Online Matching in a Ride-Sharing Platform

Chinmoy Dutta *  Chris Sholley †

Abstract

We propose a formal graph-theoretic model for studying the problem of matching rides online in a ride-sharing platform. Unlike most of the literature on online matching, our model, that we call Online Windowed Non-Bipartite Matching (OWNBM), pertains to online matching in non-bipartite graphs. We show that the edge-weighted and vertex-weighted versions of our model arise naturally in ride-sharing platforms. We provide a randomized $\frac{1}{4}$-competitive algorithm for the edge-weighted case using a beautiful result of Lehmann, Lehmann and Nisan (EC 2001) for combinatorial auctions. We also provide an $\frac{1}{2}(1 - \frac{1}{\sqrt{e}})$-competitive algorithm for the vertex-weighted case (with some constraint relaxation) using insights from an elegant randomized primal-dual analysis technique of Devanur, Jain and Kleinberg (SODA 2013).

1 Introduction

In recent times, the rapid advancement and widespread availability of mobile computing have significantly uplifted the face of urban mobility and transportation. A large part of this change is due to the impact created by ride-sharing platforms. Such platforms allow any rider to request a ride at any time and any place at the convenience of a tap on a ride-sharing app installed on her mobile device. On receiving a ride request, the platform fulfils it by choosing one of the nearby drivers available to transport the rider and dispatching her to pick up the rider. Besides the convenience of the ride-hailing process, these platforms can also do marketplace optimization to ensure lower estimated time to pick-up for riders as well as better time utilization for drivers.

The major advantage of these platforms, however, is the ride-sharing possibility that they enable. With this feature, riders can choose to share the vehicle with other riders heading in the same direction. There are tremendous socio-economical and environmental advantages to this sharing. It lets the riders share the cost of the ride, thereby providing them an economical and affordable means of transportation. Riders can also take advantage of incentives like use

*Lyft, San Francisco. Email: cdutta@lyft.com
†Lyft, San Francisco. Email: chris@lyft.com
of high occupancy lanes. Moreover, ride-sharing eases the burden on the transportation infrastructure of cities and helps reduce traffic congestion, specially during heavy commute times. Reducing traffic congestion in turn helps in cutting down commute times, thus providing great economic benefit to individuals, businesses and institutions by reducing lost time and increasing productivity. Perhaps even more importantly, ride-sharing provides great environmental benefits by reducing air pollution as a result of reduced number of cars on roads, thereby greatly reducing our carbon footprint. In fact, for some densely populated and highly congested cities, intelligent coupling of ride-sharing with smart transit systems seems the only viable and scalable transportation option.

Sharing rides with others, however, does incur some costs to the riders like increased detours and travel times as well as some loss of privacy and convenience. The task of the ride-sharing platform is to intelligently trade off these costs against the benefits mentioned above. To be able to match rides that incur minimal inconvenience to the riders while maximizing marketplace efficiency, therefore, is of paramount importance for a ride-sharing platform. Considering the central role of ride-sharing platforms for transportation and economy, and the central role of this matching problem for such platforms, we strongly believe that there is need for modeling this problem formally and studying it mathematically. To the best of our knowledge, such rigorous attempt has not been made for this problem yet.

In this paper, we identify the essential aspects of the problem of matching rides in a ride-sharing platform. This then enables us to formally model the problem in graph-theoretic online matching setting. Our modeling leads to new problems in the space of online matching that we believe are of independent theoretical interest and importance. We then provide approximation algorithms for two different variants of our proposed model that arise naturally in the ride-sharing context.

1.1 Matching Rides for Ride-sharing

We start by identifying the unique attributes of the problem of matching rides for ride-sharing. We discuss both the constraints that must be satisfied and the objectives that might be optimized subject to satisfying those constraints.

1.1.1 Constraints

**Spatial constraint**  Matching rides obviously has a spatial constraint to it. That is, rides should be matched only if there is good spatial overlap among their routes. While the exact notion and criteria of spatial overlap is for the platform to decide, this ensures that none of the riders will experience an unacceptable amount of detour or increase in their travel time. With this spatial criteria in place, the existing rides in the platform that a new ride can be matched to is known once the new ride arrives.
**Temporal constraint** Matching rides also has a temporal constraint to it. For example, two rides cannot be matched together even if they have perfect spatial overlap (that is, exact same pick-up and drop-off locations), if they arrive at very different times. The acceptable maximum period between the arrival of two rides such that they can still be matched together is for the ride-sharing platform to determine. We call this quantity the *incidence window*.

