Algorithmica
https://doi.org/10.1007/s00453-020-00698-3

Online Stochastic Matching: New Algorithms and Bounds

Brian Brubach[1] · Karthik Abinav Sankararaman[1] · Aravind Srinivasan[1] · Pan Xu[2]

Received: 19 June 2018 / Accepted: 10 March 2020

Abstract
Online matching has received significant attention in recent years due to its close connection to Internet advertising. As the seminal work of Karp, Vazirani, and Vazirani has an optimal $(1 - 1/\mathrm{e})$ 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 (*WINE*, 2011) to 0.705; and (b) the vertex-weighted case, where we improve the 0.7250 ratio of Jaillet and Lu (Math Oper Res 39(3):624–646, 2013) to 0.7299. We also consider an extension of stochastic rewards, a variant where each edge has an independent probability of being present. For the setting of stochastic rewards with non-integral arrival rates, we present a simple *optimal* non-adaptive algorithm with a ratio of $1 - 1/\mathrm{e}$. For the special case where each edge is unweighted and has a uniform constant probability of being present, we improve upon $1 - 1/\mathrm{e}$ by proposing a strengthened LP benchmark. 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 (2013) and is a tractable structure for algorithm design and analysis. The appropriate random move preserves many of the removed constraints (approximately with high probability and exactly in expectation). This underlies some of our improvements and could be of independent interest.

A preliminary version of this appeared in the European Symposium on Algorithms (ESA), 2016. Supported in part by NSF awards CCF-1422569 and CCF-1749864, and by research awards from Adobe, Amazon, and Google.

Extended author information available on the last page of the article

Published online: 27 April 2020

Keywords Online algorithms · Randomized algorithms · Bipartite matching · Internet advertising

1 Introduction

Applications to Internet advertising have driven the study of online matching problems in recent years [18]. In these problems, we consider a bipartite graph $G = (U, V, E)$ in which the set of vertices U is available *offline* while the set of vertices in V arrive *online*. Whenever some vertex v arrives, it must be matched immediately (and irrevocably) to (at most) one vertex in U. Each offline vertex u can be matched to at most one v. In the context of Internet advertising, U is the set of advertisers and V is the set of impressions. The edges E define the impressions that interest a particular advertiser. When an impression v arrives, we must choose an available advertiser (if any) to match with it. We consider the case where $v \in V$ can be matched at most once upon arriving. 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.

In the *stochastic known I.I.D.* model of arrival, we are given a bipartite graph $G = (U, V, E)$ and a finite online time horizon T (in most cases, we assume $T = |V| = n$ and say the online phase takes place over n rounds). In each round, a vertex v is sampled with replacement from a known distribution over V. The sampling distributions are independent and identical over all of the T online rounds. This captures the fact that we often have historical data about the impressions and can predict the frequency with which each type of impression will arrive. Edge-weighted matching [9] 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. Each arrival of a type $v \in V$ is considered a distinct vertex (user) that can be matched to up to one $u \in U$. For example, if the same v arrives three times, we consider this three separate vertices (or *copies* of v) that can potentially be matched to three different vertices in U. A special case of this model is vertex-weighted matching [1], where weights are associated only with the advertisers (the offline set U). 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 historical data, one can assign the probability of a particular advertisement being clicked by a type of user. Works including [20, 21] capture this notion of *stochastic rewards* by assigning a probability to each edge.

One unifying theme in most of our approaches is the use of an LP benchmark with additional valid constraints that hold for the respective stochastic-arrival models. We use the optimal solution to this LP to guide our online actions. In most cases, we use various modifications of dependent randomized rounding to convert the fractional LP solution into a suitable guide for our online algorithms.

2 Preliminaries and Technical Challenges

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 vertex set V represent the online vertices. Each edge $e \in E$ is associated with a weight w_e. Thus, this represents the *input graph*. 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 Sect. 5, this paper will focus on the integral arrival rates setting where all $r_v \in \mathbb{Z}^+$. For reasons described in [12], we can further assume WLOG that each v has $r_v = 1$ under the assumption of integral arrival rates. In particular, a vertex type v with an integral arrival rate $k > 1$, can be split into k different vertex types each with an arrival rate of 1. In this case, we have that $|V| = n$ where n is the total number of online rounds.

In the **vertex-weighted** variant, every offline vertex $u \in U$ has a weight w_u (alternatively, for any vertex $u \in U$ all edges incident to u have the same weight) and we seek a maximum weight matching. In the **edge-weighted** variant, every edge $e \in E$ has a weight w_e and we again seek a maximum weight matching. In the **stochastic rewards** variant, each edge has a probability p_e of being present once we probe edge e and we seek to maximize the expected size or weight of the matching. The edge realization process is independent for different edges. At each step, the algorithm "probes" an edge e. With probability p_e the edge e exists and with the remaining probability it does not. Once realization of an edge is determined, it does not affect the random realizations for the rest of the edges. We consider the query-commit model where an edge that is probed and found to exist must be matched. For a single arriving vertex, each edge can only be probed once. However, we remind the reader that multiple arrivals of the same vertex *type* are considered distinct vertices. Suppose the first arrival of a vertex type $v \in V$ probes an edge to some offline vertex $u \in U$ and the edge does not exist. Then later, another copy of type v might arrive and also probe an edge to u because each arrival is a distinct vertex with its own hidden edge realizations.

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 "$x \leq 1 - 1/\mathsf{e}^2$" instead of the more-accurate "$x \leq 1 - 1/\mathsf{e}^2 + o(1)$". These suppressed $o(1)$ terms will subtract at most $o(1)$ from our competitive ratios. Note the we use e for Euler's constant in contrast with e which denotes an edge. Throughout, we use "WS" to refer to the worst case instance for various algorithms.

Competitive ratio The *competitive ratio* is defined slightly differently than usual for this set of problems (similar to the notation used in [18]). In particular, it is defined as $\frac{\mathbb{E}[\text{ALG}]}{\mathbb{E}[\text{OPT}]}$. Here, $\mathbb{E}[\text{ALG}]$ is the expected performance of our online algorithm with respect to the random online vertex arrivals and any internal randomness the algorithm may use; and for the stochastic rewards variant the random edge realizations, arrival sequence and internal randomness of the algorithm. Similarly, $\mathbb{E}[\text{OPT}]$ is the expected performance of an optimal offline matching algorithm which knows the random vertex arrivals in advance. In the case of stochastic rewards, we compare

to an optimal offline stochastic matching algorithm which can probe edges in any order, but does not know the outcomes of these probes and can only probe one neighbor of each vertex from the "online" partition.

Adaptivity Algorithms can be *adaptive* or *non-adaptive*. When v arrives, an adaptive algorithm can modify its online actions based on the realization of the online vertices (and edges in the stochastic rewards model) thus far, but a non-adaptive algorithm has to specify all of its actions before the start of the online phase.

2.1 LP Benchmark for Deterministic Rewards

As in prior work (*e.g,* see [18]), we use the following LP to upper bound the optimal offline expected performance and also use it to guide our algorithm in the cases where rewards are deterministic. For the case of stochastic rewards, we use slightly modified LPs, whose definitions we defer until Sects. 5 and 6. We first show an LP for the unweighted variant, then describe the 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 $f_w = \sum_{e \in \partial(w)} f_e$. Constraint (4) is used in [12, 19].

$$\text{maximize} \quad \sum_{e \in E} f_e \tag{1}$$

$$\text{subject to} \quad \sum_{e \in \partial(u)} f_e \leq 1 \qquad \forall u \in U \tag{2}$$

$$\sum_{e \in \partial(v)} f_e \leq 1 \qquad \forall v \in V \tag{3}$$

$$0 \leq f_e \leq 1 - 1/\mathsf{e} \qquad \forall e \in E \tag{4}$$

$$f_e + f_{e'} \leq 1 - 1/\mathsf{e}^2 \qquad \forall e, e' \in \partial(u), \forall u \in U \tag{5}$$

Variants The objective function is to 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 (here w_e refers to $w_{(u,v)}$).

Lemma 1 *Let* OPT *denote the total weight obtained by the best offline algorithm. Let* $\mathbf{f}^*$ *denote the optimal solution to the above* LP. *Then* $\sum_{e \in E} f^*_e \geq \mathbb{E}[\text{OPT}]$.

Proof We prove this as follows. Let Y_e denote the indicator random variable for the event that edge $e \in E$ is matched in the optimal solution for a given arrival sequence $\mathcal{A}$. Let $y_e := \mathbb{E}_{\mathcal{A}}[Y_e]$ for every edge $e \in E$. We will now argue that the vector $\mathbf{y} := (y_e)_{e \in E}$ is a feasible solution to the LP. Consider a vertex $u \in U$. We have that $\sum_{e \in \partial(u)} Y_e \leq 1$. Taking expectations on both sides and using the linearity of expectation we have $\sum_{e \in \partial(u)} Y_e \leq 1$. This shows that $\mathbf{y}$ is feasible to the constraint (2). Let R_v

denote the random variable for the number of times a vertex $v \in V$ arrived in a given arrival sequence $\mathcal{A}$. Then we have, for every $v \in V$, $\sum_{e \in \partial(v)} Y_e \leq R_v$. From the integral arrival rates assumption, $\mathbb{E}_{\mathcal{A}}[R_v] = 1$ for every $v \in V$. Thus, from linearity of expectation we obtain $\sum_{e \in \partial(v)} Y_e \leq 1$. This shows that $\mathbf{y}$ is feasible to the constraint (3). For any edge $e = (u, v)$, let $\mathbb{I}[R_v = 0]$ be an indicator for the event that a vertex $v \in V$ never arrives in the T rounds. Thus, for any arrival sequence $\mathcal{A}$, we have $Y_e \leq \mathbb{I}[R_v \neq 0]$. Taking expectations on both sides we get $Y_e \leq \mathbb{E}_{\mathcal{A}}[\mathbb{I}[R_v \neq 0]$. The probability that a vertex v never arrives in T rounds is $\left(1 - \frac{1}{T}\right)^T \leq 1/\mathrm{e}$. Thus, $\mathbb{E}_{\mathcal{A}}[\mathbb{I}[R_v \neq 0] \leq 1 - 1/\mathrm{e}$. This shows that $\mathbf{y}$ is feasible to the constraint (4). Consider two edges $e, e' \in \partial(u)$ for some $u \in U$. Let $e = (u, v)$ and $e' = (u, v')$ and as before let $\mathbb{I}[R_v \neq 0]$ and $\mathbb{I}[R(v') \neq 0]$ denote the indicator for the events that v, v' arrives at least once in the T rounds, respectively. For any arrival sequence $\mathcal{A}$ we have that $Y_e + Y(e') \leq \mathbb{I}[R_v \neq 0] \wedge \mathbb{I}[R(v') \neq 0]$. Taking expectations on both sides we get $Y_e + y(e') \leq \mathbb{E}_{\mathcal{A}}[\mathbb{I}[R_v \neq 0] \wedge \mathbb{I}[R(v') \neq 0]]$. The probability that both v and v' never arrive in the T rounds is given by $\left(1 - \frac{2}{T}\right)^T \leq \frac{1}{\mathrm{e}^2}$. Thus, we get $Y_e + y(e') \leq 1 - \frac{1}{\mathrm{e}^2}$ which shows that $\mathbf{y}$ is feasible to the constraint 5.

The expected weight of the optimal solution is $\mathbb{E}_{\mathcal{A}}[\sum_{e \in E} w_e Y_e]$ which from linearity of expectation gives $\sum_{e \in E} w_e Y_e$. Since $\mathbf{y}$ is a feasible solution we have that the optimal value to LP is at least as large as the expected optimal solution. □

We compare the performance of our algorithm to this LP. Suppose that $\mathbf{f}^*$ is the optimal solution to the above LP. We prove the following lemma which shows that it suffices to analyze the competitive ratio *edge-wise*.

Lemma 2 *If* $\min_{e \in E, f_e^* > 0} \frac{\Pr[e \text{ is included in the matching}]}{f_e^*} \geq \alpha$, *then this implies that the competitive ratio is at least* α.

Proof From linearity of expectation we have that

$$\begin{aligned}
\mathbb{E}[\mathrm{ALG}] &= \sum_{e \in E} \Pr[e \text{ is included in the matching}] \\
&\geq \alpha \sum_{e \in E} f_e^* \\
&\geq \alpha \mathbb{E}[\mathrm{OPT}].
\end{aligned}$$

□

In what follows, we only compute a lower-bound on the probability that any edge $e \in E$ is included in the final matching (we call this quantity *competitive ratio of edge* e) which would imply a lower-bound on the competitive ratio.

In the vertex-weighted setting (Sect. 4) we compute a lower-bound on the probability that a vertex $u \in U$ is matched in any randomized online algorithm. Analogous to Lemma 2, the following lemma connects the lower-bound on this probability to the competitive ratio.

Lemma 3 *Define* $F_u := \sum_{e \in \partial(u)} f_e$. If $\min_{u \in U, F_u^* > 0} \frac{\Pr[u \text{ is matched}]}{F_u^*} \geq \alpha$, *then this implies that the competitive ratio is at least* α.

Proof From linearity of expectation we have that

$$\begin{aligned}
\mathbb{E}[\text{ALG}] &= \sum_{u \in U} \Pr[u \text{ is matched}] \\
&\geq \alpha \sum_{u \in U} F_u^* \\
&= \alpha \sum_{u \in U} \sum_{e \in \partial(u)} f_e^* \\
&\geq \alpha \mathbb{E}[\text{OPT}].
\end{aligned}$$

□

Note that the work of [19] does not use an LP to upper-bound the optimal value of the offline instance. Instead they use Monte-Carlo simulations wherein they simulate the arrival sequence and compute the vector $\mathbf{f}$ by approximating (via Monte-Carlo simulation) the probability of matching an edge e in the offline optimal solution. We do not use a similar approach for our problems for a few reasons. (1) For the *weighted* variants, namely the edge and vertex-weighted versions, the number of samples depends on the maximum value of the weight, making it expensive. (2) In the unweighted version, the running time of the sampling based algorithm is $O(|E|^2 n^4)$; on the other hand, we show in Sect. 2.5 that the LP based algorithm can be solved much faster, $\tilde{O}(|E|^2)$ time in the worst case and even faster than that in practice. (3) For the stochastic rewards setting, the offline problem is not known to be polynomial-time solvable, which is required for [19] since they rely on solving instances of the offline problem on simulated graphs. [4] show that under the assumption of constant p *and* $\text{OPT} = \omega(1/p)$, we can obtain a $(1 - \epsilon)$-approximation to the optimal solution. However, these assumptions are too strong to be used in our setting.

For the stochastic-rewards setting, one might be tempted to use an LP to achieve the same property obtained from Monte-Carlo simulation via adding extra constraints. In the context of uniform stochastic rewards where each edge e is associated with a uniform constant probability p, what we really need is:

$$\forall S \subseteq \partial(u), \quad \sum_{e \in S} f_e \leq \frac{1 - \exp(-|S|p)}{p} \tag{6}$$

To guarantee this via the LP, a straightforward approach is to add this family of constraints to the LP. However, the number of such constraints is exponential and there seems to be no *obvious* separation oracle. We overcome this challenge by showing it suffices to ensure that inequality (6) above holds for all S with $|S| \leq 2/p$, which is a constant, thus making the resultant LP polynomial-time solvable.

2.2 Overview of Edge-Weighted Algorithm and Contributions

The previous best result due to [12] for the edge-weighted problem was 0.667. They used two matchings, M_1 and M_2, from the offline graph to guide the online algorithm and leverage the *power of two choices*. When a vertex v arrives for the first time, it can be matched to its neighbor in M_1 and on its second arrival it can be matched to its neighbor in M_2. However, these two matchings may not be edge disjoint, leaving some arriving vertices with only one choice. In fact, choosing two guiding matchings that maximize both the edge weights and the number of disjoint edges is a major challenge that arises in applying the *power of two choices* to this setting.

When the same edge (u, v) is included in both matchings M_1 and M_2, 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. [12] choose two matchings in the following way. M_1 is attained by solving an LP with constraints (2), (3) and (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 $1 - 1/\mathsf{e} \approx 0.63$.

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 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 $1 - 2/\mathsf{e} \approx 0.26$.

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 Sect. 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.

2.3 Overview of Vertex-Weighted Algorithm and Contributions

The previous best results due to [13] for the vertex-weighted and unweighted problems were 0.725 and $1 - 2\mathsf{e}^{-2} \approx 0.729$, respectively. They used a clever LP which guaranteed they could find a solution wherein each edge variable was assigned a value in $\{0, 1/3, 2/3\}$ as opposed to an arbitrary fractional value. This property, which we will call a $\{0, 1/3, 2/3\}$ solution, was required by their adaptive online algorithm. However, their special LP was a slightly weaker upper bound on the optimal solution than the LP we describe in Sect. 2.1.

