Dispatching Through Pricing: Modeling Ride-Sharing and Designing Dynamic Prices

Mengjing Chen[1], **Weiran Shen**[1], **Pingzhong Tang**[1] and **Song Zuo**[2*]
[1]Institute for Interdisciplinary Information Sciences, Tsinghua University
[2]Google Research
ccchmj@qq.com, {emersonswr, kenshinping}@gmail.com, szuo@google.com

Abstract

Over the past few years, ride-sharing has emerged as an effective way to relieve traffic congestion. A key problem for the ride-sharing platforms is to come up with a revenue-optimal (or GMV-optimal) pricing scheme and a vehicle dispatching policy that incorporate geographic and temporal information. In this paper, we aim to tackle this problem via an economic approach. Modeled naively, the underlying optimization problem may be non-convex and thus hard to solve. To this end, we use a so-called "ironing" technique to convert the problem into an equivalent convex optimization one via a clean Markov decision process (MDP) formulation, where the states are the driver distributions and the decision variables are the prices for each pair of locations. Our main finding is an efficient algorithm that computes the exact revenue-optimal (or GMV-optimal) randomized pricing scheme, which naturally induces the accompany vehicle dispatching policy. We also conduct empirical evaluations of our solution through real data of a major ride-sharing platform and show its advantages over fixed pricing schemes as well as several prevalent surge-based pricing schemes.

1 Introduction

The recently established applications of shared mobility, such as ride-sharing, bike-sharing, and car-sharing, have been proven to be an effective way to utilize redundant transportation resources and to optimize social efficiency [Cramer and Krueger, 2016]. Over the past few years, intensive researches have been done on topics related to the economic aspects of shared mobility [Crawford and Meng, 2011; Pan *et al.*, 2019].

Despite these researches, the problem of designing revenue optimal pricing and vehicle dispatching schemes has been largely open and one of the main research agendas in sharing economy. There are at least two challenges when one wants to tackle this problem in the real-world applications. First of all, due to the nature of transportation, the pricing and dispatching scheme must be geographically dependent. Secondly, the pricing and dispatching scheme must take into consideration the fact that supplies and demands in these environments may change over time. As a result, it may be difficult to compute, or even to represent a pricing and dispatching scheme for such complex environments.

The dynamic ride-sharing market studied in this paper is also known to have imbalanced supply and demand, either globally in a city or locally in a particular time and location. Such imbalance in supply and demand causes severe consequences on revenues (e.g., the so-called *wild goose chase phenomenon* [Castillo *et al.*, 2017]). Surging price is a way to balance dynamic supply and demand [Chen and Sheldon, 2016] but there is no known guarantee that surge based pricing can dispatch vehicles efficiently and solve the imbalanced supply and demand. Traditional dispatching schemes [Laporte, 1992; Gendreau *et al.*, 1994; Ghiani *et al.*, 2003] focus more on the algorithmic aspect of static vehicle routing, without consider pricing. However, vehicle dispatching and pricing problem are tightly related, since a new pricing scheme will surely induce changes on supply and demand as the drivers and passengers are strategic. In this paper, our goal is to design pricing schemes that induce desirable supplies and demands.

We introduce a propose graph-based, non-strategic model for the dispatching problem that can be used for a wide range of general settings. In the graph, each node refers to a region in the city and each edge refers to a possible trip that includes a pair of origin and destination as well as a cost associated with the trip on this edge. The problem for the platform is to set a price and specify the vehicle dispatch for each edge at each time step. Drivers are considered to be non-strategic in our model, meaning that they will accept whatever offer assigned to them. The objective of the platform can either be its revenue or the Gross Merchandise Volume (GMV) or any convex combination of them.

Our model naturally induces a *Markov Decision Process* (MDP) with the driver distributions on each node as states, the price and dispatch along each edge as actions, and the revenue as immediate reward. Although the corresponding mathematical program is not convex (thus computationally hard) in general, we apply the so-called "ironing" technique that, without loss of generality, converts any non-convex in-

*This work was done while this author was in Tsinghua University.

stance to a convex one. This important step makes the exact optimal solutions can be efficiently computed and makes our result possible for real large-scale ride-sharing market. Experiment shows that our policy outperforms the two benchmarks: `FIXED` and `SURGE` pricing policies both on revenue and demand-supply balance aspects.

1.1 Related Works

Driven by real-life applications, a large number of researches have been done on ride-sharing markets. Some of them employ queuing networks to model the markets [Iglesias *et al.*, 2019; Banerjee *et al.*, 2015; Tang *et al.*, 2016]. Iglesias *et al.* [2019] describe the market as a closed, multi-class BCMP queuing network which captures the randomness of customer arrivals. They assume that the number of customers is fixed, since customers only change their locations but don't leave the network. In contrast, the number of customers is dynamic in our model and we only consider the one who asks for a ride (or sends a request to the platform). Banerjee *et al.* [2015] also use a queuing theoretic approach to analyze the ride-sharing market and mainly focus on the behaviors of drivers and customers. They assume that the drivers enter or leave the market with certain possibilities. Bimpikis *et al.* [2019] take account for the spatial dimension of pricing schemes in ride-sharing markets. They price for each region and their goal is to re-balance the supply and demand of the whole market. However, we price for each routing and aim to maximize the total revenue or social welfare of the platform. We also refer the readers to the line of researches initiated by [Ma *et al.*, 2013] for the problems about the car-pooling in the ride-sharing systems [Alonso-Mora *et al.*, 2017; Zhao *et al.*, 2014; Chan and Shaheen, 2012].

