijie 2019) have briefly touched upon common indicators associated with the timing of surge pricing, such as supply (of drivers) and demand (of deliveries). However, these indicators are only discussed at a high level to ensure the completeness of pricing structures. There is no comprehensive study on how these temporal indicators should be implemented. Furthermore, there is a lack of a cohesive end-to-end view of timing decision-making, as the timing of surge pricing is closely connected with other dispatch timings. This connection is crucial not only for establishing the right compensation mechanism but, more importantly, for ensuring on-time deliveries to customers.

The Spark Driver platform is Walmart's proprietary delivery network, launched in 2018. Crowdsourced drivers can sign up on the Spark Driver App and make deliveries for Walmart and other retailers. The platform delivers millions of customer orders from more than 17,000 pickup locations reaching more than 84% of U.S. households (Chadha 2023). The Spark Driver platform plays a crucial role in supporting our omni-channel strategy, enable Walmart to offer fast and reliable delivery options to customers on hundreds of thousands of items across Walmart and other retailers, present a consistent experience across the digital and physical worlds, and serve customers seamlessly through online, mobile, and in-store interactions. The platform had been relying on a rule-based system to define driver dispatch timing decisions. This system, which lacked consideration for variations in each delivery and environmental factors (e.g., the supply of drivers and the demand for deliveries), is suboptimal in dispatch efficiency.

Problem Statement

The business process for online scheduled deliveries is shown in Figure 1. Customers place orders and select delivery windows, for example, between 2 to 3 PM. The platform's objective is to fulfill these orders by 3 PM in an efficient manner. The platform initiates a driver search at a certain point (e.g., 1:20 PM), offering a base pay amount. If the delivery order is at risk of not finding drivers in time, the platform must decide when to increase the pay (e.g., 1:40 PM) to enhance the likelihood of securing drivers before it becomes too late for on-time delivery. When drivers are considering offers, they need to be aware of the expected arrival time (e.g., 2 PM) at the pickup locations. This is the time when the platform anticipates the order will be collected and enroute to delivery, and it should not be so late that drivers cannot complete deliveries after picking up the items. Hence the platform needs to make intelligent timing decisions to ensure on-time delivery rates while improving surge efficacy.

The timing of deliveries also needs to be considered from the drivers' perspective. Drivers see offers and decide whether to accept them or not. If they accept, they need to drive to pickup locations and collect items by the designated arrival time, then drive to the customer's location to complete the delivery.

Crowdsourced drivers spend their valuable time online to earn money, so they prefer engaged time over idle time. Timing decisions are crucial to a driver's time utilization. A poor example would be starting driver dispatch too early. The driver accepts the delivery assignment but finds that the expected arrival time is far in the future, resulting in delivery start lag time before they need to drive to the store. Another bad example is when a driver arrives at a store, only to find many drivers also waiting to pick up orders. However, the pickup locations have limited resources to handle these drivers simultaneously. Hence, they must wait a long time before picking up items. This wait time creates poor driver experiences and jeopardizes on-time delivery.

In summary, the end-to-end timing decision-making system should improve dispatch efficiency, by reducing idle time, while ensuring high on-time delivery rates. The idle time consists of two parts as discussed above, the delivery start lag time and wait time. To achieve this, it needs to address the three questions:

1. What is the optimal driver dispatch start time to avoid delivery start lag time and the risk of failed on-time delivery?
2. When is the best time to increase pay to improve surge efficacy of the platform and reduce the risk of failed on-time delivery?

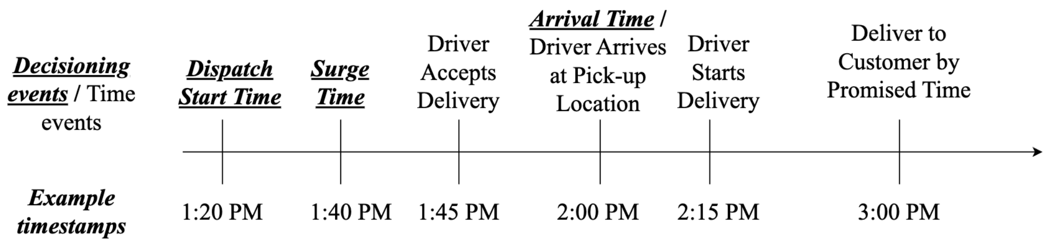

Figure 1: Delivery flow and driver dispatch timing decisioning in end-to-end

3. What is the ideal arrival time to prevent wait time due to pick-up location congestion without jeopardizing on-time delivery?