Another key challenge encountered by [13] was that solutions to their special LP could lead to length-four cycles of type C_1 shown in Fig. 1. In fact, they used this

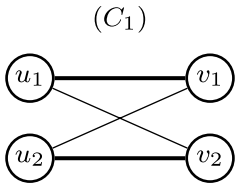

Fig. 1 This cycle is the source of the negative result described by Jaillet and Lu [13]. It results from the edge variable assignments in their special LP. Thick edges have $f_e = 2/3$ while thin edges have $f_e = 1/3$. This structure and variable assignment leads to a gap of $1 - 2\mathsf{e}^{-2}$ between the LP solution and the best possible solution of any online algorithm

case to show that no algorithm could perform better than $1 - 2\mathsf{e}^{-2} \approx 0.7293$ using their LP as an upper bound. They mentioned that tighter LP constraints such as (4) and (5) in the LP from Sect. 2.1 could avoid this bottleneck, but did not propose a technique to use them. Note that the $\{0, 1/3, 2/3\}$ solution produced by their specific LP was an essential component of their *Random List* algorithm.

To address this challenge, we show a randomized rounding algorithm to construct a similar, simplified $\{0, 1/3, 2/3\}$ vector from the solution of a stronger benchmark LP. This allows for the inclusion of additional constraints, most importantly constraint (5). Using our rounding algorithm combined with tighter constraints, we can upper-bound the probability of a vertex appearing in the cycle C_1 from Fig. 1 at $2 - 3/\mathsf{e} \approx 0.89$ (See Lemma 7). By constant, cycles of type C_1 occur deterministically in [13].

Additionally, we note briefly that there are other length four cycles with different variable weights, defined as types C_2 and C_3 (See Fig. 2 in Sect. 4.2). These cycles could be problematic, but we show how to deterministically break them in Sect. 4.2 without creating any new cycles of type C_1 (This can happen if the cycle breaking is not done carefully). Finally, we describe an algorithm which utilizes these techniques to improve previous results in both the vertex-weighted and unweighted settings.

For this problem, we first solve the LP in Sect. 2 on the input graph. In Sect. 4, we show how to use the technique in Sect. 2.6 to obtain a sparse fractional vector. We then present a randomized online algorithm (similar to the one in [13]) which uses the sparse fractional vector as a guide to achieve a competitive ratio of 0.7299.

Previously, there was a gap between the best unweighted algorithm with a ratio of $1 - 2\mathsf{e}^{-2}$ due to [13] and the negative result of $1 - \mathsf{e}^{-2}$ due to [19]. We take a step toward closing this gap by showing that an algorithm can achieve $0.7299 > 1 - 2\mathsf{e}^{-2}$ for both the unweighted and vertex-weighted variants with integral arrival rates. In doing so, we make progess on Open Questions 3 and 4 from the book [18].[1]

[1] Open Questions 3 and 4 state the following: "In general, close the gap between the upper and lower bounds. In some sense, the ratio of $1 - 2\mathsf{e}^{-2}$ achieved in [13] for the integral case, is a nice 'round' number, and one may suspect that it is the correct answer."

Table 1 Summary of contributions

Problem	Previous work	This paper
Edge-weighted (Sect. 3)	0.667 [12]	0.705
Vertex-weighted (Sect. 4)	0.725 [13]	0.7299
Unweighted	$1 - 2/e^2$ [13]	0.7299 ($> 1 - 2/e^2$)
Stochastic rewards (Sect. 5 and 6)	N/A	$1 - e^{-1}$ for general version 0.702 for the restricted version

2.4 Overview of Stochastic Rewards and Contributions

Our algorithm for the more general problem allowing stochastic rewards and non-integral arrival rates (Algorithm 9) is presented in Sects. 5 and 6. 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 (see LP (13)) specifically for this setting and show that a simple algorithm using the LP solution directly can achieve a competitive ratio of $1 - 1/e$. This ratio is optimal among all non-adaptive algorithms for the case of non-integral arrival rates even without stochastic rewards [19]. In [20], it is shown that no randomized algorithm can achieve a ratio better than $0.62 < 1 - 1/e$ in the adversarial model when comparing to a problem called Budgeted Allocation as the offline optimal. While our work instead compares to offline stochastic matching as the offline optimal, the benchmark LP we use in Sect. 5 (LP 13) upper bounds Budgeted Allocation. Hence, achieving $1 - 1/e$ for the I.I.D. model shows that this lower bound does not extend to the I.I.D. model. Further, the paper [5] shows that using LP (13) one cannot achieve a ratio better than $1 - 1/e$. We discuss some challenges relating to why the techniques used in prior work do not directly extend to this model.

To take a step toward addressing these challenges in Sect. 6, we consider a restricted version of the problem where each edge is unweighted and has a uniform constant probability $p \in (0, 1]$ under integral arrival rates. By proposing a family of valid constraints, we are able to show that in this restricted setting, one can indeed beat $1 - 1/e$. We note that this result cannot be compared to the work of [20] since we use a tighter benchmark (LP 17) which does not upper bound Budgeted Allocation. We summarize our contributions in Table 1.

2.5 Running Time of the Algorithms

In this section, we discuss the implementation details of our algorithms. All of our algorithms solve an LP in the pre-processing step. The dimension of the LP is determined by the constraint matrix which consists of $O(|E|^2 + |U| + |V|)$ rows and $O(|E|)$ columns. However, note that the number of non-zero entries in this matrix is of the order $O(|E|^2)$ because each edge is subject to $O(|E|)$ constraints primarily due to LP constraint 5. Some recent work (e.g., [17]) shows that such sparse programs

can be solved in time $\tilde{O}(|E|^2)$ using interior point methods (which are known to perform very well in practice). This sparsity in the LP is the reason we can solve very large instances of the problem. The second critical step in pre-processing is to perform randomized rounding. Note that we have $O(|E|)$ variables and that in each step of the randomized rounding due to [11], they incur a running time of $O(|E|)$. Hence the total running time to obtain a rounded solution is of the order $O(|E|^2)$. Additionally, both of these operations are part of the pre-processing step. In the online phase, the algorithm incurs a per-time-step running time of at most $O(|U|)$ for the stochastic rewards case (in fact, a smarter implementation using binary search runs as fast as $O(\log |U|)$ time) and $O(1)$ for the edge-weighted and vertex-weighted algorithms in Sects. 3 and 4.

2.6 LP Rounding Technique DR[$\mathbf{f}, k$]

For the algorithms presented, we first solve the benchmark LP in Sect. 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 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 [11] to this new vector. In this paper, it suffices to consider $k = 2$ or $k = 3$.

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 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 $F_w \in \{\lfloor kf_w \rfloor, \lceil kf_w \rceil\}$.

2.7 Related Work

The study of online matching began with the seminal work of Karp, Vazirani, Vazirani [16], 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 [18] 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 [9], where they consider an additional relaxation of "free-disposal".

In addition to the adversarial and known I.I.D. models, online matching is also studied under several other variants such as random arrival order, unknown distributions, and known adversarial distributions. In the setting of random arrival order, the arrival sequence is assumed to be a random permutation over all online vertices, see e.g., [6, 14, 15, 22]. In the case of unknown distributions, in each round an item is sampled from a fixed but unknown distribution. If the sampling distributions are required to be the same during each round, it is called unknown I.I.D. ([7, 8]); otherwise, it is called adversarial stochastic input ([7]). As for known adversarial distributions, in each round an item is sampled from a known distribution, which is allowed to change over time ([2, 3]). 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/unweighted settings The vertex-weighted and unweighted settings have many results starting with Feldman, Mehta, Mirrokni and Muthukrishnan [10] who were the first to beat $1 - 1/\mathsf{e}$ with a competitive ratio of 0.67 for the unweighted problem. This was improved by Manshadi, Gharan, and Saberi [19] 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 $1 - \mathsf{e}^{-2} \approx 0.86$. Finally, Jaillet and Lu [13] presented an adaptive algorithm which used a clever LP to achieve 0.725 and $1 - 2\mathsf{e}^{-2} \approx 0.729$ for the vertex-weighted and unweighted problems, respectively.

Related work in the edge-weighted setting For this model, Haeupler, Mirrokni, Zadimoghaddam [12] were the first to beat $1 - 1/\mathsf{e}$ 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.

Other related work Devanur et al. [8] gave an algorithm which achieves a ratio of $1 - k!/(k^k e^k)$ for the Adwords problem[2] 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 $1/k$, where k is some positive integer. Alaei et al. [2] considered 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.

[2] In the Adwords problem, every $u \in U$ has a total budget B_u. Each edge e has a bid b_e which represents the weight. The goal is to maximize the total weight of the edges matched subject to the constraint that for any vertex $u \in U$ the sum of the total weight of the edges matched to u is at most B_u.

3 Edge-Weighted Matching with Integral Arrival Rates

3.1 Warm-up: 0.688-Competitive Algorithm

As a warm-up, we describe a simple algorithm which achieves a competitive ratio of 0.688 and introduces the key ideas in our approach. We begin by solving the LP in Sect. 2.1 to get a fractional solution vector $\mathbf{f}$ and applying $\mathsf{DR}[\mathbf{f}, 2]$ as described in Sect. 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 thus we can decompose it into two matchings using a greedy algorithm and *Hall's Theorem*. The exact choice of the two matchings is not critical to the algorithm as long as the union contains all edges in $G_{\mathbf{F}}$. 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 [12]. The entire warm-up algorithm for the edge-weighted model, denoted by 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 fractional solution vector. Call $\mathsf{DR}[\mathbf{f}, 2]$ to get a random 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 as described in text.
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, attempt to match v to u_1 if $(u_1, v) \in M_1$; when v arrives for the second time, attempt to match v to 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 EW_0

We will show that 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 due to having $f_e > 1/2$, which could be rounded up to $F_e = 1$. Therefore, we consider three types of edges. We say an edge e is of type ψ_1, denoted by $e \in \psi_1$, if and only if e appears *only* in M_1. Similarly $e \in \psi_2$, if and only if e appears *only* in M_2. Finally, $e \in \psi_b$, if and only if e appears in *both* M_1 and M_2. Let P_1, P_2, and 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. [12], Lemma 4 bounds these probabilities.

Lemma 4 (Proof details in section 3 of [12]) *For any two matchings* M_1 and M_2 *steps* (5) *and* (6) *in Algorithm* 1 *implies that we have* (1) $P_1 > 0.5808$; (2) $P_2 > 0.14849$ *and* (3) $P_b > 0.6321$.

We can use Lemma 4 to prove that the warm-up algorithm EW_0 achieves a ratio of 0.688 by examining the probability that a given edge becomes type ψ_1, ψ_2, or ψ_b.

Analysis of EW_0. Consider the 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* $1/2 \le f_e \le 1 - \frac{1}{\mathsf{e}}$: 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 (noting that the first term is from case 1 while the second term is from case 2) $((1 - f_e)(P_1 + P_2) + (2f_e - 1)P_b)/f_e \ge 0.688$, where the WS is for an edge e with $f_e = 1 - \frac{1}{\mathsf{e}}$. □

3.2 Improved Algorithm: 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 (i.e. an edge with the same offline neighbor that gets matched first), 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 η be a parameter to be optimized in the analysis. For all large edges $\ell \in L$ such that $f^*_\ell > 1/2$, we set $\tilde{f}^*_\ell(\ell) = f^*_\ell + \eta$. 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 s to have two large neighbors, but we only care about the larger of the two. We set $\tilde{f}^*_s = f^*_s\left(\frac{1-\tilde{f}^*_\ell}{1-f^*_\ell}\right)$.

In other words, we increase the values of large edges while ensuring that for all $w \in U \cup V$, $f_w \le 1$ 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

Theorem 1 *For edge-weighted online stochastic matching with integral arrival rates,* EW(0.0142) *achieves a competitive ratio of at least* 0.7.

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

- *Scenario 1:* $0 \le f_s^* \le \frac{1}{2}$
  Here we have two cases
  - *Case 1 s* is not adjacent to any large edges.
    In this case, the analysis is the same as Case 1 in the warm-up analysis. Thus, the probability that edge s is added to the matching is $0.729 f_e^*$.
  - *Case 2 s* is adjacent to some large edge ℓ.
    For this case, let f_ℓ^* be the value of the largest neighboring edge in the original LP solution. Then the probability that edge s is added to the matching is

$$f_s^* \left(\frac{1 - (f_\ell^* + \eta)}{1 - f_\ell^*} \right) (0.1484 + 0.5803).$$

    This follows from Lemma 4; in particular, the first two terms are the result of how we set $\tilde{f}_s$ in the algorithm, while the two numbers, 0.1484 and 0.5803, are the probabilities that s is matched when it is in M_2 and M_1, respectively. Note that for $f_\ell^* \in [0, 1)$ this is a decreasing function in f_ℓ^*. So the worst case is when $f_\ell^* = 1 - \frac{1}{\mathsf{e}}$ (due to third constraint in the LP (4)) Thus, the probability that edge s is added to the matching is

$$f_s^* \left(\frac{1 - (1 - \frac{1}{\mathsf{e}} + \eta)}{1 - (1 - \frac{1}{\mathsf{e}})} \right) (0.1484 + 0.5803).$$

    Since $\eta = 0.0142$, this evaluates to,

$$0.701 f_s^*. \tag{7}$$

- *Scenario 2:* $\frac{1}{2} < f_\ell^* \le 1 - \frac{1}{\mathsf{e}}$
  Here, the probability that ℓ is added to the matching is, $[1 - (f_\ell^*(\ell) + \eta)][P_1 + P_2] + [2(f_\ell^* + \eta) - 1]P_b$. This can re-arranged to obtain

$$(P_1 + P_2)(1 - \eta) + (2\eta - 1)P_b + f_\ell^*[2P_b - P_1 - P_2]. \tag{8}$$

  Since $\eta = 0.0142$ using Lemma 4 we have $(P_1 + P_2)(1 - \eta) + (2\eta - 1)P_b = 0.1048$. Similarly, using Lemma 4 we have $2P_b - P_1 - P_2 = 0.535$. Thus, Eq. (8) simplifies to,

$$0.1048 + f_\ell^* 0.535 \tag{9}$$

We can write Eq. (9) as $f_\ell^*[0.1048/f_\ell^* + 0.535]$. Note that $\frac{1}{2} < f_\ell^* \leq 1 - \frac{1}{\mathsf{e}}$. Thus, Eq. (9) can be lower-bounded by

$$0.701 f_\ell^*. \tag{10}$$

Thus combining Eqs. (7) and (10) with Lemma 2 we get a competitive ratio of 0.7.

We now show that the chosen value of $\eta = 0.0142$ ensures that both $\tilde{f}_\ell^*$ and $\tilde{f}_s^*$ are less than 1 *after* modification. Since $f_\ell^* \leq 1 - \frac{1}{\mathsf{e}}$ we have that $f_\ell^* + \eta \leq 1 - \frac{1}{\mathsf{e}} + 0.0142 \leq 1$. Note that $f_\ell^* \geq 1/2$. Hence, the modified value $\tilde{f}_s^*$ is always less than or equal to the original value, since $\left(\frac{1-(f_\ell^*+\eta)}{1-f_\ell^*}\right)$ is decreasing in the range $f_\ell^* \in [1/2, 1 - \frac{1}{\mathsf{e}}]$ and has a value less than 0.98 at $f_\ell^* = 1/2$. □

3.3 Final Algorithm: Roadmap

In the next few subsections, we describe our final edge-weighted algorithm with all of the attenuation factors. To keep it modular, we give the following guide to the reader. We note that the definition of large and small edges given below in Sect. 3.3.1 is different from the definition in the previous subsection.

- Section 3.3.1 describes the main algorithm which internally invokes two algorithms, EW_1 and EW_2, which are described in Sects. 3.3.2 and 3.3.3, respectively.
- Theorem 2 proves the final competitive ratio. This proof depends on the performance guarantees of EW_1 and EW_2, which are given by Lemmas 5 and 6, respectively.
- The proof of Lemma 5 depends on Claims 9, 10, and 11 (Found in the "Appendix"). Each of those claims is a careful case-by-case analysis. Intuitively, 9 refers to the case where the offline vertex u is incident to one large edge and one small edge (here the analysis is for the large edge), 10 refers to the case where u is incident to three small edges and 11 refers to the case where u is incident to a small edge and large edge (here the analysis is for the small edge).
- The proof of Lemma 6 depends on Claims 12 and 13 (Found in the "Appendix"). Again, both of those claims are proven by a careful case-by-case analysis. Since there are many cases, we have given a diagram of the cases when we prove them.

3.3.1 Algorithm EW: 0.705-Competitive Algorithm

In this section, we describe an algorithm EW (Algorithm 2), that achieves a competitive ratio of 0.705. The algorithm first solves the benchmark LP in Sect. 2.1 and obtains a fractional optimal solution $\mathbf{f}$. By invoking $\mathsf{DR}[\mathbf{f}, 3]$, it obtains a random integral solution $\mathbf{F}$. Notice that from LP Constraint (4) we see $f_e \leq 1 - \frac{1}{\mathsf{e}} \leq 2/3$. Therefore after $\mathsf{DR}[\mathbf{f}, 3]$, each $F_e \in \{0, 1, 2\}$. We say an edge e is *large* if $F_e = 2$ and *small* if $F_e = 1$ (note that this differs from the definition of large and small in Sect. 3.2).