Another work closely related to ours is by Banerjee *et al.* [2017]. Their work is concurrent and has been developed independently from ours. In particular, the customers arrive according to a queuing model and their pricing policy is state-independent and depends on the transition volume. Both their and our models are built upon the underlying Markovian transitions between the states (the distribution of drivers over the graph). The major differences are: (i) our model is built for the dynamic environments with a very large number of customers (each of them is non-atomic) to meet the practical situations, while theirs adopts static discrete agent settings and it is not clear how their approach can extend to the dynamic case. (ii) they assume explicitly that the revenue curve is concave, while we don't have this key assumption. In fact, based on our evaluation of the real data, the revenue curve is often non-concave. We solve the problem via randomized pricing and transform the problem to a convex program. (iii) they prove approximation bounds of the relaxation problem, while we give exact optimal solutions of the problem by efficiently solving the convex program.

2 Model

A passenger (called "she" or "her" hereafter) enters the ride-sharing platform and sends a request including her origin and destination to the platform. The platform receives the request and determines a price for it. If the passenger accepts the price, then the platform may decide whether to send a driver (called "he" or "him" hereafter) to pick her up. Driver relocation is allowed, which means the platform is able to dispatch drivers from one place to another even there is no request to be served. By the pricing and dispatching methods above, the goal of maximizing revenue or social welfare of the entire platform can be achieved. Our model incorporates the two methods into a simple pricing problem. In this section, we define basic components of our model and consider two settings: dynamic environments with a finite time horizon and static environments with an infinite time horizon. Finally we reduce the action space and give a simple formulation.

Requests. We use a strongly connected digraph $G = (V, E)$ to model the geographical information of a city. Passengers can only take rides from nodes to nodes on the graph. When a passenger enters the platform, she expects to get a ride from node s to node t, and is willing to pay at most $x \geq 0$ for the ride. She then sends to the platform a request, which is associated with the tuple $e = (s, t)$. Upon receiving the request, the platform sets a price p for it. If the price is accepted by the passenger (i.e., $x \geq p$), then the platform tries to send a driver to pick her up. We say that the platform rejects the request, if no driver is available. A request is said to be *accepted* if both the passenger accepts the price p and there are available drivers. Otherwise, the request is considered to end immediately.

Drivers. Clearly, within each time period, the total number of accepted requests starting from s cannot be more than the number of drivers available at s. Formally, let $q(e)$ denote the total number of accepted requests along edge e, $\mathrm{OUT}(v)$ be the set of edges starting from v, and $w(v)$ be the number of currently available drivers at node v:

$$\textstyle\sum_{e\in\mathrm{OUT}(v)} q(e) \leq w(v),\ \forall v \in V. \tag{2.1}$$

In practice, both the total number of drivers and the number of requests are very large. Hence we assume the drivers and requests to be *non-atomic*. For simplicity, we normalize the total amount of drivers on the graph to be 1, and also normalize the number of requests accordingly. Therefore $w(v)$ is a real number in $[0, 1]$. Note that the amount of requests on an edge e can be more than 1, if the requests on e are more than the drivers on the graph.

Geographic information and travel time. For each accepted request on edge e, the platform has to cover a transportation cost $c_\tau(e)$ for the driver. In the meanwhile, the assigned driver, who is currently at node s, will be unavailable until he arrives at the destination t. Let $\Delta\tau(e)$ be the travel time from s to t and τ be the timestep of the driver leaving s. He will be available again at timestep $\tau + \Delta\tau(e)$ at node t. Formally, the amount of available drivers at $v \in V$ is evolving according to the following formula:

$$\begin{aligned} w_{\tau+1}(v) =& w_\tau(v) - \textstyle\sum_{e\in\mathrm{OUT}(v)} q_\tau(e) \\ &+ \textstyle\sum_{e\in\mathrm{IN}(v)} q_{\tau+1-\Delta\tau(e)}(e), \end{aligned} \tag{2.2}$$

where $\mathrm{IN}(v)$ is the set of edges ending at v. Here we add subscripts to emphasize the timestep. Throughout this paper, we focus on the discrete timestep setting, i.e., $\tau \in \mathbb{N}$.

