Authors are encouraged to submit new papers to INFORMS journals by means of a style file template, which includes the journal title. However, use of a template does not certify that the paper has been accepted for publication in the named journal. INFORMS journal templates are for the exclusive purpose of submitting to an INFORMS journal and should not be used to distribute the papers in print or online or to submit the papers to another publication.

Dynamic Pricing and Matching in Ride-Hailing Platforms

Nikita Korolko, Dawn Woodard, Chiwei Yan, Helin Zhu
Uber Technologies, Inc.

Problem description: Ride-hailing platforms such as Uber, Lyft and DiDi have achieved explosive growth and reshaped urban transportation. We review dynamic pricing and matching technologies — the two key levers in ride-hailing — studied in the literature. We point out that if dynamic pricing is utilized as the single marketplace lever, price can be high and volatile. To address this issue, we link the two levers together by introducing a pool-matching mechanism called dynamic waiting which varies rider waiting and walking before dispatch.

Academic/practical relevance: We focus on studying the synergy between pricing and matching technologies in ride-hailing; previous literature mainly focuses on either of the topics, but not both jointly. The introduced dynamic waiting concept is inspired by a recent carpooling product Express Pool from Uber.

Methodology: We study a steady-state model for pricing and matching in ride-hailing. The platform determines the dynamic price as well as window of time to wait before dispatching drivers to incoming ride requests. We characterize the system equilibrium of supply and demand under different prices and waiting windows. We reveal insights on system performance, as well as welfare-maximizing pricing and waiting strategies.

Results: We show that dynamic pricing and matching are critical for providing an experience with low waiting time for both riders and drivers. We calibrate the steady-state model using data from Uber. By jointly optimizing price and waiting window, we show that price can be lowered and its variability is mitigated. Furthermore, capacity utilization, trip throughput, and total welfare are increased.

Managerial implications: We highlight several key practical challenges and directions of future research in ride-hailing from a practitioner’s perspective. We also demonstrate that the joint optimization of pricing and matching can bring significant benefits to both riders and drivers, as well as the platform.

Key words: Ride-Hailing, Ride-Sharing, Sharing Economy, Matching, Dynamic Pricing, Dynamic Waiting

History: 2018/09/30, First Version
1. Introduction

Several transformative technologies have emerged in the transportation sector in recent years, including ride-hailing, autonomous driving, and electric vehicles. For example, ride-hailing platforms improve the convenience and efficiency of passenger transportation, autonomous driving promises reductions in accident rates, and electric vehicles enhance energy efficiency. The synergy of these technologies is even more powerful: maintaining a fleet of autonomous vehicles with in-network communication capability for a ride-hailing platform is projected to reduce the cost of passenger transport from $3.50 per mile to less than $0.35 per mile (ARK Invest 2016).

Among these, ride-hailing has experienced the most explosive growth. Uber has completed over 10 billion trips globally and is active in over 80 countries and 700 cities, within 9 years of its initial launch (Figure 1 and Uber (2018a)). The global ride-hailing industry is projected to grow to a $285 billion total market value by 2030 (MarketWatch 2017).

![Figure 1](image_url) Uber’s annual trip growth (y-axis scale redacted).

Ride-hailing platforms connect riders and drivers via a centralized and automated matching and pricing system in a two-sided marketplace. The rides can be either non-shared, meaning that the ride is arranged for only one customer group (e.g., UberX), or shared, meaning that several customer groups with different pickup and dropoff locations share the ride (e.g., UberPool).

Relative to taxi services, drivers with ride-hailing services spend a higher proportion of their time on trip (capacity utilization rate), as shown in data from Cramer and Krueger (2016). These authors compared the capacity utilization rates between UberX and taxi services in major metropolitan areas, as reproduced in the left plot in Figure 2. In other words, UberX drivers spend less time waiting between ride requests or driving to pick up riders, compared to taxi drivers. The story is
similar on the rider side. Feng et al. (2017) demonstrated via simulations that the average rider waiting time between request and pickup (rider waiting time) can be much lower in a ride-hailing system compared with that in a street-hailing taxi system, as reproduced in the right plot in Figure 2. Rider waiting time and capacity utilization rate are also closely related to a concept of reliability of the service: sudden spikes in demand (for instance, at the end of a concert or on New Year’s Eve) can cause a dramatic increase in the rider waiting time and drop in the capacity utilization rate, leading to a poor experience on both sides. This occurs because the available drivers are quickly dispatched, leading to low supply and causing newly available drivers to be dispatched to pick up distant riders. This phenomenon is called the Wild Goose Chase (WGC) by Castillo et al. (2017).

Two key technologies employed by ride-hailing platforms to provide high reliability and capacity utilization rate and low rider waiting time are matching and dynamic pricing (DP). Matching means the process of dispatching available drivers to pick up riders, and DP means dynamic adjustment of prices for rides based on real-time demand and supply conditions. DP is called “surge pricing” by Uber and “prime time” by Lyft. Hall et al. (2015) observed using data from Uber that DP is crucial for maintaining service reliability, reducing rider waiting time, and incentivizing drivers, in the event of a demand shock. Relatedly, Castillo et al. (2017) demonstrated, both theoretically and empirically, that DP alleviates the WGC phenomenon.

Due to the importance of matching and DP technologies to ride-hailing, they have been adopted, implemented, and constantly iterated by almost all large platforms such as Uber, Lyft, and DiDi. For example, at Uber the current matching engine generates tens of thousands of dispatch decisions per minute, and the current DP engine generates tens of millions of price decisions (surge multipliers) at the level of geographic units (Uber 2018c) every minute across the world.
We review key matching and DP algorithms proposed in the literature. Additionally, since having accurate inputs to the algorithms is as important as algorithm design, we discuss the estimation of several key inputs. The inputs for DP include short-term supply and demand forecasts, and those for matching include travel time predictions in the road network.

While DP has been demonstrated to be effective in improving system efficiency, it often comes with the downside of price volatility caused by short-term fluctuations in supply and demand conditions. This is due to the local nature of the problem in space and time: that is, only riders and drivers close to each other are eligible to be matched together. The dispatchable drivers and riders in a localized area over a short period of time exhibit much higher variability than that of the aggregated supply and demand in a large region over a longer time horizon. Price volatility can be undesirable for both riders and drivers: riders become less loyal to the platform (Harvard Business Review 2015), and drivers become frustrated because it is hard for them to relocate to areas of higher prices since the prices may drop by the time they arrive (Chen et al. 2015).

Recognizing these drawbacks, there are efforts in both industry and academia to reduce such price variability. DiDi adopted a queuing system to partially replace the role of dynamic pricing (Pingwest 2017). The queuing system makes riders wait in a queue if there are no dispatchable drivers within a certain dispatch radius. In addition to DP, Castillo et al. (2017) discussed alternative WGC mitigation strategies such as imposing a maximum dispatch radius (see Section 2) or dispatching drivers who are about to finish a trip nearby.

We show that the price volatility can be reduced by jointly optimizing DP and a pooling mechanism called dynamic waiting (DW). Inspired by a recent Uber product called Express Pool, DW works as follows. After a rider requests, they wait to be matched to a driver, for a period of time called waiting window. This window is used to attempt to pool together two (or more) requests whose origin locations as well as destination locations are close to each other (within walkable distance). The riders are then picked up together at the midpoint between their origins, and dropped off at the midpoint between their destinations. The waiting window is dynamically determined by market conditions: when supply is constrained, the waiting window increases. This mechanism has two effects. First, by asking riders to wait before being matched, the pool of eligible requests for matching is thickened, which results in a higher pool-matching probability and thus a higher trip throughput because drivers have a better chance to serve more than one request in each ride. Second, it increases rider waiting time, which suppresses demand. Together, these two effects maintain the balance between supply and demand. Note that in DW the rider waiting time is equal to the sum of the time the driver spends picking up the rider (en route time) plus any time the rider spends waiting before dispatch is made.
We show that DW is able to alleviate the WGC phenomenon. We further show that the combination of DP and DW could reduce price and its volatility, and improve capacity utilization rate, trip throughput, and welfare. Most importantly, our results showcase the benefits of jointly designing DP and matching algorithms, which is rarely studied in the existing literature to the best of our knowledge.

Matching and DP algorithms have also been leveraged by many other online platforms that connect service providers and consumers. To name a few: TaskRabbit for chores and home projects, eBay for e-commerce, SpotHero for parking spaces, StubHub for concert and sports game tickets, Turo and Getaround for car-sharing, and Airbnb for hospitality service. In contrast to ride-hailing, these services provide matching suggestions and also price guidance via recommendation systems, but the final decisions are made by the participants on the platform. Food and grocery delivery platforms (such as UberEats, Instacart, or Grubhub) are more similar to ride-hailing platforms, but they have more time and flexibility for dispatching due to a lower level of urgency for the service. The pricing and matching decisions for ride-hailing need to be solved in real-time as riders and drivers are sensitive in time. For instance, at Uber usually the upfront prices are generated within a second and matches are generated within seconds.

The remainder of this paper is organized as follows. In Sections 2 and 3, we synthesize the literature on matching and DP in ride-hailing, respectively. In Section 4 we investigate the benefits of DW, both theoretically and empirically using data from Uber. In Section 5 we review methods for estimating the inputs required for the matching and DP algorithms. Finally, in Section 6 we highlight several outstanding practical challenges and lay out promising directions of research for pricing and matching in ride-hailing.