We have developed an intelligent decision-making system to determine end-to-end timings for each delivery. The solution starts by predicting time duration of all segments throughout the delivery journey. It then optimizes the decisions on these timings using various simulation and optimization techniques, while integrating on time delivery constraints. Our solution accepts user-defined parameters to tune outputs depending on application specific objectives. The framework rooting the system is universally applicable to timing decision making for online delivery via crowd-sourced drivers. Novel contributions of this work include:

1. An innovative decision-making system to optimize dispatch timing systematically regarding dispatch efficiency, with integration of on-time delivery risk constraints.
2. A generalized framework for solving end-to-end driver dispatch timing decisions for crowdsourced online deliveries, building upon delivery duration predictions, survival modeling, simulation and real-time optimization enabled decision making.
3. Configurable decisioning through automatic market segmentation and user-defined inputs to meet various business objectives.

System Architecture

The system is composed of the following modules (Figure 2):

- Data Preprocessor: This module ingests end to end delivery data, performs standard data cleaning procedures and generates feature stores for models.
- Model:
  - Delivery Segment Duration Predictor: This module predicts time durations of end-to-end segments for each delivery.
- Decision-making optimizer: this comprises three optimization models, which determines end-to-end dispatch timing coherently in a sequential manner.
  - Arrival time optimization model: Determines the optimal arrival time for each delivery to minimize drivers' wait time while ensuring high on-time delivery rates.
  - Dispatch start time optimization model: Determines the optimal dispatch start time for each delivery to minimize the delivery start lag time while ensuring high on-time arrival rates.
  - Surge time optimization model: Determines when to increase pay to drivers for each delivery to maximize surge efficacy while ensuring high on-time delivery rates.
- Configurator: Segments all markets into several clusters, applies configs at the cluster level for decision-making optimizer.
- Evaluation flows: Comprises a simulator to validate the decisioning outputs, and an experimentation protocol to measure and quantify the impacts.

The rest of the paper is organized as follows. In Section 2, we describe components of our system (See Figure 2), detailing mathematical formulations and our solution approach. In Section 3, we perform rigorous evaluations of our system via simulation and online experimentation on. We conclude in Section 4 summarizing the importance of our work and the future directions that emanates from this work.

Methods

Delivery Segment Predictor

To make intelligent decisions regarding driver dispatch timing, it is crucial to understand the duration throughout the delivery journey. In this module, we employ predictive machine learning techniques to estimate these duration. Specifically, we consider four duration: driver search, drive to pick-up location, delivery loading time, and delivery time. Driver search is the time from the start of dispatch to when the driver accepts the delivery. Drive to pick-up location is the time the driver spends driving to the pick-up locations. Delivery loading is the necessary time needed to load all items into the drivers' cars after arriving at the pick-up locations. Delivery is the time from the driver departing the pick-up locations to the completion of deliveries.

These time duration depict one delivery from end to end, but they differ in correlating features. For instance, driver search duration heavily correlates with the supply of drivers and demand of deliveries, as well as the attractiveness of the delivery. Drive to store duration strongly correlates with drivers' locations, pick-up locations, and traffic. To address these correlations, we conducted feature engineering and

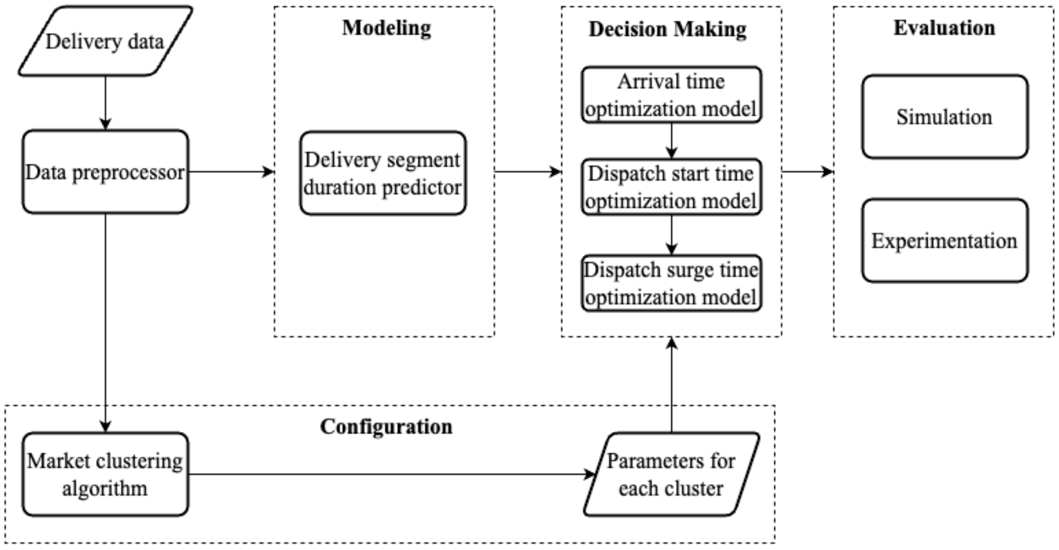
Delivery data
Data preprocessor
Modeling
Delivery segment duration predictor
Decision Making
Arrival time optimization model
Dispatch start time optimization model
Dispatch surge time optimization model
Evaluation
Simulation
Experimentation
Configuration
Market clustering algorithm
Parameters for each cluster

Figure 2: System Architecture

built four predictors for each duration. The features mainly fall into three categories: delivery attributes (such as delivery distance, number of items, pick-up location, delivery location), supply (of drivers) and demand (of deliveries), and activities at pick-up locations (such as delivery preparation speed).