We design two non-adaptive algorithms, denoted by EW_1 and EW_2, which take the sparse graph $G_{\mathbf{F}}$ as input. The difference between the two algorithms EW_1 and EW_2 is that EW_1 favors the *small edges* while EW_2 favors the *large edges*. The final algorithm is to take a convex combination of EW_1 and EW_2, i.e., run EW_1 with probability q and EW_2 with probability $1-q$.

Theorem 2 *For edge-weighted online stochastic matching with integral arrival rates, the algorithm* $\mathsf{EW}[q]$ *with* $q = 0.149251$ *achieves a competitive ratio of at least* 0.70546.

Algorithm 2: $\mathsf{EW}[q]$

1. Solve the benchmark LP in sub-section 2.1 for the input. Let $\mathbf{f}$ be the optimal solution vector.
2. Invoke $\mathsf{DR}[\mathbf{f}, 3]$ to obtain the vector $\mathbf{F}$.
3. Independently run $\mathsf{EW}_1[0.538]$ and $\mathsf{EW}_2[0.687, 1]$ with probabilities q and $1-q$ respectively on $G_{\mathbf{F}}$.

The details of algorithm EW_1 and EW_2 and the proof of Theorem 2 are presented in the following sections.

3.3.2 Sub-routine 1: Algorithm EW_1

In this section, we describe the randomized algorithm EW_1 (Algorithm 3). Let $\mathsf{PM}[\mathbf{F}, 3]$ refer to the process of constructing the graph $G_{\mathbf{F}}$ with F_e copies of each edge e, decomposing it into three matchings,[3] and randomly permuting the matchings. EW_1 first invokes $\mathsf{PM}[\mathbf{F}, 3]$ to obtain a *random ordered* triple of matchings, say $[M_1, M_2, M_3]$. Notice that from LP Constraint (4) and the properties of $\mathsf{DR}[\mathbf{f}, 3]$ and $\mathsf{PM}[\mathbf{F}, 3]$, an edge will appear in at most two of the three matchings. For a small edge $e = (u, v)$ with $F_e = 1$, we say e is of type Γ_1 if u has two other neighbors v_1 and v_2 with $F_{(u,v_1)} = F_{(u,v_2)} = 1$. We say e is of type Γ_2 if u has exactly one other neighbor v_1 with $F_{(u,v_1)} = 2$. WLOG we can assume that for every u, $F_u = \sum_{e \in \partial(u)} F_e = 3$; otherwise, we can add a dummy node v' to the neighborhood of u. Similarly, we assume $F_v = \sum_{e \in \partial(v)} F_e = 3$ by adding dummy nodes u'. Note that when we assign v to a dummy node u', it essentially means rejection of v when it arrives. Since all v has $F_v = 3$, we can safely assume that each v has exactly one edge in each of the three matchings output by $\mathsf{PM}[\mathbf{F}, 3]$. We use the terminology, *assign v to u*, to denote that *we assign v to u if u is not matched and reject v otherwise*.

[3] The natural way of repeatedly computing the maximum matching can be used to find this decomposition.

Algorithm 3: $\mathsf{EW}_1[h]$

1 Invoke $\mathsf{PM}[\mathbf{F}, 3]$ to obtain a *random ordered* triple matchings, say $[M_1, M_2, M_3]$.
2 When a vertex v comes for the first time, assign v to u_1 with $(u_1, v) \in M_1$.
3 When v comes for the second time, assign v to u_2 with $(u_2, v) \in M_2$.
4 When v comes for the third time, let $e = (u_3, v)$ be the edge in M_3. If e is either a large edge or a small edge of type Γ_1 then assign v to u_3. However, if e is a small edge of type Γ_2, then *with probability* h assign v to u_3 and do nothing otherwise.
5 When v comes for the fourth or more time, do nothing in that step.

Let $\mathsf{R}[\mathsf{EW}_1, 1/3]$ and $\mathsf{R}[\mathsf{EW}_1, 2/3]$ be the competitive ratio for a small edge and large edge respectively.

Lemma 5 *For* $h = 0.537815$, $\mathsf{EW}_1[h]$ *achieves a competitive ratio* $\mathsf{R}[\mathsf{EW}_1, 2/3] = 0.679417$, $\mathsf{R}[\mathsf{EW}_1, 1/3] = 0.751066$ *for a large and small edge respectively.*

Proof In case of the large edge e, we divide the analysis into three cases where each case corresponds to e being in one of the three matchings. And we combine these conditional probabilities using Bayes' theorem to get the final competitive ratio for e. For each of the two types of small edges, we similarly condition them based on the matching they can appear in, and combine them using Bayes' theorem. A complete version of proof can be found in Section A.1.1 of "Appendix". □

3.3.3 Sub-routine 2: Algorithm EW_2

Algorithm EW_2 (Algorithm 5) is a non-adaptive algorithm which takes $G_{\mathbf{F}}$ as input and performs well on the *large edges* with $F_e = 2$. Recall that $\mathbf{F}$ is an integral vector output by $\mathsf{DR}[\mathbf{f}, 3]$ with $F_e \in \{0, 1, 2\}$ for each e. WLOG, we can assume that $F_v = 3$ for every v in $G_{\mathbf{F}}$; otherwise we can add *dummy* vertices to ensure the case. Unlike EW_1, EW_2 will invoke a routine, denoted by $\mathsf{PM}^*[\mathbf{F}, 2]$ (Algorithm 4), to generate a pair of pseudo matchings from $\mathbf{F}$.

Algorithm 4: $\mathsf{PM}^*[\mathbf{F}, 2][y_1, y_2]$

1 Suppose v has two neighbors, say u_1, u_2, with $e_1 = (u_1, v)$ being a large edge while $e_2 = (u_2, v)$ being a small edge. Add e_1 to the primary matching M_1 and e_2 to the secondary matching M_2.
2 Suppose v has three neighbors and the incident edges are $\partial(v) = (e_1, e_2, e_3)$. Take a random permutation of $\partial(v)$, say $(\pi_1, \pi_2, *) \in \Pi(\partial(v))$. Independently add π_1 to M_1 with probability y_1 and π_2 to M_2 with probability y_2. (We use only the first two edges in the random permutation.)

Note that the pair of matchings generated by $\mathsf{PM}^*[\mathbf{F}, 2]$ can be pseudo-matchings. Consider the following case: (1) v has a large edge $e = (u, v)$; (2) u has a small edge $e' = (u, v')$ other than e; and (3) v' has two other small edges excluding

e'. From $\mathsf{PM}^*[\mathbf{F}, 2][y_1, y_2]$, we see that with probability 1, $e = (u, v) \in M_1$ and with probability $y_1/3$ (e' appears first in the random permutation and get selected in M_1), $e' = (u, v') \in M_1$. In that case, u will have two neighbors in M_1.

Algorithm 5 describes EW_2 which uses Algorithm 4 as a sub-routine.

Algorithm 5: $\mathsf{EW}_2[y_1, y_2]$

1 Invoke $\mathsf{PM}^*[\mathbf{F}, 2][y_1, y_2]$ to generate a *random ordered* pair of pseudo-matchings, say $[M_1, M_2]$.
2 When a vertex v comes for the first time, assign v to some u_1 if $(u_1, v) \in M_1$; When v comes for the second time, try to assign v to some u_2 if $(u_2, v) \in M_2$.
3 When a vertex v comes for the third or more time, do nothing in that step.

Let $\mathsf{R}[\mathsf{EW}_2, 1/3]$ and $\mathsf{R}[\mathsf{EW}_2, 2/3]$ be the competitive ratios for small edges and large edges, respectively.

Lemma 6 *For* $y_1 = 0.687$ *and* $y_2 = 1$, $\mathsf{EW}_2[y_1, y_2]$ *achieves a competitive ratio of* $\mathsf{R}[\mathsf{EW}_2, 2/3] = 0.8539$ *and* $\mathsf{R}[\mathsf{EW}_2, 1/3] = 0.4455$ *for a large and small edge respectively.*

Proof We analyze this on a case-by-case basis by considering the local neighborhood of the edge. A large edge can have two possible cases in its neighborhood, while a small edge can have eight possible cases. This is because of the fact that a large edge can have only small edges in its neighborhood while a small edge can have both large and small edges in its neighborhood. Choosing the worst case among the two for large edge and the worst case among the eight for the small edge, we prove the claim. Complete details of the proof can be found in section A.1.2 of "Appendix". □

3.3.4 Convex Combination of EW_1 and EW_2

In this section, we prove Theorem 2.

Proof Let (a_1, b_1) be the competitive ratios achieved by EW_1 for large and small edges, respectively. From Lemma 5 we have that $a_1 = 0.751$ and $b_1 = 0.679$. Similarly, let (a_2, b_2) denote the same for EW_2. From Lemma 6 we have $a_2 = 0.854$ and $b_2 = 0.445$.

We have the following two cases.

- $0 \le f_e \le \frac{1}{3}$: By marginal distribution property of $\mathsf{DR}[\mathbf{f}, 3]$, we know that $\Pr[F_e = 1] = 3f_e$. Thus, the final ratio is

$$3f_e(qb_1/3 + (1-q)b_2/3)/f_e = qb_1 + (1-q)b_2$$

- $1/3 \le f_e \le 1 - \frac{1}{\mathsf{e}}$: By the same properties of $\mathsf{DR}[\mathbf{f}, 3]$, we know that $\Pr[F_e = 2] = 3f_e - 1$ and $\Pr[F_e = 1] = 2 - 3f_e$. Thus, the final ratio is

$$\Big((3f_e - 1)(2qa_1/3 + 2(1-q)a_2/3) + (2 - 3f_e)(qb_1/3 + (1-q)b_2/3)\Big)/f_e$$

The competitive ratio of the convex combination is maximized at $q = 0.149251$ with a value of 0.70546. □

3.4 A Note on the Integral Arrival Rates Assumption

As mentioned in the introduction, we make the simplifying assumption that the arrival rates $r_v = 1$ for every online vertex $v \in V$. Our algorithms and analysis crucially rely on this assumption. Specifically, our algorithm finds two matchings in the offline graph and uses them to guide the online matching process. In doing so, the algorithm assumes that each edge in those matchings is incident to an online vertex with an arrival rate of 1. Without this assumption, two key problems arise. First, Lemma 4, which bounds the probability that each edge gets matched, is no longer true as all of the analysis in the proof relies critically on the integral arrival rates assumption. Putting it simply, when arrival rates are arbitrary, Lemma 4 does not hold. Consider an edge $e = (u, v)$ either in M_1 or M_2 with $r_v = 1/n$ for example, where n is the total number of online rounds. We observe that e will be matched with a probability no larger than the probability that v arrives at least once, which is $1 - (1 - 1/n^2)^n \sim 1/n$.

Second, the algorithm described above can have arbitrarily bad performance when the arrival rates are less than 1. This algorithm will find two matchings in the offline graph and only attempt to match edges in those matchings. However, note that when a vertex has a small arrival rate (e.g. $\frac{1}{n}$), it is unlikely to arrive at all during the online process. It is possible to construct examples where the edges added to our two matchings after our rounding procedure will be incident on online vertices that are unlikely to arrive. Thus, our online algorithm would match almost no edges while the optimal offline algorithm could find a large value matching among the vertices that actually arrived.

4 Vertex-Weighted Stochastic I.I.D. Matching with Integral Arrival Rates

In this section, we consider vertex-weighted online stochastic matching on a bipartite graph G under the known I.I.D. model with integral arrival rates. We present an algorithm in which each offline vertex u has a competitive ratio of at least $0.72998 > 1 - 2\mathsf{e}^{-2}$.

Recall from Sect. 2.6 that after invoking $\mathsf{DR}[\mathbf{f}, 3]$, we can obtain a (*random*) integral vector $\mathbf{F}$ with $F_e \in \{0, 1, 2\}$. Define $\mathbf{H} = \mathbf{F}/3$ and thus $H_e \in \{0, 1/3, 2/3\}$. Notice that $F_u = \sum_{e \in \partial(u)} F_e \leq 3$ due to the degree preservation property from $\mathsf{DR}[\mathbf{f}, 3]$ and $H_u \doteq \sum_{e \in \partial(u)} H_e \leq 1$. Let $G(\mathbf{F})$ and $G(\mathbf{H})$ be the induced sub-graphs of G determined by F_e and H_e respectively. In particular, all edges e with $F_e = 0$ and H_e=0 are removed from the respective graphs.

The main idea of our algorithm is as follows.

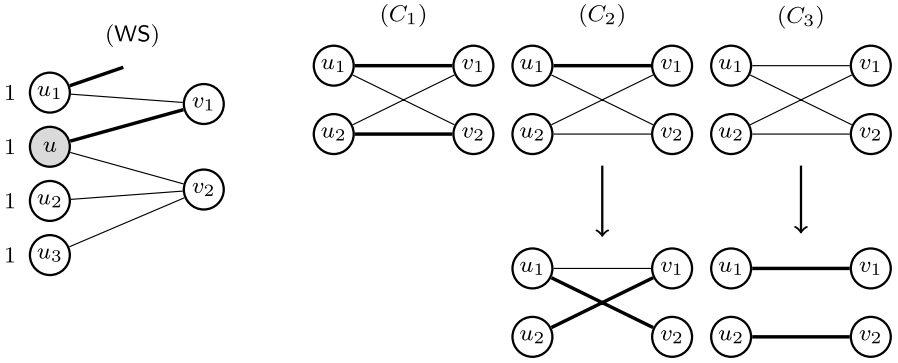

Fig. 2 Left: The WS for Jaillet and Lu [13] for their vertex-weighted case. Right: The three possible types of cycles of length 4 after applying DR[**f**, 3]. Thin edges have $H_e = 1/3$ and thick edges have $H_e = 2/3$. The arrows show how cycles C_2 and C_3 are broken

1. Solve the vertex-weighted benchmark LP in Sect. 2.1. Let **f** be an optimal solution vector.
2. Invoke DR[**f**, 3] to obtain an integral vector **F** and a fractional vector **H** with $\mathbf{H} = \mathbf{F}/3$.
3. Apply a series of modifications to **H** and transform it to another solution $\mathbf{H}'$ (See Sect. 4.2).
4. Run the Randomized List Algorithm [13] with parameter $\mathbf{H}'$, denoted by RLA[$\mathbf{H}'$], on $G(\mathbf{H})$.

We first briefly describe how we overcome the vertex-weighted and unweighted *bottleneck* cases for the algorithm in [13] and then explain the algorithm in full detail. Recall that [13] analyze their algorithm by considering cases for various neighborhood structures at a given offline vertex.

The WS for the vertex-weighted case in [13] is shown in Fig. 2 (left), which happens at a node u with a competitive ratio of 0.725. Jaillet and Lu described and analyzed this case in Claim 5 within the proof of Lemma 7 from [13]. However, also from their analysis, we have that the node u_1 in Fig. 2 (left) has a competitive ratio of at least 0.736. Hence, we can *boost* the performance of u at the cost of u_1 by making u more likely to match and u_1 less likely. Specifically, we increase the value of $H_{(u,v_1)}$ and decrease the value $H_{(u_1,v_1)}$. Cases (10) and (11) in Fig. 4 illustrate this.

After this modification, we will later show that the new WS for vertex-weighted is now the C_1 cycle shown in both Figs. 1 and 2 (right) and defined formally in Sect. 4.2.1. In fact, this is also the WS for the unweighted problem in [13] as well. Jaiillet and Lu give the following explaining in their "Tight example" section [13]:

> It is worth mentioning that the ratio of $1 - 2\mathrm{e}^{-2}$ is tight for this algorithm. The ratio can be achieved with the following example: Consider the case of the complete bipartite graph $K_{n,n}$, where n is an even number. One optimal solution to [the LP from [13]] consists of a disjoint union of $n/2$ cycles of length 4; within each cycle, two edges carry 1/3 flow, and two carry 2/3 flow. Since the

underlying graph is $K_{n,n}$, the optimal offline solution is n. On the other hand, for any cycle in the offline optimal solution, the expected number of matches is $2(1-\mathsf{e}^{-2})$. Therefore, the competitive ratio in this instance is $1-2\mathsf{e}^{-2} \approx 0.729$.

However, Lemma 7 implies that C_1 cycles can be avoided with probability at least $\frac{3}{\mathsf{e}}-1$ due to our LP and rounding procedure. This helps us improve the ratio even for the unweighted case in [13]. Lemma 7 describes this formally.

Lemma 7 *For any given $u \in U$, u appears in a C_1 cycle after* $\mathsf{DR}[\boldsymbol{f}, 3]$ *with probability at most* $2-\frac{3}{\mathsf{e}}$.

