Allocation Problems in Ride-Sharing Platforms: Online Matching with Offline Reusable Resources

John P. Dickerson
University of Maryland, College Park, USA
john@cs.umd.edu

Karthik A. Sankararaman*
University of Maryland, College Park, USA
kabinav@cs.umd.edu

Aravind Srinivasan†
University of Maryland, College Park, USA
srin@cs.umd.edu

Pan Xu‡
University of Maryland, College Park, USA
panxu@cs.umd.edu

Abstract

Bipartite matching markets pair agents on one side of a market with agents, items, or contracts on the opposing side. Prior work addresses online bipartite matching markets, where agents arrive over time and are dynamically matched to a known set of disposable resources. In this paper, we propose a new model, *Online Matching with (offline) Reusable Resources under Known Adversarial Distributions (*OM-RR-KAD*)*, in which resources on the offline side are *reusable* instead of disposable; that is, once matched, resources become available again at some point in the future. We show that our model is tractable by presenting an LP-based adaptive algorithm that achieves an online competitive ratio of $\frac{1}{2} - \epsilon$ for any given $\epsilon > 0$. We also show that no non-adaptive algorithm can achieve a ratio of $\frac{1}{2} + o(1)$ based on the same benchmark LP. Through a data-driven analysis on a massive openly-available dataset, we show our model is robust enough to capture the application of taxi dispatching services and ride-sharing systems. We also present heuristics that perform well in practice.

1 Introduction

In bipartite matching problems, agents on one side of a market are paired with agents, contracts, or transactions on the other. Classical matching problems—assigning students to schools, papers to reviewers, or medical residents to hospitals—take place in a static setting, where all agents exist at the time of matching, are simultaneously matched, and then the market concludes. In contrast, many matching problems are dynamic, where one side of the market arrives in an *online* fashion and is matched sequentially to the other side.

Online bipartite matching problems are primarily motivated by Internet advertising. In the basic version of the problem, we are given a bipartite graph $G = (U, V, E)$ where U represents the offline vertices (advertisers) and V represents the online vertices (keywords or impressions). There is an edge $e = (u, v)$ if advertiser u bids for a keyword v. When a keyword v arrives, a central clearinghouse must make an instant and irrevocable decision to either reject v or assign v to one of its "neighbors" (*i.e.,* set of incident edges) u and obtain a profit w_e for the match $e = (u, v)$. When an advertiser u is matched, it is no longer available for matches with other keywords (in the most basic case) or its budget is reduced. The goal is to design an efficient online algorithm such that the expected total weight (profit) of the matching obtained is maximized. Following the seminal work of Karp, Vazirani, and Vazirani (1990), there has been a large body of research on related variants (overviewed by Mehta (2012)). One particular flavor of problems is online matching with known identical independent distributions (OM-KIID) (Feldman et al. 2009; Haeupler, Mirrokni, and Zadimoghaddam 2011; Manshadi, Gharan, and Saberi 2012; Jaillet and Lu 2013; Brubach et al. 2016). In this flavor, agents arrive over T rounds, and their arrival distributions are assumed to be *identical and independent* over all T rounds; additionally, this distribution is known to the algorithm beforehand.

Apart from the Internet advertising application, online bipartite matching models have been used to capture a wide range of online resource allocation and scheduling problems. Typically we have an offline and an online party representing, respectively, the service providers (SP) and online users; once an online user arrives, we need to match it to an offline SP immediately. In many cases, the service is *reusable* in the sense that once an SP is matched to a user, it will be gone for some time, but will then rejoin the system afterwards. Besides that, in many real settings the arrival distributions of online users do change from time to time (*i.e.,* they are not i.i.d.). Consider the following motivational examples.

Taxi Dispatching Services and Ride-Sharing Systems. Traditional taxi services and rideshare systems like Uber and Didi Chuxing match drivers to would-be riders (Tong et al. 2016a; Lowalekar, Varakantham, and Jaillet 2016; Lee et al. 2004; Seow, Dang, and Lee 2010). Here, the offline SPs are different vehicle drivers. Once an online request (potential rider) arrives, the system matches it to a nearby driver instantly such that the rider's waiting time is minimized. In most cases, the driver will rejoin the system and can be matched again once she finishes the service. Additionally, the arrival rates of requests changes dramatically across

*Supported in part by NSF Awards CNS 1010789 and CCF 1422569

†Supported in part by NSF Awards CNS-1010789, CCF-1422569 and CCF-1749864, and by research awards from Adobe, Inc

‡Supported in part by NSF Awards CNS 1010789 and CCF 1422569

the day. Consider the online arrivals during the peak hours and off-peak hours for example: the arrival rates in the former case can be much larger than the latter.

Organ Allocation. Chronic kidney disease affects tens of millions of people worldwide at great societal and monetary cost (Neuen et al. 2013; Saran et al. 2015). Organ donation—either via a deceased or living donor—is a lifesaving alternative to organ failure. In the case of kidneys, a donor organ can last up to 15 years in a patient before failing again. Various nationwide organ donation systems exist and operate under different ethical and logistical constraints (Bertsimas, Farias, and Trichakis 2013; Dickerson and Sandholm 2015; Mattei, Saffidine, and Walsh 2017), but all share a common online structure: the offline party is the set of patients (who reappear every 5 to 15 years based on donor organ longevity), and the online party is the set of donors or donor organs, who arrive over time.

Similar scenarios can be seen in other areas such as wireless network connection management (SPs are different wireless access points) (Yiu et al. 2008) and online cloud computing service scheduling (Miller 2008; Younge et al. 2010). Inspired by the above applications, we generalize the model of OM-KIID in the following two ways.

