# New Algorithms, Better Bounds, and a Novel Model for Online Stochastic Matching*†

Brian Brubach[1], Karthik Abinav Sankararaman[2], Aravind Srinivasan[3], and Pan Xu[4]

1 Department of Computer Science, University of Maryland, College Park, USA
bbrubach@cs.umd.edu
2 Department of Computer Science, University of Maryland, College Park, USA
kabinav@cs.umd.edu
3 Department of Computer Science, University of Maryland, College Park, USA
srin@cs.umd.edu
4 Department of Computer Science, University of Maryland, College Park, USA
panxu@cs.umd.edu

## Abstract

Online matching has received significant attention over the last 15 years due to its close connection to Internet advertising. As the seminal work of Karp, Vazirani, and Vazirani has an optimal $(1 - 1/e)$ competitive ratio in the standard adversarial online model, much effort has gone into developing useful online models that incorporate some stochasticity in the arrival process. One such popular model is the "known I.I.D. model" where different customer-types arrive online from a known distribution. We develop algorithms with improved competitive ratios for some basic variants of this model with integral arrival rates, including: (a) the case of general weighted edges, where we improve the best-known ratio of 0.667 due to Haeupler, Mirrokni and Zadimoghaddam [11] to 0.705; and (b) the vertex-weighted case, where we improve the 0.7250 ratio of Jaillet and Lu [12] to 0.7299. We also consider two extensions, one is "known I.I.D." with non-integral arrival rate and stochastic rewards; the other is "known I.I.D." $b$-matching with non-integral arrival rate and stochastic rewards. We present a simple non-adaptive algorithm which works well simultaneously on the two extensions.

One of the key ingredients of our improvement is the following (offline) approach to bipartite-matching polytopes with additional constraints. We first add several valid constraints in order to get a good fractional solution $\mathbf{f}$; however, these give us less control over the structure of $\mathbf{f}$. We next *remove* all these additional constraints and randomly move from $\mathbf{f}$ to a feasible point on the matching polytope with all coordinates being from the set $\{0, 1/k, 2/k, \ldots, 1\}$ for a chosen integer $k$. The structure of this solution is inspired by Jaillet and Lu (*Mathematics of Operations Research*, 2013) and is a tractable structure for algorithm design and analysis. The appropriate random move preserves many of the removed constraints (approximately [exactly] with high probability [in expectation]). This underlies some of our improvements, and, we hope, could be of independent interest.

**1998 ACM Subject Classification** F.2 Analysis of Algorithms and Problem Complexity

**Keywords and phrases** Ad-Allocation, Online Matching, Randomized Algorithms

**Digital Object Identifier** 10.4230/LIPIcs.ESA.2016.24

* A full version of the paper is available at http://arxiv.org/abs/1606.063956.
† Supported in part by NSF Awards CNS-1010789 and CCF-1422569, and a research award from Adobe, Inc.


24th Annual European Symposium on Algorithms (ESA 2016).
Editors: Piotr Sankowski and Christos Zaroliagis; Article No. 24; pp. 24:1–24:16
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

# 1 Introduction

Applications to Internet advertising have driven the study of online matching problems in recent years [19]. In these problems, we consider a bipartite graph $G = (U, V, E)$ in which the set $U$ is available offline while the vertices in $V$ arrive online. Whenever some vertex $v$ arrives, it must be matched immediately to at most one vertex in $U$. Each offline vertex $u$ can be matched to at most one $v$ or in the $b$-matching generalization, at most $b$ vertices in $V$. In the context of Internet advertising, $U$ is the set of advertisers, $V$ is a set of impressions, and the edges $E$ define the impressions that interest a particular advertiser. When $v$ arrives, we must choose an available advertiser (if any) to match with it. Initially, we consider the case where $v \in V$ can be matched at most once. We later relax this condition to it being matched up to $b$ times. Since advertising forms the key source of revenue for many large Internet companies, finding good matching algorithms and obtaining even small performance gains can have high impact. Additionally, bipartite matching is a fundamental combinatorial optimization problem. Hence, any improvements is interesting from a theoretical standpoint.

In the *stochastic known I.I.D.* model of arrival, we are given the bipartite graph in advance and each arriving vertex $v$ is drawn with replacement from a known distribution on the vertices in $V$. This captures the fact that we often have background data about the impressions and can predict the frequency with which each type of impression will arrive. Edge-weighted matching [8] is a general model in the context of advertising: every advertiser gains a given revenue for being matched to a particular type of impression. Here, a *type* of impression refers to a class of users (e.g., a demographic group) who are interested in the same subset of advertisements. A special case of this model is vertex-weighted matching [1], where weights are associated only with the advertisers. In other words, a given advertiser has the same revenue generated for matching any of the user types interested in it.

In some modern business models, revenue is not generated upon matching advertisements, but only when a user *clicks* on the advertisement: this is the *pay-per-click* model. From background data, one can assign the probability of a particular advertisement being clicked by a type of user. Works including [20],[21] capture this notion by assigning a probability to each edge.

One unifying theme in most of our approaches is to use an LP benchmark with additional valid constraints that hold for the respective stochastic-arrival models, combined with some form of dependent rounding.

## 1.1 Related work

For readers not familiar with these problems, they are encouraged to first read parts of section 2 for formal definitions before getting into the related work. The study of online matching began with the seminal work of Karp, Vazirani, Vazirani [14], where they gave an optimal online algorithm for a version of the unweighted bipartite matching problem in which vertices arrive in adversarial order. Following that, a series of works have studied various related models. The book by Mehta [19] gives a detailed overview. The vertex-weighted version of this problem was introduced by Aggarwal, Goel and Karande [1], where they give an optimal $\left(1 - \frac{1}{e}\right)$ ratio for the adversarial arrival model. The edge-weighted setting has been studied in the adversarial model by Feldman, Korula, Mirrokni and Muthukrishnan [8], where they consider an additional relaxation of "free-disposal".

Beyond the adversarial model, these problems are studied under the name *stochastic matching*, where the online vertices either arrive in random order or are drawn I.I.D. from a known distribution. The works [5, 15, 16, 17] among others, study the random arrival order model; papers including [4, 9, 11, 12, 18, 6] study the I.I.D. arrival order model. Another variant of this problem is when the edges have stochastic rewards. Models with stochastic rewards have been previously studied by [20], [21] among others, but not in the known I.I.D. model.

**Related Work in the Vertex-Weighted and Unweighted Settings:** The vertex-weighted and unweighted settings have many results starting with Feldman, Mehta, Mirrokni and Muthukrishnan [9] who were the first to beat $1 - 1/e$ with a competitive ratio of 0.67 for the unweighted problem. This was improved by Manshadi, Gharan, and Saberi [18] to 0.705 with an adaptive algorithm. In addition, they showed that even in the unweighted variant with integral arrival rates, no algorithm can achieve a ratio better than $1 - e^{-2} \approx 0.86$. Finally, Jaillet and Lu [12] presented an adaptive algorithm which used a clever LP to achieve 0.725 and $1 - 2e^{-2} \approx 0.729$ for the vertex-weighted and unweighted problems, respectively.