Proof Consider the graph $G(\mathbf{F})$ with $\mathbf{F}$ obtained from $\mathsf{DR}[\mathbf{f}, 3]$. 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 $f_e \le 1/3$, $F_e \le 1$ after $\mathsf{DR}[\mathbf{f}, 3]$, and hence $H_e = F_e/3 \le 1/3$. 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 $\sum_{e\in\partial(u)} f_e \le 1$. 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 $f_e \le 1 - \frac{1}{\mathsf{e}}$ from LP constraint (4), we have $q_2 \le 3(1-\frac{1}{\mathsf{e}}) - 1 = 2 - \frac{3}{\mathsf{e}}$.
- *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 $H_u \le 1$. 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 (5), we know that $f_{e_1} + f_{e_2} \le 1 - \frac{1}{\mathsf{e}^2}$, and hence $q_2 \le 3(1-\frac{1}{\mathsf{e}^2}) - 2 < 2 - \frac{3}{\mathsf{e}}$.

□

Now we present the details of RLA based on a given $\mathbf{H}'$ in Sect. 4.1 and then discuss the two modifications transforming $\mathbf{H}$ to $\mathbf{H}'$ in Sect. 4.2. We give a formal statement of our algorithm in Sect. 4.3 and the related analysis.

4.1 RLA Algorithm

We describe how to apply the RLA algorithm with parameter $\mathbf{H}'$. WLOG assume that $H'_v \doteq \sum_{e\in\partial(v)} H'_e = 1$.[4] Let $\delta_{\mathbf{H}'}(v)$ be the set of neighbors of v in $G(\mathbf{H}')$ with $H'_{u,v} > 0$. Thus, $|\delta'_{\mathbf{H}}(v)| \ge 2$ since each non-zero H'_e satisfies $H'_e \in \{1/3, 2/3\}$.

[4] We can add a dummy node u' to v if $H'_v < 1$ and assignment v to u' simply means rejection of v.

Each time when a vertex v comes, RLA first generates a random list $\mathcal{R}_v$, which is a permutation over $\delta_{\mathbf{H}'}(v)$, as follows.

- If $|\delta_{\mathbf{H}'}(v)| = 2$, say $\delta_{\mathbf{H}'}(v) = (u_1, u_2)$, then sample a random list $\mathcal{R}_v$ such that

$$\Pr[\mathcal{R}_v = (u_1, u_2)] = H'_{(u_1,v)}, \quad \Pr[\mathcal{R}_v = (u_2, u_1)] = H'_{(u_2,v)} \tag{11}$$

- If $|\delta_{\mathbf{H}'}(v)| = 3$, say $\delta_{\mathbf{H}'}(v) = (u_1, u_2, u_3)$. Then we sample a permutation of (i, j, k) over $\{1, 2, 3\}$ such that

$$\Pr[\mathcal{R}_v = (u_i, u_j, u_k)] = H'_{(u_i,v)} \frac{H'_{(u_j,v)}}{H'_{(u_j,v)} + H'_{(u_k,v)}} \tag{12}$$

We can verify that the sampling distributions described in Eqs. (11) and (12) are valid since $H'_v = \sum_{e \in \partial(v)} H'_e = 1$ and no $H'_e = 1$. (Both properties are guaranteed in the two modifications shown in Sect. 4.2.) The full details of the Random List Algorithm, RLA[$\mathbf{H}'$], are shown in Algorithm 6.

Algorithm 6: ([JL13]) RLA[$\mathbf{H}'$]

1 When a vertex v comes, generate a random list $\mathcal{R}_v$ satisfying Equation (11) or (12).
2 If all u in the list are matched, then drop the vertex v; otherwise, assign v to the first unmatched u in the list.

4.2 Two Kinds of Modifications to H

As stated earlier, we first modify $\mathbf{H}$ before running the RLA algorithm. In this section, we describe the modifications.

4.2.1 The First Modification to H: Cycle Breaking

The first modification is to break the cycles of length 4 deterministically. There are three possible cycles of length 4 in the graph $G_{\mathbf{H}}$, denoted C_1, C_2, and C_3 in the right-hand side of Fig. 2 and defined as follows.

Definition 1 *(*Cycle type C_1 *)* This length 4 cycle is a complete bipartite graph on two offline vertices and two online vertices. It has two vertex-disjoint edges with $H_e = 2/3$ and the remaining edges have $H_e = 1/3$.

Definition 2 *(*Cycle type C_2 *)* This length 4 cycle is a complete bipartite graph on two offline vertices and two online vertices. It has one edge with $H_e = 2/3$ and the remaining edges have $H_e = 1/3$.

Definition 3 *(*Cycle type C_3 *)* This length 4 cycle is a complete bipartite graph on two offline vertices and two online vertices. All edges have $H_e = 1/3$.

In [13], they give an efficient way to break C_2 and C_3, as shown in Fig. 2. Cycle C_1 cannot be modified further and hence, is the bottleneck for their unweighted case. Notice that, while breaking the cycles of type C_2 and C_3, new cycles of C_1 can be created in the graph. Since our randomized construction of solution **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 not to create any new C_1 cycles. This procedure is summarized in Algorithm 7 and its correctness is proved in Lemma 8.

Algorithm 7: [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 proof of Lemma 8 will follow from three claims which we state and prove below.

Claim 3 *Breaking cycles will not change the value H_w for any $w \in U \cup V$.*

Proof As shown in Fig. 2, we increase and decrease edge values f_e in such a way that their sums H_w at any vertex w will be preserved. □

Claim 4 *After breaking a cycle of type C_2, the vertices u_1, u_2, v_1, and v_2 can never be part of any length four cycle.*

Proof Consider the structure after breaking a cycle of type C_2. Note that the edge (u_2, v_2) has been permanently removed and hence, these four vertices together can never be part of a cycle of length four. The vertices u_1 and v_1 have $H_{u_1} = 1$ and $H_{v_1} = 1$ respectively. So they cannot have any other edges and therefore cannot appear in any length four cycle. The vertices u_2 and v_2 can each have one additional edge, but since the edge (u_2, v_2) has been removed, they can never be part of any cycle with length less than six. □

Claim 5 *When all length four cycles are of type C_1 or C_3, breaking exactly one cycle of type C_3 cannot create a new cycle of type C_1.*

Proof First, we note that since no edges will be added during this process, we cannot create a new cycle of length four or join with a cycle of type C_1. Therefore, the only

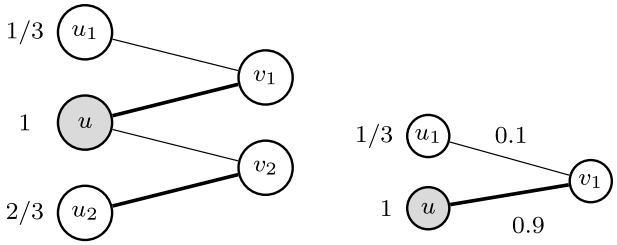

Fig. 3 An example of the need for the second modification. For the left: competitive analysis shows that in this case, u_1 and u_2 can achieve a high competitive ratio at the expense of u. For the right: an example of balancing strategy by making v_1 slightly more likely to pick u when it comes

cycles which could be affected are of type C_3. However, every cycle c of type C_3 falls into one of two cases:

Case 1 c is the cycle we are breaking. In this case, c cannot become a cycle of type C_1 since we remove two of its edges and break the cycle.

Case 2 c is not the cycle we are breaking. In this case, c can have at most one of its edges converted to a 2/3 edge. Let c' be the length four cycle we are breaking. Note that c and c' will differ by at least one vertex. When we break c', the two edges which are converted to 2/3 will cover all four vertices of c'. Therefore, at most one of these edges can be in c.

Note that breaking one cycle of type C_3 could create cycles of type C_2, but these cycles are always broken in the next iteration, before breaking another cycle of type C_3.

□

Lemma 8 *After applying Algorithm 7 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.*

Proof The proof follows from Claims 3, 4, and 5. Notice that C_2 cycles can be freely broken without creating new C_1 cycles. After removing all cycles of type C_2, removing a single cycle of type C_3 cannot create any cycles of type C_1. Hence, Algorithm 7 removes all C_2 and C_3 cycles without creating any new C_1 cycles. □

4.2.2 The Second Modification to H: Balancing the Worst Case

Informally, this second modification decreases H_e values on u with $H_u = 1/3$ or $H_u = 2/3$ and increases H_e values on u with $H_u = 1$. We will illustrate this intuition on the following example.

Consider the two graphs, denoted by G_L and G_R in Fig. 3, where thin and thick edges represent $H_e = 1/3$ and $H_e = 2/3$ respectively. We now compute the competitive ratio after applying RLA on G_L. For each node w, let $\delta(w)$ be the set of neighbors of w in G_L. Let A_u be the event that u is matched in RLA. Let $A_{u,1}$ denote the event that among the

n random arrival lists, there exists a list starting with u. For each $v \in \delta(u) = \{v_1, v_2\}$, let $A^v_{u,2}$ denote the event that among the n online arrival lists, there exists successive lists such that (I) Each of those lists starts with a $u' \neq u$ and $u' \in \delta(v)$ and (II) The lists arrive in an order which ensures u will be matched by the algorithm. From lemma 4 and Corollary 1 in [13], we have the following.

Lemma 9 ([13]) *Suppose u is not a part of any cycle of length* 4. *We have*

$$\Pr[A_u] = 1 - (1 - \Pr[A_{u,1}]) \prod_{v \in \delta(u)} (1 - \Pr[A^v_{u,2}]) + o(1/n)$$

The validity of the above lemma can be seen as follows: the probability that u is not matched ($\neg A_u$) can be approximated up to $o(1/n)$ by the probability that none of lists arrives staring with u ($\neg A_{u,1}$) and none of events described in (II) occurs ($\wedge_{v \in \delta(u)} \neg A^v_{u,2}$).

For the node u in G_L, we have $\Pr[A_{u,1}] = 1 - \mathsf{e}^{-1}$. From the definition, $A^{v_1}_{u,2}$ is the event that among the n online lists, the random list $\mathcal{R}_{v_1} = (u_1, u)$ comes at least twice. Notice that the list $\mathcal{R}_{v_1} = (u_1, u)$ arrives with probability $\frac{1}{3n}$ each round. Thus we have $\Pr[A^{v_1}_{u,2}] = \Pr[X \geq 2] = 1 - \mathsf{e}^{-1/3}(1 + 1/3)$, where $X \sim \text{Pois}(1/3)$. Similarly, we can get $\Pr[A^{v_2}_{u,2}] = 1 - \mathsf{e}^{-2/3}(1 + 2/3)$ and the resultant $\Pr[A_u] = 1 - \frac{20}{9\mathsf{e}^2} \sim 0.699$. Observe that for u_1 and u_2, $\Pr[A_{u_1} \geq \Pr[A_{u_1,1}] = 1 - \mathsf{e}^{-1/3}$ and $\Pr[A_{u_2}] \geq \Pr[A_{u_2,1}] = 1 - \mathsf{e}^{-2/3}$.

Let $\mathsf{R}[\mathsf{RLA}, 1]$, $\mathsf{R}[\mathsf{RLA}, 1/3]$ and $\mathsf{R}[\mathsf{RLA}, 2/3]$ be the competitive ratio achieved by RLA for u, u_1 and u_2 respectively. Hence, we have $\mathsf{R}[\mathsf{RLA}, 1] \sim 0.699$ while $\mathsf{R}[\mathsf{RLA}, 1/3] \geq 3(1 - \mathsf{e}^{-1/3}) \sim 0.8504$ and $\mathsf{R}[\mathsf{RLA}, 2/3] \geq 0.729$.

Intuitively, one can improve the worst case ratio by increasing the arrival rate for $\mathcal{R}_{v_1} = (u, u_1)$ while reducing that for $\mathcal{R}_{v_1} = (u_1, u)$. Suppose we modify $H_{(u_1,v_1)}$ and $H_{(u,v_1)}$ to $H'_{(u_1,v_1)} = 0.1$ and $H'_{(u,v_1)} = 0.9$ as shown in G_R, the arrival rate for $\mathcal{R}_{v_1} = (u, u_1)$ and $\mathcal{R}_{v_1} = (u_1, u)$ gets modified to $0.1/n$ and $0.9/n$ respectively. The updated values are $\Pr[A_{u,1}] = 1 - \mathsf{e}^{-0.9-1/3}$, $\Pr[A^{v_1}_{u,2}] = 1 - \mathsf{e}^{-0.1}(1 + 0.1)$, $\mathsf{R}[\mathsf{RLA}, 1] = 0.751$, $\Pr[A_{u_1,1}] = 1 - \mathsf{e}^{-1/3}$, $\Pr[A^{v_1}_{u_1,2}] \sim 0.227$ and $\mathsf{R}[\mathsf{RLA}, 1/3] \geq 0.8$. Hence, the performance on WS instance improves. Notice that after the modifications, $H'_u = H'_{(u,v_1)} + H_{(u,v_2)} = 0.9 + 1/3$.

Figure 4 describes the various modifications applied to $\mathbf{H}$ vector. The values on top of the edge, denote the new values. Cases (11) and (12) help improve upon the WS described in Fig. 2.

4.3 Vertex-Weighted Algorithm VW

4.3.1 Analysis of Algorithm VW

The full details of our vertex-weighted algorithm are stated as follows.

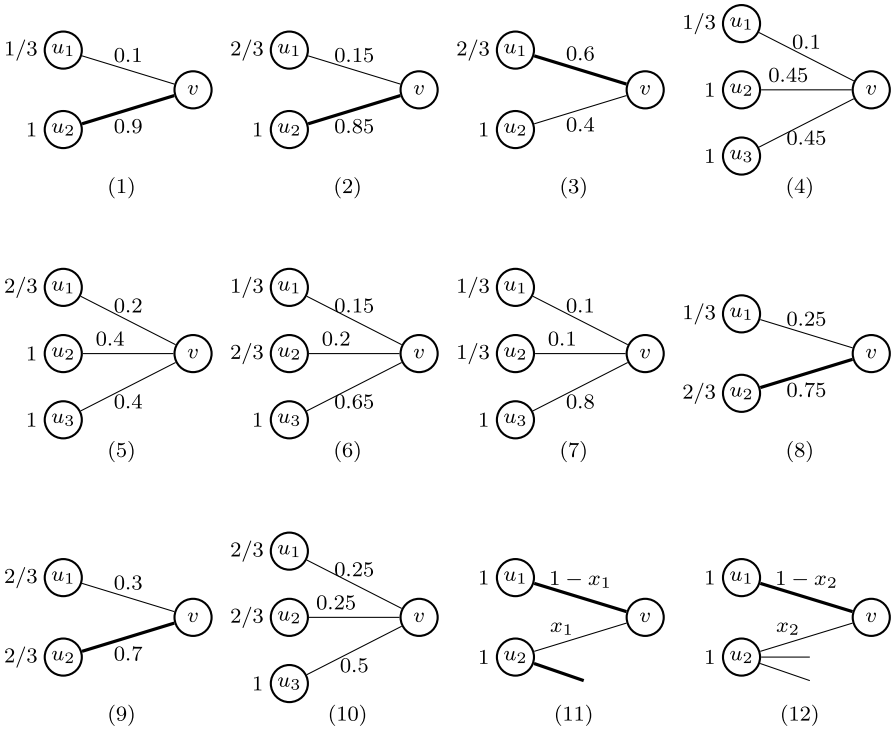

Fig. 4 Illustration for second modification to $\mathbf{H}$. The value assigned to each edge represents the value after the second modification. Here, $x_1 = 0.2744$ and $x_2 = 0.15877$

Algorithm 8: VW

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 $\mathbf{H}$.
3 Apply the first modification shown in Algorithm 7 to transfer $\mathbf{H}$ to $\widehat{\mathbf{H}}$.
4 Apply the second modification shown in Figure 4 to morph $\widehat{\mathbf{H}}$ to $\mathbf{H}'$.
5 Run RLA[$\mathbf{H}'$].

The algorithm VW consists of two different random processes: sub-routine DR[$\mathbf{f}, 3$] in the offline phase and RLA in the online phase. Consequently, the analysis consists of two parts. First, for a given graph $G_{\mathbf{H}}$, we analyze the ratio of RLA[$\mathbf{H}'$] for each node u with $H_u = 1/3, H_u = 2/3$ and $H_u = 1$. The analysis is similar to [13]. Second, we analyze the probability that DR[$\mathbf{f}, 3$] transforms each u, with fractional f_u values, into the three discrete cases seen in the first part. By combining the results from these two parts we get the final ratio.

Let us first analyze the competitive ratio for $\mathsf{RLA}[\mathbf{H}']$. For a given $\mathbf{H}$ and $G(\mathbf{H})$, let P_u be the probability that u gets matched in $\mathsf{RLA}[\mathbf{H}']$. Notice that the value P_u is determined not just by the algorithm RLA itself, but also the modifications applied to $\mathbf{H}$. We define the competitive ratio of a vertex u achieved by RLA as P_u/H_u, after modifications. Lemma 10 gives the respective ratio values. The proof can be found in section A.2.1 in the "Appendix".

