Online Stochastic Matching: New Algorithms and Bounds*

Brian Brubach†, Karthik A. Sankararaman‡, Aravind Srinivasan§, and Pan Xu¶

Department of Computer Science, University of Maryland, College Park, MD 20742, USA

Original Version: May 2016
This Version: November 2017

Abstract

Online matching has received significant attention over the last 15 years due to its close connection to Internet advertising. As the seminal work of Karp, Vazirani, and Vazirani has an optimal $(1 - 1/e)$ competitive ratio in the standard adversarial online model, much effort has gone into developing useful online models that incorporate some stochasticity in the arrival process. One such popular model is the "known I.I.D. model" where different customer-types arrive online from a known distribution. We develop algorithms with improved competitive ratios for some basic variants of this model with integral arrival rates, including: (a) the case of general weighted edges, where we improve the best-known ratio of 0.667 due to Haeupler, Mirrokni and Zadimoghaddam [12] to 0.705; and (b) the vertex-weighted case, where we improve the 0.7250 ratio of Jaillet and Lu [13] 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 $1 - 1/e$. For the special case where each edge is unweighted and has a uniform constant probability of being present, we improve upon $1 - 1/e$ 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 (*Mathematics of Operations Research*, 2013) and is a tractable structure for algorithm design and analysis. The appropriate random move preserves many of the removed constraints (approximately 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

†**Email:** bbrubach@cs.umd.edu
Supported in part by NSF Awards CNS 1010789 and CCF 1422569.

‡**Email:** kabinav@cs.umd.edu
Supported in part by NSF Awards CNS 1010789 and CCF 1422569.

§**Email:** srin@cs.umd.edu
Supported in part by NSF Awards CNS 1010789 and CCF 1422569, and by research awards from Adobe, Inc.

¶**Email:** panxu@cs.umd.edu
Supported in part by NSF Awards CNS 1010789 and CCF 1422569.

1 Introduction.

Applications to Internet advertising have driven the study of online matching problems in recent years [20]. 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. WLOG we consider the case where $v \in V$ can be matched at most once. 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 (here we assume $T = n$). 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. A special case of this model is vertex-weighted matching [1], where weights are associated only with the advertisers. In other words, a given advertiser has the same revenue generated for matching any of the user types interested in it. In some modern business models, revenue is not generated upon matching advertisements, but only when a user *clicks* on the advertisement: this is the *pay-per-click* model. From historical data, one can assign the probability of a particular advertisement being clicked by a type of user. Works including [21, 22] 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 of this LP to guide our online actions. To do so, we use various modifications of dependent randomized rounding.

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 vertices v arrive online and are drawn with replacement from an I.I.D. distribution on V. For each $v \in V$, we are given an *arrival rate* r_v, which is the expected number of times v will arrive. With the exception of Sections 5, 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 this case, we have that $|V| = n$ where n is the total number of online rounds.

In the **vertex-weighted** variant, every vertex $u \in U$ has a weight w_u and we seek a maximum weight matching. In the **edge-weighted** variant, every edge $e \in E$ has a weight w_e and we again seek a maximum weight matching. In the **stochastic rewards** variant, each edge has both a

weight w_e and a probability p_e of being present once we probe edge e[1] and we seek to maximize the expected weight of the matching.

Asymptotic assumption and notation. We will always assume n is large and analyze algorithms as n goes to infinity: e.g., if $x \leq 1-(1-2/n)^n$, we will just write this as "$x \leq 1-1/e^2$" instead of the more-accurate "$x \leq 1-1/e^2+o(1)$". These suppressed $o(1)$ terms will subtract at most $o(1)$ from our competitive ratios. Another fact to note is that the **competitive ratio** is defined slightly differently than usual for this set of problems (Similar to notation used in [20]). In particular, it is defined as $\frac{\mathbb{E}[\text{ALG}]}{\mathbb{E}[\text{OPT}]}$. 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 thus far, but a non-adaptive algorithm has to specify all of its actions before the start of the online phase. Throughout, we use "WS" to refer to the worst case instance for various algorithms.

2.1 LP benchmark for deterministic rewards.

As in prior work (*e.g,* see [20]), we use the following LP to upper bound the optimal offline expected performance and also use it to guide our algorithm in the case where rewards are deterministic. For the case of stochastic rewards, we use slightly modified LPs, whose definitions we defer until Sections 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 $f_w = \sum_{e \in \partial(w)} f_e$.

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

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

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

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

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

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 $w_{(u,v)}$).

Constraint 2 is the matching constraint for vertices in U. Constraint 3 is valid because each vertex in V has an arrival rate of 1. Constraint 4 is used in [19] and [12]. It captures the fact that the expected number of matches for any edge is at most $1-1/e$. This is valid for large n because the probability that a given vertex does not arrive during n rounds is $1/e$. Constraint 5 is similar to the previous one, but for pairs of edges. For any two neighbors of a given $u \in U$, the probability that neither of them arrive is $1/e^2$. Therefore, the sum of variables for any two distinct edges in $\partial(u)$ cannot exceed $1-1/e^2$. Notice that constraints 4 and 5 reduces the gap between the optimal LP solution and the performance of the optimal online algorithm. In fact, without constraint 4, we cannot in general achieve a competitive ratio better than $1-1/e$.

[1]The edge realization process is independent for different edges. At each step, the algorithm "probes" an edge. 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.

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 Section 2.6 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. The paper [4] shows under the assumption of constant p *and* $\text{OPT} = \omega(1/p)$, they can obtain a $(1-\epsilon)$ approximation to the optimal solution. However, these assumptions are too strong to be used in our setting.

Finally, 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 into 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 and thus the resultant LP is polynomial solvable.

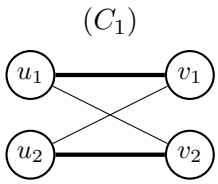
2.2 Overview of vertex-weighted algorithm and contributions.

