Putting Data in the Driver's Seat: Optimizing Earnings for On-Demand Ride-Hailing

Harshal A. Chaudhari
Boston University
harshal@cs.bu.edu

John W. Byers
Boston University
byers@cs.bu.edu

Evimaria Terzi
Boston University
evimaria@cs.bu.edu

ABSTRACT

On-demand ride-hailing platforms like Uber and Lyft are helping reshape urban transportation, by enabling car owners to become drivers for hire with minimal overhead. Although there are many studies that consider ride-hailing platforms holistically, e.g., from the perspective of supply and demand equilibria, little emphasis has been placed on optimization for the individual, self-interested drivers that currently comprise these fleets. While some individuals drive opportunistically either as their schedule allows or on a fixed schedule, we show that strategic behavior regarding when and where to drive can substantially increase driver income. In this paper, we formalize the problem of devising a driver strategy to maximize expected earnings, describe a series of dynamic programming algorithms to solve these problems under different sets of modeled actions available to the drivers, and exemplify the models and methods on a large scale simulation of driving for Uber in NYC. In our experiments, we use a newly-collected dataset that combines the NYC taxi rides dataset along with Uber API data, to build time-varying traffic and payout matrices for a representative six-month time period in greater NYC. From this input, we can reason about prospective itineraries and payoffs. Moreover, the framework enables us to rigorously reason about and analyze the sensitivity of our results to perturbations in the input data. Among our main findings is that repositioning throughout the day is key to maximizing driver earnings, whereas 'chasing surge' is typically misguided and sometimes a costly move.

ACM Reference Format:
Harshal A. Chaudhari, John W. Byers, and Evimaria Terzi. 2018. Putting Data in the Driver's Seat: Optimizing Earnings for On-Demand Ride-Hailing. In *WSDM 2018: 11th Eleventh ACM International Conference on Web Search and Data Mining, February 5–9, 2018, Marina Del Rey, CA, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3159652.3159721

1 INTRODUCTION

The proliferation of on-demand ride-hailing platforms like Lyft and Uber has begun to fundamentally change the nature of urban transit. In the last two years alone, the number of daily trips using ride-hailing platforms like Uber and Lyft in NYC has grown five-fold, to about 350,000 trips per day. Today, over 65,000 drivers drive on the streets of NYC as Uber or Lyft drivers. The explosive growth of these ride-hailing platforms has motivated a wide array of questions for academic research at the intersection of computer science and economics, ranging from the design of effective pricing mechanisms, to equilibrium analysis, to the design of reputation management systems for drivers, to algorithms for matching drivers with customers, as we discuss in our related work section.

While these studies consider the study of ride-hailing platforms holistically, little work has been done on optimizing strategies for individual drivers. Nevertheless, the challenge of how to maximize one's individual earnings as a driver for a ride-hailing platform like Uber or Lyft is a pressing question that millions of micro-entrepreneurs across the world now face. Anecdotally, many drivers spend a great deal of time strategizing about where and when to drive. However, drivers today are self-taught, using heuristics of their own devising or learning from one another, and employ relatively simple analytics dashboards such as SherpaShare. Indeed, rumors suggest that some drivers even collude in attempts to induce spikes in surge prices that they can then exploit. But in terms of concrete guidance, to date, there are only articles in the popular press and on blogs that offer (often contradictory) advice to ride-hailing drivers how to maximize their earnings [8, 10, 19].

In this paper, we formalize the problem of devising a driver strategy to maximize expected earnings and describe a series of dynamic programming algorithms to solve this problem under different sets of modeled actions available to the drivers. Our strategies take as input a detailed model of city-level data that constitutes a fine-grained weekly projection of forecasted demand for rides, comprising predicted spatiotemporal distributions of source-destination pairs, driver payments, transit times, and surge multipliers. The optimization framework we propose not only produces contingency plans in the form of highly optimized driving schedules and real-time in-course corrections to drivers, but also enables us to rigorously reason about and analyze the sensitivity of our output results to perturbations in the input data. Thus, we can justify the proposed strategies even under an uncertainty level in the collected data and the data model itself.

We then exemplify our results with a large-scale simulation of driving for Uber in NYC. For this simulation, we assemble a new dataset that uses both the publicly available NYC taxi rides dataset [1] as well as calls to the Uber API. From the former, we obtain information about over 200,000 taxi rides that occurred between different NYC zones. From the latter, we obtain representative pricing and traffic-time information for those trips, were they to reoccur on Uber. From this dataset, we construct a mathematical model to produce input to our algorithms. However, we view the dataset to

[1] http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml

WSDM 2018, February 5–9, 2018, Marina Del Rey, CA, USA

ACM ISBN 978-1-4503-5581-0/18/02…$15.00
https://doi.org/10.1145/3159652.3159721

be of independent interest that could subsequently be used for a multitude of other studies.

Our experiments with our methods on this dataset demonstrate the following findings. Being strategic about the areas they focus on picking up riders and the times they work, drivers can significantly increase their income, sometimes by as much as 1.5x, when compared to a naive optimization strategy. Moreover, we show that a pronounced difference between earnings holds even when there is large uncertainty in the input data. We argue that our results are therefore not purely an artifact of the NYC dataset we employ, but also have high potential to generalize. Finally, our experiments show that naively chasing surging prices does not typically lead to significant earnings gains, but it can actually introduce large opportunity costs, as drivers waste time driving to subsiding surges.

2 RELATED WORK

To the best of our knowledge, we are the first to formally address the problem of optimizing the driver's strategy in ride-hailing platforms like Uber and Lyft. Apart from some recent popular-press articles that offer, often contradictory, advice to ride-hailing drivers on how to maximize earnings, mostly via chasing surge [8, 10], the only relevant existing technical work studies other aspects of ride-hailing. Next, we discuss these works as well as some work related to optimization problems for taxi fleets.

Studies of ride-hailing platforms: Recent work has investigated the supply-side effects of specific incentives (e.g., surge pricing) that Uber and Lyft provide to drivers [20]. For example, Chen and Sheldon [7] showed a causal relationship that drivers on Uber respond to surges by driving more during high surge times, differentiating from previous work that suggests taxi drivers primarily focus on achieving earnings goals [3]. In another line of research, Chen *et al.* [6] measured many facets of Uber in NYC, including the prevalence and extent of surge pricing. Hall and Krueger [9] showed that drivers were attracted to the Uber platform due to the flexibility it offers, and the level of compensation, but that earnings per hour do not vary much with the number of hours worked. Finally, Castillo *et al.* [4] recently showed that surge pricing is responsible for effectively relocating drivers during periods of high-demand thereby preventing them from engaging in 'wild goose chases' to pick up distant customers, which only exacerbates the problem of low driver availability. These studies perform an *a posteriori* analysis of the data, but they do not focus on devising specific recommendations for drivers as we do.