Reusable Resources. Once we assign v to u, u will rejoin the system after C_e rounds with $e = (u, v)$, where $C_e \in \{0, 1, \ldots, T\}$ is an integral random variable with known distribution. In this paper, we call C_e the *occupation time* of u w.r.t. e. In fact, we show that our setting can directly be extended to the case when C_e is time sensitive: when matching v to u at time t, u will rejoin the system after $C_{e,t}$ rounds. This extension makes our model adaptive to nuances in real-world settings. For example, consider the taxi dispatching or ride-sharing service: the occupation time of a driver u from a matching with an online user v does depend on both the user type of v (such as destination) and the time when the matching occurs (peak hours can differ significantly from off-peak hours).

Known Adversarial Distributions (KAD**).** Suppose we have T rounds and that for each round $t \in [T]$ [1], a vertex v is sampled from V according to an arbitrary known distribution $\mathcal{D}$ where the marginal for v is $\{p_{v,t}\}$ such that $\sum_{v \in V} p_{v,t} \leq 1$ for all t. Also, the arrivals at different times are independent (and according to these given distributions). The setting of KAD was introduced by (Alaei, Hajiaghayi, and Liaghat 2012; 2013) and is known as Prophet Inequality matching.

We call our new model Online Matching with (offline) Reusable Resources under Known Adversarial Distributions (OM-RR-KAD, henceforth). Note that the OM-KIID model can be viewed as a special case when C_e is a constant (with respect to T) and $\{p_{v,t} | v \in V\}$ are the same for all $t \in [T]$.

Competitive Ratio. Let $\mathbb{E}[\mathrm{ALG}(\mathcal{I}, \mathcal{D})]$ denote the expected value obtained by an algorithm ALG on an input $\mathcal{I}$ and arrival distribution $\mathcal{D}$. Let $\mathbb{E}[\mathrm{OPT}(\mathcal{I})]$ denote the expected *offline optimal*, which refers to the optimal solution when we are allowed to make choices after observing the entire sequence of online arrival vertices. Then, competitive ratio is defined as $\min_{\mathcal{I},\mathcal{D}} \frac{\mathbb{E}[\mathrm{ALG}(\mathcal{I},\mathcal{D})]}{\mathbb{E}[\mathrm{OPT}(\mathcal{I})]}$. It is a common technique to use an LP optimal value to upper bound the $\mathbb{E}[\mathrm{OPT}(\mathcal{I})]$ (called the benchmark LP) and hence get a valid lower bound on the resulting competitive ratio.

[1] Throughout this paper, we use $[N]$ to denote the set $\{1, 2, \ldots, N\}$, for any positive integer N.

1.1 Our Contributions

First, we propose a new model of OM-RR-KAD to capture a wide range of real-world applications related to online scheduling, organ allocation, rideshare dispatch, among others. We claim that this model is tractable enough to obtain good algorithms with theoretically provable guarantees and general enough to capture many real-life instances. Our model assumptions take a significant step forward from the usual assumptions in the online matching literature where the offline side is assumed to be *single-use* or *disposable*. This leads to a larger range of potential applications which can be modeled by online matching.

Second, we show how this model can be *cleanly* analyzed under a theoretical framework. We first construct a linear program (LP henceforth) LP (1) which we show is a valid upper-bound on the expected offline optimal (note the latter is hard to characterize). Next, we propose an efficient algorithm that achieves a competitive ratio of $\frac{1}{2} - \epsilon$ for any given $\epsilon > 0$. This algorithm solves the LP and obtains an optimal fractional solution. It uses this optimal solution as a guide in the online phase. Using Monte-Carlo simulations (called simulations henceforth), and combining with this optimal solution, our algorithm makes the online decisions. In particular, Theorem 1 describes our first theoretical results formally.

Theorem 1. LP (1) *is a valid benchmark for* OM-RR-KAD. *There exists an online algorithm, based on* LP (1)*, achieving an online competitive ratio of* $\frac{1}{2} - \epsilon$ *for any given* $\epsilon > 0$.

Third, we show that our simple algorithm is nearly optimal among all non-adaptive algorithms. We show that no non-adaptive algorithm can achieve a competitive ratio better than $\frac{1}{2}$ if using LP (1) as the benchmark. Formally, Theorem 2 states this result.

Theorem 2. *No non-adaptive algorithm, based on benchmark* LP (1)*, can achieve a competitive ratio better than* $\frac{1}{2} + o(1)$ [2] *even when all* C_e *are constants.*

Finally, through a data-driven analysis on a massive openly available dataset we show that our model is robust enough to capture the setting of taxi hailing/sharing at least. Additionally, we provide certain *simpler* heuristics which also give good performance. Hence, we can *combine* these theoretically grounded algorithms with such heuristics to obtain further improved ratios in practice. Section 5 provides a detailed qualitative and quantitative discussion.

1.2 Other Related Work

In addition to the arrival assumptions of KIID and KAD, there are several other important, well-studied variants of

[2] $o(1)$ is a vanishing term when both of C_e and T/C_e are sufficiently large.

online matching problems. Under *adversarial ordering*, an adversary can arrange the arrival order of all items in an arbitrary way (e.g., online matching (Karp, Vazirani, and Vazirani 1990; Sun, Zhang, and Zhang 2016) and AdWords (Buchbinder, Jain, and Naor 2007; Mehta et al. 2007)). Under a *random arrival order*, all items arrive in a random permutation order (e.g., online matching (Mahdian and Yan 2011) and AdWords (Goel and Mehta 2008)). Finally, under *unknown distributions*, in each round, an item is sampled from a fixed but unknown distribution. (e.g., (Devanur et al. 2011)). For each of the above categories, we list only a few examples considered under that setting. For a more complete list, please refer to the book by Mehta (2012).