We used regression tree models with hyper-parameter tuning for the four predictors, as it provided satisfactory performance and interpretability among other choices we considered. We also enabled the model to store statistics (mean and standard deviation) of leaf nodes for later decision making. In many cases, using these statistics, rather than a single point prediction, provides much-needed flexibility in the decision-making model. We can take suitable thresholds (or quantiles) to finalize predictions on timing duration, to meet various operational objectives, as we'll show in the decision-making models later.

We train and deploy predictors for each market. Time duration fluctuate based on many seasonal and market factors. These fluctuations are further compounded by a rapidly growing platform. Hence, we retrain predictors on weekly basis using historical four weeks data.

Arrival Time Optimization Model

The arrival time informs drivers of the optimal time to arrive at pick-up locations. The goal is to ensure that drivers arrive at the pickup location in a manner that avoids creating undesirable congestion, while also providing them with sufficient time to complete their deliveries punctually.

First, the model utilizes duration predictors to derive the latest possible arrival time for each delivery. In other words, drivers must arrive prior to this timestamp to guarantee on-time delivery. We refer to this as the "at-risk" timestamp. It can be formulated as equation (1), where for each delivery i, t_a^i is arrival time, T_l^i is the loading (load orders into drivers' cars) time duration and T_d^i is the delivery time duration. t_e^i is the latest delivery time (i.e., delivery end time) promised to customers. The threshold α dictates the level of conservatism of the at-risk time. The higher the threshold, the earlier the at-risk time.

$$P(t_a^i + T_l^i + T_d^i > t_e^i) < \alpha \tag{1}$$

To determine the optimal arrival time for each delivery, such that congestion at pick-up locations are reduced, we aim to minimize the waiting time of drivers. Hence, we can formulate this as an optimization problem (equation (2)):

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{i=n} W_i = f(\{t_a^i\}_{i\in I}, C) \\ \text{subject to} \quad & P(t_a^i + T_l^i + T_d^i > t_e^i) < \alpha, \ \forall i \in I \\ & T_l^i > 0, \ T_d^i > 0, \ \forall i \in I \\ & C > 0, 0 < \alpha < 1 \end{aligned} \tag{2}$$

Where I is all deliveries, W_i is the wait time for each delivery, C is the dispensing capacity at pick up locations.

While this can be formulated as an optimization problem, there are three challenges to solve it. First, the wait time is a function of the arrival time for all deliveries and the dispensing capacity at pick-up locations. Establishing this function is non-trivial. It requires knowing relationships among the arrival time of all deliveries, and their interactions with dispensing capacities. Secondly, dispensing capacities are typically not readily available. While we use established three-phase traffic theory (Kerner and Lieu 2005) to enable the estimation, a method to validate the estimation accuracy is necessary. Last, it is infeasible to solve the problem through traditional optimization solver, given that many decision variables are not known simultaneously. Deliveries can occur at any time of the day, and decisions of arrival time for current deliveries need to be made without knowledge of future deliveries.

Simulation conveniently addresses these challenges. First, deliveries are dispensed to drivers following a typical queueing process. We build an agent-based, discrete-event simulator to establish the relationship. Given the arrival time for all deliveries and the dispensing capacities, the simulation engine can conveniently derive the wait time before each delivery is handled by the available dispensing resources. Second, we estimate and validate the dispensing capacity such that the wait time distribution from the simulator matches the actual observations. We then run the simulator over a list of candidate arrival patterns and select the optimal pattern that minimizes the wait time without negatively impacting on-time delivery rates. The arrival pattern is a list of distributions over a time window. Each value indicates percent of deliveries with arrival time allocated for each bucket within the time window. Finally, we use the optimal arrival pattern to guide the decisioning of the arrival time for each delivery on the fly. The process aims to match the distribution of deliveries to the targeted distribution, while following at-risk time constraints. This approach works regardless of uncertainties in future deliveries.

Thus, we effectively solve the optimization problem, but with a simulation-based approach. This allows us to tackle both the challenge of function estimations, and the infeasibility of solving the optimization problem on the fly.