A key challenge encountered by [13] was that their special LP could lead to length four cycles of type C_1 shown in Figure 1. In fact, they used this cycle to show that no algorithm could perform better than $1 - 2/e^2 \approx 0.7293$ using their LP. They mentioned that tighter LP constraints such as 4 and 5 in the LP from Section 2 could avoid this bottleneck, but they did not propose a technique to use them. Note that the $\{0, 1/3, 2/3\}$ solution produced by their LP was an essential component of their Random List algorithm.

We show a randomized rounding algorithm to construct a similar, simplified $\{0, 1/3, 2/3\}$ vector from the solution of a stricter benchmark LP. This allows for the inclusion of additional constraints, most importantly constraint 5. Using this rounding algorithm combined with tighter constraints, we will upper bound the probability of a vertex appearing in the cycle C_1 from Figure 1 at $2-3/e \approx 0.89$. (See Lemma 4) Additionally, we show how to deterministically break all other length four cycles which are not of type C_1 without creating any new cycles of type C_1. Finally, we describe an algorithm which utilizes these techniques to improve previous results in both the vertex-weighted and unweighted settings.

For this algorithm, we first solve the LP in Section 2 on the input graph. In Section 4, we show how to use the technique in sub-section 2.7 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 gap between the best unweighted algorithm with a ratio of $1 - 2e^{-2}$ due to [13] and the negative result of $1 - e^{-2}$

Figure 1: This cycle is the source of the negative result described by Jaillet and Lu [13]. Thick edges have $f_e = 2/3$ while thin edges have $f_e = 1/3$.



due to [19]. We take a step towards closing that gap by showing that an algorithm can achieve $0.7299 > 1 - 2e^{-2}$ for both the unweighted and vertex-weighted variants with integral arrival rates. In doing so, we make progess on Open Questions 3 and 4 from the book [20]. [2]

2.3 Overview of edge-weighted algorithm and contributions.

A challenge that arises in applying the *power of two choices* to this setting is when the same edge (u, v) is included in both matchings M_1 and M_2. In this case, the copy of (u, v) in M_2 can offer no benefit and a second arrival of v is wasted. To use an example from related work, Haeupler *et al.* [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 $1 - 1/e \approx 0.63$.

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 $1 - 2/e \approx 0.26$.

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

2.4 Overview of stochastic rewards and contributions.

This algorithm (Algorithm 9) is presented in Section 5 and 6. We believe the known I.I.D. model with stochastic rewards is an interesting new direction motivated by the work of [21] and [22] in the adversarial model. We introduce a new, more general LP (see LP (9)) specifically for this setting and show that a simple algorithm using the LP solution directly can achieve a competitive ratio of $1 - 1/e$, which is proved to be optimal among all non-adaptive algorithms. In [22], it is shown that no randomized algorithm can achieve a ratio better than $0.62 < 1 - 1/e$ in the adversarial

[2] 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 $1 - 2e^{-2}$ 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 (Section 3)	0.667 [12]	0.705
Vertex-Weighted (Section 4)	0.725 [13]	0.7299
Unweighted	$1-2/e^2$ [13]	0.7299 ($> 1-2/e^2$)
Stochastic Rewards (Section 5 and 6)	N/A	$1-e^{-1}$ for general version 0.702 for the restricted version

model. Hence, achieving a $1-1/e$ for the i.i.d. model shows that this lower bound does not extend to this model. Further, the paper [5] shows that using LP (9) 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. Finally, 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 $1-1/e$.

2.5 Summary of our contributions

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

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

Theorem 3. *For edge-weighted online stochastic matching with arbitrary arrival rates and stochastic rewards, online algorithm* SM (9) *achieves a competitive ratio of* $1-1/e$*, which is optimal all among all non-adaptive algorithms.*

Theorem 4. *For unweighted online stochastic matching with integral arrival rates and uniform stochastic rewards, there exists an adaptive algorithm which achieves a competitive ratio of at least* 0.702.

2.6 Runtime of algorithm.

In this section, we discuss the implementation details of our algorithm. 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 $O(|E|+|U|+|V|)$ rows and $O(|E|)$ columns. However, note that the number of non-zero entries in this matrix is of the order $O((|U|+|V|)|E|)$. 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 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 $O(|E|^2)$. Finally, recall that both these operations are part of the pre-processing step. Hence 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 $O(\log |U|)$) and $O(1)$ for the edge-weighted and the vertex-weighted algorithms in Section 3 and 4.

2.7 LP rounding technique $\mathsf{DR}[\mathbf{f}, k]$.

For the algorithms presented, we first solve the benchmark LP in sub-section 2.1 for the input instance to get a fractional solution vector $\mathbf{f}$. We then round $\mathbf{f}$ to an integral solution $\mathbf{F}$ using a two step process we call $\mathsf{DR}[\mathbf{f}, k]$. The first step is to multiply $\mathbf{f}$ by k. The second step is to apply the dependent rounding techniques of Gandhi, Khuller, Parthasarathy, and Srinivasan [11] to this new vector. In this paper, we will always choose k to be 2 or 3. This is because a vertex in V may appear more than once, but probably not more than two or three times.

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

2.8 Related work.

The study of online matching began with the seminal work of Karp, Vazirani, Vazirani [14], where they gave an optimal online algorithm for a version of the unweighted bipartite matching problem in which vertices arrive in adversarial order. Following that, a series of works have studied various related models. The book by Mehta [20] 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, 15, 16, 18]. 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 [21, 22] among others, but not in the known I.I.D. model.