2. Matching in Ride-Hailing

Drivers participating on the platform cycle through three states sequentially: “open” — waiting to be dispatched, “en route” — on the way to the pickup location, and “on-trip” — driving riders to their destination, as shown below.

\[
\cdots \rightarrow \text{open} \rightarrow \text{en route} \rightarrow \text{on-trip} \rightarrow \text{open} \rightarrow \cdots
\]

A request can be matched with a dispatchable driver using very simple algorithms, such as a non-shared ride algorithm called the first-dispatch protocol. In the first-dispatch protocol only open drivers are considered as dispatchable. Each request is immediately assigned to the open driver who is predicted to have the shortest en route time.

Feng et al. (2017) compared the average rider waiting time for street-hailing (where the rider hails the first passing driver) to that for ride-hailing with the first-dispatch protocol, in a simulation.
The authors found that ride-hailing generally has lower rider waiting time. However, in certain cases, the average rider waiting time of a street-hailing service can be lower. The intuition behind this observation is as follows. With the committed driver-rider matching used by ride-hailing, the driver once dispatched could miss the opportunity of being matched with a rider they drive passed by while en route. To mitigate this inefficiency, the authors proposed to apply a maximum dispatch radius (MDR) with carefully selected thresholds. MDR means that dispatch is made only if the en route time is below the threshold. This prevents the driver from being dispatched to pick up distant riders. Using MDR, the average rider waiting time in ride-hailing is typically substantially lower than the one in street-hailing, as illustrated in the left plot of Figure 2.

An alternative matching approach is batching as illustrated in Figure 3. Requests are collected for a short time window (e.g., a few seconds), at the end of which an optimization problem is solved to pair each request with an open driver. If there are riders that are not matched in this batch, they are carried over and re-solved in the next batching window.

A typical way to model and solve such an optimization problem is as follows. A bipartite graph is first constructed to represent all the potential matches between riders and drivers. Each node in the graph corresponds to a request or an open driver. A weighted edge connects each pair of the rider and driver nodes, and the weight represents certain reward collected by matching this pair. The edges can be trimmed via some criteria (such as MDR) to reduce the problem size before solving the optimization problem. The detailed mathematical formulation is as follows. Denote $N$ and $M$ as the sets of rider nodes and driver nodes in the same batch, respectively. The binary decision variable $x_{ij} = 1$ if rider $i$ and driver $j$ are matched, and 0 otherwise. The reward for matching rider $i$ with driver $j$ is denoted by $r_{ij}$. Then the matching problem could be formulated as an integer program below.

$$\max_x \sum_{i \in N} \sum_{j \in M} r_{ij} x_{ij} \quad (2.1)$$
s.t. \[
\sum_{j} x_{ij} \leq 1, \quad \forall i \in N, \tag{2.2}
\]
\[
\sum_{i} x_{ij} \leq 1, \quad \forall j \in M, \tag{2.3}
\]
\[
x_{ij} \in \{0, 1\}, \quad \forall i \in N, \forall j \in M.
\]

In particular, the objective in Eq. (2.1) is to maximize total rewards collected from the matching, and constraints (2.2) and (2.3) are to ensure that each rider/driver could be matched with at most one driver/rider, respectively. This integer programming formulation could be solved efficiently using its linear programming relaxation or directly using Hungarian Algorithm (Kuhn 1955). If the objective is to minimize the aggregated en route time, we could set \( r_{ij} = M - \eta_{ij} \) in Eq. (2.1), where \( \eta_{ij} \) is the en route time for driver \( j \) to rider \( i \), and \( M \) is a sufficiently large number to bound from above all potential en route time in consideration, i.e., \( M \geq \max_{i \in N, j \in M} \eta_{ij} \). For a large geographic region, we could partition the entire region into several small areas and solve a matching problem separately for each area to achieve faster computation.

Batching can further reduce the rider waiting time relative to the first-dispatch protocol, by consolidating requests and making better use of supply. Indeed, the latter can be viewed as a special case of the former when each batch consists of exactly one request. Ashlagi et al. (2018) numerically compare the performance of batching to that of the first-dispatch protocol using New York City yellow cabs dataset, and show batching outperforms the first-dispatch protocol. Due to its substantial benefits, batching has been implemented in major ride-hailing platforms (Lyft 2016a, Zhang et al. 2017). One downside of a long batching window is the prolonged rider waiting time before seeing a dispatch, which deteriorates rider experience. Selection of an optimal batching window is an interesting open question.

The matching algorithms described so far are myopic in the sense that they do not consider information about the future demand or supply. They are relatively easy to implement, if high-quality predictions of travel time in the road network are available. For this reason, these matching algorithms are very popular among ride-hailing platforms and exhibit relatively good performance. However, since demand and supply arrive in a dynamic fashion, there exist cases where such myopic algorithms could perform poorly. An example is shown in Figure 4, where there is one request \( A \) and two idle drivers \( A \) and \( B \) in the current batching window. Driver \( A \) is 3 minutes away from the rider, and Driver \( B \) is 1 minute away. Based on the en route time, rider \( A \) and driver \( B \) should be matched. Suppose that in the next batching window, another rider \( B \) shows up who is 0.5 minutes away from driver \( B \) and 4 minutes away from driver \( A \). With this new information, it would have been beneficial to hold off matching rider \( A \) with driver \( B \) in the first batching window, in order to match rider \( A \) with driver \( A \) and rider \( B \) with driver \( B \) in the next batching window. By doing
this, the total en route time of these two matches would be reduced from $1 + 4 = 5$ minutes to $3 + 0.5 = 3.5$ minutes.

Motivated by this, Ozkan and Ward (2016) developed dynamic matching policies based on a fluid linear programming formulation of the underlying stochastic matching problem where demand and supply arrive according to a non-homogeneous Poisson process, and dispatch decisions need to be made immediately after each request. The authors benchmarked their method against the first-dispatch protocol, using a simulation with synthetic inputs. They found that the first-dispatch protocol usually performs reasonably well; however, in the case where there are severe imbalances between demand and supply across regions, the proposed dynamic policies can significantly outperform the first-dispatch protocol. The intuition is that when there is a future demand spike in a specific region, drivers in nearby regions will be reserved to serve that high-demand region first instead of the regions they are currently in. Their study suggests that such dynamic matching policies could potentially serve as a complement to DP for balancing demand and supply. Similarly, Hu and Zhou (2016) introduced a batched version of dynamic matching and derived the structural properties of the optimal policy under a stylized road network topology. Most recently, Truong and Wang (2018) and Ashlagi et al. (2018) proposed online algorithms that have constant competitive ratios, i.e., they achieve at least a constant fraction of the reward from an optimal offline policy with full information of future demand and supply.

Although forward-looking algorithms bring benefits theoretically, in practice the trade-off between algorithm performance and complexity of input calibration also needs to be considered. Advanced matching algorithms usually require additional predictions such as the time-varying demand and supply arrival rates. As pointed out by Ozkan and Ward (2016), errors in these inputs can potentially deteriorate the algorithm performance, while myopic algorithms are relatively parameter-free and enjoy robust performance. Some intermediate improvements to myopic
matching algorithms have also been implemented on ride-hailing platforms. These improvements require few additional inputs and are very effective in bridging the optimality gap between myopic and forward-looking algorithms. As an example, Uber and Lyft have tested a feature called trip upgrade (Lyft 2016b) that dynamically re-assigns pickup assignments between en route drivers if all parties will be better off according to criteria such as the remaining en route time.

In practice, matching is further complicated due to the co-existence of multiple products and features on the same platform. For example, besides non-shared ride products, many ride-hailing platforms also provide pooling services such as UberPool and Lyft Line. Although we could treat the drivers with spare capacity as dispatchable drivers in the matching bipartite graph, the decision process is more complex because the pickup and dropoff sequences need to be taken into account as well. The addition of these considerations make the optimal pooling decisions combinatorially hard. We refer readers to Alonso-Mora et al. (2017) and Bertsimas et al. (2018) for some modeling and solution approaches in tackling such large-scale pool-matching and routing problems.

3. Dynamic Pricing in Ride-Hailing

One of the major challenges in ride-hailing is the continuously changing supply and demand volumes, especially on a local scale in space and time. Figure 5 shows the demand to supply ratio, based on data from Uber, for two neighborhoods in San Francisco: the Sunset and the Financial District. Demand is measured by the number of rider in-app sessions while supply is measured by the amount of time drivers spend on the app. The Sunset is a residential neighborhood, so riders

![Figure 5](image-url)  
**Figure 5**  Demand to supply ratios in San Francisco. The data is taken from a weekday on May 2018, and aggregated every 10 minutes.
often need to commute from the Sunset to the Financial District in the morning and vice versa in the evening. As a result, we see that the Sunset has a much higher demand to supply ratio than the Financial District during morning rush hour, and vice versa during evening rush hour. The demand to open supply ratio also exhibits strong variation over time. Figure 5 demonstrates that the imbalance between demand and supply is strongly localized in both space and time and exhibits high variability; in contrast, the aggregated demand to supply ratio for the entire San Francisco is relatively stable.

To mitigate such imbalance, dynamic pricing (DP) is employed by ride-hailing platforms to balance supply and demand both temporally and spatially. Uber refers to its DP decisions as *surge multipliers*, meaning that when demand is very high relative to supply the base fare is multiplied by a multiplier that is greater than one.

The utility of DP for maintaining service reliability and keeping en route times low was illustrated by Hall et al. (2015) using two examples from Uber in New York City (Figure 6). The first was at the end of an Ariana Grande concert in Madison Square Garden that led to a spike of demand and an initial increase in the en route time. Surge pricing kicked in (indicated by the yellow intervals), which mitigated the demand peak and brought down the en route time (top left two plots). As a result, the fraction of requests that received a dispatch stayed very close to one (top right plot). By contrast, a similar demand spike occurred in Times Square on New Year’s Eve 2015, but surge

![Figure 6 Surge pricing during two events, as reproduced from Hall et al. (2015). Top: after an Ariana Grande concert; Bottom: New Year’s Eve 2015 in Times Square. Left: the count of requests; Middle: the en route time; Right: the fraction of requests fulfilled.](image-url)
did not kick in due to a technical malfunction (shown in red). Requests increased to many times their usual level (bottom left plot). The en route time increased to nearly eight minutes (bottom middle plot), and fewer than a quarter of requests were fulfilled (bottom right plot).