Dispatch Start Time Optimization Model

The dispatch start time plays a crucial role in recommending the optimal initiation time for the driver search process. The primary objective is to start the search for drivers by timely publishing the delivery, thereby minimizing delivery start lag time, and ensuring high on-time arrival rates.

Using the delivery segment predictor, we get estimated driver search time and drive to pick-up location time, which are then integrated into an optimization-based decisioning algorithm. We formulate this as a stochastic programming problem for each delivery (equation (3)):

$$\begin{aligned} \text{maximize} \quad & t_s \\ \text{subject to} \quad & P(t_s + X + Y > t_a) < \alpha \\ & X + Y \sim N(\mu_x + \mu_y, \sigma_x^2 + \sigma_y^2 + 2\sigma_{xy}) \\ & \mu_x > 0,\ \mu_y > 0,\ 0 < \alpha < 1 \end{aligned} \tag{3}$$

Where t_s is the dispatch start time, X is random variable of driver search time, Y is random variable of drive to pick-up location time, t_a is the arrival time obtained from arrival time optimizer, α is the on-time arrival rates threshold, μ_x, μ_y, σ_x^2, σ_y^2, $2\sigma_{xy}$ are estimated parameters from delivery segment predictors.

The arrival time sets the reference for dispatch start time. The latter's goal is to start driver search such that delivery start lag time is minimized while on-time arrival (at pick-up locations before arrival time) constraints are satisfying. By maximizing the dispatch start time while complying the on-time arrival constraints, we start driver dispatch as late as possible. This essentially minimizes the delivery start lag time. It equals to leave just enough time for driver search and drive to pick-up locations, without rooms for delivery start lag time.

Here we use random variables to describe driver search time and drive to pick-up location time, instead of single point predictions. The reason is we make decisions for each delivery before driver dispatch starts. Hence our predictions and decisions are driver agnostic. As such, it is better to use random variables based on statistical parameters. It gives us the flexibility to set the cut-off point in predictions based on operational objectives and make decisions accordingly. We experimented with different statistical distributions and found normal distribution fits best.

Surge Time Optimization Model

The surge time pertains to the optimal time to increase the pay for drivers. Initially, drivers are offered a base pay for delivery. The process of driver dispatch, initiated by the dispatch start time model, leads to the quick acceptance of attractive deliveries. However, less attractive deliveries require a surge in pay to be accepted by drivers. This model optimizes the timing of starting surge pay.

This model introduces the concept of delay risk, which is the probability that drivers cannot be found in time to complete deliveries on time without surging. The surge timing is critical as surging too early can increase unnecessary dispatch costs, while surging too late can delay the delivery process, leading to failed on-time deliveries. The goal is to determine the optimal surge time to maximize surge efficacy while ensuring on-time delivery, by initiating the surge when the delay risk becomes unacceptable.

We first obtained the latest allowable driver found time from the delivery segment predictor. This time is the latest delivery time promised to customers minus the delivery time, loading time, and drive to pick-up location time. In essence, drivers must be found by this time to avoid risking on-time delivery.

Given the surge range, which is the time between the driver dispatch start and the latest driver found time, we use a survival regression model to estimate the delay risk for each trip at each time t within the surge range, as shown in equation (4). Survival regression, a classical method of estimating 'survival' likelihood, applies well to the driver dispatch problem. Here, not 'surviving' means deliveries are accepted by a driver. We obtain the acceptance (not surviving) probability using survival regression via the Cox proportional hazards model (Cox 1972). A key assumption for this model is the absence of timing variant factors for the sample. In our case, we adhere to this assumption by assuming no changes in the pay, aligning with our objective of modeling delay risk without surge pay.

$$h((t+t_f) \mid t, X) = h_0((t+t_f) \mid t)\, e^{\sum_{i=1}^{p} x_i \beta_i} \tag{4}$$

Where t_f represents additional search time in the future with respect to already searched time t; the term h_0 is the baseline hazard; h is the hazard function, i.e., the acceptance likelihood determined by a set of p covariates (x_i, $i = 1, \ldots, p$), which are input features such as delivery attributes, the coefficients (β_i, $i = 1, \ldots, p$) measure the impact of covariates.