Related Work in the Vertex-Weighted and Unweighted Settings. The vertex-weighted and unweighted settings have many results starting with Feldman, Mehta, Mirrokni and Muthukrishnan [10] who were the first to beat $1 - 1/e$ with a competitive ratio of 0.67 for the unweighted problem. This was improved by Manshadi, Gharan, and Saberi [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 $1 - e^{-2} \approx 0.86$. Finally, Jaillet and Lu [13] presented an adaptive algorithm which used a clever LP to achieve 0.725 and $1 - 2e^{-2} \approx 0.729$ for the vertex-weighted and unweighted problems, respectively.

Related Work in the Edge-Weighted Setting. For this model, Haeupler, Mirrokni, Zadimoghaddam [12] were the first to beat $1 - 1/e$ by achieving a competitive ratio of 0.667. They use a *discounted LP* with tighter constraints than the basic matching LP (a similar LP can be seen in 2.1) and they employ the *power of two choices* by constructing two matchings offline to guide their online algorithm.

Other Related Work. Devanur *et al* [8] gave an algorithm which achieves a ratio of $1 - 1/\sqrt{2\pi k}$ for the Adwords problem in the Unknown I.I.D. arrival model with knowledge of the optimal budget utilization and when the bid-to-budget ratios are at most $1/k$. 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.

3 Edge-weighted matching with integral arrival rates

3.1 A simple 0.688-competitive algorithm.

As a warm-up, we describe a simple algorithm which achieves a competitive ratio of 0.688 and introduces key ideas in our approach. We begin by solving the LP in sub-section 2.1 to get a fractional solution vector $\mathbf{f}$ and applying $\mathsf{DR}[\mathbf{f}, 2]$ as described in Subsection 2.7 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 *Hall's Theorem*. Finally, we randomly permute the two matchings into an ordered pair of matchings, $[M_1, M_2]$. These matchings serve as a guide for the online phase of the algorithm, similar to [12]. The entire warm-up algorithm for the edge-weighted model, denoted by EW_0, is summarized in Algorithm 1.

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 if and only if $f_e > 1/2$. Therefore, we consider three types of edges. We say an

Algorithm 1: [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 an integral vector $\mathbf{F}$.
3 Create the graph $G_{\mathbf{F}}$ with F_e copies of each edge $e \in E$ and decompose it into two matchings.
4 Randomly permute the matchings to get a *random ordered* pair of matchings, say $[M_1, M_2]$.
5 When a vertex v arrives for the first time, try to assign v to some u_1 if $(u_1, v) \in M_1$; when v arrives for the second time, try to assign v to some u_2 if $(u_2, v) \in M_2$.
6 When a vertex v arrives for the third time or more, do nothing in that step.

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 1 bounds these probabilities.

Lemma 1. *(Proof details in Section 3 of [12]) Given M_1 and M_2, in the worst case (1) $P_1 = 0.5808$; (2) $P_2 = 0.14849$ and (3) $P_b = 0.632$.*

We can use Lemma 1 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.

Proof. (Analysis for EW_0)
Consider the following two cases.

- **Case 1:** $0 \leq f_e \leq 1/2$: 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:** $1/2 \leq f_e \leq 1 - 1/e$: Similarly, by marginal distribution, $\Pr[e \in \psi_b] = \Pr[F_e = \lceil 2f_e \rceil] = 2f_e - \lfloor 2f_e \rfloor = 2f_e - 1$. It follows that $\Pr[e \in \psi_1] = \Pr[e \in \psi_2] = (1/2)(1 - (2f_e - 1)) = 1 - f_e$. Thus, the ratio is (noting that the first term is from case 1 while the second term is from case 2) $((1 - f_e)(P_1 + P_2) + (2f_e - 1)P_b)/f_e \geq 0.688$, where the WS is for an edge e with $f_e = 1 - 1/e$.

□

3.2 A 0.7-competitive algorithm.

In this section, we describe an improvement upon the previous warm-up algorithm to get a competitive ratio of 0.7. We start by making an observation about the performance of the warm-up algorithm. After solving the LP, let edges with $f_e > 1/2$ be called *large* and edges with $f_e \leq 1/2$ 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 later. For all large edges $\ell \in L$ such that $f_\ell > 1/2$, we set $f_\ell = f_\ell + \eta$. For all small edges $s \in S$ which are adjacent to some large edge, let $\ell \in L$ be the largest edge adjacent to s such that $f_\ell > 1/2$. Note that it is possible for s to have two large neighbors, but we only care about the largest one. We set $f_s = f_s \left(\frac{1-(f_\ell+\eta)}{1-f_\ell}\right)$.

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

3.2.1 Analysis.

We can now prove Theorem 2.

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

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

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

    This follows from Lemma 1; in particular, the first two terms are the result of how we set 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 with respect to f_ℓ. So the worst case is $f_\ell = 1 - 1/e$ (due to constraint 4 in LP 2.1) and we have a ratio of

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

- $\frac{1}{2} < f_\ell \leq 1 - \frac{1}{e}$:
  Here, the ratio is $((1-(f_\ell+\eta))(P_1+P_2) + (2(f_\ell+\eta)-1)P_b)/f_\ell$, where the WS is for an edge e with $f_\ell = 1 - 1/e$. This follows because of the fact that it is a decreasing function with respect to f_ℓ. To see that it is a decreasing function, note that it can be rearranged as $(P_1 + P_2 - P_b + \eta(2P_b - P_1 - P_2) + f_\ell(2P_b - P_1 - P_2))/f_\ell$. Substituting the appropriate values,

we have that the value of $(2P_b - P_1 - P_2) = 0.535$. Hence, the expression can be written as $(c_1 + 0.535 f_\ell)/f_\ell$. If we show that $c_1 \geq 0$, we are done. The optimal value of η we choose turns out to be 0.0142. Hence we have $c_1 = P_1 + P_2 - P_b + 0.0142 \cdot 0.535 = 0.1048 > 0$.