Despite the fact that our model is inspired by online bipartite matching, it also overlaps with stochastic online scheduling problems (SOS) (Megow, Uetz, and Vredeveld 2004; 2006; Skutella, Sviridenko, and Uetz 2016). We first restate our model in the language of SOS: we have $|U|$ *nonidentical* parallel machines and $|V|$ jobs; at every time-step a single job v is sampled from V with probability $p_{v,t}$; the jobs have to be assigned immediately after its arrival; additionally each job v can be processed *non-preemptively* on a specific subset of machines; once we assign v to u, we get a profit of w_e and u will be occupied for C_e rounds with $e = (u, v)$, where C_e is a random variable with known distribution. Observe that the key difference between our model and SOS is in the objective: the former is to maximize the expected profit from the completed jobs, while the latter is to minimize the total or the maximum completion time of all jobs.

2 Main Model

In this section, we present a formal statement of our main model. Suppose we have a bipartite graph $G = (U, V, E)$ where U and V represent offline and online parties respectively. We have a finite time horizon T (known beforehand) and for each time $t \in [T]$, a vertex v will be sampled (we use the term v *arrives*) from a known probability distribution $\{p_{v,t}\}$ such that $\sum_{v \in V} p_{v,t} \leq 1$[3] (noting that such a choice is made independently for each round t). The expected number of times v arrives across the T rounds, $\sum_{t \in [T]} p_{v,t}$ is called the *arrival rate* for vertex v. Once a vertex v arrives, we need to make an *irrevocable decision immediately*: either to reject v or assign v to one of its neighbors in U. For each u, once it is assigned to some v, it becomes unavailable for C_e rounds with $e = (u, v)$, and subsequently rejoins the system. Here C_e is an integral random variable taking values from $\{0, 1, \ldots, T\}$ and the distribution is known in advance. Each assignment e is associated with a weight w_e and our goal is to design an online assignment policy such that the total expected weights of all assignments made is maximized. Following prior work, we assume $|V| \gg |U|$ and $T \gg 1$. Throughout this paper, we use edge $e = (u, v)$ and assignment of v to u interchangeably.

For an assignment e, let $x_{e,t}$ be the probability that e is chosen at t in any offline optimal algorithm. For each u (likewise for v), let E_u (E_v) be the set of neighboring edges incident to u (v). We use the LP (1) as a benchmark to upper bound the offline optimal. We now interpret the constraints. For each round t, once an online vertex v arrives, we can assign it to at most one of its neighbors. Thus, we have: if v arrives at t, the total number of assignments for v at t is at most 1; if v does not arrive, the total is 0. The LHS of (2) is exactly the expected number of assignments made at t for v. It should be no more than the prob. that v arrives at t, which is the RHS of (2). Constraint (3) is the *most* novel part of our problem formulation. Consider a given u and t. In the LHS, the first term (summation over $t' < t$ and $e \in E_u$) refers to the prob. that u is not available at t while the second term (summation over $e \in E_u$) is the prob. that u is assigned to some worker at t, which is no larger than prob. u is available at t. Thus, the sum of the first term and second term on LHS is no larger than 1.[4] This argument implies that the LP forms a valid upper-bound on the offline optimal solution and hence we have Lemma 1.

Lemma 1. *The optimal value to LP* (1) *is a valid upper bound for the offline optimal.*

3 Simulation-based Algorithm

In this section, we present a simulation-based algorithm. Proofs for Lemma 2, 3 and 4 can be found in the supplementary material. The main idea is as follows. Let $\mathbf{x}^*$ denote an optimal solution to LP (1). Suppose we aim to develop an online algorithm achieving a ratio of $\gamma \in [0, 1]$. Consider an assignment $e = (u, v)$ when some v arrived at time t. Let $\mathrm{SF}_{e,t}$ be the event that e is safe at t, *i.e.,* u is available at t. By simulating the current strategy up to t, we can get an estimation of $\Pr[\mathrm{SF}_{e,t}]$, say $\beta_{e,t}$, within an arbitrary small error. Therefore in the case e is safe at t, we can sample it with probability $\frac{x^*_{e,t}}{p_{v,t}} \frac{\gamma}{\beta_{e,t}}$, which leads to the fact that e is sampled with probability $\gamma x^*_{e,t}$ unconditionally. Hence, we call any algorithm that satisfies $\gamma \leq \beta_{e,t}$ as *valid*.

The simulation-based attenuation technique has been used previously for other problems, such as stochastic knapsack (Ma 2014) and stochastic matching (Adamczyk, Grandoni, and Mukherjee 2015). Throughout the analysis, we assume that we know the exact value of $\beta_{e,t} := \Pr[\mathrm{SF}_{e,t}]$ for all t and e. (It is easy to see that the sampling error can be folded into a multiplicative factor of $(1 - \epsilon)$ in the competitive ratio by standard Chernoff bounds and hence, ignoring it leads to a cleaner presentation). The formal statement of our algorithm, denoted by $\mathrm{ADAP}(\gamma)$, is as follows. For each v and t, let $E_{v,t}$ be the set of *safe* assignments for v at t.

Lemma 2. $\mathrm{ADAP}(\gamma)$ *is valid with* $\gamma = \frac{1}{2}$.

The main Theorem 1 follows directly from Lemmas 1 and 2.

Extension from C_e to $C_{e,t}$. Consider the case when the occupation time of u from e is sensitive to t. In other words,

[3] Thus, with probability $1 - \sum_{v \in V} p_{v,t}$, none of the vertices from V will arrive at t.

[4] We would like to point out that our LP constraint (3) on u is inspired by Ma (2014). The proof is similar to that by Alaei, Hajiaghayi, and Liaghat (2012) and Alaei, Hajiaghayi, and Liaghat (2013).