Methods for DP usually leverage either steady-state economic models or dynamic programming techniques that focus on the temporal nature of the problem. These approaches usually require short-term predictions of demand and supply, including price elasticity estimates (Elmaghraby and Keskinocak 2003). In the case of a steady-state model, only overall expected supply and demand levels are needed, while dynamic programming approaches leverage the time series of forecasted demand to respond in a smooth fashion to upcoming demand or supply changes.

Castillo et al. (2017) proposed a steady-state model for DP in ride-hailing, and showed that DP is particularly important for ride-hailing due to the so-called Wild Goose Chase (WGC) phenomenon. The intuition is as follows. When there are very few drivers relative to demand, available drivers are quickly dispatched. Under the first-dispatch protocol, in such a supply-constrained situation the drivers may be dispatched to very distant pickup locations. This leads to high en route times, leaving drivers with less time for taking riders to their destinations (lower capacity utilization rate). The result is reduced earnings, leading drivers to drop off the platform, which in turn increases en route time. Such downward spiral of driver engagement further reduces trip throughput. Mathematically, the WGC phenomenon can be described by approximating the system with a steady-state economic model, and considering the implications of different price points in that model. Such an analysis shows that the combination of price and en route time that maximizes trip throughput (or similar measure of value generated, such as welfare) is one with high rider and driver participation and low en route time.

DP is a lever for preventing the marketplace from entering the WGC zone, by balancing demand with supply in supply-constrained situations. Figure 6 is an empirical evidence of the WGC phenomenon during a demand spike, and the role of DP in mitigating the WGC. This example also showcases the negative rider and driver experiences when the marketplace is in the WGC zone; drivers tend to not accept or cancel a dispatch if the en route time is high (as indicated by the low request completion rate), causing wasted driver time and an inability of riders to get a dispatch.

Next we review a key step in understanding the WGC, namely the derivation of trip throughput under the steady-state model in Castillo et al. (2017). The remainder of the derivation, with an extension to the setting of dynamic waiting, is provided in Section 4.

Consider a single-geo setting, meaning that all trips begin and end within the geographic region. We initially consider a fixed number, denoted by $L$, of drivers on the platform, who do not relocate in or out of the geo. Recall that drivers cycle through three states: open, en route, and on-trip. Using the first-dispatch protocol, only open drivers are dispatchable, and all requests are immediately
dispatched (we will relax this constraint in Section 1). In this case, en route time is equal to rider waiting time. Further suppose each trip (from pickup to dropoff) takes a given constant time \( T \) (units of time) to complete. The steady-state trip throughput, i.e., the average number of trips completed per unit of time, is denoted by \( Y \). We then have the following supply flow balance equation:

\[
L = O \cdot \frac{\text{open}}{\text{en route}} + \eta \cdot Y + T \cdot Y, \tag{3.1}
\]

where \( O \) represents the number of open drivers, \( \eta \) denotes the en route time, \( \eta \cdot Y \) represents the number of en route drivers, and \( T \cdot Y \) represents the number of on-trip drivers. The en route time \( \eta \) is determined by the number of open drivers: a lower number of open drivers \( O \) leads to a higher en route time. This relationship is captured in Proposition 1.

**Proposition 1** \((\text{Larson and Odoni (1981)})\). Suppose the open drivers are distributed uniformly in a \( n \)-dimensional \((n > 1)\) Euclidean space. Further assume a constant travel speed on the straight line between any two points in that space. Then the expected en route time, denoted by \( \eta(O) \), satisfies

\[
\eta(O) \propto O^{-\frac{1}{n}}, \tag{3.2}
\]

where \( O \) is the number of open drivers.

There is a proof of Proposition 1 in the E-Companion EC.2.1 for the case when \( n = 2 \), which is the relevant case for ride-hailing. This model also fits well empirically with data from Uber if we relax the exponent in the en route time function \( \eta(O) \) from \( -\frac{1}{n} \) to a general \( \alpha \in (-1, 0) \), as demonstrated in the left plot in Figure 7. Intuitively, \( \alpha \) is different from the theoretical value due to some violations of the assumptions in Proposition 1 in practice drivers move on a road network, travel speeds vary among road segments, and open drivers are not distributed uniformly over space.

From Proposition 1 we can rearrange the flow balance equation (3.1), and substitute in the expected value \( \eta(O) \) for \( \eta \) (noting that we are approximating a random variable with its mean), to obtain \( Y \approx \frac{L-O}{\eta(O)+T} \). To illustrate the behavior of the trip throughput \( Y \) as a function of the number of open drivers, we show this approximation in the right plot in Figure 7. We can see that there is an open driver level \( O^* \) that maximizes the trip throughput. For \( O > O^* \), the trip throughput is lower than the one at \( O^* \); however, since the en route time \( \eta(O) \) is lower, there is an experience versus throughput trade-off and there may be some cases in which it is desirable to have a value of \( O > O^* \). For example, there may be a value \( O > O^* \) that leads to a higher welfare than the one at \( O^* \). In contrast, when \( O < O^* \), the trip throughput decreases sharply, due to the fact that en route time grows quickly. This regime is undesirable from both the experience perspective and the trip
throughput perspective. As mentioned previously, it is called the WGC zone; a key function of DP
is to keep the price point above the level that would lead the marketplace to enter the WGC zone.

In the later sections we will look at $Y$ as a function of en route time $\eta$ rather than a function
of $O$ for the sake of derivation. Let $C(\eta) \propto \eta^{-n}$ indicate the inverse function of $\eta(\cdot)$, which is well-defined since $\eta(\cdot)$ is strictly decreasing. Then we could obtain the approximate trip throughput as
a function of $\eta$ and $L$:

$$Y(\eta, L) \triangleq \frac{L - C(\eta)}{\eta + T}. \quad (3.3)$$

In addition to characterizing the WGC in terms of the open driver level, Castillo et al. (2017)
also demonstrated that DP prevents the open driver level from dropping too low, and so increases
trip throughput and welfare. Here welfare is defined by rider gross utility minus driver social cost,
which can be regarded a measure of value created for the drivers and riders. The authors modeled
the response of riders to price (surge multiplier) and waiting time, and the response of drivers to
average earnings, which is a function of surge multiplier and capacity utilization rate. Such driver
response to earnings can occur over a short time horizon (Chen et al. 2015). However, the bigger
effect on driver participation is over a longer time horizon: experienced drivers on ride-hailing
platforms eventually become cognizant about the time periods and areas of high earnings due to
high demand and surge pricing, and flexibly adjust their work schedule to drive during these time
periods (Chen and Sheldon 2016, Cachon et al. 2017).

Figure 8 shows the resulting welfare and platform revenue as functions of the surge multiplier.
This figure is similar to Figure 5 from Castillo et al. (2017), except the model has been re-fit
Figure 8 welfare and revenue as a function of price, based on a model from Castillo et al. (2017) that has been fit to data from Uber in San Francisco downtown. Left: welfare and revenue under DP; Right: welfare under DP during rush and non-rush hours. Both quantities are in surge multiplier unit per \( km^2 \) and hour.

to recent data from Uber. In particular, the dashed lines in the left plot show dramatic losses of both welfare and revenue due to the WGC, when price is set too low. The right plot shows two such welfare curves, corresponding to non-rush hour (11am-noon) and rush hour (6pm-7pm), respectively. It also shows the welfare-maximizing dynamic prices during the two time periods; we can see that the welfare-maximizing price during rush hour is higher than the one during non-rush hour, due to the higher imbalance between demand and supply during rush hour. The right plot also shows the static price that maximizes the total welfare, aggregated over the two time periods. This mimics a scenario in which dynamic pricing is disallowed (for example due to regulatory constraints), so that the platform is required to select a single price across all time periods. In this example, the welfare-maximizing static price is nearly equal to the highest dynamic price during rush hour. While such a high static price avoids the WGC, it results in lower welfare relative to that under DP.

Besides DP, other mechanisms can help in alleviating the WGC phenomenon and maintaining healthy en route time. For example, Castillo et al. (2017) demonstrated that the platform can prevent long en route times by setting a MDR.

While DP brings many benefits to the marketplace, it also comes with downsides. First, the price can fluctuate due to the natural variability in local demand and supply levels that we illustrated in Figure 5. This price variability is an undesirable experience and can cause reduced engagement of both riders and drivers. For example, high surge prices during New Year’s Eve have received some negative press coverage (Lowrey 2014). Secondly, it can be hard for drivers to relocate quickly to areas of higher prices because the prices may drop by the time they arrive (Chen et al. 2015, Ming et al. 2017). For instance, Chen et al. (2015) stated that in the short term “surge prices have a small,
positive effect on vehicle supply, and a large, negative impact on passenger demand.” Another potential caveat of DP is that it creates price gradients in space and time, which in extreme cases can create an incentive for drivers to reject some dispatches. In practice, platforms such as Uber often employ smoothing techniques to create a gradual price gradient, in order to avoid selective rejection of dispatches by drivers.