In another line of work, Banerjee *et al.* [2] studied dynamic pricing strategies for ride-hailing platforms (such as Lyft) using a queuing-theoretic economic model. They showed that dynamic pricing is robust to changes in system parameters, even if it does not achieve higher performance than static pricing. More recently, Ozkan and Ward [17] looked at strategic matching between supply (individual drivers) and demand (requested rides) for Uber and Lyft. They showed that matching based on time-varying parameters like driver and customer arrival rates, and the willingness of customers to wait can achieve better performance than naively matching passengers with the closest driver. Although these works build interesting models for ride-hailing economies, they are orthogonal to ours, as they take a holistic view of such economies, while we focus on earnings of individual, self-interested drivers.

Optimization problems for taxi fleets: A considerable body of work has focused on the optimization of taxi fleets, for example building economic network models to describe demand and supply equilibria of taxi services under various tariff structures, fleet size regulations, and other policy alternatives [1, 22]. Other work seeks to optimize the allocation of taxi market resources [18]. Another direction focuses on route optimization by a centralized administrator (e.g., taxi dispatching services) [14, 16] or on maximizing occupancy and minimizing travel times in a shared-ride setting [12]. Other work has studied the supply side of the driving market from the viewpoint of behavioral economics. A seminal paper by Camerer *et al.* [3] studied cab drivers and found that inexperienced cab drivers (1) make labor supply decisions "one day at a time" instead of substituting labor and leisure across multiple days, and (2) set a loose daily income target and quit working once they reach that target. These works, however, do not focus on the design of a specific gain-optimizing strategy for drivers, as we do.

3 PROBLEM SETUP

In this section, we describe the basics of our problem setup and provide the necessary notation.

3.1 Modeling the city

Throughout the paper, we will assume that a city is divided into non-overlapping set of zones denoted by $\mathcal{X}$, and time t runs in discrete time steps. We represent a city in the form of a complete weighted directed graph $G = (\mathcal{X}, E)$ with $|\mathcal{X}| = n$ and $|E| = \binom{n}{2}$ edges, where the edge weight on edge $e(i \to j)$ corresponds to the likelihood of a driver currently at location i receiving a ride request to location j. Additionally, each edge is associated with a travel time $\tau(i, j)$, a travel cost, and a reward $r(i, j)$. In the general formulation of our problem, all of these edge attributes are time-varying, e.g., the rewards would vary with t as $r^t(i, j)$, but to avoid excess notation, we drop those superscripts in our following discussion of models and algorithms, and reintroduce them only in our experiments in Section 6. These attributes of a city, which we use as an input to our solver, are specified as follows:

Empirical transition matrix (F): Every edge $e(i \to j) \in E$ is associated with a transition probability $f(i, j) \in [0, 1]$ such that $\sum_{j \in \mathcal{X}} f(i, j) = 1, \forall i \in \mathcal{X}$.

Since the entries of **F** correspond to probabilities, the weights give rise to a *Markov Chain* with a transition matrix **F** – where each entry $f(i, j)$ denotes the probability of a passenger in zone i traveling to zone j. As we disallow trips within the same zone in our model (an assumption which could be relaxed), we let $f(i, i)$ denote the probability of a driver not finding a passenger in zone i at a given time step.

Travel time matrix (T): Every edge $e(i \to j) \in E$ is also associated with $\tau(i, j) > 0$, the travel time of a ride from zone i to zone j. These weights give us a travel time matrix **T** with entries $\tau(i, j)$.

Rewards matrix (R): Every edge $e(i \to j) \in E$ is also associated with a real valued reward $r(i, j)$ denoting the net reward for a driver delivering a passenger from zone i to zone j. The net rewards

include the driver's share of earnings from a passenger minus the sundry costs like gas, vehicle depreciation, etc. Since these earnings and costs vary with mileage and transit time, each entry in the rewards matrix $\mathbf{R}$ is of the form $r(i, j)$ = earnings(i, j) - cost(i, j).

Again, in general, all of the input matrices: $\mathbf{F}$, $\mathbf{T}$ and $\mathbf{R}$, are time-dependent, i.e., their entries could change throughout the day

3.2 Modeling the driver

Our model assumes that each driver comes with a maximum work budget of B time units, during which the driver can pick up passengers. Depending on the specific setting, the driver can work B time units consecutively or split them over a finite horizon of N time units, where $N \geq B$. As an example, a driver seeking to optimize an 8 hour work day over a 24 hour day at a ten-minute decision granularity (at most six decisions per hour), will have $B = 48$ and $N = 144$.

Home zone (i_0): Each driver has a unique home zone denoted by $i_0 \in \mathcal{X}$. We always assume that each driver starts from their home zone and returns to it at the end of each of their shifts.

Driver actions ($\mathcal{A}$): In a driver strategy, whenever faced with a choice regarding their next decision, a driver has $n + 2$ possible actions to choose from:

- *Get Passenger* (a_0): Wait for a passenger in the current zone.
- *Go Home* (a_1): Log out of the on-demand ride service, relocate to the home zone (if needed) and stop working. This action does not consume the driver's budget.
- *Relocate* ($a_2(j)$): Relocate to city zone j. This action consumes the driver's budget.

Driver policy (π): A driver policy is a sequence of time and location-dependent actions taken by a driver at different steps of the strategy. As the total number of actions taken by a driver while exhausting the budget B depends on the actual actions, the length of a driver policy π varies.

Each time and location dependent action in π, denoted by $\mathbf{a}$, can be expressed in form of a 3-tuple – $(\hat{i}, \hat{t}, \hat{a})$ where $\hat{a} \in \mathcal{A}$ refers to actual action, $\hat{i} \in \mathcal{X}$ is the zone at which action was taken and $\hat{t} \leq N$ is the time at which the action was taken. Finally, we use Π to denote the set of all possible policies.

3.3 Computing driver earnings

In this section, we describe the computation of the expected earnings of a driver who at a specific time t is in zone i and takes action a. We denote this by $E(i, t, a)$ and depending on the action a it is computed as follows.