**Related Work in the Edge-Weighted Setting:** For this model, Haeupler, Mirrokni, Zadimoghaddam [11] were the first to beat $1 - 1/e$ by achieving a competitive ratio of 0.667. They use a *discounted LP* with tighter constraints than the basic matching LP (a similar LP can be seen in 2.1) and they employ the *power of two choices* by constructing two matchings offline to guide their online algorithm.

**Related Work in Online $b$-matching:** In the model of $b$-matching, we assume each vertex $u$ has a uniform capacity of $b$, where $b$ is a parameter which is generally a large integral value. The model of unweighted $b$-matching can be viewed as a special case of Adwords or Display Ads. There is extensive literature for Adwords or Display Ads under various settings (see the book by Mehta [19]). In particular, [13] shows that their algorithm BALANCE is optimal for online $b$-matching under the adversarial model, which achieves a ratio of $1 - \frac{1}{(1+1/b)^b}$.

In this paper, we consider edge-weighted $b$-matching with stochastic rewards under the known I.I.D. model with arbitrary arrival rates. To the best of our knowledge, we are the first to consider this very general model. Devanur *et al* [7] gave an algorithm which achieves a ratio of $1 - 1/\sqrt{2\pi k}$ for the Adwords problem in the Unknown I.I.D. arrival model with knowledge of the optimal budget utilization and when the bid to budget ratios are at most $1/k$. Notice that even the problem of general edge-weighted $b$-matching with deterministic rewards cannot be captured in the Adwords model. Alaei *et al* [2] consider the Prophet-Inequality Matching problem, in which $v$ arrives from a distinct (known) distribution $\mathcal{D}_t$, in each round $t$. They gave a $1 - 1/\sqrt{k+3}$ competitive algorithm, where $k$ is the minimum capacity of $u$. They assume deterministic rewards however, and it is non-trivial to extend their result to the stochastic reward setting. In this paper, we present a very simple algorithm which achieves a ratio of $1 - b^{-1/2+\epsilon} - O(e^{-b^{2\epsilon}/3})$ for any given $\epsilon > 0$. It is worthwhile to see that our algorithm (5) can be trivially extended to the case where each vertex $u$ has a distinct capacity $b_u$. The value of $b$ in the final ratio would be replaced by $\min_{u \in U} b_u$.

## 2 Preliminaries

In the *Unweighted Online Known I.I.D. Stochastic Bipartite Matching* problem, we are given a bipartite graph $G = (U, V, E)$. The set $U$ is available offline while the vertices $v$ arrive online and are drawn with replacement from an I.I.D. distribution on $V$. For each $v \in V$, we are given an *arrival rate* $r_v$, which is the expected number of times $v$ will arrive. With the exception of Sections 5 and 6, this paper will focus on the integral-arrival-rates setting where all $r_v \in \mathbb{Z}^+$. As described in [11], WLOG we can assume in this setting that $\forall v \in V, r_v = 1$. Let $n = \sum_{v \in V} r_v$ be the expected number of vertices arriving during the online phase.

In the **vertex-weighted** variant, every vertex $u \in U$ has a weight $w_u$ and we seek a maximum weight matching. In the **edge-weighted** variant, every edge $e \in E$ has a weight $w_e$ and we seek a maximum weight matching. In the **stochastic rewards** variant [1], additionally, each edge has a probability $p_e$ and we seek to maximize the expected weight of the matching. In the **b-matching** model, every vertex in $U$ can be matched upto $b$ times. Throughout, we will use "WS" to refer to the worst case for various algorithms. Asymptotic assumption and notation: We will always assume $n$ is large and analyze algorithms as $n$ goes to infinity: e.g., if $x \leq 1 - (1 - 2/n)^n$, we will just write this as "$x \leq 1 - 1/e^2$" instead of the more-accurate "$x \leq 1 - 1/e^2 + o(1)$". These suppressed $o(1)$ terms will subtract at most $o(1)$ from our competitive ratios. Another fact to note is that the **competitive ratio** is defined slightly different than usual, for this set of problems (Similar to notation used in [19]). In particular, it is defined as $\frac{\mathbb{E}[ALG]}{\mathbb{E}[OPT]}$. Algorithms can be **adaptive** or **non-adaptive**. When $v$ arrives, an adaptive algorithm can check which neighbors are still available to be matched, but a non-adaptive algorithm cannot.

## 2.1 LP Benchmark

We will use the following LP to upper bound the optimal offline solution and guide our algorithm. We will first show an LP for the unweighted variant, then describe changes for the vertex-weighted and edge-weighted settings. As usual, we have a variable $f_e$ for each edge. Let $\partial(w)$ be the set of edges adjacent to a vertex $w \in U \cup V$ and let $f_w = \sum_{e \in \partial(w)} f_e$.

$$
\begin{aligned}
\text{maximize} \quad & \sum_{e \in E} f_e && && (2.1)\\
\text{subject to} \quad & \sum_{e \in \partial(u)} f_e \leq 1 && \forall u \in U && (2.2)\\
& \sum_{e \in \partial(v)} f_e \leq 1 && \forall v \in V && (2.3)\\
& 0 \leq f_e \leq 1 - 1/e && \forall e \in E && (2.4)\\
& f_e + f_{e'} \leq 1 - 1/e^2 && \forall e, e' \in \partial(u), \forall u \in U && (2.5)
\end{aligned}
$$

**Variants:** The objective function is: maximize $\sum_{u \in U} \sum_{e \in \partial(u)} f_e w_u$ in the vertex-weighted variant and maximize $\sum_{e \in E} f_e w_e$ in the edge-weighted variant.

Constraint 2.2 is the matching constraint for vertices in $U$. Constraint 2.3 is valid because each vertex in $V$ has an arrival rate of 1. Constraint 2.4 is used in [18] and [11]. It captures the fact that the expected number of matches for any edge is at most $1 - 1/e$. This is valid for large $n$ because the probability that a given vertex doesn't arrive after $n$ rounds is $1/e$. Constraint 2.5 is similar to the previous one, but for pairs of edges. For any two neighbors of a given $u \in U$, the probability that neither of them arrive is $1/e^2$. Therefore, the sum of variables for any two distinct edges in $\partial(u)$ cannot exceed $1 - 1/e^2$. Notice that constraints 2.4 and 2.5 reduces the gap between the optimal LP solution and the performance

[1] The edge realization process is independent from one another. At each step, the algorithm "probes" the edge. With probability $p_e$ the edge exists and with remaining probability it doesn't. Once realization of an edge is determined, it doesn't change for the rest of the algorithm

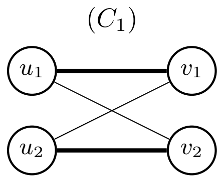


**Figure 1** This cycle is the source of the negative result described by Jaillet and Lu [12]. Thick edges have $f_e = 2/3$ while thin edges have $f_e = 1/3$.

of the optimal online algorithm. In fact, without constraint 2.4, we cannot in general achieve a competitive ratio better than $1 - 1/e$.