**Online constraint** Furthermore, the ridesharing platform cannot wait too long for making its decisions and must make them in real-time as rides arrive. In other words, the problem must be solved *online* without full knowledge of rides to arrive in future. The decision that the platform needs to make for every arriving ride is whether to *match-or-dispatch*. If the platform decides to match the ride, it must pick an existing ride to match it to. Otherwise, the platform must dispatch a driver to pick-up the ride. The maximum period the platform can wait since the arrival of a ride before it must make a match-or-dispatch decision for it is called the *matching window*.

### 1.1.2 Objectives

**Matching efficiency objective** A widely used objective to optimize for while matching rides in a ride-sharing platform is the matching efficiency. A feasible match that satisfies the constraints above provides some efficiency for the system. For example, one notion of efficiency can be the amount of savings we get by making the match as compared to transporting the riders individually. The savings may be measured in terms of distance driven, time taken, cost of trips or any combination of these. Once the platform fixes the notion of efficiency, the goal it can optimize for could be the system-wide matching efficiency which is the sum of the efficiencies derived from all the matches made.

**Matching rate objective** Another objective that the platform might want to optimize for while matching rides is the matching rate. This is especially important in the growth stages of a ride-sharing platform where it wants to maximize the volume of shared rides subject to meeting the matching constraints. Different rides can have different value for the platform depending upon various factors like distance of the route, origin-destination locations, profitability etc. In this case, the goal the platform can optimize for is the matching rate which is the sum of the values of all the rides that were matched. Both matching efficiency and matching rate are widely used and very closely watched metrics in the ride-sharing industry.

### 1.2 Our contributions

In this paper, we provide a formal model for the problem of matching rides for ride-sharing in a graph-theoretic online matching setting. We call our model *online windowed non-bipartite matching* (OWNBM). In this model, time is considered discrete and vertices arrive sequentially, one per time step. The
edges in the graph represent matchability (that is, satisfying the spatial and temporal constraints). The edges originating from a vertex are revealed once the vertex arrives and each such edge can go to one of the preceding \( d \) (incidence window) vertices. It helps to think of the vertices as ordered by arrival times and edges as directed and going from a vertex to a preceding one by that ordering. After \( w \) (matching window) time steps since the arrival of a vertex, we must match it to one of the (yet unmatched) vertices it has an edge to or from, or leave it unmatched. A matching is a subset of edges such that every vertex has at most one edge incident on it (either to or from) that is included in the subset.

We consider two variants of our model. In the \textit{edge-weighted} variant, each edge of the graph has a weight and the objective is to construct a matching to maximize the sum of the weights of the edges picked in the matching. This corresponds to the matching efficiency objective for the ride-sharing platform mentioned above. In the \textit{vertex-weighted} variant of our model, each vertex of the graph has a weight and the objective is to construct a matching to maximize the sum of the weights of the vertices that get matched (has an edge of the matching incident on it, either to or from). This corresponds to the matching rate objective mentioned above.

We provide the following approximation algorithms for the edge and vertex weighted versions of windowed online non-bipartite matching.

\textbf{Theorem 1.} There is a randomized \( \frac{1}{4} \)-approximation for edge-weighted OWNBM with \( w = d \), where \( w \) and \( d \) are the matching and incidence windows respectively.

\textbf{Theorem 2.} There is a randomized \( \frac{1}{8}(1 - \frac{1}{e}) \)-approximation for vertex-weighted OWNBM with \( w = d \), where \( w \) and \( d \) are the matching and incidence windows respectively. This is under the assumption that the online algorithm is allowed to match three rides together and is allowed to delete an edge from a 3-match.

Note that in Theorem 2 above, while the online solution is allowed to match three rides together, the competitive ratio is measured against the offline optimal which only matches two rides together. (See details in sec 2.)

To the best of our knowledge, online matching has not yet been studied for non-bipartite graphs in adversarial settings. Our model is simple and yet powerful to capture the essential aspects of matching rides in a ride-sharing platform. We believe one of the main novelties of our work is that our model allows for studying online matching in non-bipartite graphs. The notion of incidence and matching windows that we propose leads to an interesting theoretical model that we believe is of independent interest and can find further use outside of the immediate context of ride-sharing.