The variation in prices over time can lead riders and drivers to choose not to participate in the platform at the current time, in favor of waiting a few minutes for a more advantageous price. In this setting, riders (drivers) decide whether to make (accept) a request in the current time period or wait, knowing that they might be able to get better prices (earnings) in the following time periods. Addressing this, Chen and Hu (2017) investigated the case where both riders and drivers are strategic and forward-looking. More broadly, drivers may make a strategic decision about whether, when, and where to provide service, based on differences in surge price and earnings. Ride-hailing models for this case have been designed by Bimpikis et al. (2016) and Afèche et al. (2018). Relatedly, Ma et al. (2018) studied the design of incentive-aligned DP mechanisms in the presence of strategic driver behavior, under a multi-geo, multi-period setting. Here, an incentive-aligned mechanism means that the pricing decisions are spatially and temporally smooth so that it is always best for drivers to accept a dispatch rather than performing other actions such as relocating or waiting.

4. Synergy of Pricing and Matching Technologies

In the previous section we discussed the price variability that is inherent to DP, and the resulting downsides to the user experience. In this section, we address this concern by investigating the benefits of combining DP with a pool-matching mechanism called dynamic waiting (DW), which allows batching of requests and varies rider waiting time before dispatch in order to find a perfect pool-match. This mechanism is inspired by the recent ride-sharing product Express Pool (Uber 2018b) from Uber. In particular, we show that DW could be used as a complement to DP to alleviate the WGC, reduce price volatility, and increase trip throughput and welfare. By jointly optimizing DP and DW, we show how one can explicitly trade off price with rider waiting time. Our study showcases that pricing and matching algorithms are tightly connected, and their interplay can bring significant benefits to the marketplace, which is rarely studied in the literature.

4.1. Dynamic Waiting

We start with a description of how DW works. Assume each driver (car) can hold up to two riders (which can be readily generalized to more riders), and all riders opt into pooling with other riders. Two riders are pool-matched if their pickup locations as well as dropoff locations are close; i.e., the distance between the pickup locations as well as the dropoff locations is within walkable range.
(walking radius). Furthermore, they are picked up and dropped off at the same time and location, with the pickup and dropoff locations set to be the middle points so that two riders can walk equal distances. DW also asks rider to dynamically wait up to certain time (waiting window) before dispatch. Such waiting time is controlled by the length of the batching window illustrated in Figure 3. At the end of each waiting window, we examine all the batched requests and find the maximum number of matches out of all requests. The problem can be solved using a standard maximum matching algorithm such as Blossom Algorithm. The system then dispatches the closest open cars to fulfill these requests utilizing the matching formulation in (2.1) - (2.3). The resulting total rider waiting time is then the sum of the dispatch waiting time and the en route time. Note that for simplicity of the model we ignore the rider walking time by assuming it is always less than the en route time. We denote the probability that a rider is pool-matched before dispatch as pool-matching probability.

Note that the above DW mechanism is highly stylized. It is much simpler than the one used in the Express Pool product in the sense that only perfect (simultaneous pickup and dropoff of pool-matched riders) pool-matching is allowed. In contrast, the Express Pool product (and most other ride-sharing products) allows imperfect pool-matching, e.g., new riders can be pool-matched with riders who are on-trip. Thus all the benefits we illustrate through this stylized mechanism can be considered as a lower bound of the actual benefits for a fully functional ride-sharing product.

Intuitively, more pool-matching is beneficial because more drivers can complete more than one trip in a single open-enroute-ontrip cycle. The pool-matching probability is determined by the number of requests consolidated in each batch, i.e., when more requests are eligible for pool-matching, the pool-matching probability is higher. The dynamic adjustment of waiting window is able to alleviate WGC when demand outstrips supply. On one hand, under fixed trip throughput, increasing waiting window increases the number of requests eligible for pool-matching, and thus lifting the pool-matching probability and supply efficiency (fewer drivers are required to serve the same level of trips). On the other hand, increasing waiting window also increases total rider waiting time, which dampens excessive demand. The combination of these two factors helps the market stay out of WGC zone. Furthermore, the joint optimization of DP and DW enables the explicit trade-offs between price and rider waiting time. As we will show later, such trade-off leads to better welfare and less volatile prices.

4.2. Equilibrium Analysis

In this subsection, we characterize platform performance including trip throughput, capacity utilization and etc., under different prices and waiting windows. We do this by studying the market equilibrium, under an extension of the steady-state model introduced by Castillo et al. (2017) and
reviewed in Section 3. At a high level, the platform determines price, waiting window, pool-matches and dispatch decisions. Then riders participate based on price and the resulting total waiting time, and drivers participate based on average earnings. Market equilibrium refers to the fact that in steady-state the number of rides requested by riders must equal the number of trips fulfilled by drivers participating on the platform. Such equilibrium analysis under a steady-state model setting has also been utilized by quite a few literature studying ride-hailing marketplace (Banerjee et al. 2015, Taylor 2017, Bai et al. 2017, Cachon et al. 2017).

We first introduce the notations and model assumptions, extending the ones presented in Section 3. For readers’ convenience, in E-Companion EC.1 we also give a full list of notations and their meanings. Our model is based on a fluid approximation of the underlying process where riders and drivers can be matched in any fractional quantities. Suppose the rider arrival rate is $\lambda$, meaning that per unit of time there are $\lambda$ unique riders opening the app. The platform must determine the price multiplier $p$ and waiting window of $\phi$ for pool-matching. Riders make requests if they are willing to accept the price $p$ and expected total waiting time $E$, and otherwise they leave the platform. Recall that $E$ is the sum of en route time $\eta$ and dispatch waiting time which is a function that depends on the waiting window $\phi$. That is, $E(\eta, \phi) = \eta + H(\phi)$, where $H(\phi)$ denotes the expected dispatch waiting time. It is not hard to see that $H(\phi) = \phi/2$ under the batching setting because riders arrive with equal probability within the waiting window. Therefore,

$$E(\eta, \phi) = \eta + \frac{\phi}{2}. \quad (4.1)$$

The realized demand arrival (request) rate is then denoted by $D(p, E) \leq \lambda$. We make the following practical assumptions on $D(p, E)$.

**Assumption 1.** The realized demand arrival rate $D(p, E)$ satisfies:

1. It is continuously differentiable in $(p, E)$ and decreasing in both $p$ and $E$.
2. $\lim_{p \to \infty} D(p, E) = 0$ for all $E \geq 0$ and $\lim_{E \to \infty} D(p, E) = 0$ for all $p \geq 0$.
3. For all $E \geq 0$, the distribution of the maximum willingness-to-pay has finite mean.
4. For all $p > 0$, $D(p, E) \propto o(E^{-1})$ as $E$ goes to infinity.

Assumptions 1.1 to 1.3 are identical to Assumption 1 in Castillo et al. (2017), and Assumption 1.4 is necessary under the context of DW. Loosely speaking, Assumption 1.1 means riders are less likely to request when price or waiting time is higher; Assumption 1.2 means no rider will request when price or waiting time goes to infinity; Assumption 1.3 ensures that the welfare created under an arbitrary price decision is finite; Assumption 1.4 ensures that request rate goes to zero sufficiently quickly as the rider waiting time goes to infinity, which holds under common forms such as the logistic form used in the numerical study of Section 4.3 below.
On the supply side, we assume a labor market in which drivers make decisions on whether to participate on the platform based on the average earnings per unit of time, denoted by $e$. Let $l(e)$ indicate the number of drivers who will participate at earnings level $e$ (the supply elasticity curve). We assume that $l(e)$ is an increasing, bounded, and continuously differentiable function in $e$. Under the steady-state model we have $e = (1 - \tau) \cdot p \cdot Y/L$, where $\tau$ is the fraction of the price collected by the platform as revenue according to the agreed commission rate, and recall $Y$ is the trip throughput. The number of drivers participating on the platform is then $L = l((1 - \tau) \cdot p \cdot Y/L)$.

Note that this implies that if a driver is serving two riders in a pool trip, then the earnings will be doubled compared with serving a single rider in a non-pool trip.

A market equilibrium under a fixed $(\phi, p)$ pair is defined by a supply and demand pair $(\tilde{L}, \tilde{D})$ such that $\tilde{L}$ satisfies the supply elasticity curve $l(\cdot)$ with the resulting trip throughput $Y$, and $\tilde{D}$ satisfies the demand curve $D(p, E)$ with the resulting total rider waiting time $E$. To establish such a market equilibrium, let us first derive the trip throughput $Y$ as a function of $\eta$ and $L$, in the presence of DW.

Due to the existence of pool-matching, the trip throughput function (3.3) in Section 3 ceases to hold. To establish the throughput function under DW, we denote $q(Y, \phi) \in [0, 1]$ as the pool-matching probability, which is a function that depends on trip throughput $Y$ and waiting window $\phi$. Naturally, we assume $q(Y, \phi)$ is increasing in both $Y$ and $\phi$, meaning that the pool-matching probability increases as the number of eligible requests increases. The expected number of drivers consumed by a single request (driver consumption fraction) is thus

$$f(Y, \phi) = \frac{1}{2} \cdot q(Y, \phi) + (1 - q(Y, \phi)) = 1 - \frac{1}{2} \cdot q(Y, \phi),$$

(4.2)

where the first term corresponds to the case when this request is pool-matched and occupies half of the dispatched driver, and the second term corresponds to the case when this request is not pool-matched and occupies the dispatched driver fully. From the assumptions on $q(Y, \phi)$, we have $\frac{1}{2} \leq f(Y, \phi) \leq 1$, $\forall Y, \phi \geq 0$. Furthermore, $f(Y, \cdot)$ is decreasing in $\phi$ for all $Y \geq 0$, and $f(\cdot, \phi)$ is decreasing in $Y$ for all $\phi \geq 0$. The driver dispatch rate, i.e., the number of drivers dispatched per unit of time, is thus $f(Y, \phi) \cdot Y$. To facilitate the derivation, we further make the following assumptions on $f(Y, \phi)$.