## 2.2 Overview of vertex-weighted algorithm and contributions

A key challenge encountered by [12] was that their special LP could lead to length four cycles of type $C_1$ shown in Figure 1. In fact, they used this cycle to show that no algorithm could perform better than $1 - 2/e^2 \approx 0.7293$ using their LP. They mentioned that tighter LP constraints such as 2.4 and 2.5 in the LP from Section 2 could avoid this bottleneck, but they did not propose a technique to use them. Note that the $\{0, 1/3, 2/3\}$ solution produced by their LP was an essential component of their Random List algorithm.

We show a randomized rounding algorithm to construct a similar, simplified $\{0, 1/3, 2/3\}$ vector from the solution of a stricter benchmark LP. This allows for the inclusion of additional constraints, most importantly constraint 2.5. Using this rounding algorithm combined with tighter constraints, we will upper bound the probability of a vertex appearing in the cycle $C_1$ from Figure 1 at $2-3/e \approx 0.89$. (See Lemma 6) Additionally, we show how to deterministically break all other length four cycles which are not of type $C_1$ without creating any new cycles of type $C_1$. Finally, we describe an algorithm which utilizes these techniques to improve previous results in both the vertex-weighted and unweighted settings.

For this algorithm, we first solve the LP in Section 2 on the input graph. In Section 4, we show how to use the technique in sub-section 2.6 to obtain a sparse fractional vector. We then present a randomized online algorithm (similar to the one in [12]) which uses the sparse fractional vector as a guide to achieve a competitive ratio of 0.7299. Previously, there was gap between the best unweighted algorithm with a ratio of $1 - 2e^{-2}$ due to [12] and the negative result of $1 - e^{-2}$ due to [18]. We take a step towards closing that gap by showing that an algorithm can achieve $0.7299 > 1 - 2e^{-2}$ for both the unweighted and vertex-weighted variants with integral arrival rates.

## 2.3 Overview of edge-weighted algorithm and contributions

A challenge that arises in applying the *power of two choices* to this setting is when the same edge $(u, v)$ is included in both matchings $M_1$ and $M_2$. In this case, the copy of $(u, v)$ in $M_2$ can offer no benefit and a second arrival of $v$ is wasted. To use an example from related work, Haeupler *et al.* [11] choose two matchings in the following way. $M_1$ is attained by solving an LP with constraints 2.2, 2.3 and 2.4 and rounding to an integral solution. $M_2$ is constructed by finding a maximum weight matching and removing any edges which have already been included in $M_1$. A key element of their proof is showing that the probability of an edge being removed from $M_2$ is at most $1 - 1/e \approx 0.63$.

The approach in this paper is to construct two or three matchings together in a correlated manner to reduce the probability that some edge is included in all matchings. We will show a general technique to construct an ordered set of $k$ matchings where $k$ is an easily adjustable parameter. For $k = 2$, we show that the probability of an edge appearing in both $M_1$ and $M_2$ is at most $1 - 2/e \approx 0.26$.

For the algorithms presented, we first solve an LP on the input graph. We then round the LP solution vector to a sparse integral vector and use this vector to construct a randomly ordered set of matchings which will guide our algorithm during the online phase. We begin Section 3 with a simple warm-up algorithm which uses a set of two matchings as a guide to achieve a 0.688 competitive ratio, improving the best known result for this problem. We follow it up with a slight variation that improves the ratio to 0.7 and a more complex 0.705-competitive algorithm which relies on a convex combination of a 3-matching algorithm and a separate *pseudo-matching* algorithm.

## 2.4 Overview of non-integral arrival rates with stochastic rewards contributions

This algorithm is presented in Section 5. We believe the known I.I.D. model with stochastic rewards is an interesting new direction motivated by the work of [20] and [21] in the adversarial model. We introduce a new, more general LP specifically for this setting and show that a simple algorithm using the LP solution directly can achieve a competitive ratio of $1 - 1/e$. In [21], it is shown that no randomized algorithm can achieve a ratio better than 0.62 $< 1 - 1/e$ in the adversarial model. Hence, achieving a $1 - 1/e$ for the i.i.d. model shows that this lower bound does not extend to this model.

In Section 6, we extend this simple algorithm[2] to the $b$-matching generalization of this problem where each offline vertex $u$ can match with up to $b$ arriving vertices. We show that our algorithm achieves a competitive ratio of at least $1 - b^{-1/2+\epsilon} - O(e^{-b^{2\epsilon}/3})$ for any given $\epsilon > 0$. Note that this result makes progress on Open Question 14 in the online matching and ad allocation survey [19] which asks about stochastic rewards in non-adversarial models.

## 2.5 Summary of our contributions

► **Theorem 1.** *For vertex-weighted online stochastic matching with integral arrival rates, online algorithm* VW *achieves a competitive ratio of at least* 0.7299*.*

► **Theorem 2.** *For edge-weighted online stochastic matching with integral arrival rates, there exists an algorithm which achieves a competitive ratio of at least* 0.7 *and algorithm* EW$[q]$ *with* $q = 0.149251$ *achieves a competitive ratio of at least* 0.70546*.*

► **Theorem 3.** *For edge-weighted online stochastic matching with arbitrary arrival rates and stochastic rewards, online algorithm* SM (4) *achieves a competitive ratio of* $1 - 1/e$*.*

► **Theorem 4.** *For edge-weighted online stochastic $b$-matching with arbitrary arrival rates and stochastic rewards, online algorithm* $\mathsf{SM_b}$ (5) *achieves a competitive ratio of at least* $1 - b^{-1/2+\epsilon} - O(e^{-b^{2\epsilon}/3})$ *for any given* $\epsilon > 0$*.*

[2] Recently, we have come to know that the result in Section 6 can be obtained as a special case of [3]. Our approach gives an alternative, and a simpler algorithm for this special case.

**Table 1** Summary of Contributions.

| Problem | Previous Work | This Paper |
|---|---|---|
| Edge-Weighted (Section 3) | 0.667 [11] | 0.705 |
| Vertex-Weighted (Section 4) | 0.725 [12] | 0.7299 |
| Unweighted | 0.7293 [12] | 0.7299 |
| Non-integral Stochastic Rewards (Section 5) | N/A | $1 - e^{-1}$ |
| $b$-matching, Stochastic Rewards (Section 6) | N/A | $1 - b^{-1/2+\epsilon} - O(e^{-b^{2\epsilon}/3})$ |

## 2.6 LP rounding technique

For the algorithms presented, we will first solve the benchmark LP in sub-section 2.1 for the input instance to get a fractional solution vector $\mathbf{f}$. We then round $\mathbf{f}$ to an integral solution $\mathbf{F}$ using a two step process we call $\mathsf{DR}[\mathbf{f}, k]$. The first step is to multiply $\mathbf{f}$ by $k$. The second step is to apply the dependent rounding techniques of Gandhi, Khuller, Parthasarathy and Srinivasan [10] to this new vector. In this paper, we will always choose $k$ to be 2 or 3. This will help us handle the fact that a vertex in $V$ may appear more than once, but probably not more than two or three times.