- For action a_0 (*Get Passenger*), taken inside zone i at time t, the action earnings function is calculated as an expectation over possible rides,
$$E(i, t, a_0) = \mathbf{F}_i \cdot \mathbf{R}_i \tag{1}$$
where $\mathbf{F}_i$ and $\mathbf{R}_i$ denote the i-th rows of $\mathbf{F}$ and $\mathbf{R}$ respectively.
- For action a_1 (*Go Home*), taken inside zone i at time t, the action earnings function is simply
$$E(i, t, a_1) = -cost(i, i_0) \tag{2}$$
where we incur a negative reward due to the absence of a paying customer.
- Action $a_2(j)$ (*Relocate*), taken inside zone i at time t, takes the driver to zone $j \neq i$. Therefore, the action earnings function is
$$E(i, t, a_2(j)) = -cost(i, j) \tag{3}$$
where the driver again incurs a negative reward due to the absence of a paying customer.

3.4 Problem definition

Given input specification matrices $\mathbf{F}$, $\mathbf{T}$ and $\mathbf{R}$, as well as the driver's budget B, the *total expected earnings* of the driver with policy π is:
$$\mathcal{E}(\pi, \mathbf{F}, \mathbf{T}, \mathbf{R}, B) = \sum_{(\hat{i}, \hat{t}, \hat{a}) \in \pi} E(\hat{i}, \hat{t}, \hat{a}), \tag{4}$$
where $E(\hat{i}, \hat{t}, \hat{a})$ is computed using the Equations (1), (2) and (3).

As we seek to maximize the *total expected earnings* of the driver, we aim to solve the following optimization problem.

Problem 1 (MaxEarnings). *Given sets of time-evolving* $\mathbf{F}$, $\mathbf{T}$ *and* $\mathbf{R}$, *as well as the driver's budget* B, *find a* π^* *such that:*
$$\pi^* = \arg\max_{\pi \in \Pi} \mathcal{E}(\pi, \mathbf{F}, \mathbf{T}, \mathbf{R}, B).$$

4 DRIVER STRATEGIES AND OPTIMIZATION ALGORITHMS

We now describe the different driver strategies, which are defined based on the set of actions $\mathcal{A}$ at the driver's disposal. We also show how to optimally solve the MaxEarnings problem in polynomial time for different sets $\mathcal{A}$.

For the rest of the section, we will denote by $\Phi(i, b, t)$ the *total expected future earnings* of a driver who is in zone i at time t with budget b time units remaining. Hence, the *total expected earnings* of a driver can be expressed as $\Phi(i_0, B, N)$.

If a driver at zone i at time t with b budget units remaining either takes a passenger ride to zone j or relocates to zone j, that trip ends at time $t' = t + \tau^t(i, j)$ with remaining budget $b' = b - \tau^t(i, j)$. The total expected future earnings at that point for the driver is: $\Phi(j, b', t')$. Let $\mathbf{v}(i, b, t)$ denotes the vector of such cumulative earnings across different zones j induced when a driver takes an a_0 action i.e., $\mathbf{v}(i, b, t) = \left[\Phi(j, b', t')\right]_{j \in \mathcal{X}}$.

We now define the driver strategies as well as the solutions to the instances of the MaxEarnings problem they induce.

The *flexible-relocation* strategy: This is the most general strategy where a driver has complete freedom for choices regarding work schedule as well relocation to different zones. Specifically, a driver has a budget constraint of B time units to be consumed over a finite horizon N time units. An idle driver in zone i following this strategy has following set of available choices,
$$\mathcal{A} = \{a_0, a_1\} \cup \{a_2(j) | \forall j \in \mathcal{X}, j \neq i\} \tag{5}$$
Note that we restrict the *Relocate* actions to ones which do not result in $t \geq N$ or $b < 0$.

A driver following the *flexible-relocation* strategy chooses the action that maximizes *total expected earnings*. For this strategy,

the solution to the MaxEarnings problem can be found by the following dynamic programming (DP) recurrence:

$$\Phi(i,b,t) = \max_{a \in \mathcal{A}} \begin{cases} \mathbf{F}_i(\mathbf{R}_i + \mathbf{v}(i,b,t)), & \text{if } a = a_0 \\ -cost(i,i_0) + \Phi(i_0,b,t'), & \text{if } a = a_1 \\ \max_j\{-cost(i,j) + \Phi(j,b',t')\}, & \text{if } a = a_2(j) \end{cases} \quad (6)$$

Each of the $O(nNB)$ entries in the output of this dynamic program involves consideration of at most $O(n)$ actions. Hence, the solution to the MaxEarnings problem can be found in $O(n^2NB)$ time.

Other strategies: In addition to the general *flexible-relocation* strategy, we also consider the following three special cases to model other plausible strategies of ride-hailing drivers: the *naive*, the *relocation* and the *flexible* strategies.

In the *naive* strategy, a driver performs a random walk over the city on weekdays from 9AM - 5PM, with locations dictated exclusively by the passengers picked up. At the end of every passenger ride, the driver waits in the current zone for next passenger pickup. Hence, the only allowable action is *Get Passenger*.

In the *relocation* strategy, an idle driver in zone i has two choices: *Get Passenger* and *Relocate*. Hence, the set of allowable actions for a driver contains n different actions, one of which is *Get Passenger* and $(n-1)$ *Relocate* actions, one for each different city zone. Thus: $\mathcal{A} = \{a_0\} \cup \{a_2(j) | \forall j \in \mathcal{X}, j \neq i\}$. We remove from consideration the zones where relocating exhausts the budget or where $t \geq N$.

In the *flexible* strategy, a driver has the flexibility to decide working times, modeling a driver who uses heuristics to decide the most profitable times to work. As a result, we impose an additional constraint of a working time budget B that a driver can split over a finite horizon of N time units. Thus, this strategy aims to figure out an optimal in-expectation work schedule for the driver. At any stage, a driver can log out of the on-demand ride service and return to home zone. Hence, the set of allowable actions for a driver contains 2 different actions, *Get Passenger* and *Go Home*. Thus: $\mathcal{A} = \{a_0, a_1\}$. It is common for drivers to structure their day around a desired target earning, rather than a time budget. The *flexible* strategy also naturally computes a schedule that minimizes working time required for achieving the desired target earning.

Solving MaxEarnings for the *naive*, the *relocation* and the *flexible* strategies can be done by streamlined versions of the DP presented in Eq. (6); the details are omitted due to space constraints.

5 MAXIMIZING EARNINGS UNDER UNCERTAINTY

The primary source of variability in the input of the MaxEarnings problem is the set of empirical transition matrices **F**. In a typical application, we expect that predictive models would be employed to generate estimates of these matrices based upon observations from historical data (as we do in our own experiments). Empirically observed transition matrices may suffer from estimation errors due to the presence of external confounding factors (e.g., weather, special events inside the city) while gathering the data. As a result, the dynamic programming solution to MaxEarnings may also be sensitive to the transition probabilities. In this section, we address the question of how the results of the solutions we described in the previous section change under the assumption that there is some uncertainty (and thus noise) in the underlying empirical transition matrices we use as part of our input.