Lemma 10 *Consider a given* $\mathbf{H}$ *and a vertex* u*. The respective ratios achieved by* RLA *after the modifications are as described below.*

- If $H_u = 1$, then the competitive ratio $\mathsf{R}[\mathsf{RLA}, 1] = 1 - 2\mathsf{e}^{-2} \sim 0.72933$ if u is in the first cycle C_1 and $\mathsf{R}[\mathsf{RLA}, 1] \geq 0.735622$ otherwise.
- If $H_u = 2/3$, then the competitive ratio $\mathsf{R}[\mathsf{RLA}, 2/3] \geq 0.7847$.
- If $H_u = 1/3$, then competitive ratio $\mathsf{R}[\mathsf{RLA}, 1/3] \geq 0.7622$.

Now we have all ingredients to state and prove Theorem 6.

Theorem 6 *For vertex-weighted online stochastic matching with integral arrival rates, online algorithm* VW *achieves a competitive ratio of at least* 0.7299.

Proof From Lemmas 7 and 8, we know that any u is present in cycle C_1 with probability at most $(2 - \frac{3}{\mathsf{e}})$.

Consider a node u with $2/3 \leq f_u \leq 1$ and let q_1, q_2, q_3 be the probability that after $\mathsf{DR}[\mathbf{f}, 3]$ and the first modification, $H_u = 1$ and u is in the first cycle C_1, $H_u = 1$ and u is not in C_1, $H_u = 2/3$ respectively. From the marginal distribution of $\mathsf{DR}[\mathbf{f}, 3]$, we have that $q_1 + q_2 + q_3(2/3) = \mathbb{E}[\mathbf{F}_u]/3 = 3f_u/3 = f_u$. From Lemma 10, we get that the final ratio for u is

$$\begin{aligned}\frac{1}{f_u}\Pr[u \text{ is matched}] = \frac{1}{f_u}\Big(& q_1 \Pr[\text{ u is matched } | H_u = 1, u \in C_1] \\ &+ q_2 \Pr[\text{ u is matched } | H_u = 1, u \notin C_1] \\ &+ q_3 \Pr[\text{ u is matched } | H_u = 2/3, u \in C_1]\Big) \\ \geq\ & \frac{0.72933q_1 + 0.735622q_2 + (2/3) * 0.7847q_3}{q_1 + q_2 + (2/3)q_3}\end{aligned}$$

Minimizing the above expression subject to (1) $q_1 + q_2 + q_3 = 1$; (2) $0 \leq q_i, 1 \leq i \leq 3$; (3) $q_1 \leq 2 - \frac{3}{\mathsf{e}}$, we get a minimum value of 0.729982 for $q_1 = 2 - \frac{3}{\mathsf{e}}$ and $q_2 = \frac{3}{\mathsf{e}} - 1$.

For any node u with $0 \leq u \leq 2/3$, we know that the ratio is at least the min value of $\mathsf{R}[\mathsf{RLA}, 2/3]$ and $\mathsf{R}[\mathsf{RLA}, 1/3]$, which is 0.7622. This completes the proof of Theorem 6. □

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 captures the probability that the edge $e = (u, v)$ is present when we *probe* it. We assume this process is independent of the stochastic arrival of each v. We will show that the simple non-adaptive algorithm introduced in [12] can be extended to this general case. This achieves a competitive ratio of $(1 - \frac{1}{\mathsf{e}})$. Note that Manshadi *et al.* [19] show that no non-adaptive algorithm can possibly achieve a ratio better than $(1 - 1/\mathsf{e})$ 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.

We use an LP similar to [13] for the case of non-integral arrival rates. For each $e = (u, v) \in E$, let f_e be the expected number of probes on edge e. When there are multiple copies of v, we count the sum of probes among all copies of e in the offline optimal matching and thus some realizations of f_e can be greater than 1. Consider the below LP:

$$\max \sum_{e \in E} w_e f_e p_e \tag{13}$$

$$\text{s.t.} \sum_{e \in \partial(u)} f_e p_e \leq 1 \qquad \forall u \in U \tag{14}$$

$$\sum_{e \in \partial(v)} f_e \leq r_v \qquad \forall v \in V \tag{15}$$

$$0 \leq f_e \qquad \forall e \in E \tag{16}$$

Similar to Lemma 1, we have the below lemma.

Lemma 11 *Let* OPT *denote the expected weight obtained by an offline optimal algorithm. Let* $\mathbf{f}^*$ *denote the optimal solution to the above* LP. *Then* $\sum_{e \in E} w_e f_e^* p_e \geq \mathbb{E}[\text{OPT}]$.

Proof For each edge e, let Y_e indicate if e is probed (not necessarily matched) in an offline optimal algorithm after observing the full arrival sequence $\mathcal{A}$. Let $y_e \doteq \mathbb{E}_{\mathcal{A}}[Y_e]$ for every edge $e \in E$. Note that $\mathbb{E}[\text{OPT}] = \sum_{e \in E} w_e y_e p_e$. Now we show that $\mathbf{y} \doteq (y_e)_{e \in E}$ is feasible solution to LP (13).

Consider a given u. Let Z_e indicate if e is present when probed with mean p_e. Observe that $\sum_{e \in \partial(u)} Y_e Z_e$ indicate if u is matched in OPT. For any given realization of $\mathcal{A}$, we have $\sum_{e \in \partial(u)} Y_e Z_e \leq 1$ since u can be matched at most once. Thus, by linearity of expectation, we have $\mathbb{E}[\sum_{e \in \partial(u)} Y_e Z_e \leq 1] \leq 1$, which implies that $\sum_{e \in \partial(u)} y_e p_e \leq 1$. Thus, Constraint (14) is valid.

Consider a given v. Let R_v be the (random) number of copies in $\mathcal{A}$. Observe that $\sum_{e\in\partial(v)} Y_e \le R_v$. By taking expectation over randomness of $\mathcal{A}$ on both sides, we get $\mathbb{E}\left[\sum_{e\in\partial(v)} Y_e\right] \le \mathbb{E}[R_v] = r_v$. Thus, Constraint (15) is valid.

Hence, we have that the expected performance of an offline optimal is upper bounded by the optimal value to LP (13). □

Our algorithm is summarized in Algorithm 9. Notice that Constraint (15) ensures that Step 2 is valid. For a given v, recall that $\partial(v)$ is the set of edges incident to v in E.

Algorithm 9: SM

1 Construct and solve LP (13). WLOG assume $\{f_e | e \in E\}$ is an optimal solution.
2 When a vertex v arrives, sample an edge $e = (u, v) \in \partial(v)$ with probability $\frac{f_e}{r_v}$. Assign v to u if u is not matched.

Theorem 7 *For edge-weighted online stochastic matching with arbitrary arrival rates and stochastic rewards, online algorithm* SM (9) *achieves a competitive ratio of* $1 - 1/\mathsf{e}$, *which is optimal all among all non-adaptive algorithms.*

Proof Let $B(u, t)$ be the event that u is safe (i.e., u is not matched) at beginning of round t and $A(u, t)$ be the event that vertex u is matched during the round t conditioned on $B(u, t)$. From the algorithm, we know $\Pr[A(u,t)] \le \sum_{e=(u,v)\in\partial(u)} \frac{r_v}{n}\frac{f_e}{r_v} p_e \le \frac{1}{n}$, which is followed 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 a given edge $e = (u, v)$ in the graph. Notice that the probability that e gets matched in SM should be

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

Note that Manshadi et al. [19] show that no non-adaptive algorithm can possibly achieve a ratio better than $(1 - 1/\mathsf{e})$ 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. □

6 Integral Arrival Rates with Uniform Stochastic Rewards

In this section, we consider a special case of the model studied in Sect. 5 and show that we can indeed surpass the $1 - 1/\mathsf{e}$ barrier. We specialize the model in the following two ways. (1) We consider the unweighted case with uniform constant edge probabilities (i.e., $w_e = 1$ and $p_e = p$ for some constant $p \in (0, 1]$ for all $e \in E$). The constant p is arbitrary, but independent of the problem parameters. (2) Each vertex v that comes online has an integral arrival rate r_v (as usual WLOG $r_v = 1$ and $|V| = n$). We refer to this special model as *unweighted online stochastic matching with integral arrival rates and uniform stochastic rewards*. Note that even for this special case, given an offline instance (i.e., the sequence of realizations for the online arrival), it is unclear if we can efficiently solve or approximate the exact offline optimal within $(1 - \epsilon)$ without any extra assumptions. Hence we cannot directly apply the Monte-Carlo simulation technique in [19] to approximate the exact expected offline optimal within an arbitrary desired accuracy. Here we present a strengthened LP as the benchmark to upper bound the offline optimal.

$$\max \; p \cdot \sum_{e \in E} f_e \tag{17}$$

$$\text{s.t.} \sum_{e \in \partial(u)} f_e \cdot p \leq 1 \quad \forall u \in U \tag{18}$$

$$\sum_{e \in \partial(v)} f_e \leq 1 \quad \forall v \in V \tag{19}$$

$$\sum_{e \in S} f_e p \leq 1 - \exp(-|S|p) \quad \forall S \subseteq \partial(u), |S| \leq 2/p \tag{20}$$

$$0 \leq f_e \quad \forall e \in E \tag{21}$$

Lemma 12 LP (17) *is a valid upper bound for the expected offline optimal.*

Proof It suffices to show that constraint (20) is valid (the correctness of the other constraints follows from the previous section). Let f_e represent the expected number of probes on edge e in an offline optimal algorithm (denoted by OPT). Consider a given $S \subseteq \partial(u)$ and let $X_S \in \{0, 1\}^{|S|}$ be the indicators for edges in S to be matched in OPT. By definition we have $\mathbb{E}[X_S] = \sum_{e \in S} f_e \cdot p$. Let Y_S be the (random) number of arrivals of all vertices incident to edges in S during the online phase. Observe that $\mathbb{E}[X_S | Y_S] \leq 1 - (1 - p)^{Y_S}$. Thus, we have,

$$\mathbb{E}[X_S] = \mathbb{E}_Y[\mathbb{E}[X_S | Y_S]] \leq \mathbb{E}_{Y_S}[1 - (1 - p)^{Y_S}].$$

Note that for any constant size $|S| \leq 2/p$, Y_S follows a Poisson distribution with mean $|S|$ (since we assume that the total number of online rounds n is sufficiently large). Therefore, we have

$$\begin{aligned}\mathbb{E}[X_S] &\leq \mathbb{E}_{Y_S}\left[1-(1-p)^{Y_S}\right] = 1-\mathbb{E}_{Y_S}[(1-p)^{Y_S}] \\ &= 1-\exp(-|S|)\sum_{k=0}^{\infty}\frac{|S|^k}{k!}(1-p)^k = 1-\exp(-|S|)\sum_{k=0}^{\infty}\frac{\left(|S|(1-p)\right)^k}{k!} \\ &= 1-\exp\Big(-|S|+|S|(1-p)\Big) = 1-\exp(-p|S|)\end{aligned}$$

Therefore we show that $\mathbf{f}$ is feasible to constraint (20). □

Note that it is impossible to beat $1-1/\mathrm{e}$ using LP (17) as the benchmark without the extra constraint (20) (see the hardness instance shown in [5]). Our main idea in the online phase is based on [19]. In the offline phase, we first solve LP (17) and get an optimal solution $\{f_e^*\}$. When a vertex v arrives, we generate a random list of two choices based on $\{f_e^*|e \in \partial(v)\}$, denoted by $\mathcal{L}_v = (\mathcal{L}_v(1), \mathcal{L}_v(2))$, where $\mathcal{L}_v(1), \mathcal{L}(2) \in \partial(v)$. Our online decision based on $\mathcal{L}_v$ is as follows: if $\mathcal{L}_v(1) = (u, v)$ is safe, i.e., u is available, then match v to u; else if the second choice $\mathcal{L}_v(2)$ is safe match v to $\mathcal{L}_v(2)$. The random list $\mathcal{L}_v$ generated based on $\{f_e^*|e \in \partial(v)\}$ satisfies the following two properties:

(P1): $\Pr[\mathcal{L}_v(1) = e] = f_e^*$ and $\Pr[\mathcal{L}_v(2) = e] = f_e^*$ for each $e \in \partial(v)$.
(P2): $\Pr[\mathcal{L}_v(1) = e \wedge \mathcal{L}_v(2) = e] = \max\Big(2f_e - 1, 0\Big)$ for each $e \in \partial(v)$.

Algorithm 10:

1 Solve LP (17) and let $\{f_e^*|e \in E\}$ be an optimal solution.
2 When a vertex v arrives, generate a random list $\mathcal{L}_v$ of two choices based on $\{f_e^*|e \in \partial(v)\}$ such that $\mathcal{L}_v$ satisfies Property **(P1)** and **(P2)**.
3 If $\mathcal{L}_v(1) = (u, v)$ is safe, i.e., u is available, then assign v to u; else if the second choice $\mathcal{L}_v(2)$ is safe, match v to it.

There are several ways to generate $\mathcal{L}_v$ satisfying **(P1)** and **(P2)**. One simple way is shown in Section 4 of [19]. Another simple way of obtaining $\mathcal{L}_v$ as required is by running $\mathsf{DR}[\mathbf{f}^*, 2]$ and randomly permuting the two obtained matchings. We can verify that all of the calculations shown in [19] can be extended here if we incorporate the independent process that each e will be present with probability p after we assign v to u. Hence, the final ratio is as follows (this can be viewed as a counterpart to Equation (15) on page 11 of [19]).

$$\frac{\mathbb{E}[\mathrm{ALG}]}{\mathbb{E}[\mathrm{OPT}]} \geq \min_{u \in U} \left(\frac{(1-\mathrm{e}^{-f'_u}) + q'_u \mathrm{e}^{-2} - (q'_u)^2 \mathrm{e}^{-1}\left(\frac{1}{2} - \mathrm{e}^{-1}\right) - \mathrm{e}^{-2} f'_u (1-f'_u)}{f'_u} \right) \doteq F(f'_u, q'_u) \tag{22}$$

where $f'_u = \sum_{e \in \partial(u)} f^*_e \cdot p \leq 1$ and $q'_u = p \cdot \left(\sum_{e=(u,v) \in \partial(u)} \Pr[\mathcal{L}_v(2) = e \wedge \mathcal{L}_v(1) \neq e] \right)$. Observe that

$$q'_u \leq p \cdot \left(\sum_{e=(u,v) \in \partial(u)} \Pr[\mathcal{L}_v(2) = e] \right) = p \cdot \left(\sum_{e \in \partial(u)} f^*_e \right) = f'_u \leq 1$$

We can verify that for each given $f'_u \leq 1$, the RHS expression in inequality (22) is an increasing function of q'_u during the interval [0, 1]. Thus an important step is to lower bound q'_u for a given f'_u. The following key lemma can be viewed as a counterpart to Lemma 4.7 of [19]:

Lemma 13 *For each given $f'_u \geq \ln 2/2$, we have that $q'_u \geq f'_u - (1 - \ln 2)$.*

Proof Consider a given u with $f'_u \geq \ln 2/2$. Define $\Delta = f'_u - q'_u$. Thus we have the following.

$$\begin{aligned}
\Delta &= p \cdot \sum_{e=(u,v) \in \partial(u)} \left(f^*_e - \Pr[\mathcal{L}_v(2) = e \wedge \mathcal{L}_v(1) \neq e] \right) \\
&= p \cdot \sum_{e=(u,v) \in \partial(u)} \left(\Pr[\mathcal{L}_v(2) = e] - \Pr[\mathcal{L}_v(2) = e \wedge \mathcal{L}_v(1) \neq e] \right) \quad \text{From } \mathbf{P1} \\
&= p \cdot \sum_{e=(u,v) \in \partial(u)} \left(\Pr[\mathcal{L}_v(2) = e \wedge \mathcal{L}_v(1) = e] \right) \\
&= p \cdot \sum_{e \in \partial(u)} \max\left(2 f^*_e - 1, 0\right) \quad \text{From } \mathbf{P2}
\end{aligned}$$

Thus to lower bound q'_u, we essentially need to maximize Δ. Let $S^* \subseteq \partial(u)$ be the set of edges in $\partial(u)$ with $f^*_e \geq 1/2$, which is called a *contributing* edge. Thus we have

$$\Delta = p \cdot \sum_{e \in \partial(u)} \max\left(2 f^*_e - 1, 0\right) = p \cdot \sum_{e \in S^*} (2 f^*_e - 1) = \sum_{e \in S^*} 2 p f^*_e - p|S^*| \tag{23}$$

Observe that

$$\frac{p}{2}|S^*| \leq \sum_{e \in S^*} f^*_e \cdot p \leq f'_u \Rightarrow |S^*| \leq \frac{2 f'_u}{p} \leq \frac{2}{p} \tag{24}$$

From Constraint (20), we have $\sum_{e \in S^*}(pf_e^*) \leq 1 - \exp(-|S^*|p)$. Substituting this inequality back into Eq. (23), we get

$$\Delta \leq 2 - 2\exp\big(-|S^*| \cdot p\big) - |S^*| \cdot p$$

It is easy to verify that when $f'_u \geq \ln 2/2$, the above expression has a maximum value of $1 - \ln 2$ when $|S^*| \cdot p = \ln 2$. Thus we have that $\Delta \leq 1 - \ln 2$ and $q'_u \geq f'_u - (1 - \ln 2)$. □

Theorem 8 *For unweighted online stochastic matching with integral arrival rates and uniform constant stochastic rewards, there exists an adaptive algorithm which achieves a competitive ratio of at least* 0.702.