**Assumption 2.** The driver consumption fraction $f(Y, \phi)$ satisfies:

1. For all $Y \geq 0$, $f(Y, 0) = 1$ and $\lim_{\phi \to \infty} f(Y, \phi) = 1/2$; for all $\phi \geq 0$, $f(0, \phi) = 1$ and $\lim_{Y \to \infty} f(Y, \phi) = 1/2$.

2. For all $\phi \geq 0$, the driver dispatch rate $f(Y, \phi) \cdot Y$ is strictly increasing in $Y$. 
Assumption 2.1 is equivalent to the statement that no request is pool-matched when the pool of eligible requests is empty, and all requests are pool-matched when the pool of eligible requests is infinitely dense. Assumption 2.2 indicates, as trip throughput increases, the gain in pool-matching probability is not big enough to offset the extra trips — a higher driver dispatch rate is thus required. This is a mild assumption on \( f(Y, \phi) \) which essentially says that more driver dispatches are needed when more requests are satisfied.

Now we establish the trip throughput function \( Y(\eta, L) \) in the presence of DW. Note that in steady-state we have the following supply flow balance equation

\[
L = O + f(Y, \phi) \cdot Y \cdot \eta + f(Y, \phi) \cdot Y \cdot T, \tag{4.3}
\]

where the number of en route and on-trip drivers is adjusted by a factor of \( f(Y, \phi) \), compared with the original balance equation (3.1). Rearranging Eq. (4.3), we have the implicit equation for \( Y(\eta, L) \) as

\[
Y(\eta, L) \cdot f(Y(\eta, L), \phi) = \frac{L - C(\eta)}{\eta + T}, \tag{4.4}
\]

where recall that \( C(\cdot) \) is the inverse en route time function. Comparing Eq. (4.4) with Eq. (3.3), we can see that imposing a waiting window of \( \phi \) increases trip throughput by a factor of \( 1/f(Y(\eta, L), \phi) \), under the same supply level. While it is difficult to obtain an explicit formula for \( Y(\eta, L) \) from Eq. (4.4), we establish its structural properties as follows.

**Proposition 2.** Under Assumption 2, the trip throughput function \( Y(\eta, L) \) from Eq. (4.4) satisfies

1. For all \( L > 0 \), we have \( \lim_{\eta \to \eta(L)} Y(\eta, L) = 0 \) and \( \lim_{\eta \to \infty} Y(\eta, L) = 0 \). Furthermore, \( Y(\eta, L) \propto O(\eta^{-1}) \) as \( \eta \to \infty \).
2. For all \( \eta \in (\eta(L), \infty) \), \( Y(\eta, L) \) is increasing in \( L \).

We refer to E-Companion EC.2.2 for the proof of Proposition 2. On a high level, Proposition 2 exhibits the behavior of trip throughput function, which facilitates the derivation of market equilibrium later.

To this end, we have derived the formulae for trip throughput \( Y \) and total rider waiting time \( E \). Next let us establish the market clearance equation which regulates that trip throughput equals effective demand rate. For simplicity, we suppress the price variable \( p \) in the demand rate function and waiting window \( \phi \) in the total rider waiting time function. Denote

\[
\hat{D}(\eta) \triangleq D(p, E(\eta, \phi)). \tag{4.5}
\]

Then market clearance implies

\[
Y(\eta, L) = \hat{D}(\eta). \tag{4.6}
\]

We have the following Proposition 3 on the solution and its structural properties to Eq. (4.6).
Proposition 3. For all \( L > 0 \), Eq. (4.6) admits at least one solution in \( \eta > 0 \). Let \( \eta_L \) be the solution and \( \widehat{Q}(L; \phi, p) \) be the resulting trip throughput given the \((\phi, p)\) pair (if there are multiple solutions to (4.6), choose the unique solution that leads to the greatest \( \widehat{Q}(L; \phi, p) \)), i.e.,

\[
\widehat{Q}(L; \phi, p) \triangleq Y(\eta_L, L) = \widehat{D}(\eta_L).
\]

(4.7)

Then \( \widehat{Q}(L; \phi, p) \) is increasing in \( L \) for all \((\phi, p)\) pairs.

We refer to E-Companion EC.2.3 for the proof. Proposition 3 indicates that highest trip throughput \( Q \) is increasing in supply level \( L \) under the market clearance condition. Given the assumption that more drivers join the platform as average earnings increase, the demand curve and supply curve might cross with each other for certain \((L, Q)\) pairs, which leads to a market equilibrium (possibly multiple equilibria). We summarize this in Theorem 1 below. Note that we call a market equilibrium higher than another if the implied trip throughput is higher under that equilibrium.

**Theorem 1 (Market Equilibrium).** Denote \( \widehat{L}(Q; p) \) as the solution to the supply elasticity curve \( L = l((1 - \tau) \cdot p \cdot Q/L) \). Then for all \((\phi, p)\) pairs, the system of equations (on supply and demand)

\[
L = \widehat{L}(Q; p) \quad \text{and} \quad Q = \widehat{Q}(L; \phi, p)
\]

(4.8)

always admits a stable solution (a stable market equilibrium). In particular, the highest equilibrium, denoted by \((L^*, Q^*)\), is stable.

The definition of equilibrium stability and proof of the theorem are presented in E-Companion EC.2.4. Loosely speaking, an equilibrium is stable if the system of equations (4.8) always converges to that equilibrium whenever starting from a point \((L, Q)\) that is above and sufficiently close to it. Theorem 1 implies that for any given \((\phi, p)\) pair, the highest market equilibrium \((L^*, Q^*)\) is always stable. We could then compute the resulting marketplace performance metrics under \((L^*, Q^*)\), including capacity utilization rate, trip throughput, and welfare. Because \((L^*, Q^*)\) depends on \((\phi, p)\), these performance metrics also vary in accordance to the adjustments on the waiting window \( \phi \) and the price \( p \). Therefore, the platform can adjust \( \phi \) and \( p \) by maximizing a specific performance metric under \((L^*, Q^*)\).

### 4.3. Experiment with Data from Uber

In this section, we present computation results using Uber data by focusing on maximizing welfare via dynamically adjusting \( \phi \) and \( p \) in the model of Section 4.2. Note that other performance metrics could be used as the objective too, and we focus on welfare as a measure of value created for riders and drivers. We use UberX data on rider sessions, which record the experience and interactions of the rider between the time they open the app to the time they either request a ride, or a fixed period
of time goes by without a request. Session data measures potential demand for rider intent. We also use trip level data to measure actual requests. We focus on weekdays (Mondays to Thursdays) between 07/31/2017 and 09/01/2017, in San Francisco downtown. The data are aggregated by hour of day. Summary statistics for the data are provided in Section EC.3.1 of the E-Companion. Note that there is a morning peak and an evening peak in the data, corresponding to the rush hours.

Following Castillo et al. (2017), we model the rider demand function by 

\[ D(p, E) = \lambda r(p) g(E), \]

where \( \lambda \) is the total number of rider sessions, and \( r(p) \) and \( g(E) \) are the fractions of rider sessions that request rides given price \( p \) and total waiting time \( E \), respectively. Note that this form implicitly assumes that rider’s willingness to pay and willingness to wait are independent. We also specify binary logit forms for \( r(p) \) and \( g(E) \), i.e.,

\[ r(p) = \frac{e^{\beta_1 p + \kappa_1}}{1 + e^{\beta_1 p + \kappa_1}} \quad \text{and} \quad g(E) = \frac{e^{\beta_2 E + \kappa_2}}{1 + e^{\beta_2 E + \kappa_2}}, \]

where \( \beta_i, \kappa_i \) for \( i = 1, 2 \) are model parameters to be calibrated. Note that such choice of \( r(\cdot) \) and \( g(\cdot) \) ensure that \( D(p, E) \) satisfies Assumption 1.3, leading to finite welfare under arbitrary price. For each rider session, we record the quoted price and expected total waiting time the rider sees, and whether the rider requests or not. For simplicity, we use surge multiplier to measure price, instead of the actual fare in currency units that is shown to the rider when they’re making a request decision. Maximum likelihood estimation is then used to calibrate the model parameters; specifically, the following estimates are obtained:

\[ \hat{\beta}_1 = -0.67, \quad \hat{\kappa}_1 = 1.69, \quad \hat{\beta}_2 = -0.05, \quad \text{and} \quad \hat{\kappa}_2 = 1.07. \]

This approach is meant to be illustrative rather than definitive: it suffers from endogeneity due to the correlation between price and en route time, and doesn’t represent how Uber estimates price elasticity in practice. For the details of the parameter estimation, we refer to Section EC.3.1 in the E-Companion.

Following Proposition 1, the en route time function \( \eta(\cdot) \) in the number of dispatchable drivers \( O \) per km\(^2\) takes the form of \( \eta(O) = \tau \cdot O^\alpha \), where \( \tau \) and \( \alpha \) are two parameters to be estimated. We estimate \( \tau \) and \( \alpha \) using rider session data, together with dispatchable driver density around rider location for every rider session. We run a linear regression on the log scale, obtaining the estimates \( \hat{\tau} = 3.01 \) and \( \hat{\alpha} = -0.31 \). For the supply elasticity function, we follow the one used in Castillo et al. (2017) and Angrist et al. (2017), i.e., \( \ell(e) = A (e/(1 + 1/\epsilon))^{\epsilon} \) with supply elasticity \( \epsilon_1 = 1.2 \). Here the parameter \( A \) is a supply shifter which determines the general supply level. To estimate \( A \) for each hour of day, we follow the methodology in Castillo et al. (2017), where the average surge multipliers and the number of completed trips as the equilibrium throughput under these surge multipliers are used to back out \( A \). We refer to Section EC.3.1 in the E-Companion for the time series plots of average surge multipliers and supply shifters. We also estimate the average trip duration to be \( T = 14.7 \) minutes. For the driver consumption fraction \( f(Y, \phi) \), we use the form \( f(Y, \phi) = 0.5 + 0.5 \cdot e^{-\gamma Y \cdot \phi} \) in light of Assumption 2 and Proposition 2, where \( \gamma > 0 \) is
the parameter to be calibrated. In the calibration, we use 250m of walking radius and employ the batching procedure applied on historical Uber requests with various waiting windows and trip throughput to estimate the corresponding driver consumption fraction. Then a linear regression on log scale is applied to obtain the estimator \( \hat{\gamma} = 0.0006 \) with r-square value of 0.944. The upper bound of the waiting window is set to be 15 minutes.