Choosing the optimal value of $\eta = 0.0142$, yields an overall competitive ratio of 0.7 for this new algorithm. We now need to show that this value of η ensures that both f_ℓ and f_s are less than 1 *after* modification. Since $f_\ell \leq 1 - 1/e$ we have that $f_\ell + \eta \leq 1 - 1/e + 0.0142 \leq 1$. Note that $f_\ell \geq 1/2$. Hence, the modified value of 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 $f_\ell \in [1/2, 1 - 1/e]$ and has a value less than 0.98 at $f_\ell = 1/2$. □

3.3 A 0.705-competitive algorithm.

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 Subsection 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 sections 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 2 and 3, respectively.
- The proof of Lemma 2 depends on claims 4, 5, and 6 (Found in the Appendix). Each of those claims is a careful case-by-case analysis. Intuitively, 4 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), 5 refers to the case where u is incident to three small edges and 6 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 3 depends on claims 7 and 8 (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 A 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 subsection 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 $f_e \leq 1 - 1/e \leq 2/3$. Therefore after $\mathsf{DR}[\mathbf{f}, 3]$, each $F_e \in \{0, 1, 2\}$. Consider the graph $G_{\mathbf{F}}$ where each edge e is associated with the value of F_e. 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 Subsection 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 $1 - q$.

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

Algorithm 2: 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 EW_1 and EW_2 with probabilities q and $1-q$ respectively on $G_{\mathbf{F}}$.

3.3.2 Algorithm EW_1.

In this section, we describe the randomized algorithm EW_1 (Algorithm 3). Suppose we view the graph of $G_{\mathbf{F}}$ in another way where each edge has F_e copies. Let $\mathsf{PM}[\mathbf{F}, 3]$ refer to the process of constructing the graph $G_{\mathbf{F}}$ with F_e copies of each edge, decomposing it into three matchings, 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 the 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 $e = (u, v)$ in $G_{\mathbf{F}}$, we say e is of type Γ_1 if u has two other neighbors v_1 and v_2 in $G_{\mathbf{F}}$ 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.

Note, we use the terminology, assign v to u to denote that edge (u, v) is matched by the algorithm if u is not matched until that step.

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 some u_1 with $(u_1, v) \in M_1$.
3 When v comes for the second time, assign v to some u_2 with $(u_2, v) \in M_2$.
4 When v comes for the third time, if e is either a large edge or a small edge of type Γ_1 then assign v to some u_3 with $e = (u_3, v) \in M_3$. However, if e is a small edge of type Γ_2 then *with probability* h, assign v to some u_3 with $e = (u_3, v) \in M_3$; otherwise, do nothing.
5 When v comes for the fourth or more time, do nothing in that step.

Here, h is a parameter we will fix at the end of analysis. 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 2. *For $h = 0.537815$, EW_1 achieves a competitive ratio $\mathsf{R}[\mathsf{EW}_1, 2/3] = 0.679417$, $\mathsf{R}[\mathsf{EW}_1, 2/3] = 0.751066$ 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 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*. Recall that the previous algorithm, EW_1, first invokes $\mathsf{PM}[\mathbf{F}, 3]$ to obtain a

random ordered triple of matchings. In contrast, EW_2 will invoke a routine, denoted by $\mathsf{PM}^*[\mathbf{F}, 2]$ (Algorithm 4), to generate a (*random ordered*) pair of pseudo-matchings from $\mathbf{F}$. Recall that $\mathbf{F}$ is an integral solution vector where $\forall e$ we have $F_e \in \{0, 1, 2\}$. WLOG, we can assume that $F_v = 1$ for every v in $G_{\mathbf{F}}$; otherwise we can *dummy* vertex to ensure this is the case.

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

1. Suppose v has two neighbors in $G_{\mathbf{F}}$, 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 in $G_{\mathbf{F}}$ and the incident edges are $\partial(v) = (e_1, e_2, e_3)$. Take a random permutation of $\partial(v)$, say $(\pi_1, \pi_2, \pi_3) \in \Pi(\partial(v))$. Add π_1 to M_1 with probability y_1 and π_2 to M_2 with probability y_2.

Here $0 \leq y_1, y_2 \leq 1$ are parameters which will be fixed after the analysis. Algorithm 5 describes EW_2.

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 3. *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. Similarly, let (a_2, b_2) denote the same for EW_2. We have the following two cases.

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

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

- $1/3 \le f_e \le 1 - 1/e$: 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. □

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 $0.72998 > 1 - 2e^{-2}$. Recall 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 let $G_{\mathbf{H}}$ be the graph induced by $\mathbf{H}$. Each edge e in $G_{\mathbf{H}}$ thus takes a value $H_e \in \{0, 1/3, 2/3\}$. Notice that for each u, $H_u \doteq \sum_{e \in \partial(u)} H_e \le 1$, which implies that u has at most 3 neighbors in $G_{\mathbf{H}}$ (we ignore all edges e with $H_e = 0$). In this section, we focus on the sparse graph $G_{\mathbf{H}}$. The main idea of our algorithm is as follows.

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

We first briefly describe how we overcome the *bottleneck* case for the algorithm in [13] and then explain the algorithm in full detail. The WS for the vertex-weighted case in [13] is shown in Figure 2, which happens at a node u with a competitive ratio of 0.725 (recall that [13] analyze their algorithm by considering cases for various neighborhood structure at a given offline vertex). From the analysis of [13], we have that the node u_1 (in Figure 2) has a competitive ratio of at least 0.736. Hence, we can *boost* the performance of u at the cost of u_1. Specifically, we increase the value of $H_{(u,v1)}$ and decrease the value $H_{(u_1,v_1)}$. Cases (10) and (11) in Figure 4 illustrate this. After this modification, the new WS for vertex-weighted is now the C_1 cycle shown in both Figures 1 and 2. In fact, this is the WS for the unweighted case in [13] as well. However, Lemma 4 implies that C_1 cycles can be avoided with probability at least $3/e - 1$. This helps us improve the ratio even for the unweighted case in [13]. Lemma 4 describes this formally.