Demand function. As we mentioned, the platform could set different prices for the requests. Such prices may vary with the request edge e, timestep t, and the driver distribution but must be independent of the passenger's private value x as it is not observable. Formally, let $D_\tau(\cdot|e) : \mathbb{R}_+ \mapsto \mathbb{R}_+$ be the *demand function* of edge e, i.e., $D_\tau(p|e)$ is the amount of requests on edge e with private value $x \geq p$ in timestep τ.[1] Then the amount of accepted requests $q_\tau(e) \leq \mathbb{E}[D_\tau(p_\tau(e)|e)]$, where the expectation is taken over the potential randomness of the pricing rule $p_\tau(e)$.[2]

Design objectives. In this paper, we consider the class of *state-irrelevant* objective functions. A function is state-irrelevant if its value only depends on the amount of accepted requests on each edge $q(e)$ but not the driver distribution of the system $w(v)$. Note that a wide range of objectives are included in this class, such as the revenue of the platform:

$$\textsc{Rev}(p, q) = \sum_{e,\tau} \mathbb{E}[(p_\tau(e) - c_\tau(e)) \cdot q_\tau(e)],$$

and social welfare of the system:

$$\textsc{Wel}(p, q) = \sum_{e,\tau} \mathbb{E}[(x - c_\tau(e)) \cdot q_\tau(e)].$$

In fact, our techniques work for any state-irrelevant objectives. Let $g(p, q)$ denote such an objective function and the dispatching and pricing problem can be formulated as:

$$\begin{aligned} \text{maximize} \quad & \sum_{e,\tau} g(p_\tau(e), q_\tau(e)|e) \\ \text{subject to} \quad & (2.1) \text{ and } (2.2). \end{aligned} \tag{2.3}$$

Static and dynamic environment. In general, our model is defined for a dynamic environment in the sense that the demand function D^τ and the transportation cost c_τ could be different for each timestep τ. In particular, we study the problem (2.3) in general dynamic environments with finite time horizon from $\tau = 1$ to T, where the initial driver distribution $w_1(v)$ is given as input. In addition, we also study the special case with *static environment* and infinite time horizon, where $D^\tau \equiv D$ and $c_\tau \equiv c$ are consistent across each timestep.

2.1 Reducing the Action Space

We rewrite the problem to an equivalent reduced form by incorporating the action of dispatching into pricing, i.e., using p to express q. The idea is straightforward: (i) for the requests rejected by the platform, the platform could equivalently set very high prices; (ii) if the platform dispatches available drivers (without requests) from node s to t, one can create virtual requests from s to t with 0 value and let the platform set price 0 for them. In fact, we can assume without loss of generality that the total amount of requests $D(0|e) \equiv 1$, because one can always add enough virtual requests for the edges with maximum demand less than 1 or remove the requests with low values for the edges with maximum demand exceeds the total driver supply 1.

As a result, we may conclude that $q(e) \leq D(p|e)$. Since our goal is to maximize the objective $g(p, q)$, raising prices to achieve the same amount of flow $q(e)$ (such that $\mathbb{E}[D(p|e)] = q(e)$) never eliminates the optimal solution.

[1] In practice, such a demand function can be predicted from historical data [Tong *et al.*, 2017; Moreira-Matias *et al.*, 2013].

[2] The randomized pricing rule may set different prices for the requests on the same edge e.

Observation 2.1. *The original problem is equivalent to the following reduced problem, where the flow variables $q_\tau(e)$ are uniquely determined by the price variables $p_\tau(e)$:*

$$\begin{aligned} \text{maximize} \quad & \textstyle\sum_{e,\tau} g(p_\tau(e), D_\tau(p_\tau(e)|e)) \\ \text{subject to} \quad & q_\tau(e) = \mathbb{E}[D(p_\tau(e)|e)] \\ & (2.1) \textit{ and } (2.2). \end{aligned} \tag{2.4}$$

3 Problem Analysis

In this section, we demonstrate how the original problem (2.4) can be equivalently rewritten as a Markov decision process with a convex objective function. Formally,

Theorem 3.1. *The original problem (2.4) of instance $\langle G, D, g, \Delta\tau\rangle$ is equivalent to a Markov decision process problem of another instance $\langle G', D', g', \Delta\tau'\rangle$ with g' being convex.*

The proof of Theorem 3.1 is immediate after Lemma 3.2 and 3.4. The equivalent Markov decision process problem could be formulated as a convex program, and hence can be solved efficiently.

3.1 Unifying Travel Time

Note that the original problem (2.4), in general, is not a MDP by itself, because the current state $w_{\tau+1}(v)$ may depend on the action $q_{\tau+1-\Delta\tau(e)}$ in (2.2). Hence our first step is to equivalently map the original instance to another instance with travel time is always 1, i.e., $\Delta\tau(e) \equiv 1$:

Lemma 3.2 (Unifying travel time)**.** *The original problem (2.4) of a general instance $\langle G, D, g, \Delta\tau\rangle$ is equivalent to the problem of a 1-travel time instance $\langle G', D', g', \Delta\tau'\rangle$, where $\Delta\tau'(\cdot) \equiv 1$.*

A working driver cannot handle another request before finishing the current one. And different requests may require different time to finish. In order to distinguish between working drivers and idle ones (can provide service immediately), a possible solution is to add an indicator variable to each driver. Instead, we tackle this problem by adding virtual nodes into the graph to replace the original edges. This operation splits the entire trip into smaller ones, and at each timestep, all drivers become available.

Proof. For edges with travel time $\Delta\tau(e) = 1$, we are done.