Concretely, we now assume that the empirical transition matrix (**F**) is generated from an underlying traffic matrix, or count matrix, recording trips between locations i and j.

Count matrix (C): Every edge $e(i \rightarrow j) \in E$ is associated with an integer-valued weight $c(i,j)$ that denotes the number of requests at zone i that had node j as their destination. Then, we compute frequencies $f(i,j) = \frac{c(i,j)}{\sum_k c(i,k)}$, for all outbound trips from i.

With this, we now describe how to quantify uncertainty in the rows of **F** (and the underlying **C**, by construction). This will enable us to modify the MaxEarnings into the RobustEarnings problem following ideas developed by Nilim and El Ghaoui [15].

Modeling uncertainty: We now assume that there is an underlying *true* transition matrix **P**, and the question we explore is our confidence that the **C** we observe is actually generated by the true transition matrix **P**. As before, both **P** and **C** are clearly time-dependent in practice, but for ease of exposition, we ignore the time-dependency aspect of the problem here.

We consider each row of the true transition matrix and the count matrix separately; let $\mathbf{p}$ and $\mathbf{c}$ denote any particular row of **P** and **C** respectively. Following the ideas of Kullback *et al.* [13], we have a discriminatory random variable $2\hat{I}$, which follows a χ^2 distribution with $(n-1)$ degrees of freedom. Heuristically, $2\hat{I}$ can be considered as a measure of the "divergence" of $\mathbf{c}$ from $\mathbf{p}$. Thus, for $\mathbf{c}$ to be in the $(1-\alpha)$ *(or* $100(1-\alpha)\%$*) confidence interval of* $\mathbf{p}$, we need:

$$\mathrm{F}_{\chi^2_{n-1}}\left[2\hat{I}\right] = \mathrm{F}_{\chi^2_{n-1}}\left[2\sum_{i=1}^{n}\mathbf{c}(i)\log\mathbf{c}(i) - 2n\log n - 2\sum_{i=1}^{n}\mathbf{c}(i)\log\mathbf{p}(i)\right] = 1-\alpha,$$

where $\mathbf{p}(i)$ (resp. $\mathbf{c}(i)$) is the i-th element of vector $\mathbf{p}$ (resp. $\mathbf{c}$). In the above equation, α quantifies the uncertainty that one can tolerate and is an upper bound on what one believes actually exists in the set of observations $\mathbf{p}$. Thus, we call α the *input uncertainty level*.

By setting $\beta_{\max} = \sum_{i=1}^{n}\mathbf{c}(i)\log\mathbf{c}(i)$, we get

$$\sum_{i=1}^{n}\mathbf{c}(i)\log\mathbf{p}(i) = \frac{2(\beta_{\max} - n\log n) - \mathrm{F}^{-1}_{\chi^2_{n-1}}(1-\alpha)}{2}, \quad (7)$$

where $\mathrm{F}^{-1}_{\chi^2_{n-1}}$ is the inverse of the χ^2 cdf. In other words, for all vectors $\mathbf{p}$ for which Equation (7) is satisfied, $\mathbf{c}$ is within the $(1-\alpha)$-confidence interval of $\mathbf{p}$.

Thus given **C** and α, we define the *α-feasible matrices* $\mathcal{P}_\alpha$ to be the set of true transition matrices such that for every matrix **P** in $\mathcal{P}_\alpha$ and every row $\mathbf{p}$ of **P**, Equation (7) is satisfied.

The RobustEarnings problem: Our approach is to compute the *worst-case total expected earnings* for a driver, by finding the **P** among all matrices in $\mathcal{P}_\alpha$ such that the *total expected earnings* of the driver are minimized. This quantifies the worst-case difference between the earnings computed as a solution to the MaxEarnings and the worst-case earnings of the driver, given bounded uncertainty α. We formalized this as the following problem definition:

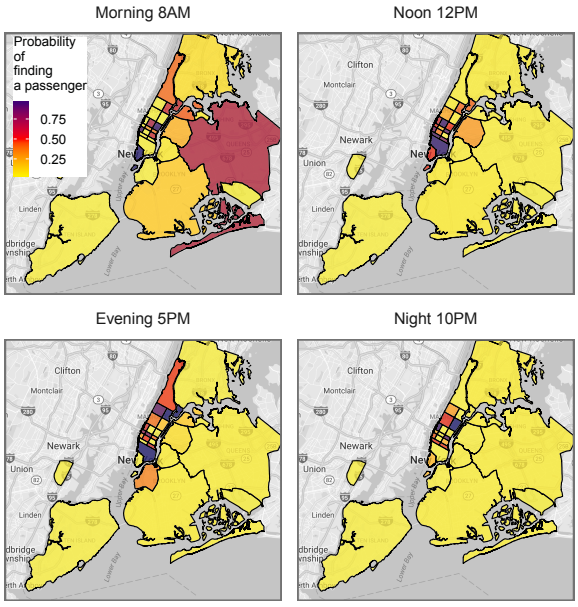

Figure 1: Probability of finding a passenger in 10 minutes across NYC zones at different times of a representative day.

Problem 2 (RobustEarnings). *Given sets of time evolving* **C**, **T** *and* **R**, *the driver's budget* B *and input uncertainty level* α, *find* $\widehat{\pi}$ *such that:*

$$\widehat{\pi} = \arg\max_{\pi \in \Pi} \min_{\mathbf{P} \in \mathcal{P}_\alpha} \mathcal{E}(\pi, \mathbf{P}, \mathbf{T}, \mathbf{R}, B).$$

Note that the above problem requires searching among all possible true transition matrices in $\mathcal{P}_\alpha$, which is a non-enumerable set. In fact, we can show (details omitted due to space constraints) that Problem 2 can be solved by enhancing the *total expected future earnings* associated with *Get Passenger* action in the dynamic-programming routines we described in Section 4 with an optimization problem. We use an off-the-shelf minimizer to solve this optimization problem. Alternatively, a bisection algorithm can approximate this problem within an accuracy δ in $O(\log(V_{\max}/\delta))$ time, where $V_{\max}$ is the maximum value of the value function [15].

6 DATA AND EXPERIMENTS

We now evaluate our strategies for drivers in practice. First, we discuss how we collect and combine the appropriate data from multiple data sources. Then, we perform a comprehensive experimental study that provides specific insights as to how NYC drivers can maximize their earnings.