!h

$$\text{maximize} \sum_{t\in[T]}\sum_{e\in E} w_e x_{e,t} \tag{1}$$

$$\text{subject to} \sum_{e\in E_v} x_{e,t} \le p_{v,t} \qquad \forall v \in V, t \in [T] \tag{2}$$

$$\sum_{t'<t}\sum_{e\in E_u} x_{e,t'} \Pr[C_e > t - t'] + \sum_{e\in E_u} x_{e,t} \le 1 \qquad \forall u \in U, t \in [T] \tag{3}$$

$$0 \le x_{e,t} \le 1 \qquad \forall e \in E, t \in [T] \tag{4}$$

Algorithm 1: Simulation-based adaptive algorithm (ADAP(γ))

1 For each time t, let v denote the request arriving at time t.

2 If $E_{v,t} = \emptyset$, then reject v; otherwise choose $e \in E_{v,t}$ with prob. $\frac{x^*_{e,t}}{p_{v,t}}\frac{\gamma}{\beta_{e,t}}$ where $e = (u, v)$.

each u will be unavailable for $C_{e,t}$ rounds from the assignment $e = (u, v)$ at t. We can accommodate the extension by simply updating the constraints (3) on u in the benchmark LP (1) to the following. We have that $\forall u \in U, t \in [T]$,

$$\sum_{t'<t}\sum_{e\in E_u} x_{e,t'} \Pr[C_{e,t'} > t - t'] + \sum_{e\in E_u} x_{e,t} \le 1 \tag{5}$$

The rest of our algorithm remains the same as before. We can verify that (1) LP (1) with constraints (3) replaced by (5) is a valid benchmark; (2) ADAP achieves a competitive ratio of $\frac{1}{2} - \epsilon$ for any given $\epsilon > 0$ for the new model based on the new valid benchmark LP. The modifications to the analysis transfer through in a straightforward way and for brevity we omit the details here.

4 Hardness Result

Consider a complete bipartite graph of $G = (U, V, E)$ where $|U| = K$, $|V| = n^2$. Suppose we have $T = n$ rounds and $p_{v,t} = \frac{1}{n^2}$ for each v and t. In other words, in each round t, each v is sampled uniformly from V. For each e, let C_e be a constant of K, which implies that each u will be unavailable for a constant K rounds after each assignment. Assume all assignments have a uniform weight (*i.e.,* $w_e = 1$ for all e). Split the whole online process of n rounds into $n - K + 1$ consecutive windows $\mathcal{W} = \{W_\ell\}$ such that $W_\ell = \{\ell, \ell + 1, \ldots, \ell+K-1\}$ for each $1 \le \ell \le n-K+1$. The benchmark LP (1) then reduces to the following.

$$\max \sum_{t\in[T]}\sum_{e\in E} x_{e,t} \tag{6}$$

$$\text{s.t.} \sum_{e\in E_v} x_{e,t} \le \frac{1}{n^2} \qquad \forall v \in V, t \in [T] \tag{7}$$

$$\sum_{t\in W_\ell}\sum_{e\in E_u} x_{e,t} \le 1 \quad \forall u \in U, 1 \le \ell \le n - K + 1 \tag{8}$$

$$0 \le x_{e,t} \le 1 \qquad \forall e \in E, t \in [T] \tag{9}$$

We can verify that an optimal solution to the above LP is as follows: $x^*_{e,t} = 1/(n^2K)$ for all e and t with the optimal objective value of n. We investigate the performance of any optimal non-adaptive algorithm. Notice that the expected arrivals of any v in the full sequence of online arrivals is $1/n$. Thus for any non-adaptive algorithm NADAP, it needs to specify the allocation distribution $\mathcal{D}_v$ for each v during the first arrival. Consider a given NADAP parameterized by $\{\alpha_{u,v} \in [0, 1]\}$ for each v and $u \in E_v$ such that $\sum_{u\in E_v} \alpha_{u,v} \le 1$ for each v. In other words, NADAP will assign v to u with probability $\alpha_{u,v}$ when v comes for the first time and u is available.

Let $\beta_u = \sum_{v\in E_u} \alpha_{u,v} * \frac{1}{n^2}$, which is the probability that u is matched in each round if it is safe at the beginning of that round, when running NADAP. Hence,

$$\sum_{u\in U} \beta_u = \sum_{u\in U}\sum_{v\in E_u} \alpha_{u,v} * \frac{1}{n^2} = \sum_{v\in V}\sum_{u\in E_v} \alpha_{u,v} * \frac{1}{n^2} \le 1$$

Consider a given u with β_u and let $\gamma_{u,t}$ be the probability that u is available at t. Then the expected number of matches of u after the n rounds is $\sum_t \beta_u \gamma_{u,t}$. We have the recursive inequalities on $\gamma_{u,t}$ as in Lemma 3, with $\gamma_{u,t} = 1, t = 1$.

Lemma 3. *$\forall 1 < t \le n$, we have*

$$\gamma_{u,t} + \beta_u \sum_{t-K+1\le t'<t} \gamma_{u,t'} = 1$$

Note that the OPT of our benchmark LP is n while the performance of NADAP is $\sum_u \sum_t \beta_u \gamma_{u,t}$. The resulting competitive ratio achieved by an optimal NADAP is captured by the following maximization program.

$$\begin{aligned}
\max \quad & \frac{\sum_u \sum_t \beta_u \gamma_{u,t}}{n} \\
\text{s.t.} \quad & \sum_{u \in U} \beta_u \leq 1 \\
& \gamma_{u,t} + \beta_u \sum_{t-K+1 \leq t' < t} \gamma_{u,t'} = 1 \quad \forall 1 < t \leq n, u \in U \\
& \beta_u \geq 0, \gamma_{u,1} = 1 \qquad \forall u \in U
\end{aligned} \tag{10}$$

We prove the following Lemma which implies Theorem 2.

Lemma 4. *The optimal value to the program* (10) *is at most* $\frac{1}{2-1/K} + o(1)$ *when* $K = o(n)$.

Unconditional Hardness. Manshadi, Gharan, and Saberi (2012) prove that for the online matching problem under known distribution (but disposable offline vertices), no algorithm can achieve a ratio better than 0.823. Since our setting generalizes this, the hardness results directly apply to our problem as well.