While dependent rounding is typically applied to values between 0 and 1, the useful properties extend naturally to our case in which $kf_e$ may be greater than 1 for some edge $e$. To understand this process, it is easiest to imagine splitting each $kf_e$ into two edges with the integer value $f'_e = \lfloor kf_e \rfloor$ and fractional value $f''_e = kf_e - \lfloor kf_e \rfloor$. The former will remain unchanged by the dependent rounding since it is already an integer while the latter will be rounded to 1 with probability $f''_e$ and 0 otherwise. Our final value $F_e$ would be the sum of those two rounded values. The two properties of dependent rounding we will use are:

1. **Marginal distribution:** For every edge $e$, let $p_e = kf_e - \lfloor kf_e \rfloor$. Then, $\Pr[F_e = \lceil kf_e \rceil] = p_e$ and $\Pr[F_e = \lfloor kf_e \rfloor] = 1 - p_e$.
2. **Degree-preservation:** For any vertex $w \in U \cup V$, let its fractional degree $kf_w$ be $\sum_{e \in \partial(w)} kf_e$ and integral degree be the random variable $F_w = \sum_{e \in \partial(w)} F_e$. Then $F_w \in \{\lfloor kf_w \rfloor, \lceil kf_w \rceil\}$.

# 3 Edge-weighted matching with integral arrival rates

## 3.1 A simple 0.688-competitive algorithm

As a warm-up, we will describe a simple algorithm which achieves a competitive ratio of 0.688 and introduces key ideas in our approach. We begin by solving the LP in sub-section 2.1 to get a fractional solution vector $\mathbf{f}$ and applying $\mathsf{DR}[\mathbf{f}, 2]$ as described in Subsection 2.6 to get an integral vector $\mathbf{F}$. We construct a bipartite graph $G_{\mathbf{F}}$ with $F_e$ copies of each edge $e$. Note that $G_{\mathbf{F}}$ will have max degree 2 since for all $w \in U \cup V$, $F_w \leq \lceil 2f_w \rceil \leq 2$ and therefore we can decompose it into two matchings using *Hall's Theorem*. Finally, we randomly permute the two matchings into an ordered pair of matchings, $[M_1, M_2]$. These matchings serve as a guide for the online phase of the algorithm, similar to [11].

The entire warm-up algorithm for the edge-weighted model, denoted by $\mathsf{EW}_0$, is summarized in Algorithm 1.

**Algorithm 1:** [$\mathsf{EW}_0$]

1. Construct and solve the benchmark LP in sub-section 2.1 for the input instance.
2. Let $\mathbf{f}$ be an optimal fraction solution vector. Call $\mathsf{DR}[\mathbf{f}, 2]$ to get an integral vector $\mathbf{F}$.
3. Create the graph $G_{\mathbf{F}}$ with $F_e$ copies of each edge $e \in E$ and decompose it into two matchings.
4. Randomly permute the matchings to get a *random ordered* pair of matchings, say $[M_1, M_2]$.
5. When a vertex $v$ arrives for the first time, try to assign $v$ to some $u_1$ if $(u_1, v) \in M_1$; when $v$ arrives for the second time, try to assign $v$ to some $u_2$ if $(u_2, v) \in M_2$.
6. When a vertex $v$ arrives for the third time or more, do nothing in that step.

### 3.1.1 Analysis of algorithm $\mathsf{EW}_0$

We will show that $\mathsf{EW}_0$ (Algorithm 1) achieves a competitive ratio of 0.688. Let $[M_1, M_2]$ be our randomly ordered pair of matchings. Note that there might exist some edge $e$ which appears in both matchings if $f_e > 1/2$. Therefore, we consider three types of edges. We say an edge $e$ is of type $\psi_1$, denoted by $e \in \psi_1$, iff $e$ appears *only* in $M_1$. Similarly $e \in \psi_2$, iff $e$ appears *only* in $M_2$ and $e \in \psi_b$, iff $e$ appears in *both* $M_1$ and $M_2$.

Let $P_1, P_2, P_b$ be the probabilities of getting matched for $e \in \psi_1$, $e \in \psi_2$, and $e \in \psi_b$ respectively. According to the result in Haeupler *et al.* [11], the respective values are shown as follows.

▶ **Lemma 5.** *(Proof details in Section 3 of [11]) Given $M_1$ and $M_2$, in the worst case* (1) $P_1 = 0.5808$*;* (2) $P_2 = 0.14849$ *and* (3) $P_b = 0.632$*.*

**Proof.** (Analysis for $\mathsf{EW}_0$) Consider following two cases.

**Case 1:** $0 \le f_e \le 1/2$: By the marginal distribution property of dependent rounding, there can be at most one copy of $e$ in $G_{\mathbf{F}}$ and the probability of including $e$ in $G_{\mathbf{F}}$ is $2f_e$. Since an edge in $G_{\mathbf{F}}$ can appear in either $M_1$ or $M_2$ with equal probability $1/2$, we have $\Pr[e \in \psi_1] = \Pr[e \in \psi_2] = f_e$. Thus, the ratio is $(f_e P_1 + f_e P_2)/f_e = P_1 + P_2 = 0.729$.

**Case 2:** $1/2 \le f_e \le 1 - 1/e$: Similarly, by marginal distribution, $\Pr[e \in \psi_b] = \Pr[F_e = \lceil 2f_e \rceil] = 2f_e - \lfloor 2f_e \rfloor = 2f_e - 1$. It follows that $\Pr[e \in \psi_1] = \Pr[e \in \psi_2] = (1/2)(1 - (2f_e - 1)) = 1 - f_e$. Thus, the ratio is $((1 - f_e)(P_1 + P_2) + (2f_e - 1)P_b)/f_e \ge 0.688$, where the $\mathsf{WS}$ is for an edge $e$ with $f_e = 1 - 1/e$. ◀

## 3.2 A 0.7-competitive algorithm

In this section, we describe an improvement upon the previous warm-up algorithm to get a competitive ratio of 0.7. We start by making an observation about the performance of the warm-up algorithm. After solving the LP, let edges with $f_e > 1/2$ be called *large* and edges with $f_e \le 1/2$ be called *small*. Let $L$ and $S$, be the sets of large and small edges, respectively. Notice that in the previous analysis, small edges achieved a much higher competitive ratio of 0.729 versus 0.688 for large edges. This is primarily due to the fact that we may get two copies of a large edge in $G_{\mathbf{F}}$. In this case, the copy in $M_1$ has a better chance of being matched, since there is no edge which can block it, but the copy that is in $M_2$ has no chance of being matched.