6.1 Data collection and preparation

In order to evaluate our strategies, we need to construct time-evolving matrices **F**, **T** and **R** as defined in Section 3, and **C** as defined in Section 5. For this, we use two data sources: (1) the NYC taxi rides dataset[2] and (2) information we obtain from the Uber platform via queries to the Uber API.[3]

[2]http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml
[3]https://developer.uber.com/docs/riders/ride-requests/tutorials/api/introduction

Forming time-evolving matrices C and F: Our starting point is the NYC Taxi dataset (2015-2016), which contains yellow street-hail records of over 200,000 taxi rides per day with fields capturing pickup and dropoff times, location co-ordinates, trip distances, and fares. Each taxi record is accompanied with a taxi location ID for the pick-up and drop-off locations. Each location ID is associated with one of 29 non-overlapping city zones, as defined in the dataset. While the set of taxi rides is undoubtedly produced from a different ridership than Uber, it nonetheless provides a useful baseline that reflects many of the broader dynamics of ridership demand in NYC.

Given this data, we divide each 24-hour day of the week into 144 time-slices of duration 10 minutes each, indexed by their start time. To model traffic demand in the city at time t, the $c(i,j)$ entry of count matrix $\mathbf{C}^t$ is the total number of rides from zone i to zone j in a 30-minute long time window centered at time t. For example, $c(i,j)$ for the time slot [10:40, 10:50] on a Wednesday is a count of all rides from i to j that were initiated between 10:30 and 11:00 on *any* Wednesday in the dataset. Since our model disallows rides within the same zone, we ignore such rides while populating the entries of the matrix $\mathbf{C}^t$, resulting in all diagonal entries of the count matrix being zero.

To populate the entries of the empirical transition matrix $\mathbf{F}^t$, as defined in Section 3, we must estimate its diagonal entries, which correspond to the probability of not finding a ride, as well as the transition probabilities. We derive these from the data as follows. Assuming that the parameters do not change significantly within a single time-slice, let $N(\lambda)$ and $N(\mu)$ denote the number of passenger and driver arrivals in zone i in one time unit, with independent[4] Poisson arrival rates λ and μ respectively. Hence, the random variable $K = N(\lambda) - N(\mu)$ follows a Skellam distribution such that:

$$\Pr[K = k] = e^{-(\lambda+\mu)} \left(\frac{\lambda}{\mu}\right) I_k\left(2\sqrt{\lambda\mu}\right)$$

where $I_k(z)$ is the modified Bessel function of the first kind [21].

Whenever $K < 0$, there are more drivers than passengers. We assume a worst case scenario in which a driver (conceptually) joins the end of a FIFO queue for that zone. Hence, for $k \leq 0$, the driver has to wait for $(|k|+1)$ passenger arrivals for a successful passenger pickup. Then, the probability of a successful passenger pickup is:

$$\Pr[N(\lambda) = |k| + 1] = \frac{\lambda^{\left(|k|+1\right)} e^{-\lambda}}{\left(|k|+1\right)!}.$$

Thus, we can express a diagonal entry $f^t(i,i)$ as follows:

$$f^t(i,i) = 1 - \sum_{k \leq 0} \Pr[K = k] \times \Pr[N(\lambda) \geq |k| + 1].$$

For **F** to be stochastic, we set every other entry $f^t(i,j)$ to:

$$f^t(i,j) = (1 - f^t(i,i)) \times \frac{c^t(i,j)}{\sum_j c^t(i,j)}.$$

The matrix $\mathbf{F}^t$ built in this manner satisfies all our assumptions.

Figure 1 shows an example of varying estimated probabilities of successful pickups in different zones at various times of the day derived from the NYC data using the methods above. As expected,

[4]Although we assume the independence of the passenger and driver arrival processes, we can also accommodate correlated processes with slight modification.

we see that the probability of a successful pickup is higher outside Manhattan in the morning, and this trend reverses in the evening.

Forming time-evolving matrices T and R: We obtain information regarding travel times and rewards using the `estimates/price` endpoint of the Uber API. The API takes longitude and latitude of pick-up and drop-off locations and returns price estimates for all types of Uber products – UberX, UberXL and UberBlack – together with the active surge multiplier rate at the pick-up location at the time of query. We only focus on UberX, the most popular Uber product. We also use the `/products` API endpoint to get information on the base fare, minimum fare, cost per minute and cost per unit distance for UberX. However, none of the Uber API endpoints provide information about the supply of drivers or demand of passengers; we impute this information from the NYC taxi rides dataset.

To create a representative sample of the data, we "recreated" NYC taxi rides virtually on the Uber platform. Using the Uber API, we were able to take a NYC taxi ride recorded in 2015, and capture the Uber attributes of that ride exactly one year later, collecting price estimates and other data above for that virtual ride. To respect the Uber rate limit of 1,000 API requests per hour per account, we sub-sampled one ride between each pair of zones in the city every 15 minutes. We implicitly assume that price estimates, travel times, and distance of preferred travel paths by drivers do not vary significantly in 15 minutes. Every 5 minutes, we also queried the surge multiplier active within each zone.[5] Using this approach, we collected data from the Uber API for a 6-month period (Oct. 2016–Mar 2017), recreating rides that originally occurred from Oct. 2015 to Mar 2016. Thus, we built realistic estimates for $r(i, j)$ and $\tau(i, j)$ for all pairs of zones[6]. Finally, we maintained same-day of week estimates, so that, for example, travel time estimates and rewards computed for Sunday, Oct 16, 2016, were paired with frequency estimates drawn from the NYC taxi rides dataset for Sunday, Oct. 18, 2015. In the remainder of this section, we provide results for driving during one representative week in October. Our results do not vary qualitatively across different weeks, with the exception of seasonal peak days, such as New Year's Eve.

6.2 Experimental results

For all our experiments we use a single process implementation of our algorithms on a 24-core 2.9GHz Intel Xeon E5 processor with 512GB memory. Running time for *naive* and *relocation* is less than a minute, and about 5 minutes for *flexible* and *flexible-relocation*. Uncertainty analysis (Section 6.4) with an off-the-shelf minimizer takes around 3 hours. Our code has been made publicly available in order to encourage reproducible research [5].

Comparison of strategies: First, we address the question: *what is the best driver strategy?* Intuitively, it is clear that *flexible-relocation* is the best strategy, as it takes advantage of spatial as well as temporal variations in the passenger demand across NYC. In order to verify this intuition, we compare driver earnings across different strategies. Drivers following the *naive* and the *relocation* strategies are assumed to drive from 9 AM to 5 PM, a standard 8 hour workday, while those following the *flexible* or the *flexible-relocation* strategies drive for a total of 8 hours each day with a flexible schedule.