Lemma 4. *For any given $u \in U$, u appears in a C_1 cycle after $\mathsf{DR}[\boldsymbol{f}, 3]$ with probability at most $2 - 3/e$.*

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

- **Case 1:** $\partial(u)$ contains only one edge e with $f_e > 1/3$. Let $q_1 = \Pr[H_e = 1/3]$ and $q_2 = \Pr[H_e = 2/3]$ after $\mathsf{DR}[\mathbf{f}, 3]$. By $\mathsf{DR}[\mathbf{f}, 3]$, we know that $\mathbb{E}[H_e] = \mathbb{E}[F_e]/3 = q_2(2/3) + q_1(1/3) = f_e$.

  Notice that $q_1 + q_2 = 1$ and hence $q_2 = 3f_e - 1$. Since this is an increasing function of f_e and $f_e \leq 1 - 1/e$ from LP constraint 4, we have $q_2 \leq 3(1 - 1/e) - 1 = 2 - 3/e$.

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

  From LP constraint 5, we know that $f_{e_1} + f_{e_2} \leq 1 - 1/e^2$, and hence $q_2 \leq 3(1 - 1/e^2) - 2 < 2 - 3/e$.

□

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

4.1 RLA algorithm.

Now we discuss how to apply RLA based on $\mathbf{H}'$ to the sparse graph $G_{\mathbf{H}}$. Let $\delta_{\mathbf{H}}(v)$ be the set of neighboring nodes of v in $G_{\mathbf{H}}$. Here we assume WLOG that $\mathbf{H}_v \doteq \sum_{e \in \partial(v)} H_e = 1$ and thus each v has at least 2 neighbors in $G_{\mathbf{H}}$ since each non-zero H_e satisfies $H_e \in \{1/3, 2/3\}$ (we are in a better situation when $\mathbf{H}_v < 1$). Additionally, we will see in Section 4.2 that during the two modifications, we have the sum of all edge values incident to v (i.e., $\mathbf{H}_v$) unchanged and hence we have $\mathbf{H}'_v = \mathbf{H}_v = 1$ for each v.

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

$$\Pr[\mathcal{R}_v = (u_1, u_2)] = H'_{(u_1,v)}, \quad \Pr[\mathcal{R}_v = (u_2, u_1)] = H'_{(u_2,v)} \tag{7}$$

- 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

$$\Pr[\mathcal{R}_v = (u_i, u_j, u_k)] = H'_{(u_i,v)} \frac{H'_{(u_j,v)}}{H'_{(u_j,v)} + H'_{(u_k,v)}} \tag{8}$$

We can verify that the sampling distributions described in Equations (7) and (8) are valid since $\mathbf{H}'_v = \sum_{e \in \partial(v)} H'_e = 1$.

The full details of the Random List Algorithm, $\mathsf{RLA}[\mathbf{H}']$, are shown in Algorithm 6.

Algorithm 6: RLA[$\mathbf{H}'$] (Random List Algorithm based on $\mathbf{H}'$)

1 When a vertex v comes, generate a random list $\mathcal{R}_v$ satisfying Equation (7) or (8)
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.

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.

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

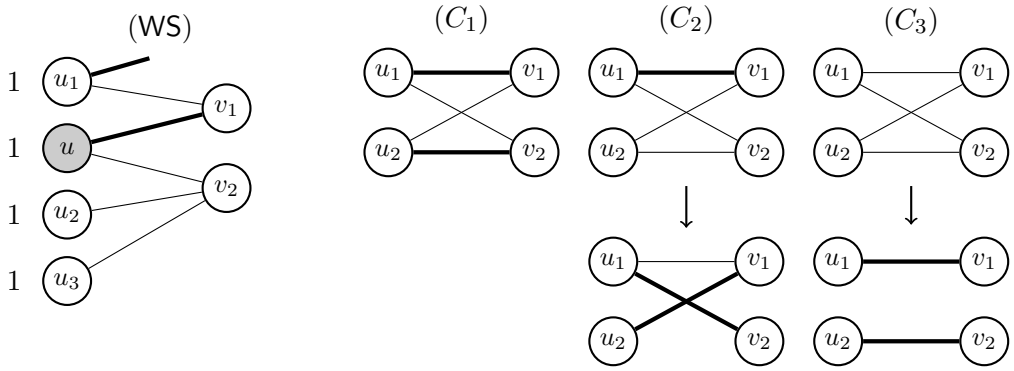


Figure 2: Left: The WS for Jaillet and Lu [13] for their vertex-weighted case. Right: The three possible types of cycles of length 4 after applying DR[$\mathbf{f}, 3$]. Thin edges have $H_e = 1/3$ and thick edges have $H_e = 2/3$.

The first modification is to break the cycles deterministically. There are three possible cycles of length 4 in the graph $G_{\mathbf{H}}$, denoted C_1, C_2, and C_3. In [13], they give an efficient way to break C_2 and C_3, as shown in Figure 2. Cycle C_1 cannot be modified further and hence, is the bottleneck for their unweighted case. Notice that, while breaking the cycles of C_2 and C_3, new cycles of C_1 can be created in the graph. Since our randomized construction of solution $\mathbf{H}$ gives us control on the probability of cycles C_1 occurring, we would like to break C_2 and C_3 in a controlled way, so as not to create any new C_1 cycles. This procedure is summarized in Algorithm 7 and its correctness is proved in Lemma 6.