To correct this imbalance, we make an additional modification to the $f_e$ values *before* applying $\mathsf{DR}[\mathbf{f}, k]$. The rest of the algorithm is exactly the same. Let $\eta$ be a parameter to be optimized later. For all large edges $\ell \in L$ such that $f_\ell > 1/2$, we set $f_\ell = f_\ell + \eta$. For all small edges $s \in S$ which are adjacent to some large edge, let $\ell \in L$ be the largest edge adjacent to $s$ such that $f_\ell > 1/2$. Note that it is possible for $e$ to have two large neighbors, but we only care about the largest one. We set $f_s = f_s\left(\frac{1-(f_\ell+\eta)}{1-f_\ell}\right)$.

In other words, we increase the values of large edges while ensuring that for all $w \in U \cup V$, $f_w \leq 1$ by reducing the values of neighboring small edges proportional to their original values. Note that it is not possible for two large edges to be adjacent since they must both have $f_e > 1/2$. For all other small edges which are not adjacent to any large edges, we leave their values unchanged. We then apply $\mathsf{DR}[\mathbf{f}, 2]$ to this new vector, multiplying by 2 and applying dependent rounding as before.

### 3.2.1 Analysis

We can now prove Theorem 2.

**Proof.** As in the warm-up analysis, we'll consider large and small edges separately

- $0 \leq f_s \leq \frac{1}{2}$: Here we have two cases
  - Case 1: $s$ is not adjacent to any large edges. In this case, the analysis is the same as the warm-up algorithm and we still get a 0.729 competitive ratio for these edges.
  - Case 2: $s$ is adjacent to some large edge $\ell$. For this case, let $f_\ell$ be the value of the largest neighboring edge in the original LP solution. Then $s$ achieves a ratio of

$$f_s\left(\frac{1-(f_\ell+\eta)}{1-f_\ell}\right)(0.1484+0.5803)/f_s = \left(\frac{1-(f_\ell+\eta)}{1-f_\ell}\right)(0.1484+0.5803)$$

    Note that for $f_\ell \in [0, 1)$ this is a decreasing function with respect to $f_\ell$. So the worst case is $f_\ell = 1 - 1/e$ and we have a ratio of

$$\left(\frac{1-(1-1/e+\eta)}{1-(1-1/e)}\right)(0.1484+0.5803) = \left(\frac{1/e-\eta}{1/e}\right)(0.1484+0.5803)$$

- $\frac{1}{2} < f_\ell \leq 1 - \frac{1}{e}$: Here, the ratio is $((1-(f_\ell+\eta))(P_1+P_2) + (2(f_\ell+\eta)-1)P_b)/f_\ell$, where the WS is for an edge $e$ with $f_\ell = 1 - 1/e$ since this is a decreasing function with respect to $f_\ell$.

Choosing the optimal value of $\eta = 0.0142$, yields an overall competitive ratio of 0.7 for this new algorithm. ◀

## 3.3 A 0.705-competitive algorithm

The details of algorithm and the proof of Theorem 2 can be found in the full version of this paper.

# 4 Vertex-weighted stochastic I.I.D. matching with integral arrival rates

In this section, we will consider vertex-weighted online stochastic matching on a bipartite graph $G$ under known $I.I.D.$ model with integral arrival rates. We will present an algorithm in which each $u$ has a competitive ratio of at least 0.72998. Recall that after invoking $\mathsf{DR}[\mathbf{f}, 3]$,

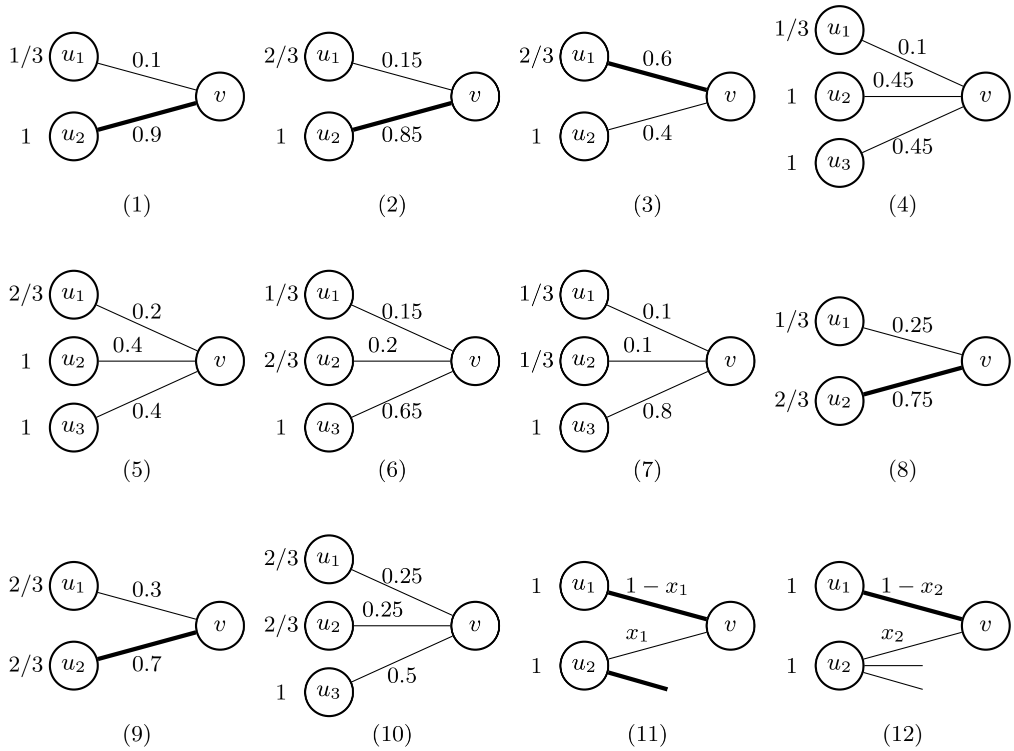


**Figure 2** Illustration for second modification to $\mathbf{H}$. The value assigned to each edge represents the value after the second modification. Here, $x_1 = 0.2744$ and $x_2 = 0.15877$.

we can obtain a (*random*) integral vector $\mathbf{F}$ with $F_e \in \{0, 1, 2\}$. Define $\mathbf{H} = \mathbf{F}/3$ and let $G_{\mathbf{H}}$ be the graph induced by $\mathbf{H}$ and each edge takes the value $H_e \in \{0, 1/3, 2/3\}$.

In this section, we focus on the sparse graph $G_{\mathbf{H}}$. The main steps of the algorithm are:

1. Solve the vertex-weighted benchmark LP in sub-section 2.1. Let $\mathbf{f}$ be an optimal solution vector.
2. Invoke DR$[\mathbf{f}, 3]$ to obtain an integral vector $\mathbf{F}$ and a fractional vector $\mathbf{H}$ with $\mathbf{H} = \mathbf{F}/3$.
3. Apply a series of modifications to $\mathbf{H}$ and transform it to another solution $\mathbf{H}'$. See sub-section 4.1.
4. Run the randomized list algorithm (RLA) [12] induced by $\mathbf{H}'$ on the graph $G_{\mathbf{H}}$. See the details in full version of this paper.