### 1.3 Related work

Recently, there has been a lot of research interest in ride-sharing and associated challenges and problems. Much of the work has been done in transportation-related literature. A recent study by Santi et al [18] showed about 80% of rides in Manhattan can be shared by two riders. A lot of studies related to fleet...
management considers ride-sharing without pooling requests, e.g. [17, 20, 18, 19]. There has also been a lot of research interest in studying autonomous ride-sharing systems [17, 19, 6]. Heuristic-based solutions to the rider and driver assignment problem were studied in [14]. Alonso-Mora et al. [4] studied real-time high-capacity ride-sharing and route generation. However, their work does not provide a theoretical graph-theoretic online matching formulation for matching rides.

Starting with the seminal work of Karp, Vazirani and Vazirani [12], online matching has been widely studied and a vast amount of literature exists on the topic. However, such literature, being mostly motivated by the real-life applications such as online advertising, kidney exchange and online dating, considers the problem in the bipartite setting. The book by Mehta [16] gives a detailed overview of this literature. The work of Karp, Vazirani and Vazirani [12] gave an optimal online algorithm with competitive ratio \((1 - \frac{1}{e})\) for unweighted bipartite matching in the adversarial arrival model. The vertex-weighted version was studied by Aggarwal, Goel, Karande and Mehta [2] where they gave an optimal \((1 - \frac{1}{e})\) ratio. The edge-weighted version was studied under the additional economic assumption of free disposal by Feldman, Korula, Mirrokni, Muthukrishnan and Pl [8] where they gave a \((1 - \frac{1}{e})\) competitive ratio.

Feldman, Mehta, Mirrokni and Muthukrishnan [9] were the first to beat the \((1 - \frac{1}{e})\) bound for the unweighted version assuming IID arrival model and gave a competitive ratio of 0.67. This was improved by Manshadi, Gharan and Saberi [15] to 0.705. Jaliet and Lu [11] presented adaptive algorithms to further improve the ratio to 0.729 for the unweighted variant, and also achieved a ratio of 0.725 for the vertex-weighted variant. Haeupler, Mirrokni and Zadimoghadam [10] first beat the \((1 - \frac{1}{e})\) bound for the edge-weighted variant and achieved a ratio of 0.667.

Online matching literature for the non-bipartite setting is extremely limited. Anderson, Ashlagi, Gamarnik and Kanoria [5] studied a dynamic model of matching under homogeneous and independent stochastic assumptions to minimize average wait-times for agents. Akbarpour, Li and Gharan [3] studied dynamic matching with stochastically arriving and departing agents. However, as stated earlier, there has not been any work on online matching in the adversarial setting to the best of our knowledge.

1.4 Organization of the paper

We formally present our model, definitions and results in Section 2. The proof of our result for edge-weighted OWNBM (Theorem 1) is presented in Section 3 and that for the vertex-weighted case (Theorem 2) is presented in Section 4. Finally, we conclude in Section 5 with discussion of open problems and future directions for research.
2 Our Model

In this section, we formalize definitions and notations related to our model. We also restate our results formally.

Definition 3 (Online Windowed Non-Bipartite Matching (OWNBM)). An instance \( I \) of the Online Windowed Non-Bipartite Matching problem is a tuple \((G, d)\) where \( G \) is a graph and \( d \) is a non-negative integer. It satisfies the following properties.

- Vertices of \( G \) arrive sequentially, one at each time step.
- (Directed) edges of \( G \) originating from a vertex are revealed when the vertex arrives.
- [incidence window \((d)\)] Every (directed) edge in \( G \) originating from a vertex terminates at one of the \( d \) preceding vertices.

A matching \( M \) in \( I \) is a subset of edges of \( G \) such that for every vertex \( v \) of \( G \), at most one edge of \( M \) is incident on \( v \) (either originating from it or terminating at it).

Definition 4 (Edge-weighted OWNBM). An edge-weighted OWNBM is an instance \( I = (G, d) \) of OWNBM, where every edge \( e \) of \( G \) has an associated non-negative weight \( w_e \). The weight of a matching \( M \) in \( I \) is \( \sum_{e \in M} w_e \).

Definition 5 (Vertex-weighted OWNBM). A vertex-weighted OWNBM is an instance \( I = (G, d) \) of OWNBM, where every vertex \( v \) of \( G \) has an associated non-negative weight \( w_v \). The weight of a matching \( M \) in \( I \) is \( \sum_{(u,v) \in M} w_u + w_v \).

Definition 6 ((\( r, w \))-Competitive algorithm for OWNBM). An \((r, w)\)-Competitive algorithm \( ALG \) for OWNBM with \( r \leq 1 \) and \( w \geq 0 \) is an algorithm that, for every instance \( I = (G, d) \) of OWNBM, constructs a matching \( M \) in \( I \) online such that

- [matching window \((w)\)] any edge \( e = (j, i) \) terminating at a vertex \( i \) included in \( M \) is picked no later than \( w \) time steps after the arrival of \( i \) (and hence also \( j \)). An edge once picked cannot be deleted from \( M \) later.