Lemma 5. *After applying Algorithm 7 to $G_{\boldsymbol{H}}$, 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 of Lemma 5. Lemma 5 follows from the following three Claims:

Claim 1. *Breaking cycles will not change the value H_w for any $w \in U \cup V$.*

Claim 2. *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.*

Claim 3. *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 of Claim 1. As shown in Figure 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.

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.

Proof of Claim 2. 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.

Proof of Claim 3. 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 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.

4.2.2 The Second modification to H.

Informally, this second modification decreases the rates of lists associated with those nodes u with $H_u = 1/3$ or $H_u = 2/3$ and increases the rates of lists associated with nodes u with $H_u = 1$. We will illustrate our intuition on the following example.

Consider the graph G in Figure 3. Let thin and thick edges represent $H_e = 1/3$ and $H_e = 2/3$ respectively. We will now calculate the competitive ratio after applying RLA on G. Let $\mathsf{P_u}$ denote the probability that u gets matched after the algorithm. Let B_u denote the event that among the n random lists, there exists a list starting with u and G_u^v denote the event that among the n lists, there exists successive lists such that (1) Each of those lists starts with a $u' \neq u$ and $u' \in \delta(v)$ and (2) The lists arrive in an order which ensures u will be matched by the algorithm. From lemma 4 and Corollary 1 in [13], the following lemma follows:

Figure 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.

Lemma 6. *Suppose u is not a part of any cycle of length 4. We have*

$$\mathsf{P}_u = 1 - (1 - \Pr[B_u]) \prod_{v \sim u} (1 - \Pr[G_u^v]) + o(1/n)$$

For the node u, we have $\Pr[B_u] = 1 - e^{-1}$. From definition, $G_u^{v_1}$ is the event that among the n lists, the random list $\mathcal{R}_{v_1} = (u_1, u)$ comes at least twice. Notice that the list $\mathcal{R}_{v_1} = (u_1, u)$ comes with probability $\frac{1}{3n}$. Thus we have $\Pr[G_u^{v_1}] = \Pr[X \geq 2] = 1 - e^{-1/3}(1 + 1/3)$, where $X \sim \text{Pois}(1/3)$. Similarly, we can get $\Pr[G_u^{v_2}] = 1 - e^{-2/3}(1 + 2/3)$ and the resultant $\mathsf{P}_u = 1 - \frac{20}{9e^2} \sim 0.699$. Observe that $\mathsf{P}_{u_1} \geq \Pr[B_{u_1}] = 1 - e^{-1/3}$ and $\mathsf{P}_{u_2} \geq \Pr[B_{u_2}] = 1 - e^{-2/3}$. 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 $\mathsf{R}[\mathsf{RLA}, 1/3] \geq 3(1 - e^{-1/3}) \sim 0.8504$ 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 one modifies $H_{(u_1,v_1)}$ and $H_{(u,v_1)}$ to $H'_{(u_1,v_1)} = 0.1$ and $H'_{(u,v_1)} = 0.9$, 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 resulting changes are $\Pr[B_u] = 1 - e^{-0.9-1/3}$, $\Pr[G_u^{v_1}] = 1 - e^{-0.1}(1 + 0.1)$, $\mathsf{R}[\mathsf{RLA}, 1] = 0.751$, $\Pr[B_{u_1}] = 1 - e^{-1/3}$, $\Pr[G_{u_1}^{v_1}] \sim 0.227$ 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 Figure 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.

Algorithm 8: VW

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

The algorithm VW consists of two different random processes: sub-routine $\mathsf{DR}[\mathbf{f}, 3]$ in the offline phase and RLA in the online phase. Consequently, the analysis consists of two parts. First, for a

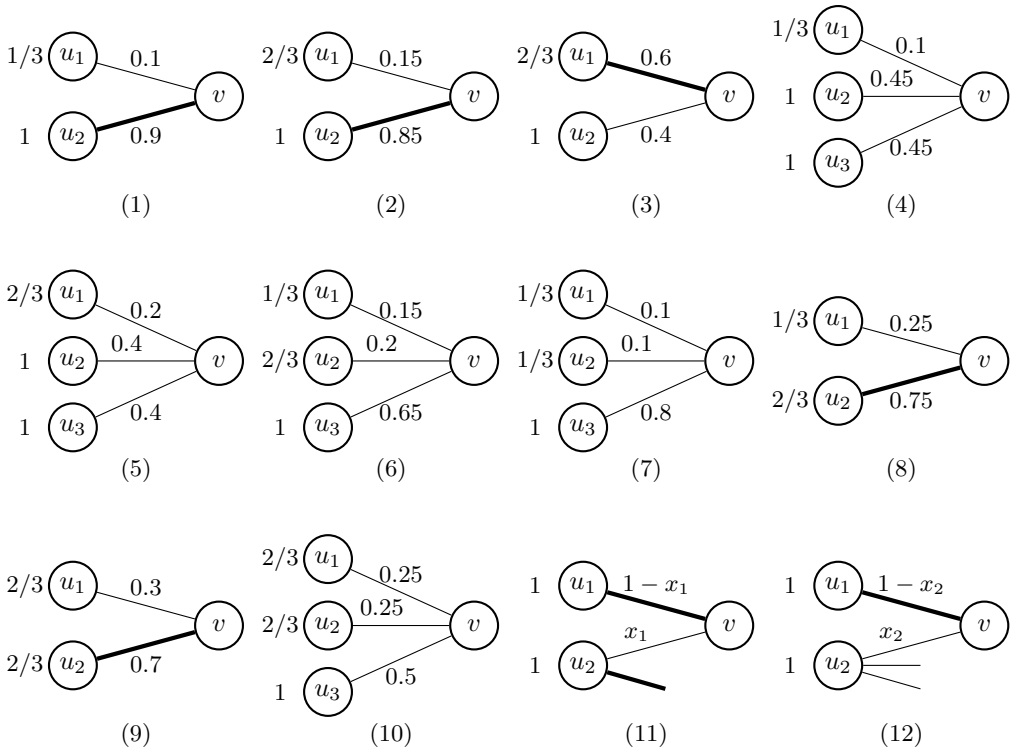


Figure 4: Illustration for second modification to $\mathbf{H}$. The value assigned to each edge represents the value after the second modification. Here, $x_1 = 0.2744$ and $x_2 = 0.15877$.

given graph $G_{\mathbf{H}}$, we analyze the ratio of $\mathsf{RLA}[\mathbf{H}']$ 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 $\mathsf{DR}[\mathbf{f}, 3]$ 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 7 gives the respective ratio values. The proof can be found in section A.2.1 in the Appendix.