For edges with travel time $\Delta\tau(e) > 1$, we add $\Delta\tau(e) - 1$ virtual nodes into the graph, i.e., $v^e_1, \ldots, v^e_{\Delta\tau(e)-1}$, and the directed edges connecting them to replace the original edge e, i.e.,

$$\begin{aligned} E'(e) = \{&(s, v^e_1), (v^e_1, v^e_2), \ldots, (v^e_{\Delta\tau(e)-2}, \\ &v^e_{\Delta\tau(e)-1}), (v^e_{\Delta\tau(e)-1}, t)\}, \\ E' = \cup_{e\in E} E'(e), \quad & V' = \cup_{e\in E}\{v^e_1, \ldots, v^e_{\Delta\tau(e)-1}\} \cup V. \end{aligned}$$

We set the demand function of each new edge $e' \in E'(e)$ to be identical to those of the original edge e: $D'(\cdot|e') \equiv D(\cdot|e)$.

An important but natural constraint is that if a driver handles a request on edge e of the original graph, then he must go along all edges in $E'(e)$ of the new graph, because he cannot leave the passenger halfway. To guarantee this, we only

need to guarantee that all edges in $E'(e)$ have the same price. Also, we need to split the objective of traveling along e into the new edges, i.e., each new edge has objective function

$$g'(p, q|e') = g(p, q|e)/\Delta\tau(e), \forall e' \in E'(e).$$

One can easily verify that the above operations increase the graph size to at most $\max_{e\in E} \Delta\tau(e)^*$ times of that of the original one. In particular, there is a straightforward bijection between the dispatching behaviors of the original $G = (V, E)$ and the new graph $G' = (V', E')$. Hence we can always recover the solution to the original problem. □

3.2 Flow Formulation and Randomized Pricing

By Lemma 3.2, the original problem (2.4) can be formulated as a MDP:

Definition 3.3 (Markov Decision Process)**.** *The vehicle pricing and dispatching problem is a Markov decision process, denoted by a tuple (G, D, g, S, A, W), where $G = (V, E)$ is the given graph, D is the demand function, objective g is the reward function, $S = \Delta(V)$ is the state space including all possible driver distributions over the nodes, A is the action space, and W is the state transition rule:*

$$w_{\tau+1}(v) - w_\tau(v) = \textstyle\sum_{e\in \mathrm{IN}(v)} q_\tau(e) - \sum_{e\in \mathrm{OUT}(v)} q_\tau(e). \tag{3.1}$$

By using the pricing functions $p_\tau(e)$ as the actions, the induced flow $q_\tau(e) = \mathbb{E}[D_\tau(p_\tau(e)|e)]$, in general, is neither convex nor concave. In other words, both the reward g and the state transition W of the MDP is non-convex. With randomized pricing and the "ironing" technique, we will show that by formulating the MDP with the *flows* $q_\tau(e)$ as actions, the corresponding MDP is convex. This is a key step that enables efficient computation of the exact optimal pricing scheme.

Lemma 3.4 (Flow-based MDP)**.** *In MDP (G, D, g, S, A, W) with all possible flows as the action set A, i.e., $A = [0, 1]^{|E|}$, the state transition rules are linear functions of the flows and the reward functions g are convex functions of the flows.*

Proof. We first rewrite the prices $p_\tau(e)$ as functions of the flows $q_\tau(e)$. In general, since the prices could be randomized, the inverse function of $q_\tau(e) = \mathbb{E}[D_\tau(p_\tau(e)|e)]$ is not unique.

Note that conditional on fixed flows $q_\tau(e)$, the state transition of the MDP is also fixed. In this case, different prices yielding such specific flows only differ in the rewards. In other words, it is without loss of generality to let the inverse function of prices be as follows:

$$p_\tau(e) = \arg\max_p g_\tau(p_\tau(e), q_\tau(e)|e),$$
$$\text{s.t. } q_\tau(e) = \mathbb{E}[D_\tau(p_\tau(e)|e)].$$

In particular, since the objective function g we studied in this paper is linear and weakly increasing in the prices p and the demand function $D(p|e)$ is decreasing in p, the inversed price function could be defined as follows:

- Let $g_\tau(q|e) = g_\tau\left(D_\tau^{-1}(q|e), q|e\right)$, i.e., the objective obtained by setting the maximum fixed price $p = D_\tau^{-1}(q|e)$ such that the induced flow is exactly q;
- Let $\hat{g}_\tau(q|e)$ be the *ironed objective function*, i.e., the smallest concave function that upper-bounds $g_\tau(q|e)$ (see Figure 1);

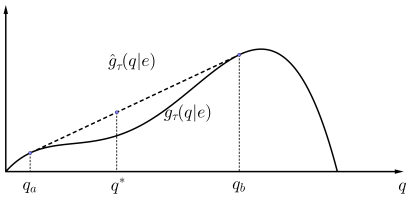


Figure 1: Ironed objective function

- For any given $q_\tau(e)$, the maximum objective on edge e is $\hat{g}_\tau(q_\tau(e)|e)$ and could be achieve by setting the price to be randomized over $D_\tau^{-1}(q'|e)$ and $D_\tau^{-1}(q''|e)$.