We then define delay risk at any time t as equation (5). It is the likelihood of survival (not finding drivers) by latest

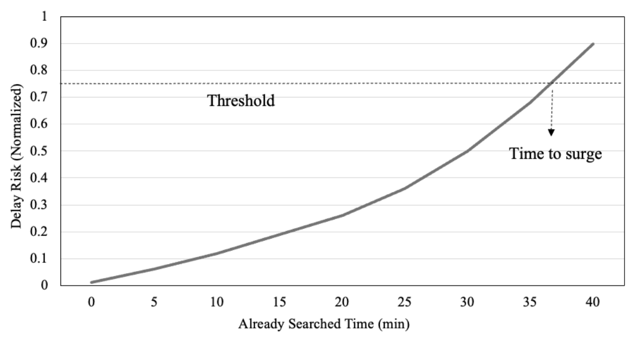

Figure 3: Delay risk over time and corresponding surge timing decisioning

allowable driver found time t_{df}, given already searching for drivers for t. The delay risk is monotonically increasing over time. This is expected as the longer the delivery is still in the market, the less likely (higher risk) it can find drivers before a fixed timestamp in the future.

$$F(t, X) = h(t_{df} \mid t, X) \tag{5}$$

Finally, we get the optimal surge time by comparing the delay risk with the thresholds, as shown in Figure 3. The thresholds are generated by automatic learning of the delivery risk from historical data in the same and similar markets. It also accepts user defined configurations for flexible cut-off and corresponding decisioning.

Configurator

Our system operates in a complex environment with more than thousands of markets in United States, each exhibiting unique characteristics. For instance, markets with ample driver supplies typically experience shorter times to find drivers and fewer deliveries incurring surge costs. Conversely, markets with low delivery volumes and high dispensing resources at pick-up locations are less concerned about the concentration of drivers' arrival times. These varied patterns necessitate different operational strategies and objectives. In markets with sufficient driver supplies, we can afford to relax on-time delivery rate constraints, providing more flexibility to optimize other objectives such as dispatch efficiency. However, in markets with limited supplies, we need to enforce stringent on-time delivery constraints.

To cater to these differentiated strategies, we designed a configurator that segments markets into several clusters using k-means clustering. This clustering is based on key performance metrics such as demand volume of deliveries, supply volume of drivers, historical on-time delivery rates, and driver dispatch cost. The number of clusters is determined using the elbow method and business feedback.

Once the markets are segmented, we assign configurations at the cluster level for each decision-making model. The decisioning model integrates on-time delivery as constraints, with parameters defining the conservative or aggressive levels. Based on market cluster attributes, we assign corresponding levels for these parameters. The assignment of parameters is two-fold. First, we have set up pipelines to periodically inspect the metrics and determine the appropriate settings such that the on-time delivery rates are no worse than historical levels. Second, we take business feedback into consideration to fine-tune settings for some clusters from time to time. This approach ensures automated parameter settings while maintaining the flexibility to incorporate user-defined inputs.

Evaluation Flows

We build simulation and experimentation flows to evaluate the system performance. The simulation evaluates the effectiveness of the solution in comparison to existing rule-based processes. The simulation flow utilizes an agent-based, discrete-event simulation technique to emulate the complete delivery journey, starting from the driver dispatch to the delivery completion. The simulated results allow us to evaluate the impacts of the timing decisions on both directly impacted metrics (e.g., driver wait time, delivery start lag time) and indirectly impacted metrics (e.g., dispatch cost, on-time delivery rates). To ensure the accuracy of the simulation engine, it is validated against real-world datasets, based on the similarity of primary metrics of interests, such as wait time, delivery start lag time, and on-time delivery rates.

We have also built a Difference-in-Differences (DID) (Angrist and Krueger 1999) experimentation flow to monitor and measure impacts. Given the nature of timing decisions, which are agnostic to users (i.e., drivers) and have potential carry-over effects (decisions from one hour can impact deliveries in the next hour), DID is the most suitable experimentation framework. Each market is treated as one unit.

The flow is built in several steps. First, it selects test and control groups via stratified sampling. The test groups are chosen to be representative of the entire market, based on primary metrics of interest. The control groups are selected to exhibit parallel trends in recent times, on weekly or daily frequencies. Once the experiment has been launched for a sufficient duration with enough data inputs, the experimentation framework examines the impacts through visual inspection on dashboards and statistical analysis via DID modeling, shown in equation (6), where Y_{gt} are observations on metrics of interests, a_g are group-fixed effects, b_t is time-fixed effects, δ is the coefficient of the cross-over term (experiment group in post-launch period) D_{gt}, ϵ_{gt} is the correlated error within each market. The DID model evaluates the impacts through the coefficient of the cross-over term. The statistical significance is determined by the p-value of the coefficient δ. The model considers the fixed effects on markets and weekly patterns, and error correlations within each market through cluster robust standard error.

$$Y_{gt} = a_g + b_t + \delta D_{gt} + \epsilon_{gt} \tag{6}$$

System Performance Evaluation

Arrival Time Evaluation

The effectiveness of the arrival time model is evaluated using the simulation platform. The primary question we aim

Parameter settings (conservative level)	Relative changes in wait time	Relative changes in OTD
0.0	-65%	+0.5%
-0.25 (optimal)	-68%	+1.1%
-0.5	-64%	+1.3%
-0.75	-57%	+1.4%
-1.0	-55%	+1.4%

Table 1: Wait time and OTD changes under different candidate arrival time distribution

to answer is: Can the model generate an optimal arrival time distribution that minimizes wait time without negatively impacting on-time delivery rates? This question is addressed in two steps. Initially, within a specific market and hour, we observe how the model evaluates candidate distributions and determines the optimal arrival time distribution. Subsequently, we apply the model across various markets and hour slots to verify its effectiveness in a broad range of scenarios.