Proof We need to prove that $F(f'_u, q'_u)$ defined in (22) has a lower bound of 0.702 for all $f'_u \in [0, 1]$.

Consider the first case when $f'_u \leq \ln 2/2$. It is easy to verify that $F(f'_u, q'_u) \geq F(f'_u, 0) \geq F(\ln 2/2, 0) \sim 0.8$. Consider the second case when $f'_u \geq \ln 2/2$. From Lemma 13, we have $q'_u \geq f'_u - (1 - \ln 2)$. Once again, simple calculations show that

$$F(f'_u, q'_u) \geq F\big(f'_u, f'_u - (1 - \ln 2)\big) \geq F(1, 1 - (1 - \ln 2)) \sim 0.702$$

□

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 $1 - 2\mathsf{e}^{-2}$ due to [13] and the negative result of $1 - \mathsf{e}^{-2}$ due to [19]. We took a step towards closing that gap by showing that an algorithm can achieve $0.7299 > 1 - 2\mathsf{e}^{-2}$ for both the unweighted and vertex-weighted variants with integral arrival rates. In doing so, we made progress on Open Questions 3 and 4 in the online matching and ad allocation survey [18]. 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 $(1 - 1/\mathsf{e})$-competitive algorithm. This showed that the $0.62 < 1 - 1/\mathsf{e}$ bound given in [21] for the adversarial model with stochastic rewards does not extend to the known I.I.D. model.

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 $\{0, 1/k, 2/k, \ldots, \lceil k(1 - 1/\mathsf{e})\rceil/k\}$; 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 in the general case. In this work, we showed how for certain restrictions it is possible to beat $1 - 1/\mathsf{e}$. However, the serious limitation comes from the fact that a polynomial sized LP is insufficient to capture the complexity of the problem.

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.

A Appendix

A.1 Supplementary Materials in Sect. 3 (Edge-Weighted Model)

A.1.1 Proof of Lemma 5

We will prove Lemma 5 using the following three Claims. Recall that we had one kind of large edge, while two kinds of small edges. Hence, the following claim characterizes the performance of each of them.

Claim 9 *For a large edge e, $\mathsf{EW}_1[h]$ (3) with parameter h achieves a competitive ratio of $\mathsf{R}[\mathsf{EW}_1, 2/3] = 0.67529 + (1 - h) * 0.00446$.*

Claim 10 *For a small edge e of type Γ_1, $\mathsf{EW}_1[h]$ (3) achieves a competitive ratio of $\mathsf{R}[\mathsf{EW}_1, 1/3] = 0.751066$, regardless of the value h.*

Claim 11 *For a small edge e of type Γ_2, $\mathsf{EW}_1[h]$ (3) achieves a competitive ratio of $\mathsf{R}[\mathsf{EW}_1, 1/3] = 0.72933 + h * 0.040415$.*

By setting $h = 0.537815$, the two types of small edges have the same ratio and we get that $\mathsf{EW}_1[h]$ achieves $(\mathsf{R}[\mathsf{EW}_1, 2/3], \mathsf{R}[\mathsf{EW}_1, 1/3]) = (0.679417, 0.751066)$. Thus, this proves Lemma 5.

Proof of Claim 9 Consider a large edge $e = (u, v_1)$ in the graph $G_{\mathbf{F}}$. Let $e' = (u, v_2)$ be the other small edge incident to u. Edges e and e' can appear in $[M_1, M_2, M_3]$ in the following three ways.

- α_1: $e \in M_1, e' \in M_2, e \in M_3$.
- α_2: $e' \in M_1, e \in M_2, e \in M_3$.
- α_3: $e \in M_1, e \in M_2, e' \in M_3$.

Notice that the random triple of matchings $[M_1, M_2, M_3]$ is generated by invoking $\mathsf{PM}[\mathbf{F}, 3]$. Since $\mathsf{PM}[\mathbf{F}, 3]$ considers a uniform random permutation we have that α_i will occur with probability 1/3 for $1 \leq i \leq 3$. For α_1 and α_2, we can ignore the second copy of e in M_3 and from Lemma 4 we have

$$\Pr[e \text{ is matched} \mid \alpha_1]=0.580831 \text{ and } \Pr[e \text{ is matched} \mid \alpha_2]=0.148499$$

For α_3, we have

$$\begin{aligned}\Pr[e \text{ is matched} \mid \alpha_3] \geq & \sum_{t=1}^{n} \frac{1}{n}\left(1-\frac{2}{n}\right)^{t-1} + \sum_{t=1}^{n} \frac{1}{n}\left(\frac{t-1}{n}\right)\left(1-\frac{2}{n}\right)^{t-2} \\ & + \sum_{t=1}^{n} \frac{1}{n}\left(\frac{(t-1)(t-2)}{2n^2}\right)\left(1-\frac{2}{n}\right)^{t-3} \\ & + (1-h)\sum_{t=1}^{n} \frac{1}{n}\left(\frac{1}{n^3}\right)\binom{t-1}{3}\left(1-\frac{2}{n}\right)^{t-4} \\ \geq & 0.621246 + (1-h) * 0.00892978\end{aligned}$$

There are four terms in the summation above. The four terms denote the probabilities that v_1 comes for *the first time* at some time $t \in [T]$ and v_2 arrives for 0, 1, 2 and 3 times before t respectively. Note that in the last term when v_2 comes for a third time at some time before t, we need to ensure that v_3 never matches u which occurs with probability $1-h$ as described in EW_1.

Recall that $\mathsf{R}[\mathsf{EW}_1, 2/3]$ denotes the competitive ratio for a large edge. By definition, we have

$$\begin{aligned}\mathsf{R}[\mathsf{EW}_1, 2/3] &= \frac{\Pr[e \text{ is matched}]}{2/3} \\ &= \frac{\frac{1}{3}\sum_{i=1}^{3} \Pr[e \text{ is matched}| \alpha_i]}{2/3} \\ &\geq 0.67529 + (1-h) * 0.00446489\end{aligned}$$

Proof of Claims 10 and 11 Consider a small edge $e=(u,v)$ of type Γ_1. Let e_1 and e_2 be the two other small edges incident to u. For a given triple of matchings $[M_1, M_2, M_3]$, we say e is of type ψ_1 if e appears in M_1 while the other two in the remaining two matchings. Similarly, we define the type ψ_2 and ψ_3 for the case where e appears in M_2 and M_3 respectively. Notice that the probability that e is of type ψ_i, $1 \leq i \leq 3$ is $1/3$. □

Similar to the calculations in the proof of Claim 9, we have $\Pr[e \text{ is matched} \mid \psi_1] \geq 0.571861$, $\Pr[e \text{ is matched} \mid \psi_2] \geq 0.144776$ and $\Pr[e \text{ is matched} \mid \psi_3] \geq 0.0344288$. Therefore we have

$$\Pr[e \text{ is matched}] = \frac{1}{3}\sum_{i=1}^{3} \Pr[e \text{ is matched} \mid \psi_i] \geq \frac{1}{3}\mathsf{R}[\mathsf{EW}_1, 1/3]$$

where $\mathsf{R}[\mathsf{EW}_1, 1/3] = 0.751066$.

Consider a small edge $e=(u,v)$ of type Γ_2, we define type $\beta_i, 1 \leq i \leq 3$, if e appears in M_i while the large edge e' incident to u appears in the

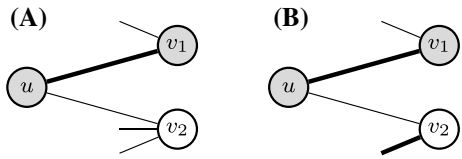

Fig. 5 Diagram of configurations for a large edge $e = (u, v_1)$. Thin and Thick lines represent small and large edges respectively

remaining two matchings. Similarly, we have $\Pr[e \text{ is matched} | \psi_1] \geq 0.580831$, $\Pr[e \text{ is matched} | \psi_2] \geq 0.148499$ and $\Pr[e \text{ is matched} | \psi_3] \geq h * 0.0404154$.

Hence, the ratio for a small edge of type Γ_2 is $\mathsf{R}[\mathsf{EW}_1, 1/3] = 0.72933 + h * 0.0404154$.

A.1.2 Proof of Lemma 6

We will prove Lemma 6 using the following two Claims.

Claim 12 *For a large edge e, $\mathsf{EW}_2[y_1, y_2]$ (5) achieves a competitive ratio of*

$$\mathsf{R}[\mathsf{EW}_2, 2/3] = \min\left(0.948183 - 0.099895y_1 - 0.025646y_2, 0.871245\right)$$

Claim 13 *For a small edge e, $\mathsf{EW}_2[y_1, y_2]$ (5) achieves a competitive ratio of $\mathsf{R}[\mathsf{EW}_2, 1/3] = 0.4455$, when $y_1 = 0.687, y_2 = 1$.*

Therefore, by setting $y_1 = 0.687, y_2 = 1$ we get that $\mathsf{R}[\mathsf{EW}_2, 2/3] = 0.8539$ and $\mathsf{R}[\mathsf{EW}_2, 1/3] = 0.4455$, which proves Lemma 6.

Proof of Claim 12 Figure 5 shows the two possible configurations for a large edge.

Consider a large edge $e = (u, v_1)$ with the configuration (A). From $\mathsf{PM}^*[\mathbf{F}, 2][y_1, y_2]$, we know that e will always be in M_1 while $e' = (u, v_2)$ will be in M_1 and M_2 with probability $y_1/3$ and $y_2/3$ respectively.

We now have the following cases:

- α_1: $e \in M_1$ and $e' \in M_1$. This happens with probability $y_1/3$. In this case, e is matched if v_1 comes for the first time at some time $t \in [T]$ and v_2 never comes before t. Thus,

$$\Pr[e \text{ is matched} \mid \alpha_1] = \sum_{t=1}^{n} \frac{1}{n}\left(1 - \frac{2}{n}\right)^{t-1} \geq 0.432332$$

- α_2: $e \in M_1$ and $e' \in M_2$. This happens with probability $y_2/3$. In this case, e is matched if v_1 comes for the first time at some time $t \in [T]$ and v_2 comes at most once before t. Note that this case is essentially the same as P_1 described in Lemma 4. Thus, we have

$$\Pr[e \text{ is matched} \mid \alpha_2] \geq 0.580831$$

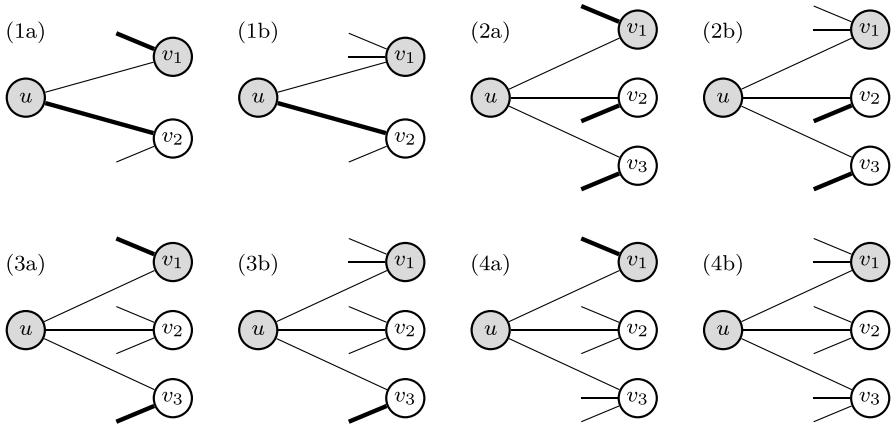

Fig. 6 Diagram of configurations for a small edge $e = (u, v_1)$. Thin and Thick lines represent small and large edges respectively

- α_3: $e \in M_1$ and $e' \notin M_1, e' \notin M_2$. This happens with probability $(1 - y_1/3 - y_2/3)$. In this case, e is matched if v_1 comes at least once. Thus, $\Pr[e \text{ is matched} | \alpha_1] = 1 - 1/\mathsf{e} \geq 0.632121$.

Therefore we have

$$\begin{aligned}\Pr[e \text{ is matched}] =& \Big(\frac{y_1}{3}\Pr[e \text{ is matched} \mid \alpha_1] + \frac{y_2}{3}\Pr[e \text{ is matched} \mid \alpha_2] \\ &+ (1 - \frac{y_1}{3} - \frac{y_2}{3})\Pr[e \text{ is matched} \mid \alpha_3]\Big) \\ \geq& \frac{2}{3}(0.948183 - 0.099895y_1 - 0.025646y_2)\end{aligned}$$

Consider the configuration (B). From $\mathsf{PM}^*[\mathbf{F}, 2][y_1, y_2]$, we know that e will always be in M_1 and $e' = (u, v_2)$ will always be in M_2. Thus we have

$$\Pr[e \text{ is matched}] = \Pr[e \text{ is matched} \mid \alpha_2] = \frac{2}{3} * 0.871245$$

Hence, this completes the proof of Claim 12.

Proof of Claim 13 Figure 6 shows all possible configurations for a small edge.

Similar to the proof of Claim 12, we will do a case-by-case analysis on the various configurations. Let $e_i = (u, v_i)$ for $1 \leq i \leq 3$ and $\mathcal{E}$ be the event that e_1 gets matched. For a given e_i, denote $e_i \in M_0$ if $e_i \notin M_1, e_i \notin M_2$.

- $(1a)$. Observe that $e_1 \in M_2$ and $e_2 \in M_1$. Thus we have $\Pr[\mathcal{E}] = \frac{1}{3} * 0.44550$.
- $(1b)$. Observe that we have two cases: $\{\alpha_1 : e_2 \in M_1, e_1 \in M_1\}$ and $\{\alpha_2 : e_2 \in M_1, e_1 \in M_2\}$. Case α_1 happens with probability $y_1/3$ and the con-

ditional probability is $\Pr[\mathcal{E}\,|\alpha_1] = 0.432332$. Case α_2 happens with probability $y_2/3$ and the conditional is $\Pr[\mathcal{E}\,|\alpha_2] = 0.148499$. Thus we have

$$\Pr[\mathcal{E}] = y_1/3 * \Pr[\mathcal{E}\,|\alpha_1] + y_2/3 * \Pr[\mathcal{E}\,|\alpha_2] \geq \frac{1}{3}(0.432332y_1 + 0.148499y_2)$$

- $(2a)$. Observe that $e_1 \in M_2, e_2 \in M_2, e_3 \in M_2$. $\Pr[\mathcal{E}] = \frac{1}{3} * 0.601704$
- $(2b)$. Observe that we have two cases: $\{\alpha_1 : e_1 \in M_1, e_2 \in M_2, e_3 \in M_2\}$ and $\{\alpha_2 : e_1 \in M_2, e_2 \in M_2, e_3 \in M_2\}$. Case α_1 happens with probability $y_1/3$ and the conditional is $\Pr[\mathcal{E}\,|\alpha_1] = 0.537432$. Case α_2 happens with probability $y_2/3$ and conditional is $\Pr[\mathcal{E}\,|\alpha_2] = 0.200568$. Thus we have

$$\Pr[\mathcal{E}] = y_1/3 * \Pr[\mathcal{E}\,|\alpha_1] + y_2/3 * \Pr[\mathcal{E}\,|\alpha_2] \geq \frac{1}{3}(0.537432y_1 + 0.200568y_2)$$

- $(3a)$. Observe that we have three cases: $\{\alpha_1 : e_1 \in M_2, e_2 \in M_1, e_3 \in M_2\}$, $\{\alpha_2 : e_1 \in M_2, e_2 \in M_2, e_3 \in M_2\}$ and $\{\alpha_2 : e_1 \in M_2, e_2 \in M_0, e_3 \in M_2\}$. Case α_1 happens with probability $y_1/3$ and conditional is $\Pr[\mathcal{E}\,|\alpha_1] = 0.13171$. Case α_2 happens with probability $y_2/3$ and conditional is $\Pr[\mathcal{E}\,|\alpha_2] = 0.200568$. Case α_3 happens with probability $(1 - y_1/3 - y_2/3)$ and conditional is $\Pr[\mathcal{E}\,|\alpha_3] = 0.22933$.