5 Experiments

To validate the approaches presented in this paper, we use the New York City yellow cabs dataset,[5] which contains the trip records for trips in Manhattan, Brooklyn, and Queens for the year 2013. The dataset is split into 12 months. For each month we have numerous records each corresponding to a single trip. Each record has the following structure. We have an anonymized license number which is the primary key corresponding to a car. For privacy purposes a long string is used as opposed to the actual license number. We then have the time at which the trip was initiated, the time at which the trip ended, and the total time of the trip in seconds. This is followed by the starting coordinates (*i.e.,* latitude and longitude) of the trip and the destination coordinates of the trip.

Assumptions. We make two assumptions specific to our experimental setup. Firstly, we assume that every *car* starts and ends at the same location, for *all* trips that it makes. Initially, we assign every car a location (potentially the same) which corresponds to its *docking* position. On receiving a request, the car leaves from this docking position to the point of pick-up, executes the trip and returns to this docking position. Secondly, we assume that occupation time distributions (OTD) associated with all matches are identically (and independently) distributed, *i.e.,* $\{C_e\}$ follow the same distribution. Note that this is a much stronger assumption than what we made in the model, and is completely inspired by the dataset (see Section 5.2). We test our model on two specific distributions, namely a *normal* distribution and the *power-law* distribution (see Figure 5). The docking position of each car and parameters associated with each distribution are all learned from the training dataset (described below in the **Training** discussion).

[5] http://www.andresmh.com/nyctaxitrips/

5.1 Experimental Setup

For our experimental setup, we randomly select 30 cabs (each cab is denoted by u). We discretize the Manhattan map into cells such that each cell is approximately 4 miles (increments of 0.15 degrees in latitude and longitude). For each pair of locations, say (a, b), we create a request *type* v, which represents all trips with starting and ending locations falling into a and b respectively. In our model, we have $|U| = 30$ and $|V| \approx 550$ (variations depending on day to day requests with low variance). We focus on the month of January 2013. We split the records into 31 parts, each corresponding to a day of January. We choose a random set of 12 parts for *training* purposes and use the remaining for *testing* purposes.

The edge weight w_e on $e = (u, v)$ (*i.e.,* edge from a car u to type v) is set as a function of two distances in our setup. The first is the trip distance (*i.e.,* the distance from the starting location to the ending location of v, denoted L_1) while the second is the docking distance (*i.e.,* the distance from the docking position of u to the starting/ending location of v, denoted L_2). We set $w_e = \max(L_1 - \alpha L_2, 0)$, where α is a parameter capturing the subtle balance between the positive contribution from the trip distance and negative contribution from the docking distance to the final profit. We set $\alpha = 0.5$ for the experiments. We consider each single day as the time horizon and set the total number of rounds $T = \frac{24*60}{5} = 288$ by discretizing the 24-hour period into a time-step of 5 minutes. Throughout this section, we use time-step and round interchangeably.

Training. We use the training dataset of 12 days to learn various parameters. As for the arrival rates $\{p_{v,t}\}$, we count the total number of appearances of each request type v at time-step t in the 12 parts (denote it by $c_{v,t}$) and set $p_{v,t} = c_{v,t}/12$ under KAD. When assuming KIID, we set $p_v = p_{v,t} = (c/12)/T$ (*i.e.,* the arrival distributions are assumed the same across all the time-steps for each v). The estimation of parameters for the two different occupation time distributions are processed as follows. We first compute the average number of seconds between two *requests* in the dataset (note this was 5 minutes in the experimental setup). We then assume that each *time-step* of our online process corresponds to a time-difference of this average in seconds. We then compute the sample mean and sample variance of the trip lengths (as number of seconds taken by the trip divided by five minutes) in the 12 parts. Hence we use the normal distribution obtained by this sample mean and standard deviation as the distribution with which a car is unavailable. We assign the docking position of each car to the location (in the discretized space) in which the majority of the requests were initiated (*i.e.,* starting location of a request) and matched to this car.

5.2 Justifying The Two Important Model Assumptions

Known Adversarial Distributions. Figure 4 plots the number of arrivals of a particular type at various times during the day. Notice the significant increase in the number of requests in the middle of the day as opposed to the mornings and

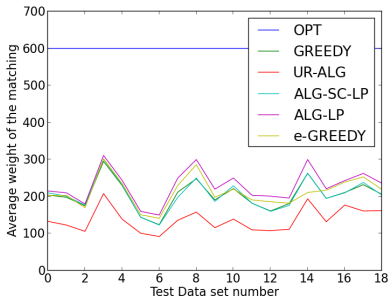

Figure 1: OTD is normal distribution under KIID

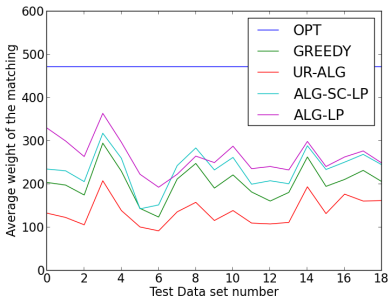

Figure 2: OTD is normal distribution under KAD

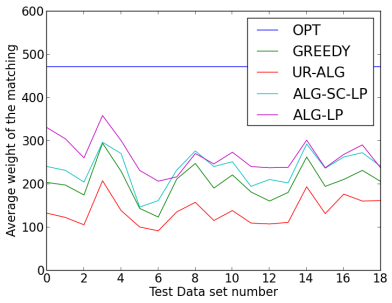

Figure 3: OTD is power law distribution under KAD

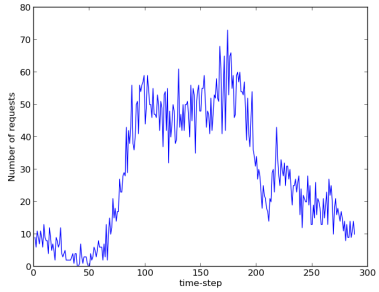

Figure 4: The number of requests of a given type at various time-steps. x-axis: time-step, y-aixs: number of requests