Finally, we prove the above claim to complete the proof of Lemma 3.4.

By the definition of $\hat{g}_\tau(q|e)$, for any randomized price p,

$$\mathbb{E}_p[g_\tau(D_\tau(p|e)|e)] \le \mathbb{E}_p[\hat{g}_\tau(D_\tau(p|e)|e)].$$

Since $\hat{g}$ is concave, applying Jensen's inequality yields:

$$\mathbb{E}_p[\hat{g}_\tau(D_\tau(p|e)|e)] \le \hat{g}_\tau\left(\mathbb{E}_p[D_\tau(p|e)]|e\right) = \hat{g}_\tau(\bar{q}|e)$$

Now it suffices to show that the upper bound $\hat{g}_\tau(\bar{q}|e)$ is attainable.

If $\hat{g}_\tau(\bar{q}|e) = g_\tau(\bar{q}|e)$, the right-hand-side can be achieved by letting $p_\tau(e)$ be the deterministic price $D_\tau^{-1}(\bar{q}|e)$.

Otherwise, let $I = (q', q'')$ be the ironed interval (where $\hat{g}_\tau(q|e) > g_\tau(q|e), \forall q \in I$ but $\hat{g}_\tau(q'|e) = g_\tau(q'|e)$ and $\hat{g}_\tau(q''|e) = g_\tau(q''|e)$) containing $\bar{q}$. Thus $\bar{q}$ can be written as a convex combination of the end points q' and q'': $\bar{q} = \lambda q' + (1 - \lambda)q''$. Note that the function $\hat{g}_\tau$ is linear within the interval I. Therefore

$$\lambda g_\tau(q'|e) + (1 - \lambda)g_\tau(q''|e) = \lambda\hat{g}_\tau(q'|e) + (1 - \lambda)\hat{g}_\tau(q''|e)$$
$$= \hat{g}_\tau(\lambda q' + (1 - \lambda)q''|e) = \hat{g}_\tau(\bar{q}|e).$$

In other words, the upper bound $\hat{g}_\tau(\bar{q}|e)$ could be achieved by setting the price to be q' with probability λ and q'' with probability $1 - \lambda$. In the meanwhile, the flow $q_\tau(e)$ would retain the same. □

Proof of Theorem 3.1. The theorem is implied by Lemma 3.2 and Lemma 3.4. In particular, the reward function is the ironed objective function $\hat{g}$. □

In the rest, we will focus on the equivalent problem:

$$\begin{aligned} \text{maximize} \quad & \textstyle\sum_{e,\tau} \hat{g}_\tau(q_\tau(e)|e) \\ \text{subject to} \quad & (2.1) \text{ and } (3.1). \end{aligned} \tag{3.2}$$

4 Optimal Solution in Static Environment

We focus on the case where the environment is static, hence the objective function is unchanging over time, i.e., $\forall\tau \in [T], \hat{g}_\tau(q|e) \equiv \hat{g}(q|e)$. Our goal is to find the optimal stationary policy that maximizes the objective function, i.e., the decisions q_τ depend only on the current state w_τ.

In this section, we discretize the MDP problem and focus on *stable policies*. We show that for any discretization scheme, the optimal stationary policy of the induced *discretized MDP* is dominated by a stable dispatching scheme. Then we formulate the stable dispatching scheme as a convex problem, which means the optimal stationary policy can be found in polynomial time.

Definition 4.1. *A stable dispatching scheme is a pair of state and policy (w_τ, π), such that if policy π is applied, the distribution of available drivers does not change over time, i.e., $w_{\tau+1}(v) = w_\tau(v)$.*

In particular, under a stable dispatching scheme, the state transition rule (3.1) is equivalent to the following form:

$$\sum_{e\in \mathrm{OUT}(v)} q(e) = \sum_{e\in \mathrm{IN}(v)} q(e). \tag{4.1}$$

Definition 4.2. *Let $\mathcal{M} = (G, D, \hat{g}, S, A, W)$ be the original MDP problem. A discretized MDP $\mathcal{DM}$ with respect to $\mathcal{M}$ is a tuple $(G_d, D_d, \hat{g}_d, S_d, A_d, W_d)$, where $G_d = G$, $D_d = D$, $\hat{g}_d = \hat{g}$, $W_d = W$, S_d is a finite subset of S, and A_d is a finite subset of A that contains all feasible transition flows between every two states in S_d.*

Theorem 4.3. *Let $\mathcal{DM}$ and $\mathcal{M}$ be a discretized MDP and the corresponding original MDP. Let $\pi_d : S_d \to A_d$ be an optimal stationary policy of $\mathcal{DM}$. Then there exists a stable dispatching scheme (w, π), such that the time-average objective of π in $\mathcal{M}$ is no less than that of π_d in $\mathcal{DM}$.*