In order to evaluate the performance of our strategies, we find the solution to MaxEarnings and simulate 100 drivers, each randomly assigned a home zone, operating on these strategies on the same day of the following week, for a total of 10 weeks. Figure 2 presents a box-plot of the resulting earnings.[7].

We observe that all "smart" strategies consistently outperform *naive*; as expected. On most days, *flexible-relocation* is the strategy with the highest earnings. The median earning of a driver following the *naive* strategy on a Sunday is $104 while that of a driver following *flexible-relocation* is $177, representing a 70% increase in median earnings. Averaged over all days of the week, this results in a 47% increase in median earnings per work day when following the *flexible-relocation* strategy. Thus, our strategies do exploit the spatial and the temporal variation in demand across NYC. The results also show that for a part-time Uber driver in NYC, it is more beneficial to drive midweek, from Wednesday to Friday, and Sunday than during Saturday and Monday.

Spatial dynamics of strategies: Next, we address the question - *what are the benefits of the Relocate action?* Figure 1 already shows the spatial variation in the demand across different NYC zones at different times of the day. Intuitively, this spatial variation can cause a disparity in the driver earnings based on the zone of the driver. For example, drivers based in Manhattan should be expected to earn more than those based in Brooklyn due to persistently higher demand in Manhattan. Similarly, Figure 2 shows temporal variation in earnings across days of the week. We observe that on the days of low-demand, not only are the median earnings for *relocation* consistently higher than those for *naive* but also the inter-quartile range (IQR) and the length of whiskers for *relocation* are narrower. On days with high but localized demand like Fridays, the *relocation* strategy performs on par with the *flexible-relocation* strategy and significantly outperforms *naive*.

These observations indicate that the location-based disparity in earnings for the *naive* strategy is much larger than the *relocation* strategy. Thus, we conclude that smart relocations throughout the day prevent a driver from becoming "trapped" in low-earning neighborhoods, translating into significant increases in the earnings. This may be counterintuitive to some drivers, as a *Relocate* action (essentially an empty ride) incurs a cost to the driver. Yet, the results demonstrate that these actions, when timed appropriately, lead to earnings far higher than the costs they incur.

Temporal dynamics of strategies: Intuitively, due to the periodicity of demand, we expect driver earnings to strongly depend on the time of the day they are driving. Thus, we address the question: *what is the best time of the day to drive in order to maximize earnings?* To answer this, we simulate 1000 drivers, each randomly assigned a home zone, for each of the *flexible* and *flexible-relocation* strategies. We solve the MaxEarnings problem for both strategies and create a recommended plan of action for the simulated drivers. Then, at every step of the simulation, a driver undertakes the personalized action recommended by the strategy, corresponding to their location, the time of day and their budget remaining.

[5] Chen *et al.* [6] have observed that 90% of the surges on Uber platform have durations lasting multiples of 5 minutes.

[6] We take into account the Uber fee structure in NYC as reported by the Uber API, as well as the overall *cost per mile* estimates provided by the American Automobile Association (AAA) in order to build realistic estimates for $r(i, j)$.

[7] The lower and upper edges of the boxes in Figure 2 indicate quartiles Q1 and Q3 respectively, and length of whisker is 1.5 times IQR.

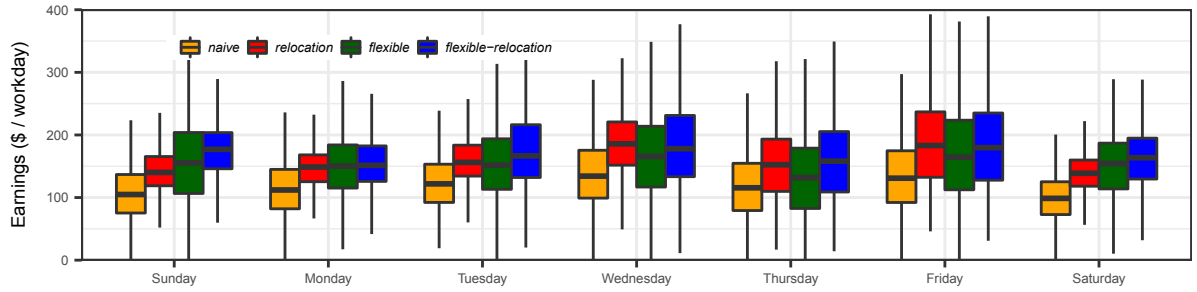

Figure 2: Daily driver earnings for different strategies averaged over different home zones on a representative day.

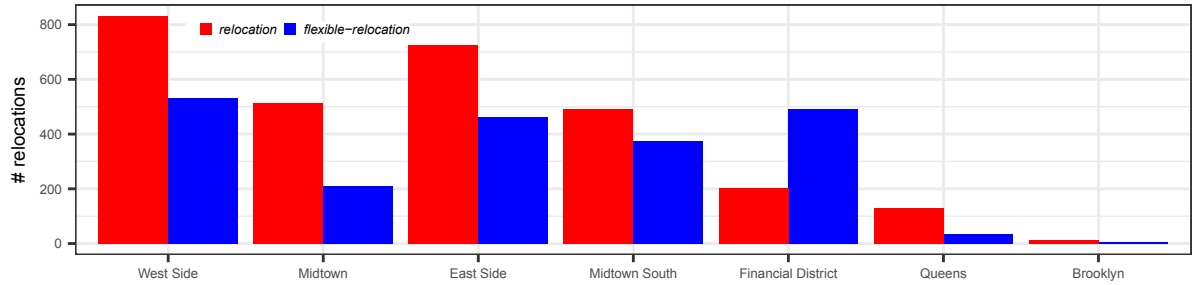

Figure 3: Contrast between preferred relocation destinations for drivers with *relocation* and *flexible-relocation* strategies on a representative day.

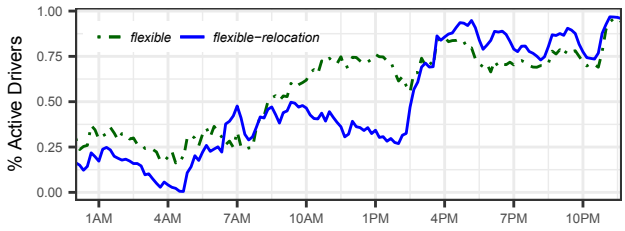

Figure 4: Active drivers with *flexible* and *flexible-relocation* strategies at different times of a representative day.