- [competitive ratio \((r)\)] \( \frac{ALG_I}{OPT_I} \geq r \), where \( ALG_I \) is the weight of the matching \( M \) and \( OPT_I \) is the maximum (optimal) weight of a matching in \( I \). The weight of the matching is defined depending on whether it is an edge-weighted or a vertex-weighted instance of OWNBM. In case \( ALG \) is randomized, \( M \) is a random variable and \( ALG_I \) is the expected weight of \( M \) over the randomness used in \( ALG \).

We now restate our main results that we already presented earlier.

Theorem (Restatement of Theorem 1). There is a randomized \((\frac{1}{4}, d)\)-competitive algorithm for edge-weighted OWNBM where \( d \) is the incidence window of the OWNBM instance.
Theorem (Restatement of Theorem 2). There is a randomized \((\frac{1}{2}(1 - \frac{1}{e}), d)\)-competitive algorithm for vertex-weighted OWNBM where \(d\) is the incidence window of the OWNBM instance, if we allow the online algorithm (but not the offline optimal algorithm) to

1. form a 3-matching, and
2. delete an edge \((j, i)\) from a 3-set no later than \(d\) steps since the arrival of vertex \(j\).

A 3-matching \(M\) in \(I\) is a set of disjoint subsets of vertices of \(G\), each of size 2 or 3, such that each 2-set is an edge in \(G\), and the induced subgraph of each 3-set has at least 2 edges. The weight of a 3-matching is defined as the sum of weight of all vertices included a 2-set or a 3-set.

Note that we do not assume any distribution on edges for our results above. In particular, our results hold for the adversarial model for the arriving vertices and edges.

It will be convenient to define a notion of semi-matching for the proofs of both the above theorems.

Definition 7 (Semi-matching). Given an instance \(I = (G, d)\) of OWNBM, a semi-matching \(M'\) in \(I\) is a subset of edges of \(G\) such that for every vertex \(v\) of \(G\), at most one edge of \(M'\) originates from \(v\) and at most one edge of \(M'\) terminates at \(v\).

3 Edge-Weighted OWNBM

In this section, we prove Theorem 1. We will need a beautiful result from combinatorial auction theory by Lehmann, Lehmann and Nisan [13]. First, we define an instance of a combinatorial auction from a given instance of OWNBM.

Definition 8. Let \(I = (G, d)\) be an instance of edge-weighted OWNBM, and let \(G = (V, E)\) where \(V\) and \(E\) are the sets of vertices and edges of \(G\) respectively. Let \(|V| = n\). Define a combinatorial auction with \(n\) bidders and \(n\) items. Without loss of generality and slightly abusing notation, we assume the vertex set, the item set and the bidder set to be \([n]\). The valuation \(\text{val}_i\) of the \(i\)'th bidder for a set \(S\) of items is given by the following.

\[
\text{val}_i(S) = \max_{j \in S; (j, i) \in E} \max_{(j, i) \in E} w(j, i), 0
\]

The following lemma upper bounds the weight of optimal matching in a edge-weighted OWNBM by the valuation of optimal allocation of the associated combinatorial auction.

Lemma 9. Let \(I = (G, d)\) be an instance of edge-weighted OWNBM and \(\text{OPT}_I\) be the weight of an optimal matching. Then the optimal allocation for the combinatorial auction as defined in Definition 8 has valuation at least \(\text{OPT}_I\).
Algorithm 1: $\frac{1}{2}$-approximation for combinatorial auctions with submodular valuations

**Input:** $n$ bidders with submodular valuations and $m$ items 

**Output:** A $\frac{1}{2}$-approximation to the optimal allocation 

$S_1 = S_2 = \ldots = S_n \leftarrow \emptyset$; 

for each item $j = 1, 2, \ldots, m$ do 

let $i$ be the bidder with highest marginal valuation $\text{val}_i(j|S_i)$ for item $j$; 

$S_i \leftarrow S_i \cup \{j\}$; 

end 

return $S_1 = S_2 = \ldots = S_n$;

Proof. To prove the lemma, we produce an allocation with valuation $\text{OPT}_{I}$. Consider an optimal matching $M$ in $I$. For any edge $(j, i) \in M$, set $S_i \leftarrow \{i, j\}$ and $S_j \leftarrow \emptyset$. For any unmatched vertex $k$ of $G$, set $S_k \leftarrow \{k\}$. It is easy to see that the valuation of this allocation is $\text{OPT}_{I}$, which proves the lemma.