Recall that welfare is defined by gross utility minus social cost, where gross utility is the integral of inverse demand curve with respect to price, and social cost is the integral of inverse supply curve with respect to earnings. Using the above demand and supply curves, we obtain gross utility as \( U(p, E) = \lambda \cdot g(E) \cdot \left( \int_{p'}^{+\infty} r(p') dp' + p \cdot r(p) \right) \), and social cost as \( S(L) = A \cdot \left( \frac{L}{3} \right)^{1+\frac{1}{\epsilon_L}} \). We further use \( W(\phi, p) \) to denote the welfare at the highest market equilibrium \( (L^*, Q^*) \) given a \( (\phi, p) \) pair. Then we have \( W(\phi, p) = U(p, \hat{E}(\eta_{L^*}, L^*)) - S(L^*) \), where note that \( \hat{E}(\eta_{L^*}, L^*) \) and \( L^* \) represent the implied total rider waiting time and supply level under \( (L^*, Q^*) \), respectively; \( \eta_{L^*} \) is the en route time at the highest market equilibrium.

With the calibrated input parameters above, we first compute welfare under the market equilibria corresponding to different prices during morning rush hour, with and without DW (left plot in Figure 9). For the scenarios without DW (blue curve), we compute the market equilibria following Castillo et al. (2017). For the scenarios with DW (red curve), we compute the market equilibria following the derivations in Section 4.2, and the welfare is calculated as the maximum one achieved among all possible waiting windows, i.e., the one under the optimal waiting window. We could see

![Figure 9](image-url)

**Figure 9**  Welfare and optimal waiting window. Left: welfare as a function of surge multipliers with and without DW; Right: optimal waiting window as a function of surge multipliers under static pricing. The data is from San Francisco downtown during morning rush hour.

that the welfare-maximizing (optimal) price without DW is around \( p = 1.46 \), and the marketplace enters the WGC zone sharply when price falls below the optimal one. When DW is present, the
optimal price is around $p = 1.14$, which is about 22% lower than the one when DW is absent. Furthermore, the marketplace enters the WGC zone at a much lower price $p = 0.78$, which implies that DW can alleviate the WGC when demand outstrips supply, fully replacing the role of DP. Lastly, by jointly optimizing DP and DW, welfare is improved in this example by about 1.4%, from 339.6 to 344.3 in surge multiplier unit per hour and km$^2$.

The right plot in Figure 9 illustrates the optimal waiting windows under different prices. We can see that there is an explicit trade-off between price and optimal waiting window, in the sense that the optimal waiting window decreases as price increases. In particular, the optimal waiting window stays at the maximum possible value $T = 15.0$ minutes until $p = 0.78$, at which the WGC phenomenon is mitigated when DW is present; then the optimal waiting window decreases until the price reaches $p = 1.46$, at which the WGC phenomenon is mitigated when DW is absent; finally when price is sufficiently high ($p \geq 1.46$), the optimal waiting window remains at zero.

Next we conduct similar experiments for all hours of day to understand how different pricing and waiting approaches perform under various market conditions. In particular, we compare four different approaches: (1) DP & DW, where we use dynamic price $p$ and waiting window $\phi$ that maximize welfare for each hour of day; (2) DP only, where we use dynamic $p$ with $\phi = 0$ that maximizes welfare for each hour of day; (3) DW only, where we use dynamic $\phi$ and the static $p$ that maximize the aggregated welfare over all hours of day; (4) static pricing and no waiting, where we use the static $p$ with $\phi = 0$ that maximizes the aggregated welfare over all hours of day. The results are summarized as follows.

We first examine the aggregated results over all hours of day. In Figure 10 we plot the aggregated capacity utilization rate, trip throughput, and welfare. Overall, we could see that either DP or DW is able to significantly improve these performance metrics. This suggests that DW could be beneficial in cities where DP is not allowed due to regulatory constraints. Additionally, DW actually achieves a slightly higher level of capacity utilization rate, trip throughput and welfare than DP does (comparing the second and third columns in all figures), implying that DW is a more effective marketplace lever under this pool-matching context. Furthermore, a higher capacity utilization rate implies lower driver dead heading, which is beneficial for reducing traffic congestion in a busy market such as NYC or San Francisco. Finally, the combination of DP and DW could further improve trip throughput and welfare, which demonstrates the synergy of pricing and matching technologies when they are jointly optimized.

Next we look at the optimal prices and waiting windows under the four approaches, as summarized in the left plot of Figure 11. The surge multipliers are lower bounded by 1.0 as practice. The solid blue curve shows the optimal dynamic prices without DW over the course of a day: Prices are high during the morning peak (8am - 9am) and evening peak (5pm - 6pm), which is expected since
Figure 10  Comparisons of capacity utilization rate, trip throughput, and welfare under four different approaches. Top left: capacity utilization rate; Bottom middle: trip throughput; Top right: welfare. All the quantities are aggregated over the 24 hours of day.

Figure 11  Optimal prices and waiting windows by hour of day. Left: optimal dynamic and static prices; Right: optimal DW windows under DP and static pricing.
in general the market is more supply constrained during rush hour. The dot blue line represents
the optimal static price without DW, which is very close to the highest optimal dynamic price.
The solid red curve shows the optimal dynamic prices with optimal DW, which are consistently
lower (by about 5% on average) than the ones without DW. This implies that DW could decrease
price and its volatility under a range of market conditions. The dotted red line represents the
optimal static price with DW, which is higher than the optimal dynamic prices with DW during
non-rush hour but lower than the ones during rush hour. Moreover, it is much lower (30%) than
the one without DW, which implies DW could reduce the optimal static price. Another interesting
observation is that under DW, the optimal surge multiplier during evening rush hour is 1.0. This
implies that waiting brings enough liquidity that excessive demand can be effectively pool-matched
to avoid WGC even without increasing price.

We finally examine the optimal DW windows, as shown in the right plot in Figure 11. The
solid and dot red lines represent the optimal waiting windows under dynamic and static pricing,
respectively. When DP is absent the optimal waiting window can spike over 8 minutes during rush
hour, while when DP is present it is relatively stable and mostly does not exceed 6 minutes. Based
on the two plots in Figure 11 we can see that DP or DW alone leads to either high price or high
rider waiting time, while the combination of DP and DW provides a beneficial trade-off, leading
to a better rider experience.

Additional numerical results are presented in Sections EC.3.2 and EC.3.3 in the E-Companion.
Overall, the results above demonstrate that DW could alleviate the WGC, reduce price and its
volatility, and improve welfare and trip throughput. Furthermore, the combination of DP and DW
provides a beneficial trade-off between price and rider waiting time.

5. Input Data Estimation

The DP and matching decisions rely on several key inputs that are estimated from data. As
mentioned previously, short-term demand and supply forecasts are needed as inputs to DP algo-
rithms. Travel time prediction is a critical input for matching algorithms that take into account
the predicted en route time. In this section, we briefly review methods for demand and travel time
predictions.

5.1. Demand Prediction

Short-term demand forecasting is needed for a variety of large-scale transportation applications,
ranging from ride-hailing, taxi (Ke et al. 2017 Yao et al. 2018), and ambulance services (Zhou
et al. 2015). A variety of forecasting techniques have been applied to these problems, including
In one example, Daulton (2015) leveraged a popular and publicly available dataset with 440 million NYC taxi trips occurring over 2.5 years, which includes information about pickup and dropoff locations as well as time, distance and passenger counts. The authors applied machine learning techniques such as random forests to predict the pickup density and dropoff location. Figure 12 shows the predicted demand during an average Monday rush hour. The major advantage of this approach is its flexibility in space and time granularity. For example, they used geo-hashing techniques to discretize space, which is convenient to control the precision of location estimation and therefore the size of grid cells. Their approach could be further improved by incorporating features such as events and holidays which are big factors for demand shocks, and more sophisticated machine learning techniques such as deep learning (Yao et al. 2018).

5.2. Travel Time Prediction
Travel time prediction is not unique to ride-hailing platforms, and has been used extensively by map services such as Google Maps and Waze. However, in logistics services like ride-hailing, taking into account the uncertainty of the prediction becomes more important. This is due to the fact that users tend to react negatively to the most extreme experiences, and such experiences can be made less common by building decision systems that are robust to travel time variability. For example, a carpooling matching system could enforce an upper bound on the 95th percentile of rider time to destination rather than an upper bound on the expected rider time to destination.

First we discuss how to obtain a prediction of the expected travel time, and then we review extensions that provide uncertainty estimates (such as predictive intervals or probability distributions). Specifically, the goal is to predict the expected travel time given the vehicle’s origin, destination, and trip start time. A classic and widely-used algorithm for this problem is as follows.
Start by predicting the travel times on individual road segments (edges in the road graph), using for example historical and recent travel times for drivers in the road network. In Figure 13 we illustrate this step by showing a heatmap of Uber's traffic speed predictions for individual road segments in San Francisco, at a particular time. Second, find the fastest route from the origin to the destination using a shortest-path finding algorithm such as hub-labeling (Abraham et al. 2012). Finally, add up the estimated travel times over all the road segments in the fastest route to obtain the trip-level travel time prediction. Drivers do not always take the fastest route, so this approach tends to underestimate travel time. To address this issue, one can instead take an average over several likely routes, or apply a post-hoc bias adjustment on the trip-level travel time predictions (Westgate et al. 2016).