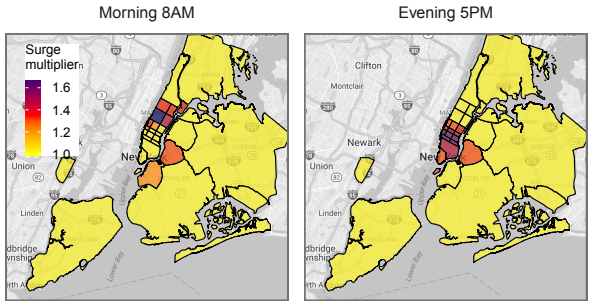

Figure 5: Active surge multiplier across NYC zones at different times of a representative day.

In Figure 4, we plot the percentage of simulated drivers driving in the city at different times of the day. We observe a noticeable difference between the "preferred" driving schedules output by *flexible* and *flexible-relocation*. In particular, a high percentage of *flexible* schedule drivers are active during the standard working hours of the day from 9AM to 6PM. This also supports our choice to evaluate fixed schedule strategies in the interval 9AM to 5PM. In contrast, the number of active drivers that follow *flexible-relocation* exhibits two distinct peaks, corresponding to the morning and the evening rush hours. Furthermore, over 50% of *flexible-relocation* drivers use their driving budget in the latter half of the day starting approximately at 3PM, continuing through until midnight. Since *flexible* and *flexible-relocation* only differ in the *Relocate* action, all observed differences are due to this action. Hence, we can conclude that the *Relocate* action is most effective in the evening hours, thereby prompting higher active percentages of *flexible-relocation* drivers at that time.

Preferred relocation zones: By simulating drivers, we can also compare the *Relocate* actions between drivers following the *relocation* strategy and those following the *flexible-relocation* strategy. The contrast between popular destinations of *Relocate* actions for drivers following the two strategies can be seen in Figure 3. Drivers following the *relocation* strategy predominantly relocate themselves to the center of Manhattan. In contrast, the drivers following the *flexible-relocation* strategy do not exhibit a clear most-preferred relocation destination. Furthermore, the *number* of relocations performed by the *relocation* strategy drivers, is, surprisingly, significantly higher than those performed by the *flexible-relocation* strategy drivers. This is due to the flexible work schedule of the latter, which allows them to drive continuously during the hours of highest demand, reducing the frequency of *Relocate* actions they take.

6.3 Surge chasing

We now turn our attention to surge pricing. Surge pricing is a feature of the Uber platform aimed at matching supply with passenger demand by increasing prices at times of high demand.According to Uber, it incentivizes drivers to start driving during the peak hours in order to efficiently meet demand with supply, albeit at a higher cost

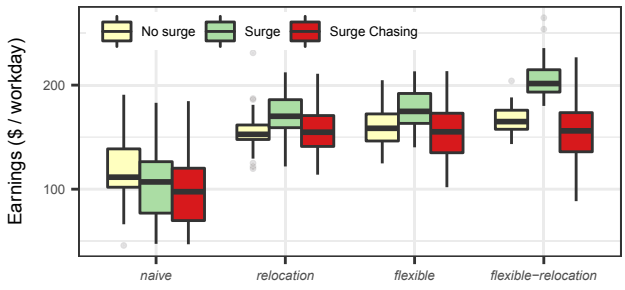

Figure 6: Exploring surge: Simulated earnings for drivers across different strategies on a representative day.

to passengers. It also decreases demand, since more price-sensitive customers drop out, as surge prices rise.

Figure 5 shows the active surge multiplier across different neighborhoods of NYC at different times of the day. This information is readily available to the drivers; however, due to uncertainty in the duration of surges as well as the proprietary nature of Uber's surge pricing algorithm, it is unclear whether drivers should relocate themselves to surging areas in order to maximize their earnings.

Next, we address the question– *Should drivers engage in surge chasing?* In order to do so, we evaluate earnings of simulated drivers in three scenarios viz., "no surge" - where we disable the surge multiplier to compute earnings; "surge" - where the multiplier is used while calculating earnings; and "surge chasing" - wherein a driver located in a non-surging zone always relocates to the zone with highest surge multiplier within a 10-minute drive radius. Simulated driver earnings in these three scenarios for each of the strategies are shown in Figure 6. We observe that blind "surge chasing" leads to lower earnings irrespective of the strategy being followed. Figure 6 reinforces our previous observation regarding the high variance of the *naive* strategy. At times, drivers following the *naive* strategy with surge multiplier enabled may earn less than when it is disabled. For other strategies, "surge chasing" consistently fails to provide any tangible benefits as compared to following the predetermined strategy. We conclude that actively and blindly chasing the surge is an ill-advised strategy and may lead to losses. Furthermore, surges last for short durations, and an unsuccessful surge chase may land a driver in a sub-optimal location with respect to longer term earnings. Note that although the NYC taxi demand data strongly correlates with active surge multipliers, we do currently model the impact of surge multiplier on consumer demand. This should be considered a limitation of our study.

6.4 Effect of uncertainty

Our experiments indicate that our strategies always outperform a *naive* strategy that is likely prevalent among Uber drivers. However, all our strategies use historical data. Consequently, our results can potentially be sensitive to perturbations of the empirically-observed transition matrices. Thus, we can only conclude that our results are robust if the drivers following one of the *relocation*, *flexible* and *flexible-relocation* strategies have higher earnings than those following *naive*, even when the input data is perturbed.

Hence, the question we have to address is the following: *Are the conclusions we drew above robust to perturbations of the empirical transition matrices?* We do so using the framework we developed in Section 5: we solve the RobustEarnings problem for each of the four strategies for increasing levels of uncertainty (α) using the Sequential Least Squares Programming (SLSQP) minimizer implementation provided by Jones *et al.* [11].

Figure 7 shows the effect of increasing uncertainty on the earnings of drivers for each of the four strategies. We observe two main takeaways. First, we find that all strategies suffer a loss under small amounts of uncertainty, even at levels of α in the range of 0.02, so all strategies are tuned closely to the empirical data. However, all strategies then remain resilient to a wide range of additional uncertainty, and we find that the *relocation*, *flexible* and *flexible-relocation* strategies are most tolerant to uncertainty in the input transition matrices. Interestingly, even with 99% uncertainty, the *flexible-relocation* strategy significantly outperforms the *naive* strategy with no uncertainty. This observation further supports our claim that being strategic using historical data can significantly improve driver earnings in on-demand ride-hailing platforms.