Next, we show that the valuation functions for the auction derived from an edge-weighted OWNBM are sub-modular.

Lemma 10. The valuation functions for all the bidders in Definition 8 are sub-modular.

Proof. Fix bidder $i$ and consider its valuation function $\text{val}_i$ as defined in Definition 8. Also consider sets $S \subseteq T$ of items and an item $k$. We will show that the marginal valuation of item $k$ given $T$ is at most the marginal valuation of $k$ given $S$, i.e., $\text{val}_i(k|T) \leq \text{val}_i(k|S)$. Assume the contrary. It is not hard to see that this will imply that

$$\max_{j \in S, (j,i) \in E} w(j,i) > \max_{j \in T, (j,i) \in E} w(j,i).$$

Since $S \subseteq T$, this is an impossibility.

Lemma 10 lets us use the following 2-approximation from [13] for combinatorial auctions with sub-modular valuations. (Holds for $n$ bidders and $m$ items; in our case, $n = m$.)

Theorem (Theorem 11 of [13]). If the valuation functions of all the bidders are submodular, then Algorithm 1 is a $\frac{1}{2}$-approximation for the optimal allocation.

A useful property of the above theorem is that it works in our online setting as shown in the following lemma.

Lemma 11. Let $I = (G, d)$ be an instance of edge-weighted OWNBM and $\text{OPT}_{I}$ be the weight of an optimal matching. Then we can construct an allocation for the combinatorial auction as defined in Definition 8 online with valuation at least $\frac{1}{2} \text{OPT}_{I}$. Here, the $i$’th bidder and the $i$’th item arrives with the arrival of vertex $i$. 

8
Proof. From Theorem 3 and Lemma 10 we immediately see that we can construct an allocation with valuation at least half that of the optimal allocation offline once all the vertices of $G$ have arrived. Combining with Lemma 9 gives the valuation of this offline allocation to be at least $\frac{1}{2} \OPT_I$. The lemma follows by noting that the allocation can be done online (that is, items and bidders corresponding to a vertex $i$ arrives with the arrival of vertex $i$ and item $i$ is allocated to a bidder as soon as the item arrives) due to the following two facts.

1. the $\frac{1}{2}$-approximation algorithm presented above can be run sequentially as items arrive.
2. the marginal valuation of an item for a bidder that has not arrived yet when the item arrives is 0.

The following lemma shows that we can do an approximation-preserving transformation from a valuation to a semi-matching online.

**Lemma 12.** Let $\mathcal{I} = (G, d)$ be an instance of edge-weighted OWNBM and $\OPT_I$ be the weight of an optimal matching. Then we can construct a semi-matching $\mathcal{M}'$ of weight $\frac{1}{2} \OPT_I$ online such that any edge $(j, i) \in \mathcal{M}'$ terminating at vertex $i$ is picked no later than $d$ time steps after its arrival.

**Proof.** Start by setting $\mathcal{M}' \leftarrow \emptyset$. Construct the allocation for the combinatorial auction online as in the proof of Lemma 11. Note that an item $j$ is allocated as soon as it arrives. As a result, given the incidence window property of $\mathcal{I}$, the allocation of bidder $i$, $S_i$ is frozen $d$ time steps after arrival of vertex $i$. For a vertex $i$, if its valuation $\text{val}(S_i) = 0$ after $d$ steps from its arrival, then do not pick any edge terminating at vertex $i$. Otherwise, set $\mathcal{M}' \leftarrow \mathcal{M}' \cup \{(j, i)\}$, where

$$j = \arg\max_{k \in S_i} w(k, i).$$

It is clear that $\mathcal{M}'$ is a semi-matching as every item is allocated to at most one bidder which means at most one edge of $\mathcal{M}'$ originates from a given vertex $j$. Also, by construction, we include at most one edge terminating at a given vertex $i$ in $\mathcal{M}'$. It is also clear that the weight of the semi-matching thus constructed is the same as the valuation of the allocation, because for any $i$, the valuation of the allocation for bidder $i$ equals the weight of the edge terminating at vertex $i$ included in $\mathcal{M}'$. Therefore, by Lemma 11 weight of the constructed semi-matching $\mathcal{M}'$ is at least $\frac{1}{2} \OPT_I$.

The remaining part in the proof of Theorem 4 is to show how can we go from a semi-matching of weight $w$ to a matching of weight at least $w/2$ online. Consider Algorithm 2 that takes a semi-matching $\mathcal{M}'$ and outputs a matching $\mathcal{M}$ in an edge-weighted OWNBM.