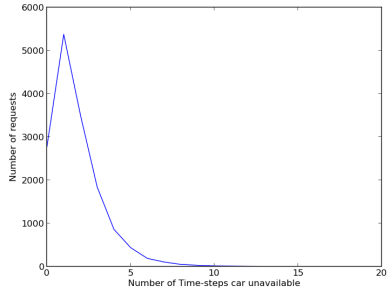

Figure 5: Occupation time distribution of all cars. x-axis: number of time-steps, y-axis: number of requests

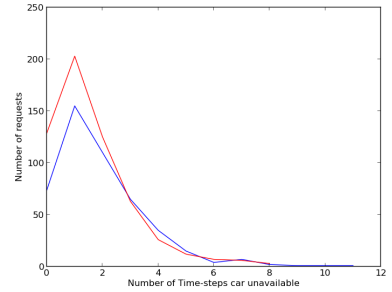

Figure 6: Occupation time distribution of two different cars. x-axis: number of time-steps, y-axis: number of requests

nights. This justified our arrival assumption of KAD which assumes different arrival distributions at different time-steps. Hence the LP (and the corresponding algorithm) can exploit this vast difference in the arrival rates and potentially obtain improved results compared to the assumption of Known Identical Independent Distributions (KIID). This is confirmed by our experimental results shown in Figures 1 and 2.

Identical Occupation Time Distribution. We assume each car will be available again via an independent and identical random process regardless of the matches it received. The validity of our assumptions can be seen in Figures 5 and 6, where the x-axis represents the different occupation time and the y-axis represents the corresponding number of requests in the dataset responsible for each occupation time. It is clear that for most requests the occupation time is around 2-3 time-steps and dropping drastically beyond that with a long tail. Figure 6 displays occupation times for two representative (we chose two out of the many cars we plotted, at random) cars in the dataset; we see that the distributions roughly coincide with each other, suggesting that such distributions can be learned from historical data and used as a guide for future matches.

5.3 Results

Inspired by the experimental setup by (Tong et al. 2016a; 2016b), we run five different algorithms on our dataset. The first algorithm is the ALG-LP. In this algorithm, when a request v arrives, we choose a neighbor u with probability $x^*_{e,t}/p_{v,t}$ with $e = (u, v)$ if u is available. Here $x^*_{e,t}$ is an optimal solution to our benchmark LP and $p_{v,t}$ is the arrival rate of type v at time-step t. The second algorithm is called ALG-SC-LP. Recall that $E_{v,t}$ is the set of "safe" or available assignments with respect to v when the type v arrives at t. Let $x_{v,t} = \sum_{e \in E_{v,t}} x^*_{e,t}$. In ALG-SC-LP, we sample a safe assignment for v with probability $x^*_{e,t}/x_{v,t}$. The next two algorithms are heuristics oblivious to the underlying LP. Our third algorithm is called GREEDY which is as follows. When a request v comes, match it to the safe neighbor u with the highest edge weight. Our fourth algorithm is called UR-ALG which chooses one of the safe neighbors uniformly at random. Finally, we use a combination of LP-oblivious algorithm and LP-based algorithm called ϵ-GREEDY. In this algorithm when a type v comes, with probability ϵ we use the greedy choice and with probability $1 - \epsilon$ we use the optimal LP choice. In our algorithm, we optimized the value of ϵ and set it to $\epsilon = 0.1$. We summarize our results in the following plots. Figures 1, 2, and 3 show the performance of the five algorithms and OPT (optimal value of the benchmark LP) under the different assumptions of the OTD (normal or power law) and online arrives (KIID or KAD). In all three figures the x-axis represents test data-set number and the y-axis represents average weight of matching.

Discussion. From the figures, it is clear that both the LP-based solutions, namely ALG-LP and ALG-SC-LP, do better than choosing a free neighbor uniformly at random. Additionally, with distributional assumptions the LP-based solutions outperform greedy algorithm as well. We would like to draw attention to a few interesting details in these results. Firstly, compared to the LP optimal solution, our LP-based algorithms have a competitive ratio in the range of 0.5 to 0.7. We believe this is because of our experimental setup. In particular, we have that the rates are high (> 0.1) only in a few time-steps while in all other time-steps the rates are very close to 0. This means that it resembles the structure of the *theoretical* worst case example we showed in Section 4. In future experiments, running our algorithms during *peak* periods (where the request rates are significantly larger than 0) may show that competitive ratios in those cases approach 1. Secondly, it is surprising that our algorithm is fairly robust to the *actual* distributional assumption we made. In particular, from Figures 2 and 3 it is clear that the difference between the assumption of normal distribution versus power-law distribution for the unavailability of cars is *negligible*. This is important since it might not be easy to learn the *exact* distribution in many cases (*e.g.,* cases where the sample complexity is high) and this shows that a close approximation will still be as good.

6 Conclusion and Future Directions

In this work, we provide a model that captures the application of assignment in ride-sharing platforms. One key aspect in our model is to consider the *reusable* aspect of the offline resources. This helps in modeling many other important applications where agents enter and leave the system *multiple* times (*e.g.,* organ allocation, crowdsourcing markets (Ho and Vaughan 2012), and so on). Our work opens several important research directions. The first direction is to generalize the online model to the *batch* setting. In other words, in each round we assume multiple arrivals from V. This assumption is useful in crowdsourcing markets (for example) where multiple tasks—but not all—become available at some time. The second direction is to consider a Markov model on the driver starting position. In this work, we assumed that each driver returns to her docking position. However, in many ride-sharing systems, drivers start a new trip from the position of the last-drop off. This leads to a Markovian system on the offline types, as opposed to the assumed static types in the present work. Finally, pairing our current work with more applied stochastic optimization and reinforcement learning approaches would be of practical interest to policymakers running taxi and bikeshare services (Singhvi et al. 2015; O'Mahony and Shmoys 2015; Lowalekar, Varakantham, and Jaillet 2016; Verma et al. 2017; Ghosh et al. 2017).