  Similarly, we have

$$\begin{aligned}\Pr[\mathcal{E}] &= y_1/3 * \Pr[\mathcal{E}\,|\alpha_1] + y_2/3 * \Pr[\mathcal{E}\,|\alpha_2] + (1 - y_1/3 - y_2/3) * \Pr[\mathcal{E}\,|\alpha_3] \\ &\geq \frac{1}{3}(0.13171y_1 + 0.200568y_2 + (3 - y_1 - y_2)0.22933)\end{aligned}$$

- $(3b)$. Observe that we have six cases.
  - $\alpha_1 : e_1 \in M_1, e_2 \in M_1, e_3 \in M_2$. $\Pr[\alpha_1] = y_1^2/9$ and $\Pr[\mathcal{E}\,|\alpha_1] = 0.4057$.
  - $\alpha_2 : e_1 \in M_1, e_2 \in M_2, e_3 \in M_2$. $\Pr[\alpha_2] = y_1y_2/9$ and $\Pr[\mathcal{E}\,|\alpha_2] = 0.5374$.
  - $\alpha_3 : e_1 \in M_1, e_2 \in M_0, e_3 \in M_2$. $\Pr[\alpha_3] = y_1/3(1 - y_1/3 - y_2/3)$ and $\Pr[\mathcal{E}\,|\alpha_3] = 0.58083$.
  - $\alpha_4 : e_1 \in M_2, e_2 \in M_1, e_3 \in M_2$. $\Pr[\alpha_4] = y_1y_2/9, \Pr[\mathcal{E}\,|\alpha_4] = 0.1317$.
  - $\alpha_5 : e_1 \in M_2, e_2 \in M_2, e_3 \in M_2$. $\Pr[\alpha_5] = y_2^2/9, \Pr[\mathcal{E}\,|\alpha_5] = 0.2006$.
  - $\alpha_6 : e_1 \in M_2, e_2 \in M_0, e_3 \in M_2$. $\Pr[\alpha_6] = y_2/3(1 - y_1/3 - y_2/3)/3$ and $\Pr[\mathcal{E}\,|\alpha_6] = 0.22933$.

  Therefore we have

$$\begin{aligned}\Pr[\mathcal{E}] \geq \frac{1}{3}\Big(&0.135241y_1^2 + 0.223033y_1y_2 + 0.066856y_2^2 \\ &+ y_1(3 - y_1 - y_2)0.193610 + y_2(3 - y_1 - y_2)0.076443\Big)\end{aligned}$$

- $(4a)$. Observe that we have following six cases.
  - $\alpha_1 : e_1 \in M_2, e_2 \in M_1, e_3 \in M_1$. $\Pr[\alpha_1] = y_1^2/9$ and $\Pr[\mathcal{E}\,|\alpha_1] = 0.08898$.
  - $\alpha_2 : e_1 \in M_2, e_2 \in M_2, e_3 \in M_2$. $\Pr[\alpha_2] = y_2^2/9$ and $\Pr[\mathcal{E}\,|\alpha_2] = 0.2006$.

- $\alpha_3 : e_1 \in M_2, e_2 \in M_0, e_3 \in M_0$. $\Pr[\alpha_3] = (1 - y_1/3 - y_1/3)^2$, and $\Pr[\mathcal{E} \mid \alpha_3] = 0.2642$.
- α_4: $e_1 \in M_2$ while either $e_2 \in M_1, e_3 \in M_2$ or $e_2 \in M_2, e_3 \in M_1$. $\Pr[\alpha_2] = 2y_1y_2/9$ and $\Pr[\mathcal{E} \mid \alpha_4] = 0.1317$.
- α_5: $e_1 \in M_2$ while either $e_2 \in M_1, e_3 \in M_0$ or $e_2 \in M_0, e_3 \in M_1$. $\Pr[\alpha_5] = 2y_1/3(1 - y_1/3 - y_2/3)$ and $\Pr[\mathcal{E} \mid \alpha_5] = 0.14849$.
- α_6: $e_1 \in M_2$ while either $e_2 \in M_2, e_3 \in M_0$ or $e_2 \in M_0, e_3 \in M_2$. $\Pr[\alpha_5] = 2y_2/3(1 - y_1/3 - y_2/3)$ and $\Pr[\mathcal{E} \mid \alpha_6] = 0.22933$.

  Therefore we have

$$\Pr[\mathcal{E}] \geq \frac{1}{3}\Big(0.029661y_1^2 + 2 * 0.043903y_1y_2 + 0.066856y_2^2 + 2y_1(3 - y_1 - y_2)0.0494997 + 2y_2(3 - y_1 - y_2)(0.076443) + (3 - y_1 - y_2)^2 0.0880803\Big)$$

- (4*b*). Observe that in this configuration, we have additional six cases to the ones discussed in (4*a*). Let α_i be the cases defined in (4*a*) for each $1 \leq i \leq 6$. Notice that each $\Pr[\alpha_i]$ has a multiplicative factor of $y_2/3$. Now, consider the six new cases.
  - $\beta_1 : e_1 \in M_1, e_2 \in M_1, e_3 \in M_1$. $\Pr[\alpha_1] = y_1^3/27$ and $\Pr[\mathcal{E} \mid \alpha_1] = 0.3167$.
  - $\beta_2 : e_1 \in M_1, e_2 \in M_2, e_3 \in M_2$. $\Pr[\alpha_2] = y_1y_2^2/27$ and $\Pr[\mathcal{E} \mid \alpha_2] = 0.5374$.
  - $\beta_3 : e_1 \in M_1, e_2 \in M_0, e_3 \in M_0$. $\Pr[\alpha_3] = y_1/3 * (1 - y_1/3 - y_2/3)^2$ and $\Pr[\mathcal{E} \mid \alpha_3] = 0.632$.
  - β_4: $e_1 \in M_1$ and either $e_2 \in M_1, e_3 \in M_2$ or $e_2 \in M_2, e_3 \in M_1$. $\Pr[\alpha_2] = 2y_1^2y_2/27$ and $\Pr[\mathcal{E} \mid \alpha_4] = 0.4057$.
  - β_5: $e_1 \in M_1$ and either $e_2 \in M_1, e_3 \in M_0$ or $e_2 \in M_0, e_3 \in M_1$. $\Pr[\alpha_5] = 2y_1^2/9 * (1 - y_1/3 - y_2/3)$ and $\Pr[\mathcal{E} \mid \alpha_5] = 0.4323$.
  - β_6: $e_1 \in M_1$ and either $e_2 \in M_2, e_3 \in M_0$ or $e_2 \in M_0, e_3 \in M_2$. $\Pr[\alpha_5] = 2y_1y_2/9 * (1 - y_1/3 - y_2/3)$ and $\Pr[\mathcal{E} \mid \alpha_6] = 0.58083$.

    Hence, we have

$$\Pr[\mathcal{E}] \geq \frac{1}{3}\Big(0.632y_1 - 0.133133y_1^2 + 0.0093y_1^3 + 0.264241y_2 - 0.11127y_1y_2 + 0.01170y_1^2y_2 - 0.0232746y_2^2 + 0.00488y_1y_2^2 + 0.00068y_2^3\Big)$$

Setting $y_1 = 0.687$, $y_2 = 1$, we get that the competitive ratio for a small edge is 0.44550. The bottleneck cases are configurations (1*a*) and (1*b*).

A.2 Supplemental Materials in Sect. 4

A.2.1 Proof of Lemma 10 (Vertex-Weighted and Unweighted)

When $H_u = 1$ and u is in the cycle C_1, [13] show that the competitive ratio of u is $1 - 2e^{-2}$. Hence, for the remaining cases, we use the following Claims.

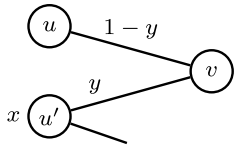

Fig. 7 Case 1 in calculation of $\Pr[A^v_{u,2}]$

Claim 14 *If $H_u = 1$ and u is not in C_1, then we have* $\mathsf{R}[\mathsf{RLA}, 1] \geq 0.735622$.

Claim 15 $\mathsf{R}[\mathsf{RLA}, 2/3] \geq 0.7870$.

Claim 16 $\mathsf{R}[\mathsf{RLA}, 1/3] \geq 0.8107$.

Recall that $A_{u,1}$ is the event that among the n random lists, there exists a list starting with u and $A^v_{u,2}$ is the event that among the n lists, there exist successive lists such that (1) all start with some u' which are different from u but are neighbors of v; and (2) they ensure u will be matched.

Notice that A_u is the probability that u gets matched in $\mathsf{RLA}[\mathbf{H}']$. For each u, we compute $\Pr[A_{u,1}]$ and $\Pr[A^v_{u,2}]$ for all possibilities of $v \sim u$ and using Lemma 9 we get A_u. We first discuss two different ways to calculate $\Pr[A^v_{u,2}]$ when v has different neighboring structures.

Two ways to compute the value $\Pr[A^v_{u,2}]$.

1. *Case 1 v has two neighbors.* Consider the case when v has two neighbors as shown in Fig. 7. In this case we choose a slightly direct approach to computing $\Pr[A^v_{u,2}]$.

   Assume v has two neighbors u and u' as shown in Fig. 7. After modifications, assume $H'_{(u',v)} = y$, $H'_{(u,v)} = 1 - y$ and $H'_{u'} = x$. Thus, the second certificate event $A^v_{u,2}$ corresponds to the event (1) a list starting with u' comes at some time $1 \leq i < n$; (2) the list $\mathcal{R}_v = (u', u)$ comes for a second time at some j with $i < j \leq n$. Note that the arrival rate of a list starting with u' is $H'_{u'} = x/n$ and the rate of list $\mathcal{R}_v = (u', u)$ is y/n. Therefore we have

$$\Pr[A^v_{u,2}] = \sum_{i=1}^{n-1} \left(x/n(1 - x/n)^{(i-1)}(1 - (1 - y/n)^{(n-i)} \right) \tag{25}$$

$$\sim \frac{x - \mathsf{e}^{-y}x + (-1 + \mathsf{e}^{-x})y}{x - y} \quad (\text{if } x \neq y) \tag{26}$$

$$\sim 1 - \mathsf{e}^{-x}(1 + x) \quad (\text{if } x = y) \tag{27}$$

2. *Case 2 v has three neighbors.* Consider the case when v has three neighbors as shown in Fig. 8. In this case, we approximate the value $\Pr[A^v_{u,2}]$ using the Markov Chain method, similar to [13].

   Focus on the case shown in Fig. 8 where v has three neighbors u, u_1 and u_2 with $H_u = 1, H_{u_1} = 1/3$ and $H_{u_2} = 2/3$. Recall that after modifications, we have $H'_{(u_1,v)} = b = 0.1, H'_{(u_2,v)} = c = 0.15$ and $H'_{(u,v)} = d = 0.75$. We simulate the process

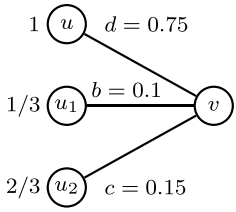

Fig. 8 Case 2 in calculation of $\Pr[A^v_{u,2}]$

of u getting matched resulting from several successive random lists starting from either u_1 or u_2 by an n-step Markov Chain as follows. We have 5 states: $s_1 = (0,0,0), s_2 = (0,1,0), s_3 = (0,0,1), s_4 = (0,1,1)$ and $s_5 = (1,*,*)$ and the three numbers in each triple correspond to u, u_1 and u_2 being matched(or not) respectively. State s_5 corresponds to u being matched; the matched status of u_1 and u_2 is irrelevant. The chain initially starts in s_1 and the probability of being in state s_5 after n steps gives an approximation to $\Pr[A^v_{u,2}]$. The one-step transition probability matrix M is shown as follows.

$$
\begin{aligned}
&M_{1,2} = \frac{b}{n}, M_{1,3} = \frac{c+1/3}{n}, M_{1,1} = 1 - M_{1,2} - M_{1,3}\\
&M_{2,4} = \frac{c+1/3}{n} + \frac{bc}{(c+d)n}, M_{2,5} = \frac{bd}{(c+d)n},\\
&M_{2,3} = 1 - M_{2,4} - M_{2,5}\\
&M_{3,4} = \frac{b}{n} + \frac{cb}{(b+d)n}, M_{3,5} = \frac{cd}{(b+d)n}\\
&M_{3,3} = 1 - M_{3,4} - M_{3,5}\\
&M_{4,5} = \frac{b+c}{n}, M_{4,4} = 1 - M_{4,5}\\
&M_{5,5} = 1\\
&M_{i,j} = 0 \text{ for all other } i,j
\end{aligned}
$$

Notice that $M_{1,3} = \frac{c+1/3}{n}$ and not $\frac{2}{3n}$ since after modifications, the arrival rate of a list starting with u_2 decreases correspondingly.

Let us now prove the three Claims 14, 15 and 16. Here we give the explicit analysis for the case when $H_u = 1$. For the remaining cases, similar methods can be applied. Hence, we omit the analysis and only present the related computational results which leads to the conclusion.

Proof of Claim 14 Notice that u is not in the cycle C_1 and thus Lemma 9 can be used. Figure 9 describes all possible cases when a node $u \in U$ has $H_u = 1$. (We ignore all those cases when $H_u < 1$, since they will not appear in the WS.)

Let v_1 and v_2 be the two neighbors of u with $H_{(u,v_1)} = 2/3$ and $H_{(u,v_2)} = 1/3$. In total, there are 4×10 combinations, where v_1 is chosen from some $\alpha_i, 1 \le i \le 4$ and v_2 is chosen from some $\beta_i, 1 \le i \le 9$. For $H_u = 1$, we need to find the worst

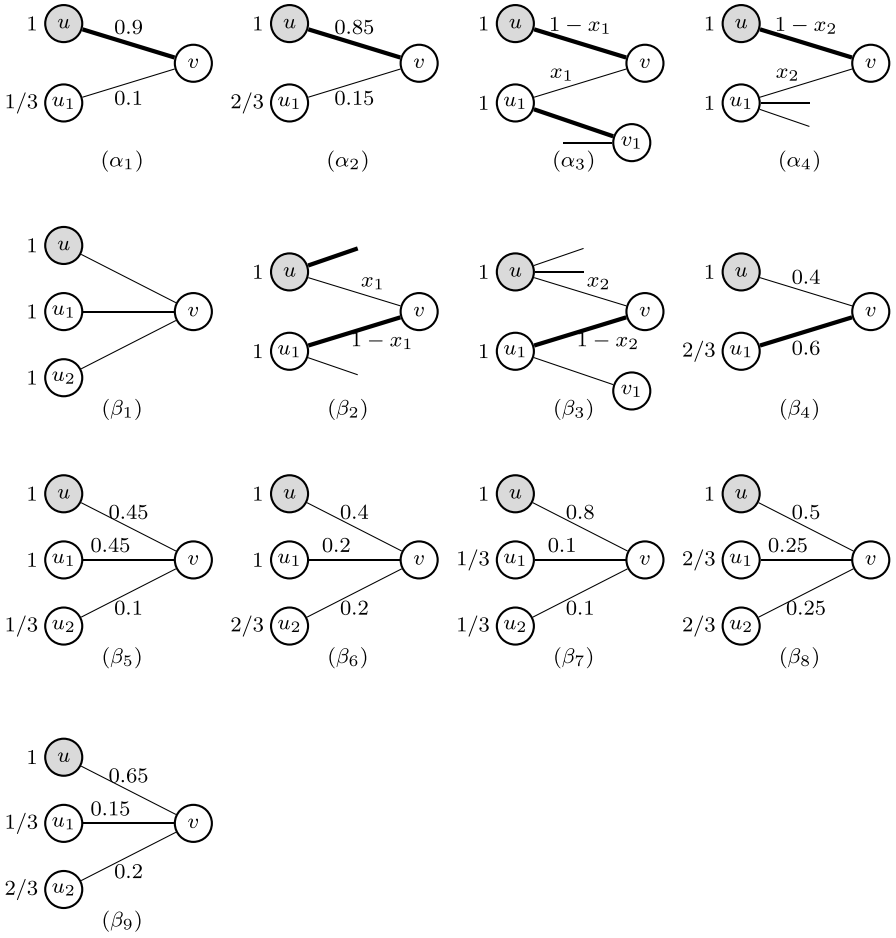

Fig. 9 Vertex-weighted $H_u = 1$ cases. The value assigned to each edge represents the value after the second modification. No value indicates no modification. Here, $x_1 = 0.2744$ and $x_2 = 0.15877$

combination among these such that the value A_u is minimized. We can find this WS using the Lemma 9.