Many types of data could be used for estimating the travel times on individual road segments, such as historical speed data that usually have time of day and day of week patterns, official speed limits, likely speeds derived from road types, and real-time traffic information transformed from GPS and device sensor signals. For example, Jenelius and Koutsopoulos (2013) applied a statistical model to predict road segment speeds based on low-frequency GPS observations, road segment attributes such as speed limits, and contextual information such as weather, and taking into account the correlation in speeds between connected road segments. In Figure 13, for example, freeway and arterial roads have generally higher predicted speeds than local roads.

Many approaches have been proposed to predict the variability in travel time estimation, in addition to the expected travel time. This can be done by providing the predicted probability distribution of travel time, based on either a parametric or non-parametric model. Woodard et al.
(2017) fit a statistical model to predict the distribution of travel time, taking into account local traffic patterns by time of week (obtained from historical and real-time travel speed data). In order to account for the correlation of travel speed across road segments (e.g., due to traffic congestion), they modeled travel time variability both at the trip level and the road segment level.

6. Practical Challenges and Conclusions

Here we have focused on scalable, straightforward DP and matching algorithms because they are practical for implementation by major ride-hailing platforms, while retaining most of the benefits. More complex algorithms may exhibit better performance in theory, but typically require sophisticated modeling of the marketplace dynamics as well as more complex inputs. From a practitioner’s perspective, one of the biggest challenges when designing and implementing matching and pricing algorithms at industrial scales is striking a balance between model complexity and accurate description of the marketplace dynamics. Here are a few areas in which this trade-off should be further explored:

1. Local versus Network Models. The ride-hailing marketplace exhibits micro (local) characteristics as well as macro (global) dynamics. As mentioned previously, the local characteristics come from the fact that the en route time is determined by local supply and demand conditions. For example, the single-geo model proposed by Castillo et al. (2017) captures such local characteristics when determining prices. Meanwhile, drivers move across different geos due to dispatch or self-relocation, influencing the future supply distribution over the entire network. Riders could also be incentivized through price to stimulate more demand going towards supply-constrained areas to further improve marketplace efficiency (Bimpikis et al. 2016, Banerjee et al. 2017). Similarly, drivers can be dispatched based on their network value (Banerjee et al. 2018). Although such network models have potential, they would require detailed inputs about supply and demand over the whole city, and could be challenging to calibrate and compute.

2. Single versus Multiple Products. Ride-hailing platforms often offer more than one product. For example, Uber offers UberX (main product), UberPool (ride-sharing), Express Pool (ride-sharing with waiting and walking), UberXL (extra seats), and Uber Black (luxury service). On the rider side, riders exhibit heavy substitution behavior across products, so that changing the price or waiting time for one product affects riders’ willingness to select other products. On the driver side, the supply pool is shared among some of the products; e.g., UberX, POOL, and Express Pool share the same supply base. Hence, dispatching a driver to one product impacts the supply level and thus the en route time of other products. Therefore, ideally the pricing and matching decisions should be determined simultaneously for all the products on the platform. This is challenging because it requires modeling the rider/driver behavior and the market dynamics across multiple products.
3. **Deterministic versus Stochastic Inputs.** Most of the algorithms we have reviewed assume deterministic models and inputs. As discussed in Section 5, these model inputs often include supply and demand forecasts, prediction of travel times, etc., which are challenging to estimate without error or variability. On one hand, the variability in inputs are propagated to algorithm outputs and thus impacting the solution quality. On the other hand, modeling and calibrating such stochasticity are non-trivial problems and potentially increase model complexity. It is important yet challenging to take the uncertainty of these inputs into account when designing pricing and matching algorithms.

To conclude, we reviewed the recent advancements in matching and pricing for ride-hailing. We showed that matching and DP are two key levers for ride-hailing platforms to maintain service reliability in the two-sided marketplace. While we demonstrated the benefits of DP, we also noted that it can lead to price volatility, which could be a poor experience for both riders and drivers. Motivated by this, we studied the interactions between DP and matching algorithms; in particular, we investigated the benefits from a pool-matching mechanism called dynamic waiting (DW), which varies rider waiting and walking to allow more pool-matches. By explicitly trading-off price and waiting time rider experiences, we showed that the synergy between DP and DW could reduce price volatility, and further improve capacity utilization rate, trip throughput, and welfare. This showcases that the joint optimization of pricing and matching technologies could bring additional benefits to both riders and drivers, as well as the platform. Along the way, we have highlighted how these technologies draw from a broad set of fields: economics, operations research, machine learning, statistics, and transportation engineering.

**Acknowledgments**

The authors would like to thank Ali Aouad at London Business School, and Cameron Blume, Chris Wilkins, Dan Knoepfle, Eoin O’Mahoney, Lior Seeman, Duncan Gilchrist, Kane Sweeney and Jonathan Hall at Uber for helpful discussions and insights in improving this manuscript.

**References**


Daulton, Samuel. 2015. NYC taxi data prediction. URL http://sdaulton.github.io/TaxiPrediction


Uber. 2018a. 10 billion trips. URL https://www.uber.com/newsroom/10-billion


E-Companion

EC.1. Nomenclature

\begin{tabular}{ll}
L & Total number of drivers \\
O & Number of open drivers \\
T & Duration of a trip \\
Y & Trip throughput \\
\eta & En route time \\
\eta(\cdot) & En route time as a function of open drivers \\
C(\cdot) & Inverse function of \eta(\cdot) \\
\phi & Waiting window \\
\lambda & Rider arrival rate \\
D & Actual request rate \\
p & Trip price \\
E & Expected total rider waiting time \\
e & Expected driver earnings per unit of time \\
\tau & Commission rate of the platform \\
l(\cdot) & Supply elasticity curve \\
q & Pool-matching probability \\
f & Expected number of open drivers consumed by a single request \\
\eta_L & The en route time that satisfies market clearance equation \\
\hat{Q} & Highest trip throughput \\
\end{tabular}

EC.2. Proof of Propositions and Theorems

EC.2.1. Proof of Proposition \[1\]

Proof. Let us show the en route time as a function of the number of open drivers in the form of

\[ \eta(O) \propto O^{-\frac{1}{2}}, \]  

(1)
on a two-dimensional plane. Note that the open drivers are assumed to be uniformly distributed on a two-dimensional plane. Suppose the intensity of drivers per unit of area is \( \mu \), and without of loss of generality, assume the pickup location is at the origin. Then the number of drivers which are within distance \( r \) from the pickup locations follow a Poisson distribution with rate \( \mu \pi r^2 \). Furthermore, the probability that the closet driver to the pickup location is at least \( r \) away is equivalent to the probability that zero drivers are within distance \( r \) from the pickup location. Denote the random variable pickup distance as \( D \). Then we have

\[ P(D \leq r) = 1 - \exp(-\mu \pi r^2) \]

Therefore, the probability density distribution (p.d.f.) of \( D \) is

\[ P(D = r) = \frac{dP(D \leq r)}{dr} = 2\mu \pi r \exp(-\mu \pi r^2). \]

Therefore, the expected pickup distance, i.e., the expectation of \( D \) satisfies

\[ E[D] = \int_0^\infty 2\mu \pi r^2 \exp(-\mu \pi r^2)dr = \frac{1}{2}\mu^{-\frac{1}{2}}. \]  

(2)

Note that the total number of open drivers \( O \) is proportional to \( \mu \), and by assuming a constant travel speed the en route time \( \eta \) is proportional to the expected pickup distance \( E[D] \). By Eq. \[ 2 \] we could see Eq. \[ 1 \] holds. \( \Box \)
EC.2.2. Proof of Proposition 2

Proof. Recall the trip throughput equation Eq. (4.4)

\[ Y(\eta, L) \cdot f(Y(\eta, L), \phi) = \frac{L - C(\eta)}{\eta + T}. \]  

(3)

Notice that \( \lim_{\eta \to \eta(L)} C(\eta) = L \) and \( \lim_{\eta \to \infty} C(\eta) = 0 \), we have

\[ \lim_{\eta \to \eta(L)} \frac{L - C(\eta)}{\eta + T} = 0, \quad \text{and} \quad \lim_{\eta \to \infty} \frac{L - C(\eta)}{\eta + T} = 0. \]

Recall that the driver consumption fraction \( f(Y, \phi) \) in defined by Eq. (4.2) is bounded between \( 1/2 \) and 1. This implies

\[ \frac{L - C(\eta)}{\eta + T} \leq Y(\eta, L) \leq 2 \cdot \frac{L - C(\eta)}{\eta + T}. \]

Therefore,

\[ \lim_{\eta \to \eta(L)} Y(\eta, L) = 0, \quad \text{and} \quad \lim_{\eta \to \infty} Y(\eta, L) = 0. \]

Furthermore, note that \( \frac{L - C(\eta)}{\eta + T} \propto O(\eta^{-1}) \) as \( \eta \to \infty \), we have \( Y(\eta, L) \propto O(\eta^{-1}) \) as \( \eta \to \infty \). The first statement in Proposition 2 holds.