Lemma 7. *Consider a given $\boldsymbol{H}$ and a vertex u in $G_{\boldsymbol{H}}$. The respective ratios achieved by* RLA *after the modifications are as described below.*

- *If $H_u = 1$, then the competitive ratio $\mathsf{R}[\mathsf{RLA}, 1] = 1 - 2e^{-2} \sim 0.72933$ 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 essentials to prove Theorem 1.

Proof. From Lemmas 4 and 5, we know that any u is present in cycle C_1 with probability at most $(2-3/e)$.

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 Lemma 7, we get that the final ratio for u should be at least

$$(0.72933q_1 + 0.735622q_2 + (2/3) * 0.7847q_3)/(q_1 + q_2 + (2/3)q_3)$$

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) $q_1 \leq 2 - 3/e$, we get a minimum value of 0.729982 for $q_1 = 2 - 3/e$ and $q_2 = 3/e - 1$.

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 1. □

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, i.e., we try to assign v to u. We assume this process is independent of the stochastic arrival of each v. We will show that the simple non-adaptive algorithm introduced in [12] can be extended to this general case. This achieves a competitive ratio of $(1 - \frac{1}{e})$. Note that Manshadi *et al.* [19] show that no non-adaptive algorithm can possibly achieve a ratio better than $(1 - 1/e)$ for the non-integral arrival rates, even for the case of all $p_e = 1$. Thus, our algorithm is an optimal non-adaptive algorithm for this model.

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 and thus some realizations of f_e can be greater than 1.

$$\max \quad \sum_{e \in E} w_e f_e p_e : \tag{9}$$

$$\text{s.t.} \quad \sum_{e \in \partial(u)} f_e p_e \leq 1, \qquad \forall u \in U \tag{10}$$

$$\sum_{e \in \partial(v)} f_e \leq r_v, \qquad \forall v \in V \tag{11}$$

Our algorithm is summarized in Algorithm 9. Notice that the last constraint ensures that step 2 in the algorithm is valid. Let us now prove theorem 3.

Proof. Let $B(u, t)$ be the event that u is safe at beginning of round t and $A(u, t)$ to be the event that vertex u is matched during the round t conditioned on $B(u, t)$. From the algorithm, we know $\Pr[A(u,t)] \leq \sum_{v \sim u} \frac{r_v}{n} \frac{f_{u,v}}{r_v} p_e \leq \frac{1}{n}$, which follows by $Pr[B(u,t)] = \Pr\left[\bigwedge_{i=1}^{t-1} (\neg A(u,i))\right] \geq \left(1 - \frac{1}{n}\right)^{t-1}$.

Algorithm 9: SM

1 Construct and solve LP (9). WLOG assume $\{f_e | e \in E\}$ is an optimal solution.

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

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

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

□

6 Integral arrival rates with uniform stochastic rewards.

In this section, we consider a special case of the model studied in Section 5 and show that we can indeed surpass the $1 - 1/e$ 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.

$$\max \quad p \cdot \sum_{e \in E} f_e : \tag{12}$$

$$\text{s.t.} \sum_{e \in \partial(u)} f_e \cdot p \leq 1, \qquad \forall u \in U \tag{13}$$

$$\sum_{e \in \partial(v)} f_e \leq 1, \qquad \forall v \in V \tag{14}$$

$$\sum_{S \subseteq \partial(u)} f_e p \leq 1 - \exp(-|S|p), \qquad \forall S \subseteq \partial(u), |S| \leq 2/p \tag{15}$$

Lemma 8. *LP (12) is a valid upper bound for the expected offline optimal.*

Proof. It suffices to show that the constraint (15) is valid (the correctness of the other constraints follows from the previous section). Consider a given $S \subseteq \partial(u)$. Let $X(I)$ be the number of copies of all edges in S in an offline instance I and $Y(I)$ be the probability that any edge in S is included in

an offline optimal on I. From a straightforward calculation, it follows that $Y(I) \leq 1-(1-p)^{X(I)}$ and using linearity of expectation we get,

$$\mathbb{E}[Y(I)] \leq 1 - \mathbb{E}[(1-p)^{X(I)}]$$

By definition we have $\mathbb{E}[Y(I)] = \sum_{e \in S} f_e \cdot p$. Since n is assumed to be sufficiently large, $X(I)$ follows a Poisson distribution with mean $|S|$. Substituting the fact that $\mathbb{E}[(1-p)^{X(I)}] = \exp(-p|S|)$ into the equation above, we prove the lemma. □

Note that it is impossible to beat $1-1/e$ using LP (12) as the benchmark without the extra constraint (15) (see the hardness instance shown in [5]). Our main idea in the online phase is primarily based on [19]. In the offline phase, we first solve LP (12) 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 (12) 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]. 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]).

$$\frac{\mathbb{E}[\text{ALG}]}{\mathbb{E}[\text{OPT}]} \geq \min_{u \in U} \left(\frac{(1-e^{-f'_u}) + q'_u e^{-2} - (q'_u)^2 e^{-1}\left(\frac{1}{2} - e^{-1}\right) - e^{-2} f'_u (1-f'_u)}{f'_u} \right) \doteq F(f'_u, q'_u) \qquad (16)$$