Proof. Consider policy π_d in $\mathcal{DM}$. Starting from any state in S_d with policy π_d, let $\{w_\tau\}_0^\infty$ be the subsequent state sequence. Since $\mathcal{DM}$ has finitely many states and policy π_d is a stationary policy, there must be an integer n, such that $w_n = w_m$ for some $m < n$ and from timestep m on, the state sequence becomes a periodic sequence. Define

$$\bar{w} = \tfrac{1}{n-m}\sum_{k=m}^{n-1} w_k, \quad \bar{q} = \tfrac{1}{n-m}\sum_{k=m}^{n-1} \pi_d(w_k)$$

Denote by $\pi_d(w_k|e)$ or $q_d(e)$ the flow at edge e of the decision $\pi_d(w_k)$. Sum the transition equations for all the timesteps $m \le k < n$, and we get:

$$\begin{aligned}&\textstyle\sum_{k=m}^{n-1} w_{k+1}(v) - \sum_{k=m}^{n-1} w_k(v)\\ =&\textstyle\sum_{k=m}^{n-1}\left(\sum_{\mathrm{IN}(v)} \pi_d(w_k|e)\right) - \sum_{k=m}^{n-1}\left(\sum_{\mathrm{OUT}(v)} \pi_d(w_k|e)\right)\end{aligned}$$

$$\bar{w}(v) = \bar{w}(v) - \left(\textstyle\sum_{\mathrm{OUT}(v)} \bar{q}(e)\right) + \left(\sum_{\mathrm{IN}(v)} \bar{q}(e)\right)$$

Policy π_d is a valid policy, so $\forall v \in V$ and $\forall m \le k < n$:

$$\textstyle\sum_{\mathrm{OUT}(v)} q_k(e) \le w_k(v)$$

Summing over k, we have $\sum_{\mathrm{OUT}(v)} \bar{q}(e) \le \bar{w}(v)$.

Now consider the original problem $\mathcal{M}$. Let $w = \bar{w}$ and π be any stationary policy such that:

- $\pi(w) = \bar{q}$;
- starting from any state $w' \neq w$, policy π leads to state w within finitely many steps.

Note that the second condition can be easily satisfied since the graph G is strongly connected.

With the above definitions, we know that (w, π) is a stable dispatching scheme. Now we compare the objectives of the two policies π_d and π. The time-average objective function is not sensitive about the first finitely many immediate objectives. And since the state sequences of both policies π_d and π are periodic, Their time-average objectives can be written as:

$$\begin{aligned}\mathrm{OBJ}(\pi_d) &= \tfrac{1}{n-m}\textstyle\sum_{k=m}^{n-1}\sum_{e\in E} \hat{g}(q_d(e)|e)\\ \mathrm{OBJ}(\pi) &= \textstyle\sum_{e\in E} \hat{g}(\bar{q}(e)|e)\end{aligned}$$

order	driver	user	origin	dest	price	timestamp
hash	hash	hash	hash	hash	37.5	01-15 00:35:11

Table 1: An example of a row in the dataset, where "hash" means hash strings of exact values that we don't show here.

By Jensen's inequality, we have:

$$\begin{aligned}\mathrm{OBJ}(\pi_d) &= \tfrac{1}{n-m}\textstyle\sum_{k=m}^{n-1}\sum_{e\in E} \hat{g}(q_d(e)|e)\\ &\le \textstyle\sum_{e\in E} \hat{g}\left[\left(\tfrac{1}{n-m}\sum_{k=m}^{n} q_d(e)\right)|e\right]\\ &= \textstyle\sum_{e\in E} \hat{g}(\bar{q}(e)|e) = \mathrm{OBJ}(\pi)\end{aligned}$$

□

With Theorem 4.3, we know there exists a stable dispatching scheme that dominates the optimal stationary policy of the our discretized MDP. Thus we now only focus on stable dispatching schemes. The problem of finding an optimal stable dispatching scheme can be formulated as a convex program with linear constraints:

$$\begin{aligned}\text{maximize}\quad & \textstyle\sum_{e\in E} \hat{g}(q|e)\\ \text{subject to}\quad & (2.1) \text{ and } (4.1).\end{aligned} \tag{4.2}$$

Because $\hat{g}(q|e)$ is concave, the program is convex. Since all convex programs can be solved in polynomial time, our algorithm for finding optimal stationary policy of maximizing the objective functions is efficient.

5 Empirical Analysis

We conduct experiments to demonstrate the performance of our algorithms. Two benchmark policies, `FIXED` and `SURGE`, are compared with our pricing policy. The analysis results include demand-supply balance and instantaneous revenue in both static and dynamic environments.

5.1 Dataset

We perform our empirical analysis based on a public dataset from a major ride-sharing company.[3] The dataset includes the orders in a city for three consecutive weeks (Jan. 1st, 2016 $\sim$ Jan. 21st, 2016) and the total number of orders is more than 8.5 million. An order is created when a passenger sends a ride request to the platform.

Each order consists of a unique order ID, a passenger ID, a driver ID, an origin, a destination, an estimated price, and the timestamp when the order is created (see Table 1 for example). The driver ID might be empty if no driver was assigned to pick up the passenger. For ease of presentation, we relabel the region IDs in descending order of their popularities.