For each type of α_i, β_j, we compute the values it will contribute to the term $(1 - A_{u,1}) \prod_{v \sim u}(1 - \Pr[A^v_{u,2}])$. For example, assume v_1 is of type α_1, denoted by $v_1(\alpha_1)$. It contributes $\mathsf{e}^{-0.9}$ to the term $(1 - A_{u,1})$ and $(1 - \Pr[A^{v_1}_{u,2}])$ to $\prod_{v \sim u}(1 - \Pr[A^v_{u,2}])$, thus the total value it contributes is $\gamma(v_1, \alpha_1) = \mathsf{e}^{-0.9}(1 - \Pr[A^{v_1}_{u,2}])$. Similarly, we can compute all $\gamma(v_1, \alpha_i)$ and $\gamma(v_2, \beta_j)$. Let $i^* = \arg\max_i \gamma(v_1, \alpha_i)$ and $j^* = \arg\max_j \gamma(v_2, \beta_j)$. The WS is for the combination $\{v_1(\alpha_{i^*}), v_2(\beta_{j^*})\}$ and the resulting value of A_u and R[RLA, 1] is as follows:

$$A_u = 1 - \gamma(v_1, \alpha_{i^*})\gamma(v_2, \beta_{j^*})$$
$$\mathsf{R}[\mathsf{RLA}, 1] = A_u / H_u = A_u$$

Here is a list of $\gamma(v_1, \alpha_i)$ and $\gamma(v_2, \beta_j)$, for each $1 \le i \le 4$ and $1 \le j \le 9$.

- α_1: We have $\Pr[A^v_{u,2}] = 1 - \mathsf{e}^{-0.1} * 1.1$ and $\gamma(v, \alpha_1) = \mathsf{e}^{-0.1} * 1.1 * \mathsf{e}^{-0.9} = 0.404667$.
- α_2: $\Pr[A^v_{u,2}] \geq 1 - \mathsf{e}^{-0.15} * 1.15$ and $\gamma(v, \alpha_2) \leq 0.423$.
  Notice that after modifications, $H'_{u_1} \geq 0.15$. Hence, we use this and Eq. (25) to compute the lower bound of $\Pr[A^v_{u,2}]$.
- α_3: $\Pr[A^v_{u,2}] \geq 0.0916792$ and $\gamma(v, \alpha_3) \leq 0.439667$.
  Notice that for any large edge e incident to a node u with $H_u = 1$ (before modification), we have after modification, $H'_e \geq 1 - 0.2744 = 0.7256$. Thus we have $H'_{(u_1,v_1)} \geq 0.7256$ and $H'_{u_1} \geq 1$. From Eq. (25), we get $\Pr[A^v_{u,2}] \geq 0.0916792$.
- α_4: $\Pr[A^v_{u,2}] \geq 0.0307466$ and $\gamma(v, \alpha_4) \leq 0.417923$.
  Notice that for any small edge e incident to a node u with $H_u = 1$ (before modification), we have after modification, $H'_e \geq 0.15877$. Thus, we have $H'_{u_1} \geq 3 * 0.15877$.
- β_1: $\Pr[A^v_{u,2}] = 0.1608$ and $\gamma(v, \beta_1) = 0.601313$.
- β_2: $\Pr[A^v_{u,2}] \geq 0.208812$ and $\gamma(v, \beta_2) \leq 0.601313$.
  After modifications, we have $H'_{(u_1,v_1)} \geq 0.2744$ and thus we get $H'_{u_1} \geq 1$.
- β_3: $\Pr[A^v_{u,2}] \geq 0.251611$ and $\gamma(v, \beta_2) \leq 0.63852$.
  After modifications, we have $H'_{(u_1,v_1)} \geq 0.2744$ and thus we get $H'_{u_1} \geq 1 - 0.15877 + 0.2744$.
- β_4: $\Pr[A^v_{u,2}] = 0.121901$ and $\gamma(v, \beta_4) = 0.588607$.
- β_5: $\Pr[A^v_{u,2}] = 0.1346$ and $\gamma(v, \beta_5) = 0.551803$.
- β_6: $\Pr[A^v_{u,2}] \geq 0.1140$ and $\gamma(v, \beta_6) \leq 0.593904$.
- β_7: $\Pr[A^v_{u,2}] = 0.0084$ and $\gamma(v, \beta_7) = 0.4455$.
- β_8: $\Pr[A^v_{u,2}] \geq 0.0397$ and $\gamma(v, \beta_8) \leq 0.582451$.
- β_9: $\Pr[A^v_{u,2}] \geq 0.0230$ and $\gamma(v, \beta_9) \leq 0.510039$.

Using the computed values above, let us compute the ratio of a node u with $H_u = 1$.

- If u has three neighbors, then the WS configuration is when each of the three neighbors of u is of type β_3. This is because, the value of $\gamma(v, \beta_3)$ is the largest. The resultant ratio is 0.73967.
- If u has two neighbors, then the WS configuration is when one of the neighbor is of type β_1 (or β_2) and the other is of type α_3. The resultant ratio is 0.735622.

Proof of Claim 15 The proof is similar to that of Claim 14. The Fig. 10 shows all possible configurations of a node u with $H_u = 2/3$. Note that the WS cannot have $F(v) < 1$ and hence we omit them here. For a neighbor v of u, if $H_{(u,v)} = 2/3$, then v is in one of α_i, $1 \leq i \leq 3$; if $H_{(u,v)} = 1/3$, then v is in one of β_i, $1 \leq i \leq 8$. We now list the values $\gamma(v, \alpha_i)$ and $\gamma(v, \beta_j)$, for each $1 \leq i \leq 3$ and $1 \leq j \leq 8$.

- α_1: We have $\Pr[A^v_{u,2}] = 1 - \mathsf{e}^{-0.25} * 1.25$ and $\gamma(v, \alpha_1) = \mathsf{e}^{-0.25} * 1.25 * \mathsf{e}^{-0.75} = 0.459849$.
- α_2: We have $\Pr[A^v_{u,2}] \geq 0.0528016$ and $\gamma(v, \alpha_1) \leq 0.470365$.
- α_3. We have $\Pr[A^v_{u,2}] \geq 0.13398$ and $\gamma(v, \alpha_3) \leq 0.475282$.

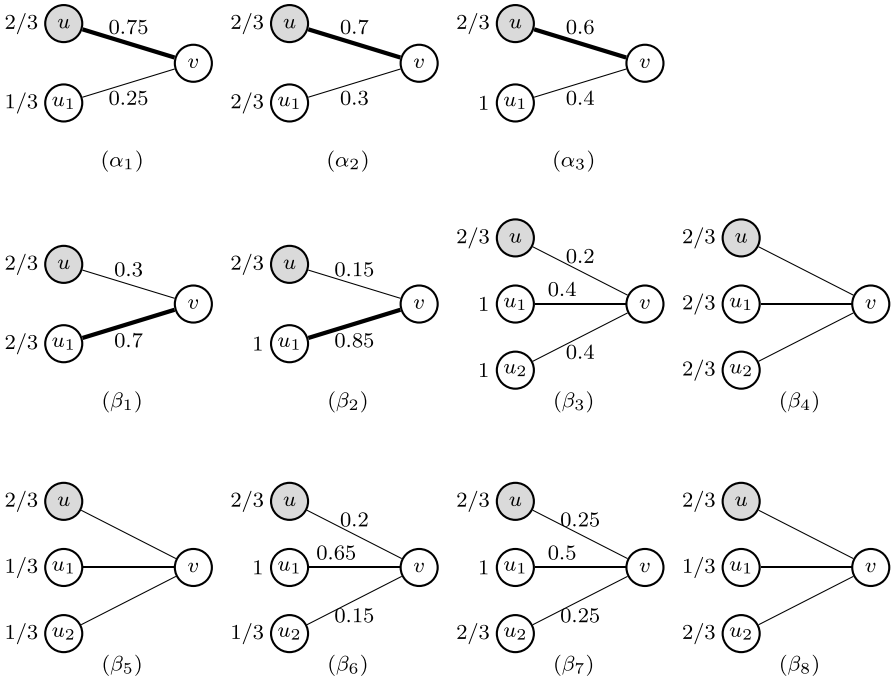

Fig. 10 Vertex-weighted $H_u = 2/3$ cases. The value assigned to each edge represents the value after the second modification. No value indicates no modification

- β_1: We have $\Pr[A^v_{u,2}] = 1 - \mathsf{e}^{-0.7} * 1.7$ and $\gamma(v, \beta_1) = 0.625395$.
- β_2: We have $\Pr[A^v_{u,2}] \geq 0.226356$ and $\gamma(v, \beta_2) \leq 0.665882$.
- β_3: We have $\Pr[A^v_{u,2}] \geq 0.1819$ and $\gamma(v, \beta_3) \leq 0.669804$.
- β_4: We have $\Pr[A^v_{u,2}] \geq 0.1130$ and $\gamma(v, \beta_4) \leq 0.635563$.
- β_5: We have $\Pr[A^v_{u,2}] \geq 0.0587$ and $\gamma(v, \beta_5) \leq 0.674471$.
- β_6: We have $\Pr[A^v_{u,2}] \geq 0.1688$ and $\gamma(v, \beta_6) \leq 0.680529$.
- β_7: We have $\Pr[A^v_{u,2}] \geq 0.1318$ and $\gamma(v, \beta_7) \leq 0.676155$.
- β_8: We have $\Pr[A^v_{u,2}] \geq 0.0587$ and $\gamma(v, \beta_8) \leq 0.674471$.

Hence, the WS structure is when u is such that $H_u = 2/3$ and has one neighbor of type α_3. The resultant ratio is 0.7870.

Proof of Claim 16 The Fig. 11 shows the possible configurations of a node u with $H_u = 1/3$. Again, we omit those cases where $H_v < 1$.

We now list the values $\gamma(v, \alpha_i)$, for each $1 \leq i \leq 8$.

- α_1: We have $\Pr[A^v_{u,2}] = 1 - \mathsf{e}^{-0.75} * 1.75$ and $\gamma(v, \alpha_1) = 0.643789$.
- α_2: We have $\Pr[A^v_{u,2}] \geq 0.282256$ and $\gamma(v, \alpha_2) \leq 0.649443$.
- α_3: We have $\Pr[A^v_{u,2}] \geq 0.1935$ and $\gamma(v, \alpha_3) \leq 0.729751$.

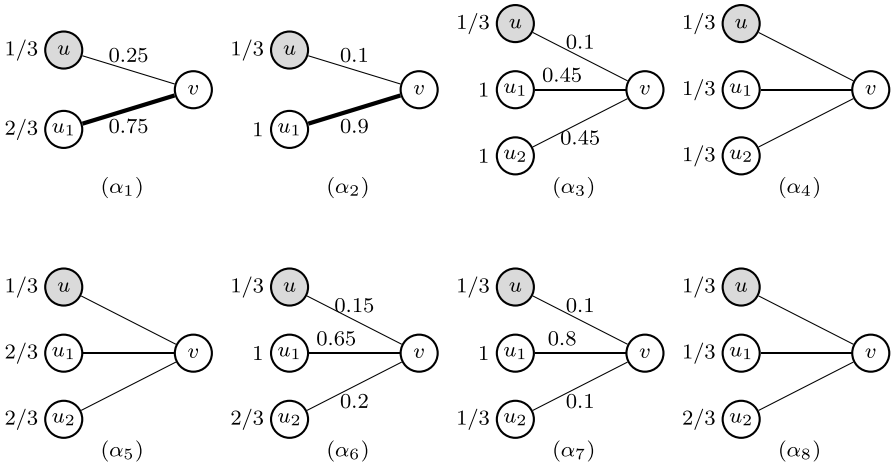

Fig. 11 Vertex-weighted $H_u = 1/3$ cases. The value assigned to each edge represents the value after the second modification. No value indicates no modification

- α_4: We have $\Pr[A^v_{u,2}] \geq 0.0587$ and $\gamma(v, \alpha_4) \leq 0.674471$.
- α_5: $\gamma(v, \alpha_5) \leq 0.674471$.
- α_6: We have $\Pr[A^v_{u,2}] \geq 0.1546$ and $\gamma(v, \alpha_6) \leq 0.727643$.
- α_7: We have $\Pr[A^v_{u,2}] \geq 0.1938$ and $\gamma(v, \alpha_7) \leq 0.72948$.
- α_8: $\gamma(v, \alpha_8) \leq 0.674471$.

Hence, the WS for node u with $H_u = 1/3$ is when u has one neighbor of type α_3. The resultant ratio is 0.8107.

References

1. Aggarwal, G., Goel, G., Karande, C., Mehta, A.: Online vertex-weighted bipartite matching and single-bid budgeted allocations. In: Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 1253–1264. SIAM (2011)
2. Alaei, S., Hajiaghayi, M.T., Liaghat, V.: Online prophet-inequality matching with applications to ad allocation. In: Proceedings of the 13th ACM Conference on Electronic Commerce, pp. 18–35. ACM (2012)
3. Alaei, S., Hajiaghayi, M.T., Liaghat, V.: The online stochastic generalized assignment problem. In: Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, pp. 11–25. Springer (2013)
4. Assadi, S., Khanna, S., Li, Y.: The stochastic matching problem with (very) few queries. In: Proceedings of the 2016 ACM Conference on Economics and Computation, pp. 43–60. ACM (2016)
5. Brubach, B., Sankararaman, K.A., Srinivasan, A., Xu, P.: Attenuate locally, win globally: an attenuation-based framework for online stochastic matching with timeouts. In: Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems, AAMAS '17, pp. 1223–1231. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC (2017)

6. Devanur, N.R., Hayes, T.P.: The adwords problem: online keyword matching with budgeted bidders under random permutations. In: Proceedings of the 10th ACM Conference on Electronic Commerce, pp. 71–78. ACM (2009)
7. Devanur, N.R., Jain, K., Sivan, B., Wilkens, C.A.: Near optimal online algorithms and fast approximation algorithms for resource allocation problems. In: Proceedings of the 12th ACM Conference on Electronic Commerce, pp. 29–38. ACM 2011
8. Devanur, N.R., Sivan, B., Azar, Y.: Asymptotically optimal algorithm for stochastic adwords. In: Proceedings of the 13th ACM Conference on Electronic Commerce, pp. 388–404. ACM (2012)
9. Feldman, J., Korula, N., Mirrokni, V., Muthukrishnan, S., Pál, M.: Online ad assignment with free disposal. In: Internet and Network Economics, pp. 374–385. Springer (2009)
10. Feldman, J., Mehta, A., Mirrokni, V., Muthukrishnan, S.: Online stochastic matching: beating 1-1/e. In: Foundations of Computer Science (FOCS), pp. 117–126. IEEE (2009)
11. Gandhi, R., Khuller, S., Parthasarathy, S., Srinivasan, A.: Dependent rounding and its applications to approximation algorithms. J. ACM (JACM) **53**(3), 324–360 (2006)
12. Haeupler, Bernhard, Mirrokni, Vahab S., Zadimoghaddam, Morteza: Online stochastic weighted matching: improved approximation algorithms. Internet and Network Economics, Volume 7090 of Lecture Notes in Computer Science, pp. 170–181. Springer, Berlin (2011)
13. Jaillet, P., Lu, X.: Online stochastic matching: new algorithms with better bounds. Math. Oper. Res. **39**(3), 624–646 (2013)
14. Korula, N., Pál, M.: Algorithms for secretary problems on graphs and hypergraphs. In: Automata, Languages and Programming, pp. 508–520. Springer (2009)
15. Kesselheim, T., Radke, K., Tönnis, A., Vöcking, B.: An optimal online algorithm for weighted bipartite matching and extensions to combinatorial auctions. In: European Symposium on Algorithms (ESA), pp. 589–600. Springer (2013)
16. Karp, R.M., Vazirani, U.V., Vazirani, V.V.: An optimal algorithm for on-line bipartite matching. In: Proceedings of the Twenty-Second Annual ACM Symposium on Theory of Computing, pp. 352–358. ACM (1990)
17. Lee, Y.T., Sidford, A.: Efficient inverse maintenance and faster algorithms for linear programming. In: 2015 IEEE 56th Annual Symposium on Foundations of Computer Science (FOCS), pp. 230–249. IEEE (2015)
18. Mehta, A.: Online matching and ad allocation. Found. Trends Theor. Comput. Sci. **8**(4), 265–368 (2012)
19. Manshadi, V.H., Gharan, S.O., Saberi, A.: Online stochastic matching: online actions based on offline statistics. Math. Oper. Res. **37**(4), 559–573 (2012)
20. Mehta, A., Panigrahi, D.: Online matching with stochastic rewards. In: Foundations of Computer Science (FOCS), pp. 728–737. IEEE (2012)
21. Mehta, A., Waggoner, B., Zadimoghaddam, M.: Online stochastic matching with unequal probabilities. In: Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms. SIAM (2015)
22. Mahdian, M., Yan, Q.: 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, pp. 597–606. ACM (2011)

Affiliations

Brian Brubach[1] · Karthik Abinav Sankararaman[1] · Aravind Srinivasan[1] · Pan Xu[2]

✉ Karthik Abinav Sankararaman
karthikabinavs@gmail.com

Brian Brubach
bbrubach@cs.umd.edu

Aravind Srinivasan
srin@cs.umd.edu

Pan Xu
pxu@njit.edu

[1] University of Maryland, College Park, USA

[2] New Jersey Institute of Technology, Newark, USA