**Lemma 13.** Let $\mathcal{I} = (G, d)$ be an instance of edge-weighted OWNBM and $\mathcal{M}'$ be a semi-matching in $\mathcal{I}$ of weight $w$. Then Algorithm 2 constructs a random matching $\mathcal{M}$ in $\mathcal{I}$ of expected weight $w/2$.  

9
Algorithm 2: Constructing matching from semi-matching for edge-weighted OWNBM

**Input:** A semi-matching $M'$ in an edge-weighted OWNBM $I$

**Output:** A (random) matching $M$ in $I$ with expected weight at least half the weight of $M'$

$M \leftarrow \emptyset$;

for each vertex $i$ in increasing order do

if $(j, i) \in M'$ for some vertex $j$ then

if $(i, k) \in M'$ for some vertex $k$ then

if color$(k) == \text{green}$ then

color$(i) = \text{red}$;

end

if color$(k) == \text{red}$ then

color$(i) = \text{green}$;

end

else

choose color$(i)$ green or red with probability $1/2$ each;

end

if color$(i) == \text{green}$ then

$M \leftarrow M \cup (j, i)$;

end

end

return $M$;

**Proof.** We need to show that

1. the constructed set $M$ is a matching, and

2. weight of $M$ is $1/2$ of the weight of $M'$.

Notice that we include edge $(j, i)$ in $M$ iff color$(i)$ is green. For (1), assume contrary. Then there exist vertices $i$, $j$ and $k$ such that both the edges $(i, k)$ and $(j, i)$ are included in $M$. Since we process vertices in order and vertex $k$ must have been processed first. Since edge $(i, k)$ was included in $M$, color$(k)$ must be green. This implies color$(i)$ is red, which contradicts that edge $(j, i)$ was also included in $M$.

For (2), it can be easily shown by induction that for any vertex $i$, such that there exists some edge $(j, i)$ in $M'$, color$(i) = \text{green}$ with probability $1/2$. This means the edge $(j, i)$ gets included in $M$ with probability $1/2$, which proves that the weight of $M$ is $1/2$ of the weight of $M'$.

**Proof of Theorem 1.** Combining Lemma 12 and Lemma 13, we immediately see that given an instance $I = (G, d)$ of edge-weighted OWNBM, we can construct a matching $M$ in $I$ of weight $\frac{1}{4}\text{OPT}_I$. Since the semi-matching constructed in
Lemma 12 picks an edge \((j, i)\) no later than \(d\) time steps after the arrival of \(i\), and since Algorithm 2 processes edges in increasing order of the terminal vertex, this yields a \((\frac{1}{4}, d)\)-competitive algorithm for edge-weighted OWNBM.

\[ \qed \]

## 4 Vertex-Weighted OWNBM

In this section, we prove Theorem 2.

**Lemma 14.** Let \(I = (G, d)\) be an instance of vertex-weighted OWNBM. Then Algorithm 3 constructs a (random) semi-matching \(M'\) in \(I\) online such that any edge \((j, i) \in M'\) terminating at vertex \(i\) is picked no later than \(d\) time steps after its arrival.

**Proof.** Obvious from the algorithm. \(\qed\)

We could lower bound the weight of the semi-matchings constructed by Algorithm 3. But for our proof, we need a stronger claim for which we define the notion of half-weight.

**Definition 15.** A semi-matching \(M'\) in an instance \(I\) of vertex-weighted OWNBM constructed by Algorithm 3 is said to be of origin-type if the random variable \(type\) took the value \(origin\); else it is said to be of destination-type. The half-weight of an origin-type semi-matching is defined as

\[
\sum_{(j, i) \in M'} w_j;
\]

while the half-weight of a destination-type semi-matching is defined as

\[
\sum_{(j, i) \in M'} w_i.
\]

We use a beautiful primal-dual proof technique introduced by Devanur, Jain and Kleinberg [7] for lower bounding the expected half-weight of a semi-matching constructed by the algorithm. However, the proof is omitted in this shorter exposition.

**Lemma 16.** Let \(I = (G, d)\) be an instance of vertex-weighted OWNBM and \(OPT_I\) be the weight of an optimal matching. Then Algorithm 3 constructs a (random) semi-matching \(M'\) of expected half-weight \(\frac{1}{2}(1 - \frac{1}{e})OPT_I\).