Table 1 presents a comparison of wait times and on-time delivery (OTD) rates under different candidate solutions. The final choice made by the model is indicated in brackets. Each candidate solution is governed by a parameter that signifies conservative level. The lower the value, the more conservative the setting, meaning more deliveries are assigned with an earlier arrival time. A value of zero represents the most aggressive scenario, where a uniform distribution across all times is adopted. We can see that the solution chosen by the model achieves the greatest reduction (68%) in wait time without compromising OTD. Another observation is that as we adopt a more conservative setting to candidate distribution, the reduction in wait time decreases. This outcome is expected and validates the model's effectiveness. As the setting becomes more conservative, more deliveries are assigned to earlier times, increasing the likelihood of congestion at pick-up points, which increases wait time.

This method was then applied across several markets and hour slots, with the results summarized in Table 2. These different markets and hour slots represent clusters identified by the clustering algorithm. They generally have different patterns of driver supply versus delivery demand, dispensing resources at pick-up locations. Consequently, different configurations are adopted, through either automatic learning or user-defined inputs. For instance, in markets and times where store dispensing capacities are sufficient and delivery demands are low, a more conservative distribution setting is used. This is used in anticipation that the wait time will remain manageably low while significant gains are made on OTD. The results indicate a consistent improvement in reducing wait time and no negative impacts on OTD rates, demonstrating the effectiveness of the model.

Surge Time Evaluation

The surge timing model's effectiveness is evaluated by its capability to reasonably differentiate surge time when delay risk varies. To achieve this, we applied the model across various scenarios and evaluated the outputs.

First, we assessed how surge timing results vary under different delay risks. Figure 4 illustrates the surge timing (relative to the promised delivery time) under low versus high supply-demand ratio. When ratios of driver supply to delivery demand are low, delay risks of deliveries increase as drivers are harder to find. Consequently, we observe earlier surge timing (3.3 minutes on average). Figure 5 depicts the surge timing for deliveries with short versus long delivery distances. When delivery distances are high, delay risks of deliveries increase as they need more time to fulfill. As expected, deliveries with longer distances have earlier surge timings (9.7 minutes on average). This confirms that the model can reasonably differentiate surge timing based on delay risk.

Cluster	Relative changes in wait time	Relative changes in OTD
0	-68%	+1.9%
1	-28%	+4.6%
2	-47%	0%
3	-63%	+2.0%

Table 2: Wait time and OTD changes under different clusters with optimal arrival-time distribution

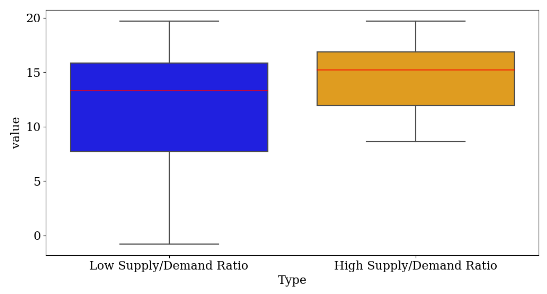

Figure 4: Surge time under different supply (of drivers) and demand (of deliveries) ratio

Next, we investigated how the model outputs more diversified surge timing, as compared to existing rule-based decisions. Figure 6 compares surge timing distributions under these two cases. The existing decisions are rule based, fixed timestamps for all deliveries (with bifurcation based on delivery types), while the model yields more diversified distributions based on underlying delay risks of each delivery. We further applied two configurations on the delay risk threshold and observed two distributions differentiating the overall decision-making of all deliveries. A higher delay risk threshold implies more conservative decision-making, or earlier surge timing, to focus more on protecting on-time delivery rates. This confirms that the model can generate more diversified surge timing than existing solutions and shift that decision-making reasonably, based on configurations.