References

Adamczyk, M.; Grandoni, F.; and Mukherjee, J. 2015. Improved approximation algorithms for stochastic matching. In *ESA-15*.

Alaei, S.; Hajiaghayi, M.; and Liaghat, V. 2012. Online prophet-inequality matching with applications to ad allocation. In *EC-12*.

Alaei, S.; Hajiaghayi, M.; and Liaghat, V. 2013. The online stochastic generalized assignment problem. In *APPROX-RANDOM-13*.

Bertsimas, D.; Farias, V. F.; and Trichakis, N. 2013. Fairness, efficiency, and flexibility in organ allocation for kidney transplantation. *Operations Research* 61(1).

Brubach, B.; Sankararaman, K. A.; Srinivasan, A.; and Xu, P. 2016. New algorithms, better bounds, and a novel model for online stochastic matching. In *ESA-16*.

Buchbinder, N.; Jain, K.; and Naor, J. S. 2007. Online primal-dual algorithms for maximizing ad-auctions revenue. In *ESA-07*.

Devanur, N. R.; Jain, K.; Sivan, B.; and Wilkens, C. A. 2011. Near optimal online algorithms and fast approximation algorithms for resource allocation problems. In *EC-11*.

Dickerson, J. P., and Sandholm, T. 2015. FutureMatch: Combining human value judgments and machine learning to match in dynamic environments. In *AAAI-15*.

Feldman, J.; Mehta, A.; Mirrokni, V.; and Muthukrishnan, S. 2009. Online stochastic matching: Beating 1-1/e. In *FOCS-09*.

Ghosh, S.; Varakantham, P.; Adulyasak, Y.; and Jaillet, P. 2017. Dynamic repositioning to reduce lost demand in bike sharing systems. *Journal of Artificial Intelligence Research (JAIR)* 58:387–430.

Goel, G., and Mehta, A. 2008. Online budgeted matching in random input models with applications to adwords. In *SODA-08*.

Haeupler, B.; Mirrokni, V. S.; and Zadimoghaddam, M. 2011. Online stochastic weighted matching: Improved approximation algorithms. In *WINE-11*.

Ho, C.-J., and Vaughan, J. W. 2012. Online task assignment in crowdsourcing markets. In *AAAI-12*.

Jaillet, P., and Lu, X. 2013. Online stochastic matching: New algorithms with better bounds. *Mathematics of Operations Research* 39(3).

Karp, R. M.; Vazirani, U. V.; and Vazirani, V. V. 1990. An optimal algorithm for on-line bipartite matching. In *STOC-90*.

Lee, D.-H.; Wang, H.; Cheu, R.; and Teo, S. 2004. Taxi dispatch system based on current demands and real-time traffic conditions. *Transportation Research Record: Journal of the Transportation Research Board* (1882).

Lowalekar, M.; Varakantham, P.; and Jaillet, P. 2016. Online spatio-temporal matching in stochastic and dynamic domains. In *AAAI-16*.

Ma, W. 2014. Improvements and generalizations of stochastic knapsack and multi-armed bandit approximation algorithms. In *SODA-14*.

Mahdian, M., and Yan, Q. 2011. Online bipartite matching with random arrivals: an approach based on strongly factor-revealing lps. In *STOC-11*.

Manshadi, V. H.; Gharan, S. O.; and Saberi, A. 2012. Online stochastic matching: Online actions based on offline statistics. *Mathematics of Operations Research* 37(4).

Mattei, N.; Saffidine, A.; and Walsh, T. 2017. Mechanisms for online organ matching. In *IJCAI-17*.

Megow, N.; Uetz, M.; and Vredeveld, T. 2004. Stochastic online scheduling on parallel machines. In *WAOA-04*.

Megow, N.; Uetz, M.; and Vredeveld, T. 2006. Models and algorithms for stochastic online scheduling. *Mathematics of Operations Research* 31(3).

Mehta, A.; Saberi, A.; Vazirani, U.; and Vazirani, V. 2007. Adwords and generalized online matching. *Journal of the ACM (JACM)* 54(5).

Mehta, A. 2012. Online matching and ad allocation. *Theoretical Computer Science* 8(4).

Miller, M. 2008. *Cloud computing: Web-based applications that change the way you work and collaborate online*. Que publishing.

Neuen, B. L.; Taylor, G. E.; Demaio, A. R.; and Perkovic, V. 2013. Global kidney disease. *The Lancet* 382(9900).

O'Mahony, E., and Shmoys, D. B. 2015. Data analysis and optimization for (citi) bike sharing. In *AAAI-15*.

Saran, R.; Li, Y.; Robinson, B.; Ayanian, J.; Balkrishnan, R.; Bragg-Gresham, J.; Chen, J.; Cope, E.; Gipson, D.; He, K.; et al. 2015. US renal data system 2014 annual data report: Epidemiology of kidney disease in the United States. *American Journal of Kidney Diseases* 65(6 Suppl 1).

Seow, K. T.; Dang, N. H.; and Lee, D.-H. 2010. A collaborative multiagent taxi-dispatch system. *IEEE Transactions on Automation Science and Engineering* 7(3).

Singhvi, D.; Singhvi, S.; Frazier, P. I.; Henderson, S. G.; O'Mahony, E.; Shmoys, D. B.; and Woodard, D. B. 2015. Predicting bike usage for new york city's bike sharing system. In *AAAI-15 Workshop on Computational Sustainability*.

Skutella, M.; Sviridenko, M.; and Uetz, M. 2016. Unrelated machine scheduling with stochastic processing times. *Mathematics of operations research* 41(3).