Next let us show the second statement. To show \( Y(\eta, L) \) is increasing in \( L \), it is sufficient to show \( \frac{\partial Y}{\partial L} > 0 \). Taking derivative with respect to \( L \) on both sides for Eq. (4.4), we have

\[ \frac{\partial Y}{\partial L} \cdot \left( f(Y, \phi) + \frac{\partial f}{\partial Y}(Y, \phi) \cdot Y \right) = \frac{1}{\eta + Y} > 0. \]

Therefore, it is sufficient to show \( f(Y, \phi) + \frac{\partial f}{\partial Y}(Y, \phi) \cdot Y > 0 \). From Assumption 2.2, we know \( f(Y, \phi) \cdot Y \) is increasing in \( Y \). Taking the derivative of \( f(Y, \phi) \cdot Y \) with respect to \( Y \), we have \( f(Y, \phi) + \frac{\partial f}{\partial Y}(Y, \phi) \cdot Y > 0 \). Proof is thus complete.

EC.2.3. Proof of Proposition 3

![Figure EC.1 Intersection of trip throughput and effective demand rate functions.](image-url)
Proof. In order to prove this proposition, we leverage properties of functions \( \hat{D}(\eta) \) and \( Y(\eta, L) \) discussed in Section 4. Most of these properties are schematically visualized in Figure EC.1.

Let us first show for any fixed \( L \), the market clearance equation (4.6) admits at least one solution. Due to the finiteness of \( \eta(L) \), we also have \( \hat{D}(\eta(L)) > 0 \). Therefore,

\[
\hat{D}(\eta(L)) > Y(\eta(L), L).
\]

On the other hand, from Assumption 1.4 and Eq. (4.1), we know \( \hat{D}(\eta) \propto o(\eta^{-1}) \) as \( \eta \to \infty \). At the same time, \( Y(\eta, L) \propto O(\eta^{-1}) \) as \( \eta \to \infty \) according to Proposition 2. Therefore, there exists a sufficiently large \( \eta_a \) such that

\[
\hat{D}(\eta_a) < Y(\eta_a, L).
\]

Due to the continuity of \( \hat{D}(\cdot) \) and \( Y(\cdot, L) \), there exists at least one solution \( \eta \in (\eta(L), \infty) \) to the market clearance equation (4.6).

Next, let us show the highest trip throughput resulted from the equation (if there are multiple) is increasing in \( L \). As denoted in the statement of Proposition 3, let \( \hat{Q}(L; \phi, p) \) be the highest trip throughput resulted from the equation. Further denote the corresponding solution of \( \eta \) by \( \eta_L \). Thus, we have

\[
\hat{Q}(L; \phi, p) \overset{\Delta}{=} Y(\eta_L, L) = \hat{D}(\eta_L).
\]

It is sufficient to show \( \forall L' > L, \)

\[
\hat{Q}(L'; \phi, p) > \hat{Q}(L; \phi, p).
\]

From the second statement of Proposition 2, \( Y(\eta, L) \) is increasing in \( L \) for any \( \eta \). It follows that

\[
\hat{D}(\eta_L) = Y(\eta_L, L) < Y(\eta_L, L').
\]

Therefore, there exists an \( \eta_{L'} < \eta_L \) such that

\[
\hat{D}(\eta_{L'}) = Y(\eta_{L'}, L').
\]

Therefore

\[
\hat{Q}(L'; p, \phi) \geq Y(\eta_{L'}, L') = \hat{D}(\eta_{L'}) = \hat{D}(\eta_L) = \hat{Q}(L; p, \phi),
\]

noting that function \( \hat{D}(\cdot) \) is decreasing in \( \eta \). Proof is thus complete.

EC.2.4. Proof of Theorem 1

Proof. The proof is similar to the one in Castillo et al. (2017). Note that below we suppress the dependency of \( \hat{Q}(\cdot) \) and \( \hat{L}(\cdot) \) on \( (\phi, p) \) for simplicity. We define a point \( (L^*, Q^*) \) as a stable market equilibrium, i.e., a stable solution of (4.8), if there exists a small neighborhood \( \epsilon > 0 \) such that, starting from any point \( (L, Q) \) that satisfies \( L \in [L^*, L^* + \epsilon] \) and \( Q \in [Q^*, Q^* + \epsilon] \), the following two sequences

\[
L \to \hat{L}(\hat{Q}(L)) \to \hat{L}(\hat{Q}(\hat{L}(\hat{Q}(L)))) \to \ldots \quad (6a)
\]

\[
Q \to \hat{Q}(\hat{L}(Q)) \to \hat{Q}(\hat{L}(\hat{Q}(\hat{L}(Q)))) \to \ldots \quad (6b)
\]

converge to \( L^* \) and \( Q^* \), respectively. (We consider only positive perturbations \( (L, Q) > (L^*, Q^*) \) in the definition of stability because we are interested in the highest equilibrium, while other perturbations may lead to another equilibrium which is lower.)

In light of the fact that \( \hat{L}(0) = 0, \hat{Q}(0) = 0 \), and both \( \hat{L}(\cdot) \) and \( \hat{Q}(\cdot) \) are increasing (Proposition 3 and assumption on the supply curve \( I(\cdot) \)), we show that an equilibrium exists either at \((0, 0)\) or at certain value pair \((L^*, Q^*)\) such that \( L^* > 0 \) and \( Q^* > 0 \). In particular, we will show that the
highest equilibrium is always stable. Note that \((0,0)\) is the highest if \(\hat{Q}(-)\) and \(\hat{L}(-)\) do not cross at \(L,Q > 0\).

Proof by contradiction. Suppose \((L^*,Q^*)\) is the highest equilibrium, and it is not stable. Then by the above definition of stable equilibrium we know \(\exists L' > L^*\) such that \(\hat{L}(\hat{Q}(L')) > L'\). Otherwise, since the two function \(\hat{L}(-)\) and \(\hat{Q}(-)\) are increasing, the following two sequences

\[
L' \rightarrow \hat{L}(\hat{Q}(L')) \rightarrow \hat{L}(\hat{Q}(\hat{L}(\hat{Q}(L')))) \rightarrow \cdots
\]

\[
\hat{Q}(L') \rightarrow \hat{Q}(\hat{L}(\hat{Q}(L'))) \rightarrow \hat{Q}(\hat{L}(\hat{Q}(\hat{L}(\hat{Q}(L'))))) \rightarrow \cdots
\]

are non-increasing and hence converging to an equilibrium solution above or equal to \((L^*,Q^*)\). By the assumption that \((L^*,Q^*)\) is the highest equilibrium, the two sequences above converge to \((L^*,Q^*)\), which implies \((L^*,Q^*)\) is stable. Therefore, such a \(L'\) exists. Combined with the fact that the function \(\hat{L}(\hat{Q}(-))\) is bounded from above (since \(l(-)\) is bounded), then there exists a \(L'' > L' > L^*\) such that \(\hat{L}(\hat{Q}(L'')) = L''\). Then it follows \(\hat{Q}(L'') = \hat{Q}(\hat{L}(\hat{Q}(L'')))\). Therefore, by definition \((L'',\hat{Q}(L''))\) is an equilibrium, which contradicts with the assumption that \((L^*,Q^*)\) is the highest equilibrium. Proof is thus complete. \(\square\)

**EC.3. Additional Numerical Results**

**EC.3.1. Data and Model Calibration**

We present the summary statistic of the data used to calibrate the model in Figure EC.2 below, including rider sessions, completed trips, average surge multipliers, and supply shifters. The plots in the top of Figure EC.2 show the number of rider sessions and completed trips per hour and km². Note that there are two peaks in the statistics, corresponding to the morning and evening rush hours, respectively. The plots in the bottom of Figure EC.2 show the average surge multipliers and supply shifters by hour of day, respectively. Note that the supply shifters follow a similar hour-of-day trend as rider sessions where it has a morning peak and an evening peak.

Figure EC.3 depicts the empirical fractions of rider sessions that request rides, as functions of price and waiting time in blue lines. The red lines are the estimated curves using maximum likelihood estimation, and the light blue areas represent confidence interval for the empirical fractions. We could also observe that the noises are higher in the estimation when the price and waiting time are higher, due to the natural bias towards low price and waiting time.

**EC.3.2. DW as A Single Lever**

We examine the performance of DW as a single marketplace lever when DP is absent. Specifically, we randomly fix the price at \(p = 1.02\), and compute the market equilibria for different waiting windows, as illustrated in the left plot in Figure EC.4. We could see that under fixed price \(p = 1.02\), the marketplace exits the WGC zone when the waiting window is slightly above 8 minutes. This implies that DW is an effective marketplace lever to alleviate the WGC phenomenon.

We also investigate the interactions between DP and a fixed waiting window and the resulting impact to welfare. In the right plot of Figure EC.4 we show welfare as a function of price under fixed waiting windows \(\phi = 2\) minutes and \(\phi = 0\), respectively. We can see that having a short waiting window is able to reduce the optimal price slightly, indicating a beneficial trade-off between price and rider waiting time while maintaining the same level of welfare. Another observation is that having a waiting window reduces welfare when price is sufficiently high, which implies that jointly optimizing DP and DW is necessary to not hurt the marketplace in some cases.

**EC.3.3. Additional Performance Metrics**

We examine the total rider waiting time, driver waiting time, en route time, and total driver off-trip time (driver waiting time plus en route time) by hour of day under the four approaches in Section 4.3, as summarized in Figure EC.5 below. In all the plots, the solid red line represents the metric.
under DP with DW, the solid blue line represents the metric under DP without DW, the dot blue line represents the metric under static pricing without DW, and the dot red line represents the metric under static pricing with DW. We make the following observations:

- Rider needs to wait the longest time during rush hour under static pricing with DW, while the rider waiting time under the rest three approaches are relatively stable. This is to trade-off
with driver off-trip time, noting that driver off-trip time is the lowest during rush hour under static pricing with DW.

- DP is able to stabilize both ride waiting time and driver off-trip time.