The travel time from nodes to nodes and demand curves for edges are known in our model. By applying a standard linear regression, the unit price per minute can be calculated and then travel time can be inferred from the order price. For the demand curves, we observe the values of each edge and fit them to lognormal distributions. The data process is described in the full version due to the lack of space.

[3]The dataset is provided by Didi for an algorithm competition.

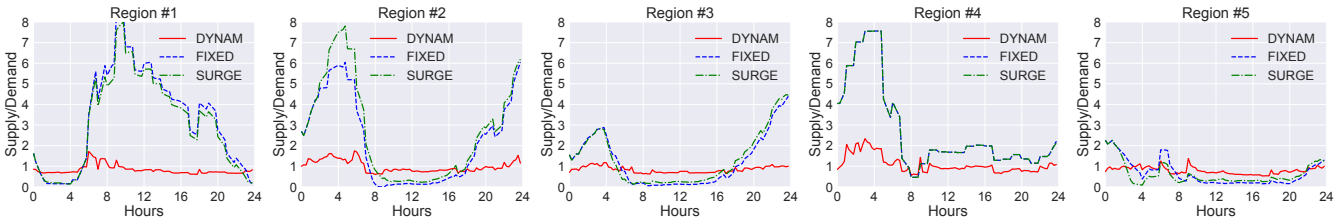
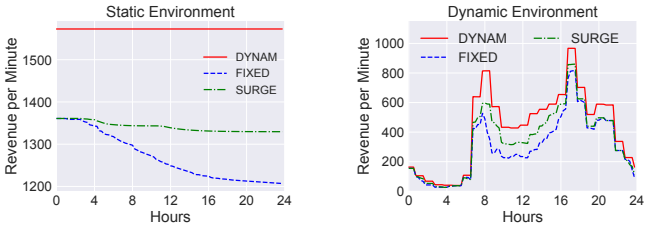


Figure 2: Instantaneous supply ratios for different regions.



(a) Static environment (b) Dynamic environment

Figure 3: Instantaneous revenue in different environments.

5.2 Benchmarks

We consider two benchmark policies:

- FIXED: fixed per-minute pricing, i.e., the price of a ride equals to the estimated travel time multiplied by a constant per-minute price α.
- SURGE: the price of a ride equals to the estimated travel time times $\alpha\beta$, with α the same as FIXED and $\beta \geq 1$ as a surge multiplier that depends on demand and supply .

We compare our dynamic pricing policy DYNAM with these two benchmarks in both static and dynamic environments.

5.3 Performance

In the static environment, we use the average of the statistics of all 21 days as the inputs to our model. We can instantiate the convex program (4.2) and solve via standard gradient descent algorithms. The length of each timestep is set to be 15 minutes and the number of steps in simulation is 96 (so 24 hours in total). For both FIXED and SURGE, we use the per-minute price fitted from data as the base price, $\alpha = 0.5117$, and allow the surge ratio β to be in $[1.0, 5.0]$. Figure 3(a) shows how the instantaneous revenues evolve as the time goes by, where DYNAM on average outperforms FIXED and SURGE by roughly 24% and 17%, respectively.

Our policy DYNAM is stationary under the static environment, so the instantaneous revenue is constant (the red horizontal line). The instantaneous revenue curves of both FIXED and SURGE are decreasing and FIXED is decreasing much faster. The observation reflects that both FIXED and SURGE are not doing well in dispatching the vehicles: FIXED simply never balances the supply and demand, while SURGE shows better control in the balance of supply and demand because the policy seeks to balance the demand with local supply when supply can not meet the demand. However, neither of them really balance the global supply and demand, so the instantaneous revenue decrease as the supply and demand become more unbalanced.

In the dynamic environment, limited by computing resource, the parameters (i.e., the demand functions and the total number of requests) are estimated based on the statistics of *each hour* but averaged over different days and we only use the data from the weekdays (14 days in total)[4] among the most popular 5 regions (covering over 50% requests). We instantiate the convex program (3.2) for the dynamic environment and solve via the `fmincon` function. FIXED and SURGE are set the same as the static environment.

Figure 3(b) shows the instantaneous revenue along the simulation. The relationship DYNAM $\succ$ SURGE $\succ$ FIXED holds almost surely. Moreover, the advantages of DYNAM over the other two policies are more significant at the high-demand "peak times". For example, at 8 a.m., DYNAM ($\sim$800) outperforms SURGE ($\sim$600) and FIXED ($\sim$500) by roughly 33% and 60%, respectively.

Demand-Supply Balance

Balancing the demand and supply is not the goal of our dispatching policy. However, such balancing abilities are important for the real market. In Figure 2, we plot the *supply ratios* (defined as the local instantaneous supply divided by the local instantaneous demand) for all the 5 regions during the 24 hours of the simulation.

From the figures, we can easily check that comparing with the other two lines, the red line (the supply ratio of DYNAM) tightly surrounds the "balance" line of 1, which means that the number of available drivers at any time and at each region is close to the number of requests sent from that region at that time. The lines of other two policies sometimes could be very far from the "balance" line, that is, the drivers under policy FIXED and SURGE are not in the location where many passengers need the service.