Online Experimentation

We deployed the system to Walmart's proprietary delivery platform and conducted online experiments to measure its impact. After selecting test and control groups with simi-

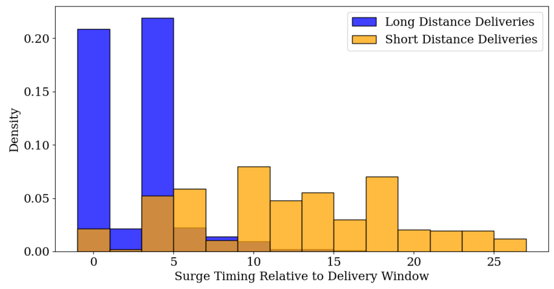

Figure 5: Surge time under different delivery distance

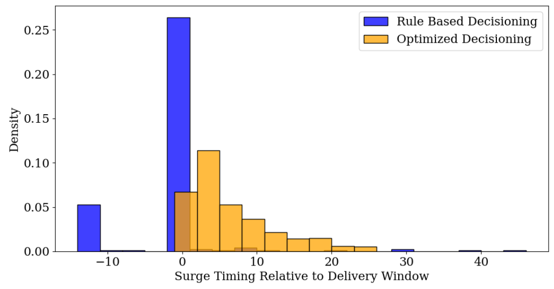

Figure 6: Surge time comparison with optimized and rule based decisioning

lar characteristics, we launched the service in the test group, while keeping the control groups unchanged. We monitored metrics on a weekly basis. Once we collected enough samples for statistical analysis, we used the DID model to quantify the impacts and their statistical significance.

The primary questions we seek to answer through online experimentation are: 1) Does the system reduce dispatch idle time, without negative impacts on on-time delivery rates? 2) What is the magnitude of these impacts and are they statistically significant?

Figure 7 illustrates the changes between the test and control groups, before and after the launch. It reveals that the system significantly reduces the delivery dispatch idle time for drivers. This reduction is attributed to two factors. Firstly, the optimal dispatch start timing minimizes the delivery start lag time (due to drivers are less likely to idle given dispatch starts not too early) and wait time (due to less congested arrival time). Further, the optimal timing increases the likelihood of drivers accepting deliveries earlier, as they see their time utilization get improved. These two factors contributed to the overall reduction in driver dispatch time. Getting more drivers accepting deliveries earlier also leads to more deliveries dispatched before the surge price is applied. Consequently, surge efficacy is improved since less unnecessary surge is incurred.

Lastly, all these decisions resulted in a neutral impact on on-time delivery rates. The reason for this is that the system is designed to consider delivery risk as constraints in all decision-making models. The decisions are made to main-

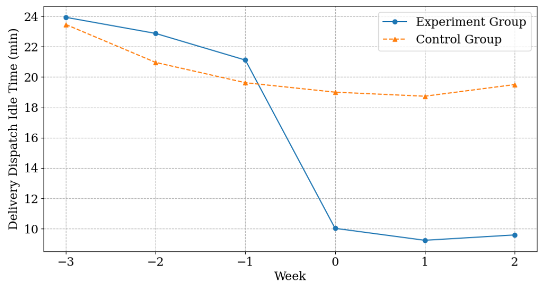

Figure 7: Comparison of delivery dispatch idle time through DID experimentation

tain on-time delivery rates at least intact. That said, there exist parameters that can be tuned or accept user-defined inputs. The parameters can then guide the decision-making process more conservative or aggressive.

We further quantified the impacts and their statistical significance using the DID model. It is concluded that the system reduced dispatch time by 55%, as compared to previous solutions, while keeping high on-time delivery rates consistent for Walmart's last mile delivery platform. All the changes were statistically significant (p values less than 0.05), except for the changes in on-time delivery rates, indicating no impacts as expected .

Conclusion

In this paper, we proposed a comprehensive system for optimizing end-to-end driver dispatch timing decisions, to improve dispatch efficiency in the context of crowdsourced online deliveries. This system uses predictive modeling to estimate delivery journey duration. The decision-making models then leverage stochastic programming, simulation, and survival modeling to determine optimal dispatch timings. Specifically, it decides 1) when drivers should arrive to minimize wait time at pick-up locations while ensuring on-time delivery rates, 2) when to start driver dispatch to minimize delivery start lag time while adhering to on-time arrival constraints, and 3) when to start surge to improve surge efficacy while ensuring high on-time delivery rates.

We conducted simulation and experimentation studies on Walmart's proprietary crowdsourced last-mile delivery platform to demonstrate the solution's effectiveness. These include reducing wait time at pickup locations, reducing the delivery start lag time for drivers, and maintaining satisfactory on-time delivery rates. These accomplishments have culminated in highly efficient dispatch processes: 55% reduction in dispatch time, as compared to previous solutions, while meeting on-time delivery promises to customers.

This decision-making system has proven to be fundamental to successful driver dispatch, and consequently, to the success of Walmart's crowdsourced online delivery platform. The framework rooting the system is applicable to other crowdsourced platforms. It addresses the gap in end-to-end timing decisions for crowdsourced online deliveries with promised delivery time.