To complete the proof of Theorem 2 we need to show how can we go from a semi-matching of half-weight \(w\) to a 3-matching of weight at least \(w\) online. The proof of the following lemma is omitted.

**Lemma 17.** Let \(I = (G, d)\) be an instance of vertex-weighted OWNBM and \(M'\) be a semi-matching in \(I\) of weight at least \(w\). Then we can construct a 3-matching \(M\) in \(I\) of weight at least \(w\) by processing vertices in order. An edge \((j, i)\) is picked in a 2-set or a 3-set while processing vertex \(i\), and may get deleted from a 3-set while processing vertex \(j\).

11
Proof of Theorem 2. Combining Lemma 16 and Lemma 13, we immediately see that given an instance \( I = (G, d) \) of vertex-weighted OWNBM, we can construct a 3-matching \( \mathcal{M} \) in \( I \) of weight \( \frac{1}{2}(1 - \frac{1}{e})\text{OPT}_I \). Since the semi-matching constructed in Algorithm 3 picks an edge \((j, i)\) no later than \( d \) time steps after the arrival of \( i \), and since Lemma 17 processes edges in increasing order of the terminal vertex, this yields a randomized \((\frac{1}{2}(1 - \frac{1}{e}), d)\)-competitive algorithm for vertex-weighted OWNBM where an edge \((j, i)\) picked in a 3-set may get deleted no later than \( d \) steps since the arrival of vertex \( j \). \( \square \)

5 Conclusions

In this work, we initiated the study of matching rides in a ride-sharing platform in a formal graph-theoretic online matching setting. For this purpose, we proposed a model that we call Online Windowed Non-Bipartite Matching (OWNBM). Our model is simple and elegant, and yet remarkably powerful to capture the important attributes of the rider matching problem in ride-sharing platforms. We showed how the edge-weighted and vertex-weighted versions of OWNBM capture the objectives of matching efficiency and matching rate, which are important and widely used metrics in ride-sharing industry.

We provided a randomized \( \frac{1}{4} \)-competitive algorithm for the edge-weighted version, and a randomized \( \frac{1}{2}(1 - \frac{1}{e}) \)-competitive algorithm for the vertex-weighted version of OWNBM (with some constraints relaxed). While we believe our bounds might be further improved, these are first results for online matching in non-bipartite graphs in the adversarial setting to the best of our knowledge. Our result for the edge-weighted version uses a beautiful result from combinatorial auction theory, while that for the vertex-weighted version uses insights and techniques from a very elegant randomized primal-dual analysis technique in matching theory.

We believe our model is of independent theoretical interest and can find use outside of the immediate context of ride-sharing. In fact, one of the novelties of our work is proposing a model for studying online matching in non-bipartite graphs. We leave a range of open questions and future research directions. We believe it is possible to improve the bounds we presented. Our model can also be studied in the stochastic setting where edges are drawn from a distribution which is known or can be learned. We considered two-party matches in this work. One can consider hyper-edges in matching which correspond to matching more than two parties together for sharing a car. We believe that these and other related challenging problems are both theoretically interesting and practically relevant.

References

[1] Niels Agatz, Alan L. Erera, Martin W. P. Savelsbergh, and Xing Wang. Dynamic ride-sharing: a simulation study in metro atlanta. Transp Res
[2] Gagan Aggarwal, Gagan Goel, Chinmay Karande, and Aranyak Mehta. Online vertex weighted bipartite matching and single-bid budgeted allocations. In Proceedings of the twenty-second annual ACM-SIAM symposium on Discrete Algorithms, 2011.

[3] Mohammad Akbarpour, Shengwu Li, and Shayan Oveis Gharan. A dynamic model of barter exchange. In Proceedings of the ACM conference on Economics and computation, 2014.

[4] Javier Alonso-Mora, Samitha Samaranayake, Alex Wallar, Emilio Frazzoli, and Daniela Rus. On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment. Proc Natl Acad Sci USA, 2017.

[5] Ross Anderson, Itai Ashlagi, David Gamarnik, and Yash Kanoria. A dynamic model of barter exchange. In Proceedings of the ACM-SIAM Symposium on Discrete Algorithms, 2015.

[6] Gonalo Homem de Almeida Correia and Bart van Arem. Solving the user optimum privately owned automated vehicles assignment problem (uopoavap): a model to explore the impacts of self-driving vehicles on urban mobility. Transp Res Part B Methodological, 2016.

[7] Nikhil R. Devanur, Kamal Jain, and Robert D. Kleinberg. Randomized primal-dual analysis of ranking for online bipartite matching. In Proceedings of the twenty-fourth annual ACM-SIAM symposium on Discrete algorithms, 2013.