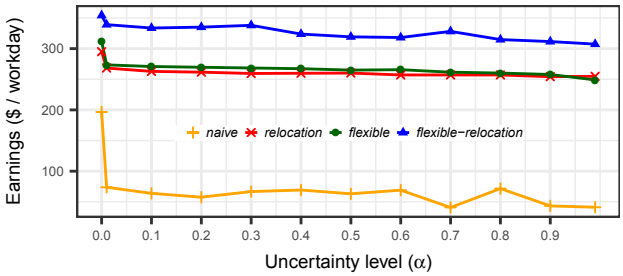

Figure 7: Sensitivity to uncertainty in parameters.

7 CONCLUSIONS

In this paper, we focused on the problem of maximizing a driver's individual earnings on ride-hailing platforms like Uber or Lyft. Our work confirms the power of strategic driving behavior using data-driven projections of ridership in the NYC area. Our first key takeaway is that a naive driver, armed with no data, and driving a 9-5 random walk schedule, is leaving roughly a 50% pay raise on the table by not driving more strategically. In contrast, a data-savvy driver armed with good historical data can build a forecast and optimal contingency driving plans with relatively little computational overhead using our dynamic programming algorithms, that have provable resilience to input uncertainty. Our experimental results yield insights into the structure of highly-optimized schedules, including relatively frequent relocation, working at specific peak periods, and taking advantage of surges when the time is ripe.

An obvious limitation of our work is that it is tailored to the setting when the methods are employed by self-interested individuals. If a significant percentage of the labor supply employs sophisticated optimization methods for driving, one would need to consider different strategies that achieve equilibria or other global objectives. Indeed, in the long run, as drivers for ride-hailing platforms like Uber and Lyft are put out of work by fleets of autonomous vehicles, the formulation and solution of new sets of optimization problems along those lines are likely to become relevant as well.

Acknowledgments: This work was supported by NSF awards: CAREER 1253393 and III 1421759. The authors also thank the anonymous reviewers for their helpful comments.

REFERENCES

[1] William A. Bailey Jr. and Thomas D. Clark Jr. 1987. A simulation analysis of demand and fleet size effects on taxicab service rates. In *Proceedings of the 19th Conference on Winter Simulation*. ACM, 838–844.

[2] Siddhartha Banerjee, Ramesh Johari, and Carlos Riquelme. 2015. Pricing in Ride-Sharing Platforms: A Queueing-Theoretic Approach. https://dl.acm.org/citation.cfm?id=2764527. Abstract appeared in ACM EC-2015.

[3] Colin Camerer, Linda Babcock, George Loewenstein, and Richard Thaler. 1997. Labor Supply of New York City Cabdrivers: One day at a Time. *The Quarterly Journal of Economics* 112, 2 (1997), 407–441.

[4] Juan Camilo Castillo, Dan Knoepfle, and Glen Weyl. 2017. Surge pricing solves the wild goose chase. In *Proceedings of the 2017 ACM Conference on Economics and Computation*. ACM, 241–242.

[5] Harshal A. Chaudhari, John W. Byers, and Evimaria Terzi. 2017. Project Webpage: Putting Data in the Driver's Seat. https://www.bu.edu/cs/groups/dblab/ride-hailing. (2017).

[6] Le Chen, Alan Mislove, and Christo Wilson. 2015. Peeking beneath the hood of Uber. In *Proceedings of the 2015 ACM Internet Measurement Conference*. ACM, 495–508.

[7] M Keith Chen and Michael Sheldon. 2016. Dynamic Pricing in a Labor Market: Surge Pricing and Flexible Work on the Uber Platform. http://www.anderson.ucla.edu/faculty_pages/keith.chen/papers/SurgeAndFlexibleWork_WorkingPaper.pdf. Working paper. Abstract appeared in ACM EC-2016.

[8] The Rideshare Guy. 2016. Advice For New Uber Drivers-Don't Chase The Surge! http://maximumridesharingprofits.com/advice-new-uber-drivers-dont-chase-surge/. (2016).

[9] Jonathan V. Hall and Alan B. Krueger. 2016. *An Analysis of the Labor Market for Uber's Driver-Partners in the United States*. Technical Report No. w22843. National Bureau of Economic Research.

[10] Waster Hudson. 2016. Chasing the Surge: 3 Tips for Maximizing Uber Earnings. https://pjmedia.com/lifestyle/2016/07/19/chasing-the-surge-3-tips-for-maximizing-uber-earnings/1/. (2016).

[11] Eric Jones, Travis Oliphant, Pearu Peterson, and others. 2001–2017. SciPy: Open source scientific tools for Python. (2001–2017). http://www.scipy.org/

[12] Jaeyoung Jung, R. Jayakrishnan, and Ji Young Park. 2013. Design and Modeling of Real-time Shared-taxi Dispatch Algorithms. In *Proc. Transportation Research Board 92nd Annual Meeting*.

[13] S. Kullback, M. Kupperman, and H. H. Ku. 1962. Tests for Contingency Tables and Markov Chains. *Technometrics* 4, 4 (1962), 573–608.

[14] Michal Maciejewski and Kai Nagel. 2013. Simulation and dynamic optimization of taxi services in MATSim. *VSP Working Paper 13-0. TU Berlin, Transport Systems Planning and Transport Telematics, 2013* (2013).

[15] Arnab Nilim and Laurent El Ghaoui. 2004. Robustness in Markov decision problems with uncertain transition matrices. In *Advances in Neural Information Processing Systems*. 839–846.

[16] Jorge Nunes, Luís Matos, and António Trigo. 2011. Taxi Pick-Ups Route Optimization Using Genetic Algorithms. *Adaptive and Natural Computing Algorithms* (2011), 410–419.

[17] Erhun Ozkan and Amy R. Ward. 2016. Dynamic Matching for Real-time Ridesharing. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2844451, (2016). Working paper.

[18] Ying Shi and Zhaotong Lian. 2016. Optimization and strategic behavior in a passenger–taxi service system. *European Journal of Operational Research* 249, 3 (2016), 1024–1032.

[19] The New York Times. 2015. An App That Helps Drivers Earn the Most From Their Trips. https://www.nytimes.com/2015/05/10/technology/a-dashboard-management-consultant.html. (2015).

[20] The New York Times. 2017. How Uber Uses Psychological Tricks to Push Its Drivers' Buttons. https://www.nytimes.com/interactive/2017/04/02/technology/uber-drivers-psychological-tricks.html. (2017).

[21] Wikipedia. 2017. Skellam distribution — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Skellam_distribution. (2017).

[22] Hai Yang, Sze Chun Wong, and Ki Wong. 2002. Demand–supply equilibrium of taxi services in a network under competition and regulation. *Transportation Research Part B: Methodological* 36, 9 (2002), 799–819.