Acknowledgements

We would like to extend our sincere appreciation to our esteemed leadership for their unwavering support and guidance throughout this work. Their visionary leadership, encouragement, and guidance have been instrumental in shaping the direction of this work and facilitating its successful execution. We extend our special thanks to Nav Chadha, Angela Gray Samuelson, Manish Rathi, Deepak Deshpande, Kristen Quinn, Jonathan Dale, Parvez Musani and Aniket Athawale for their outstanding support. Additionally, our heartfelt gratitude extends to our exceptional business, product, supply chain data science and engineering teams. Their collective expertise and unwavering dedication have significantly accelerated our progress and enriched our findings. We offer our special thanks to Francisco Medina Medina, Josie Sullivan, Shayan Mahinfallah, Shreya Tara Kannan, Mohit Agarwal, Kanishk Sharma, Anchal Gupta, Caroline Michaud, Anupam Garge, Hengchao Wang, Justin VanHouten, Tom Stiefel, Gopinath Maditheti, Viresh Jivane, Aruna Amin, Richard Williamson, Elly Rath, Ayush Ganeriwal, Hitansu Sahoo, Navneet Nandi and Anupriya Chakraborty, for their exceptional contributions.

References

Angrist, J. D.; and Krueger, A. B., eds. 1999. *Empirical Strategies in Labor Economics*. Elsevier.

Araujo, A. C. d.; and Etemad, A. 2021. End-to-End Prediction of Parcel Delivery Time With Deep Learning for Smart-City Applications. *IEEE Internet of Things Journal*, 8: 17043–17056.

Barbosa, M.; Pedroso, J. P.; and Viana, A. 2023. A data-driven compensation scheme for last-mile delivery with crowdsourcing. *Computers and Operations Research*, 150: 106059.

Chadha, N. 2023. The Spark Driver Platform Celebrates 5 Years of Growth. https://corporate.walmart.com/news/2023/06/07/the-spark-driver-platform-celebrates-5-years-of-growth. Accessed: 2023-11-15.

Cox, D. R. 1972. Regression Models and Life-Tables. *Journal of the Royal Statistical Society. Series B (Methodological)*, 34: 187–220.

Ding, Y.; Guo, B.; Zheng, L.; Lu, M.; Zhang, D.; Wang, S.; Son, S. H.; and He, T. 2021. A City-Wide Crowdsourcing Delivery System with Reinforcement Learning. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 5: 1–22.

Hu, X.; Cirit, O.; Binaykiya, T.; and Hora, R. 2022. DeepETA: How Uber Predicts Arrival Times Using Deep Learning. https://www.uber.com/blog/deepeta-how-uber-predicts-arrival-times/. Accessed: 2023-11-14.

Kerner, B. S.; and Lieu, H. 2005. The Physics of Traffic: Empirical Freeway Pattern Features, Engineering Applications; and Theory. *Physics Today*, 58: 54–56.

Lei, Y.; Jasin, S.; Wang, J.; Deng, H.; and Putrevu, J. 2020. Dynamic Workforce Acquisition for Crowdsourced Last-Mile Delivery Platforms. *SSRN*, 3532844.

Liu, S.; He, L.; and Shen, Z.-J. M. 2020. On-Time Last-Mile Delivery: Order Assignment with Travel-Time Predictors. *Management Science*, 67: 4095–4119.

Miao, X.; Peng, H.; Gao, Y.; Zhang, Z.; and Yin, J. 2023. On Dynamically Pricing Crowdsourcing Tasks. *ACM Transactions on Knowledge Discovery from Data*, 17: 1–27.

Silva, M.; and Pedroso, J. P. 2022. Deep Reinforcement Learning for Crowdshipping Last-Mile Delivery with Endogenous Uncertainty. *Mathematics*, 3902.

Skiver, R. L.; and Godfrey, M. 2017. Crowdserving: A Last Mile Delivery Method for Brick-and-Mortar Retailers. *Global Journal of Business Research*, 11: 67–77.

Tong, Y.; Wang, L.; Zhou, Z.; Chen, L.; Du, B.; and Ye, J. 2018. Dynamic Pricing in Spatial Crowdsourcing: A Matching-Based Approach. In *SIGMOD '18: Proceedings of the 2018 International Conference on Management of Data*. Houston TX USA: ACM.

Wang, F.; Zhu, Y.; Wang, F.; and Liu, J. 2018. Ridesharing as a Service: Exploring Crowdsourced Connected Vehicle Information for Intelligent Package Delivery. In *2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS)*. Banff, AB, Canada: IEEE.

Wang, W.; Wang, H.; and Shuaijie, J. 2019. Surge Pricing Optimization of Crowdsourcing Logistics Service Based on Sharing Economy. In *2019 International Conference on Industrial Engineering and Systems Management (IESM)*. Shanghai, China: IEEE.

Zehtabian, S.; Larsen, C.; and Wøhlk, S. 2022. Estimation of the arrival time of deliveries by occasional drivers in a crowd-shipping setting. *European Journal of Operational Research*, 303: 616–632.