The WS for vertex-weighted case in [12] is shown in Figure 3, which arrives at node $u$ with a competitive ratio of 0.725. From their analysis, we find node $u_1$ has a competitive ratio of at least 0.736. Hence, we *boost* the performance of $u$ at the cost of $u_1$. In other words, we increase the value of $H_{(u,v1)}$ and decrease the value $H_{(u_1,v_1)}$. Case (10) and (11) in Figure 2 illustrates this. After this modification, the new WS for vertex-weighted is now the $C_1$ cycle shown in Figure 1. In fact, this is the WS for the unweighted case in [12]. However, Lemma 6 and the cycle breaking algorithm, implies that $C_1$ cycle can be avoided with probability at least $3/e - 1$. This helps us improve the ratio even for the unweighted case in [12].

▶ **Lemma 6.** *For any given $u \in U$, $u$ appears in a $C_1$ cycle after* DR$[\boldsymbol{f}, 3]$ *with probability at most $2 - 3/e$.*

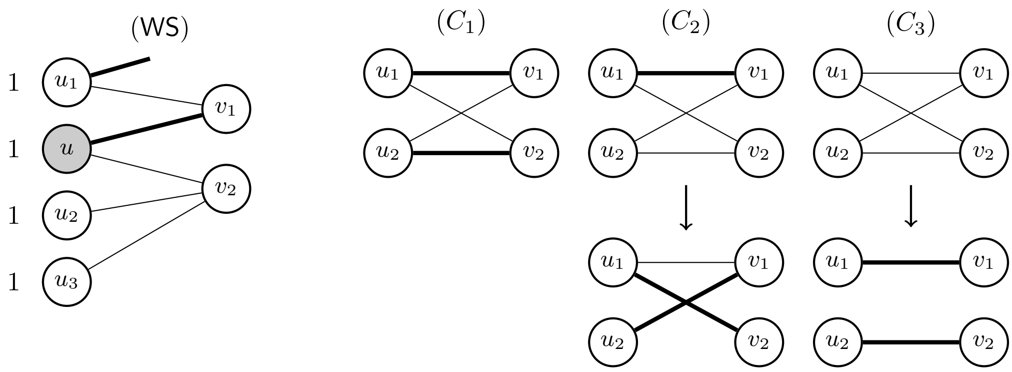


**Figure 3** Left: The WS for Jaillet and Lu [12] for their vertex-weighted case. Right: The three possible types of cycles of length 4 after applying $\mathsf{DR}[\mathbf{f}, 3]$. Thin edges have $H_e = 1/3$ and thick edges have $H_e = 2/3$.

**Proof.** Consider the graph $G_{\mathbf{H}}$ obtained after $\mathsf{DR}[\mathbf{f}, 3]$. Notice that for some vertex $u$ to appear in a $C_1$ cycle, it must have a neighboring edge with $H_e = 2/3$. Now we try to bound the probability of this event. It is easy to see that for some $e \in \partial(u)$ with $f_e \leq 1/3$, $F_e \leq 1$ after $\mathsf{DR}[\mathbf{f}, 3]$, and hence $H_e = F_e/3 \leq 1/3$. Thus only those edges $e \in \partial(u)$ with $f_e > 1/3$ will possibly be rounded to $H_e = 2/3$. Note that, there can be at most two such edges in $\partial(u)$, since $\sum_{e \in \partial(u)} f_e \leq 1$. Hence, we have the following two cases.

**Case 1:** $\partial(u)$ contains only one edge $e$ with $f_e > 1/3$. Let $q_1 = \Pr[H_e = 1/3]$ and $q_2 = \Pr[H_e = 2/3]$ after $\mathsf{DR}[\mathbf{f}, 3]$. By $\mathsf{DR}[\mathbf{f}, 3]$, we know that $\mathbb{E}[H_e] = \mathbb{E}[F_e]/3 = q_2(2/3) + q_1(1/3) = f_e$.
Notice that $q_1 + q_2 = 1$ and hence $q_2 = 3f_e - 1$. Since this is an increasing function of $f_e$ and $f_e \leq 1 - 1/e$ from LP constraint 2.4, we have $q_2 \leq 3(1 - 1/e) - 1 = 2 - 3/e$.

**Case 2:** $\partial(u)$ contains two edges $e_1$ and $e_2$ with $f_{e_1} > 1/3$ and $f_{e_2} > 1/3$. Let $q_2$ be the probability that after $\mathsf{DR}[\mathbf{f}, 3]$, either $H_{e_1} = 2/3$ or $H_{e_2} = 2/3$. Note that, these two events are mutually exclusive since $H_u \leq 1$. Using the analysis from case 1, it follows that $q_2 = (3f_{e_1} - 1) + (3f_{e_2} - 1) = 3(f_{e_1} + f_{e_2}) - 2$.
From LP constraint 2.5, we know that $f_{e_1} + f_{e_2} \leq 1 - 1/e^2$, and hence $q_2 \leq 3(1 - 1/e^2) - 2 < 2 - 3/e$. ◀

## 4.1 Two kinds of Modifications to H

The first modification is to break the cycles deterministically.

There are three possible cycles of length 4 in the graph $G_{\mathbf{H}}$, denoted $C_1$, $C_2$, and $C_3$. In [12], they give an efficient way to break $C_2$ and $C_3$, as shown in Figure 3. Cycle $C_1$ cannot be modified further and hence, is the bottleneck for their unweighted case. Notice that, while breaking the cycles of $C_2$ and $C_3$, new cycles of $C_1$ can be created in the graph. Since our randomized construction of solution $\mathbf{H}$ gives us control on the probability of cycles $C_1$ occurring, we would like to break $C_2$ and $C_3$ in a controlled way, so as to not create any new $C_1$ cycles. This procedure is summarized in Algorithm 2. The proof of Lemma 7 can be found in the full version of this paper.

▶ **Lemma 7.** *After applying Algorithm 2 to $G_{\mathbf{H}}$, we have (1) the value $H_w$ is preserved for each $w \in U \cup V$; (2) no cycle of type $C_2$ or $C_3$ exists; (3) no new cycle of type $C_1$ is added.*

**Algorithm 2:** [Cycle breaking algorithm] Offline Phase

1 While there is some cycle of type $C_2$ or $C_3$, Do:
2 Break all cycles of type $C_2$.
3 Break one cycle of type $C_3$ and return to the first step.

The second modification is to decrease the rates of lists associated with those nodes $u$ with $H_u = 1/3$ or $H_u = 2/3$ and increase the rates of lists associated with nodes $u$ with $H_u = 1$. All details can be found in the full version. Let $\mathbf{H}'$ be the solution vector obtained by applying two kinds of modifications to $\mathbf{H}$. The algorithm for the vertex-weighted case, denoted by VW, is summarized below. The detailed analysis can be found in the full version of this paper.