Sun, X.; Zhang, J.; and Zhang, J. 2016. Near optimal algorithms for online weighted bipartite matching in adversary model. *Journal of Combinatorial Optimization*.

Tong, Y.; She, J.; Ding, B.; Chen, L.; Wo, T.; and Xu, K. 2016a. Online minimum matching in real-time spatial data: experiments and analysis. *Proceedings of the VLDB Endowment* 9(12).

Tong, Y.; She, J.; Ding, B.; Wang, L.; and Chen, L. 2016b. Online mobile micro-task allocation in spatial crowdsourcing. In *ICDE-16*.

Verma, T.; Varakantham, P.; Kraus, S.; and Lau, H. C. 2017. Augmenting decisions of taxi drivers through reinforcement learning for improving revenues.

Yiu, M. L.; Mouratidis, K.; Mamoulis, N.; et al. 2008. Capacity constrained assignment in spatial databases. In *SIGMOD-08*.

Younge, A. J.; Von Laszewski, G.; Wang, L.; Lopez-Alarcon, S.; and Carithers, W. 2010. Efficient resource management for cloud computing environments. In *International Green Computing Conference*.

7 Supplementary Materials

7.1 Proof of Lemma 2

We show by induction on t as follows. When $t = 1$, $\beta_{e,t} = 1$ for all $e = (u, *)$ and we are done since

$$\sum_{e \in E_{v,t}} \frac{x^*_{e,t}}{p_{v,t}} \frac{\gamma}{\beta_{e,t}} \leq \sum_{e \in E_v} \frac{x^*_{e,t}}{p_{v,t}} \gamma \leq \frac{1}{2}$$

Assume for all $t' < t$, $\beta_{e,t'} \geq 1/2$ and ADAP(γ) is valid for all rounds before t. In other words, each e is made with probability equal to $x^*_{e,t'} * \frac{1}{2}$ for all $t' < t$. Now consider a given $e = (u, v)$. Observe that e is unsafe at t iff u is assigned with some v' at $t' < t$ such that the assignment $e' = (u, v')$ makes u unavailable at t. Therefore

$$1 - \beta_{e,t} = 1 - \Pr[\mathrm{SF}_{e,t}] = \sum_{t' < t} \sum_{e \in E_u} \frac{x^*_{e,t'}}{2} \Pr[C_e > t - t']$$

Thus from the constraints (3) in our benchmark LP, we see

$$\beta_{e,t} = 1 - \sum_{t' < t} \sum_{e \in E_u} \frac{x^*_{e,t'}}{2} \Pr[C_e > t - t'] \geq \frac{1}{2} + \frac{1}{2} \sum_{e \in E_u} x^*_{e,t} \geq \frac{1}{2}$$

Thus we are done since, $\sum_{e \in E_{v,t}} \frac{x^*_{e,t}}{p_{v,t}} \frac{\gamma}{\beta_{e,t}} \leq \sum_{e \in E_v} \frac{x^*_{e,t}}{p_{v,t}} \leq 1$.

7.2 Proof of Lemma 3

The inequality for $t = 1$ is due to the fact that u is safe at $t = 1$. For each time $t > 1$, Let $\mathrm{SF}_{u,t}$ be the event that u is safe at t and $A_{u,t}$ be the event that u is matched at t. Observe that for each window of time slots K, $\{\mathrm{SF}_{u,t}, A_{u,t'}, t - K + 1 \leq t' < t\}$ are disjoint events. Therefore,

$$\begin{aligned} 1 &= \Pr[\mathrm{SF}_{u,t}] + \sum_{t-K+1 \leq t' < t} \Pr[A_{u,t'}] \\ &= \gamma_{u,t} + \beta_u \sum_{t-K+1 \leq t' < t} \gamma_{u,t} \end{aligned}$$

7.3 Proof of Lemma 4

Proof. Focus on a given u. Notice that $\gamma_{u,t} + \beta_u \sum_{t-K+1 \leq t' < t} \gamma_{u,t'} = 1$ for all $1 \leq t \leq n$. Sum all equations over $t \in [n]$, we have

$$\begin{aligned} \Big(1 + \beta_u(K-1)\Big) \sum_{t \in [n]} \gamma_{u,t} &= n + \beta_u(K-1)\gamma_{u,n} + \beta_u(K-2)\gamma_{u,n-1} + \cdots + \beta_u \gamma_{u,n-K+2} \\ &\leq n + K - 1 \end{aligned}$$

Therefore we have

$$\sum_{t \in [n]} \gamma_{u,t} \leq \frac{n}{1 + \beta_u(K-1)} + \frac{K-1}{1 + \beta_u(K-1)} \leq \frac{n}{1 + \beta_u(K-1)} + \frac{1}{\beta_u}$$

Define $H_u \doteq \sum_t \beta_u \gamma_{u,t}$. From the above analysis, we have $H_u \leq \frac{n\beta_u}{1+\beta_u(K-1)} + 1$. Thus the objective value in the program (10) should be at most

$$\frac{\sum_u \sum_t \beta_u \gamma_{u,t}}{n} = \sum_{u \in U} \frac{H_u}{n} \leq \sum_{u \in U} \frac{\beta_u}{1 + \beta_u(K-1)} + \frac{K}{n}$$

We claim that the optimal value to the program (10) can be upper bounded by the below maximization program:

$$\left\{ \max \sum_{u \in [U]} \frac{\beta_u}{1 + \beta_u(K-1)} + \frac{K}{n} : \quad \sum_{u \in U} \beta_u = 1, \beta_u \geq 0, \forall u \in U \right\}$$

According to our assumption $K = o(n)$, the second term can be ignored. Let $g(x) = x/(1 + x(K-1))$. For any $K \geq 2$, it is a concave function, which implies that maximization of g subject to $\sum_u \beta_u = 1$ will be achieved when all $\beta_u = 1/K$. The resultant value is $\frac{1}{2-1/K} + o(1)$. Thus we are done. □