As a result, our policy DYNAM shows much stronger power in vehicle dispatching and balancing demand and supply in dynamic ride-sharing systems. Such advanced techniques in dispatching can in turn help the platform to gain higher revenue through serving more passengers.

Acknowledgements

This paper is supported in part by the National Natural Science Foundation of China Grant 61561146398, a China Youth 1000-talent Program, and an Alibaba Innovative Research Program. We thank the anonymous reviewers for their helpful comments.

[4]The reason that we only use data from weekdays is that the dynamics of demands and supplies in weekdays do have similar patterns but quite different from the patterns of weekends.

References

[Alonso-Mora *et al.*, 2017] Javier Alonso-Mora, Samitha Samaranayake, Alex Wallar, Emilio Frazzoli, and Daniela Rus. On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment. *PNAS*, page 201611675, 2017.

[Banerjee *et al.*, 2015] Siddhartha Banerjee, Carlos Riquelme, and Ramesh Johari. Pricing in ride-share platforms: A queueing-theoretic approach. 2015.

[Banerjee *et al.*, 2017] Siddhartha Banerjee, Daniel Freund, and Thodoris Lykouris. Pricing and optimization in shared vehicle systems: An approximation framework. In *Proceedings of the 2017 ACM Conference on Economics and Computation*, pages 517–517. ACM, 2017.

[Bimpikis *et al.*, 2019] Kostas Bimpikis, Ozan Candogan, and Daniela Saban. Spatial pricing in ride-sharing networks. *Operations Research*, 2019.

[Castillo *et al.*, 2017] Juan Camilo Castillo, Dan Knoepfle, and Glen Weyl. Surge pricing solves the wild goose chase. In *EC 2017*, pages 241–242. ACM, 2017.

[Chan and Shaheen, 2012] Nelson D Chan and Susan A Shaheen. Ridesharing in north america: Past, present, and future. *Transport Reviews*, 32(1):93–112, 2012.

[Chen and Sheldon, 2016] M Keith Chen and Michael Sheldon. Dynamic pricing in a labor market: Surge pricing and flexible work on the uber platform. In *Ec*, page 455, 2016.

[Cramer and Krueger, 2016] Judd Cramer and Alan B Krueger. Disruptive change in the taxi business: The case of uber. *The American Economic Review*, 106(5):177–182, 2016.

[Crawford and Meng, 2011] Vincent P Crawford and Juanjuan Meng. New york city cab drivers' labor supply revisited: Reference-dependent preferences with rationalexpectations targets for hours and income. *AER*, 101(5):1912–1932, 2011.

[Gendreau *et al.*, 1994] Michel Gendreau, Alain Hertz, and Gilbert Laporte. A tabu search heuristic for the vehicle routing problem. *Management science*, 40(10):1276–1290, 1994.

[Ghiani *et al.*, 2003] Gianpaolo Ghiani, Francesca Guerriero, Gilbert Laporte, and Roberto Musmanno. Real-time vehicle routing: Solution concepts, algorithms and parallel computing strategies. *European Journal of Operational Research*, 151(1):1–11, 2003.

[Iglesias *et al.*, 2019] Ramon Iglesias, Federico Rossi, Rick Zhang, and Marco Pavone. A bcmp network approach to modeling and controlling autonomous mobility-on-demand systems. *The International Journal of Robotics Research*, 38(2-3):357–374, 2019.

[Laporte, 1992] Gilbert Laporte. The vehicle routing problem: An overview of exact and approximate algorithms. *European journal of operational research*, 59(3):345–358, 1992.

[Ma *et al.*, 2013] Shuo Ma, Yu Zheng, and Ouri Wolfson. T-share: A large-scale dynamic taxi ridesharing service. In *ICDE*, pages 410–421. IEEE, 2013.

[Moreira-Matias *et al.*, 2013] Luis Moreira-Matias, Joao Gama, Michel Ferreira, Joao Mendes-Moreira, and Luis Damas. Predicting taxi–passenger demand using streaming data. *IEEE Transactions on Intelligent Transportation Systems*, 14(3):1393–1402, 2013.

[Pan *et al.*, 2019] Ling Pan, Qingpeng Cai, Zhixuan Fang, Pingzhong Tang, and Longbo Huang. Rebalancing dockless bike sharing systems. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, 2019.

[Tang *et al.*, 2016] Christopher S Tang, Jiaru Bai, Kut C So, Xiqun Michael Chen, and Hai Wang. Coordinating supply and demand on an on-demand platform: Price, wage, and payout ratio. 2016.

[Tong *et al.*, 2017] Yongxin Tong, Yuqiang Chen, Zimu Zhou, Lei Chen, Jie Wang, Qiang Yang, Jieping Ye, and Weifeng Lv. The simpler the better: a unified approach to predicting original taxi demands based on large-scale online platforms. In *KDD 2017*, pages 1653–1662. ACM, 2017.

[Zhao *et al.*, 2014] Dengji Zhao, Dongmo Zhang, Enrico H Gerding, Yuko Sakurai, and Makoto Yokoo. Incentives in ridesharing with deficit control. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, pages 1021–1028. International Foundation for Autonomous Agents and Multiagent Systems, 2014.