**Algorithm 3:** VW [Vertex Weighted]

1 Construct and solve the LP in sub-section 2.1 for the input instance.
2 Invoke DR[$\mathbf{f}$, 3] to output $\mathbf{F}$ and $\mathbf{H}$. Apply the two kinds of modifications to morph $\mathbf{H}$ to $\mathbf{H}'$.
3 Run RLA[$\mathbf{H}'$] on the graph $G_{\mathbf{H}}$.

## 5 Non-integral arrival rates with stochastic rewards

The setting here is strictly generalized over the previous sections in the following ways. Firstly, it allows an arbitrary arrival rate (say $r_v$) which can be fractional for each stochastic vertex $v$. Notice that, $\sum_v r_v = n$ where $n$ is the total number of rounds.

Secondly, each $e = (v, u) \in E$ is associated with a value $p_e$, which indicates the probability that edge $e = (u, v)$ is present when we assign $v$ to $u$. We assume this process is independent of the stochastic arrival of each $v$. We will show that the simple non-adaptive algorithm introduced in [11] can be extended to this general case. This achieves a competitive ratio of $(1 - \frac{1}{e})$. Note that Manshadi *et al.* [18] show that no non-adaptive algorithm can possibly achieve a ratio better than $(1 - 1/e)$ for the non-integral arrival rates, even for the case of all $p_e = 1$. Thus, our algorithm is an optimal non-adaptive algorithm for this model.

$$\max \quad \sum_{e \in E} w_e f_e p_e : \tag{5.1}$$

$$\text{s.t.} \quad \sum_{e \in \partial(u)} f_e p_e \leq 1, \forall u \in U \tag{5.2}$$

$$\sum_{e \in \partial(v)} f_e \leq r_v, \forall v \in V \tag{5.3}$$

We use a similar LP as [12] for the case of non-integral arrival rates. For each $e \in E$, let $f_e$ be the probability that $e$ gets matched in the offline optimal algorithm.

Our algorithm is summarized in Algorithm 4. Notice that the last constraint ensures that step 2 in the algorithm is valid. Let us now prove theorem 3.

**Proof.** Let $B(u, t)$ be the event that $u$ is safe at beginning of round $t$ and $A(u, t)$ to be the event that vertex $u$ is matched during the round $t$ conditioned on $B(u, t)$. From

**Algorithm 4:** SM

1 Construct and solve LP (5.1). WLOG assume $\{f_e | e \in E\}$ is an optimal solution.

2 When a vertex $v$ arrives, assign $v$ to each of its neighbor $u$ with a probability $\frac{f_{(u,v)}}{r_v}$.

the algorithm, we know $\Pr[A(u,t)] \le \sum_{v \sim u} \frac{r_v}{n} \frac{f_{u,v}}{r_v} p_e \le \frac{1}{n}$, which follows by $Pr[B(u,t)] = \Pr\left[\bigwedge_{i=1}^{t-1} (\neg A(u,i))\right] \ge \left(1 - \frac{1}{n}\right)^{t-1}$.

Consider an edge $e = (u,v)$ in the graph. Notice that the probability that $e$ gets matched in SM should be

$$\Pr[e \text{ is matched}] = \sum_{t=1}^{n} \Pr[\text{v arrives at } t \text{ and } B(u,t)\ ] \cdot \frac{f_e p_e}{r_v}$$
$$\ge \sum_{t=1}^{n} \left(1 - \frac{1}{n}\right)^{t-1} \frac{r_v}{n} \frac{f_e p_e}{r_v} \ge \left(1 - \frac{1}{e}\right) f_e p_e \,. \qquad \blacktriangleleft$$

# 6 Extension to b-matching with stochastic rewards

In this section, we further generalize the model in Section 5 to the case where each $u$ in the offline set $U$ has a uniform integral capacity $b$ (i.e., each vertex $u$ can be matched at most $b$ times). Otherwise, we retain the same setting as Section 5; we allow non-integral arrival rates and stochastic rewards. We will generalize the simple algorithm used in the previous setting (i.e., Section 5) to this new setting. Consider the following updated LP:

$$\max \quad \sum_{e \in E} w_e f_e p_e : \tag{6.1}$$
$$\text{s.t.} \quad \sum_{e \in \partial(u)} f_e p_e \le b, \forall u \in U \tag{6.2}$$
$$\sum_{e \in \partial(v)} f_e \le r_v, \forall v \in V \tag{6.3}$$

We modify Algorithm 4 for the $b$-matching problem as shown in Algorithm 5. Let us now prove Theorem 4.

**Algorithm 5:** $\mathsf{SM_b}$

1 Construct and solve LP (6.1). WLOG assume $\{f_e | e \in E\}$ is an optimal solution.

2 When a vertex $v$ arrives, assign $v$ to each of its neighbor $u$ with a probability $\frac{f_{(u,v)}}{r_v}$.

**Proof.** The proof is similar to that of Theorem 3. Let $A_t$ be the number of times $u$ has been matched at the beginning of round $t$.

Let $B(u,t)$ be the event that $u$ is safe at the beginning of round $t$, which is defined as $A_t \le b - 1$. For any given edge $e$, let $X_e$ be the number of times that $e$ gets matched over the $n$ rounds. Thus we have

$$\mathbb{E}[X_e] = \sum_{t=1}^{n} \Pr[B(u,t)] \frac{r_v}{n} \frac{f_e}{r_v} p_e = \frac{f_e p_e}{n} \sum_{t=1}^{n} \Pr[A_t \le b - 1] \,.$$

Now we upper bound the value of $\Pr[A_t \geq b]$. For each $1 \leq i \leq t$, let $Z_i$ be the indicator random variable for $u$ to be matched during round $i$. Thus $A_{t+1} = \sum_{i=1}^{t} Z_i$. Notice that for each $i$, we have

$$\mathbb{E}[Z_i] \leq \sum_{v \sim u} \frac{r_v}{n} \frac{f_{(u,v)}}{r_v} p_{(u,v)} \leq \frac{b}{n}.$$

It follows that for any $t \leq n(1-\tau)$ with $0 < \tau < 1$, we have $\mathbb{E}[A_{t+1}] \leq (1-\tau)b$. By applying Chernoff-Hoeffding bounds, we get $\Pr[A_{t+1} \geq b] \leq e^{-b\tau^2/3}$. Therefore

$$\begin{aligned}\mathbb{E}[X_e] &= \frac{f_e p_e}{n} \sum_{t=1}^{n} \Pr[A_t \leq b-1] \\ &\geq \frac{f_e p_e}{n} \sum_{t=1}^{n(1-\tau)} (1 - e^{-b\tau^2/3}) = f_e p_e (1-\tau)(1 - e^{-b\tau^2/3})\end{aligned}$$