[8] Jon Feldman, Nitish Korula, Vahab Mirrokni, S. Muthukrishnan, and Martin Pl. Online ad assignment with free disposal. In Internet and network economics, 2009.

[9] Jon Feldman, Aranyak Mehta, Vahab Mirrokni, and S. Muthukrishnan. Online stochastic matching: Beating 1-1/e. In Proceedings of the Foundations of Computer Science, 2009.

[10] Bernhard Haeupler, Vahab S. Mirrokni, and Morteza Zadimoghaddam. Online stochastic weighted matching: Improved approximation algorithms. Internet and Network Economics, 2011.

[11] Patrick Jaillet and Xin Lu. Online stochastic matching: New algorithms with better bounds. Mathematics of Operations Research, 2013.

[12] Richard M. Karp, Umesh V. Vazirani, and Vijay V. Vazirani. An optimal algorithm for on-line bipartite matching. In Proceedings of the twenty-second annual ACM symposium on Theory of computing, 1990.

[13] Benny Lehmann, Daniel Lehmann, and Noam Nisan. Combinatorial auctions with decreasing marginal utilities. In Proceedings of the 3rd ACM conference on Electronic Commerce, 2001.
[14] Shuo Ma, Yu Zheng, and Ouri Wolfson. A large scale dynamic taxi ridesharing service. In *Proc of the 29th International Conference on Data Engineering*, 2013.

[15] Vahideh H. Manshadi, Shayan Oveis Gharan, and Amin Saberi. Online stochastic matching: Online actions based on offline statistics. *Mathematics of Operations Research*, 2012.

[16] Aranyak Mehta. *Online matching and ad allocation*. Foundations and Trends in Theoretical Computer Science, 2012.

[17] Marco Pavone, Stephen L Smith, Emilio Frazzoli, and Daniela Rus. Robotic load balancing for mobility-on-demand systems. *Int J Rob Res*, 2012.

[18] Paolo Santi, Giovanni Resta, Michael Szell, Stanislav Sobolevsky, Steven H. Strogatz, and Carlo Ratti. Quantifying the benefits of vehicle pooling with shareability networks. In *Proc Natl Acad Sci USA*, 2014.

[19] Kevin Spieser, Saniitha Samaranayake, Wolfgang Gruel, and Emilio Frazzoli. Shared-vehicle mobility-on-demand systems: A fleet operators guide to rebalancing empty vehicles. In *Transportation Research Board 95th Annual Meeting*, 2016.

[20] Rick Zhang and Marco Pavone. Control of robotic mobility-on-demand systems: a queueing-theoretical perspective. In *Proceedings of Robotics: Science and Systems Conference*, 2014.
Algorithm 3: Constructing semi-matching for vertex-weighted OWNBM

Input: An instance $\mathcal{I} = (\mathcal{G}, d)$ of vertex-weighted OWNBM
Output: A semi-matching $\mathcal{M}'$ in $\mathcal{I}$

let $n$ be the number of vertices in $\mathcal{G}$;

define function $g(Y) = e^{Y-1}$;

$\mathcal{M}' \leftarrow \emptyset$;

assume all vertices in $\mathcal{G}$ are colored white;
pick type as origin or destination with probability $1/2$ each;

if type == destination then
    for vertex $j$ in increasing order do
        pick $Y_j \in [0, 1]$ uniformly at random;
        let $N(j)$ be the set of vertices $k$ of color white such that $(j, k)$ is an edge of $\mathcal{G}$;
        if $Nj \neq \emptyset$ then
            let $i = \arg\max \{w_k (1 - g(Y_k)) : k \in N(j)\}$;
            $\mathcal{M}' \leftarrow \mathcal{M}' \cup \{(j, i)\}$;
            color vertex $i$ black;
        end
    end
else
    add $d$ dummy vertices of weight 0 with no edges last in the ordering;

    for vertex $t$ in increasing order do
        pick $Y_t \in [0, 1]$ uniformly at random;
        let $i = t - d$;
        if $i > 0$ then
            let $N(i)$ be the set of vertices $k$ of color white such that $(k, i)$ is an edge of $\mathcal{G}$;
            if $Ni \neq \emptyset$ then
                let $j = \arg\max \{w_k(1 - g(Y_k)) : k \in N(i)\}$;
                $\mathcal{M}' \leftarrow \mathcal{M}' \cup \{(j, i)\}$;
                color vertex $j$ black;
            end
        end
    end
end

return $\mathcal{M}'$;