For any given $\epsilon > 0$, choose $\tau = b^{-1/2+\epsilon}$ to get a competitive ratio of $1 - b^{-1/2+\epsilon} - O(e^{-b^{2\epsilon}/3})$. ◀

## 7 Conclusion and Future Directions

In this paper, we gave improved algorithms for the Edge-Weighted and Vertex-Weighted models. Previously, there was a gap between the best unweighted algorithm with a ratio of $1 - 2e^{-2}$ due to [12] and the negative result of $1 - e^{-2}$ due to [18]. We took a step towards closing that gap by showing that an algorithm can achieve $0.7299 > 1 - 2e^{-2}$ for both the unweighted and vertex-weighted variants with integral arrival rates. In doing so, we made progess on Open Questions 3 and 4 in the online matching and ad allocation survey [19]. This was possible because our approach of rounding to a simpler fractional solution allowed us to employ a stricter LP. For the edge-weighted variant, we showed that one can significantly improve the power of two choices approach by generating two matchings from the same LP solution. For the variant with edge weights, non-integral arrival rates, and stochastic rewards, we presented a $(1 - 1/e)$-competitive algorithm. This showed that the $0.62 < 1 - 1/e$ bound given in [21] for the adversarial model with stochastic rewards does not extend to the known I.I.D. model. Furthermore, we considered the online edge-weighted $b$-matching problem with stochastic rewards under the known IID setting. We gave a very simple non-adaptive algorithm which achieves a ratio of $1 - b^{-1/2+\epsilon} - O(e^{-b^{2\epsilon}/3})$ for any given $\epsilon > 0$.

A natural next step in the edge-weighted setting is to use an *adaptive* strategy. For the vertex-weighted problem, one can easily see that the stricter LP we use still has a gap. In addition, we only utilize fractional solutions $\{0, 1/3, 2/3\}$. However, dependent rounding gives solutions in $\{0, 1/k, 2/k, \ldots, \lceil k(1-1/e) \rceil / k\}$; allowing for random lists of length greater than three. Stricter LPs and longer lists could both yield improved results. In the stochastic rewards model with non-integral arrival rates, an open question is to either improve upon the $\left(1 - \frac{1}{e}\right)$ ratio or consider a simpler model with integral arrival rates and improve the ratio for this restricted model. Lastly, there is a gap between our result for $b$-matching with stochastic rewards and the results of [7] and [2] for similar problems with deterministic rewards. It would be nice to see a result for this problem that is $1 - O(k^{-1/2})$.

**Acknowledgements.** The authors would like to thank Aranyak Mehta and the anonymous reviewers for their valuable comments, which have significantly helped improve the presentation of this paper.

## References


1. 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*, pages 1253–1264. SIAM, 2011.
2. Saeed Alaei, MohammadTaghi Hajiaghayi, and Vahid Liaghat. Online prophet-inequality matching with applications to ad allocation. In *Proceedings of the 13th ACM Conference on Electronic Commerce*, pages 18–35. ACM, 2012.
3. Saeed Alaei, MohammadTaghi Hajiaghayi, and Vahid Liaghat. The online stochastic generalized assignment problem. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques: 16th International Workshop, APPROX, and 17th International Workshop, RANDOM*, pages 11–25. Springer Berlin Heidelberg, 2013.
4. Bahman Bahmani and Michael Kapralov. Improved bounds for online stochastic matching. In *European Symposium on Algorithms (ESA)*, pages 170–181. Springer, 2010.
5. Nikhil R Devanur and Thomas P Hayes. The adwords problem: online keyword matching with budgeted bidders under random permutations. In *Proceedings of the 10th ACM conference on Electronic commerce*, pages 71–78. ACM, 2009.
6. Nikhil R. Devanur, Kamal Jain, Balasubramanian Sivan, and Christopher A. Wilkens. Near optimal online algorithms and fast approximation algorithms for resource allocation problems. In *Proceedings of the 12th ACM Conference on Electronic Commerce*, pages 29–38. ACM, 2011.
7. Nikhil R Devanur, Balasubramanian Sivan, and Yossi Azar. Asymptotically optimal algorithm for stochastic adwords. In *Proceedings of the 13th ACM Conference on Electronic Commerce*, pages 388–404. ACM, 2012.
8. Jon Feldman, Nitish Korula, Vahab Mirrokni, S Muthukrishnan, and Martin Pál. Online ad assignment with free disposal. In *Internet and network economics*, pages 374–385. Springer, 2009.
9. Jon Feldman, Aranyak Mehta, Vahab Mirrokni, and S Muthukrishnan. Online stochastic matching: Beating 1-1/e. In *Foundations of Computer Science (FOCS)*, pages 117–126. IEEE, 2009.
10. Rajiv Gandhi, Samir Khuller, Srinivasan Parthasarathy, and Aravind Srinivasan. Dependent rounding and its applications to approximation algorithms. *Journal of the ACM (JACM)*, 53(3):324–360, 2006.
11. Bernhard Haeupler, Vahab S. Mirrokni, and Morteza Zadimoghaddam. Online stochastic weighted matching: Improved approximation algorithms. In *Internet and Network Economics*, volume 7090 of *Lecture Notes in Computer Science*, pages 170–181. Springer Berlin Heidelberg, 2011.
12. Patrick Jaillet and Xin Lu. Online stochastic matching: New algorithms with better bounds. *Mathematics of Operations Research*, 39(3):624–646, 2013.
13. Bala Kalyanasundaram and Kirk R Pruhs. An optimal deterministic algorithm for online b-matching. *Theoretical Computer Science*, 233(1):319–325, 2000.
14. 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*, pages 352–358. ACM, 1990.
15. Thomas Kesselheim, Klaus Radke, Andreas Tönnis, and Berthold Vöcking. An optimal online algorithm for weighted bipartite matching and extensions to combinatorial auctions. In *European Symposium on Algorithms (ESA)*, pages 589–600. Springer, 2013.
16. Nitish Korula and Martin Pál. Algorithms for secretary problems on graphs and hypergraphs. In *Automata, Languages and Programming*, pages 508–520. Springer, 2009.

**17** Mohammad Mahdian and Qiqi Yan. Online bipartite matching with random arrivals: an approach based on strongly factor-revealing LPs. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 597–606. ACM, 2011.

**18** Vahideh H Manshadi, Shayan Oveis Gharan, and Amin Saberi. Online stochastic matching: Online actions based on offline statistics. *Mathematics of Operations Research*, 37(4):559–573, 2012.

**19** Aranyak Mehta. Online matching and ad allocation. *Foundations and Trends in Theoretical Computer Science*, 8(4):265–368, 2012.

**20** Aranyak Mehta and Debmalya Panigrahi. Online matching with stochastic rewards. In *Foundations of Computer Science (FOCS)*, pages 728–737. IEEE, 2012.

**21** Aranyak Mehta, Bo Waggoner, and Morteza Zadimoghaddam. Online stochastic matching with unequal probabilities. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 2015.