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

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

We can verify that for each given $f'_u \leq 1$, the RHS expression in inequality (16) 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 9. *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:

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

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

$$\Delta = p \cdot \sum_{e\in\partial(u)} \max\Big(2f^*_e - 1, 0\Big) = p \cdot \sum_{e\in S^*} (2f^*_e - 1) = \sum_{e\in S^*} 2pf^*_e - p|S^*| \tag{17}$$

Observe that

$$\frac{p}{2}|S^*| \leq \sum_{e\in S^*} f^*_e \cdot p \leq f'_u \Rightarrow |S^*| \leq \frac{2f'_u}{p} \leq \frac{2}{p} \tag{18}$$

From Constraint (15), we have $\sum_{e\in S^*}(pf^*_e) \leq 1 - \exp(-|S^*|p)$. Substituting this inequality back into Equation (17), 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 $q'_u \geq f'_u - (1 - \ln 2)$. □

Let us now complete the proof of Theorem 4.

Proof. We need to prove that $F(f'_u, q'_u)$ defined in 16 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 9, we have $q'_u \geq f'_u - (1 - \ln 2)$. Once again, simple calculations show that

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

□

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 $1-2e^{-2}$ due to [13] and the negative result of $1 - e^{-2}$ due to [19]. We took a step towards closing that gap by showing that an algorithm can achieve $0.7299 > 1 - 2e^{-2}$ for both the unweighted and vertex-weighted variants with integral arrival rates. In doing so, we made progess on Open Questions 3 and 4 in the

online matching and ad allocation survey [20]. This was possible because our approach of rounding to a simpler fractional solution allowed us to employ a stricter LP. For the edge-weighted variant, we showed that one can significantly improve the power of two choices approach by generating two matchings from the same LP solution. For the variant with edge weights, non-integral arrival rates, and stochastic rewards, we presented a $(1 - 1/e)$-competitive algorithm. This showed that the $0.62 < 1 - 1/e$ bound given in [22] 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 $\{0, 1/k, 2/k, \ldots, \lceil k(1-1/e)\rceil/k\}$; 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 $1 - 1/e$. However, the serious limitation comes from the fact that a polynomial sized LP is insufficient to capture the complexity of the problem.

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

References

[1] Aggarwal, G., Goel, G., Karande, C., and 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* (2011), SIAM, pp. 1253–1264.

[2] Alaei, S., Hajiaghayi, M., and Liaghat, V. Online prophet-inequality matching with applications to ad allocation. In *Proceedings of the 13th ACM Conference on Electronic Commerce* (2012), ACM, pp. 18–35.

[3] Alaei, S., Hajiaghayi, M., and Liaghat, V. The online stochastic generalized assignment problem. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*. Springer, 2013, pp. 11–25.

[4] Assadi, S., Khanna, S., and Li, Y. The stochastic matching problem with (very) few queries. In *Proceedings of the 2016 ACM Conference on Economics and Computation* (2016), ACM, pp. 43–60.

[5] Brubach, B., Sankararaman, K. A., Srinivasan, A., and 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* (Richland, SC, 2017), AAMAS '17, International Foundation for Autonomous Agents and Multiagent Systems, pp. 1223–1231.

[6] Devanur, N. R., and 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* (2009), ACM, pp. 71–78.

[7] Devanur, N. R., Jain, K., Sivan, B., and 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* (2011), ACM, pp. 29–38.

[8] Devanur, N. R., Sivan, B., and Azar, Y. Asymptotically optimal algorithm for stochastic adwords. In *Proceedings of the 13th ACM Conference on Electronic Commerce* (2012), ACM, pp. 388–404.

[9] Feldman, J., Korula, N., Mirrokni, V., Muthukrishnan, S., and Pál, M. Online ad assignment with free disposal. In *Internet and network economics*. Springer, 2009, pp. 374–385.

[10] Feldman, J., Mehta, A., Mirrokni, V., and Muthukrishnan, S. Online stochastic matching: Beating 1-1/e. In *Foundations of Computer Science (FOCS)* (2009), IEEE, pp. 117–126.

[11] Gandhi, R., Khuller, S., Parthasarathy, S., and Srinivasan, A. Dependent rounding and its applications to approximation algorithms. *Journal of the ACM (JACM) 53*, 3 (2006), 324–360.

[12] Haeupler, B., Mirrokni, V. S., and Zadimoghaddam, M. Online stochastic weighted matching: Improved approximation algorithms. In *Internet and Network Economics*, vol. 7090 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2011, pp. 170–181.

[13] Jaillet, P., and Lu, X. Online stochastic matching: New algorithms with better bounds. *Mathematics of Operations Research 39*, 3 (2013), 624–646.

[14] Karp, R. M., Vazirani, U. V., and Vazirani, V. V. An optimal algorithm for on-line bipartite matching. In *Proceedings of the twenty-second annual ACM symposium on Theory of computing* (1990), ACM, pp. 352–358.

[15] Kesselheim, T., Radke, K., Tönnis, A., and Vöcking, B. An optimal online algorithm for weighted bipartite matching and extensions to combinatorial auctions. In *European Symposium on Algorithms (ESA)*. Springer, 2013, pp. 589–600.

[16] Korula, N., and Pál, M. Algorithms for secretary problems on graphs and hypergraphs. In *Automata, Languages and Programming*. Springer, 2009, pp. 508–520.

[17] Lee, Y. T., and Sidford, A. Efficient inverse maintenance and faster algorithms for linear programming. In *Foundations of Computer Science (FOCS), 2015 IEEE 56th Annual Symposium on* (2015), IEEE, pp. 230–249.

[18] Mahdian, M., and 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* (2011), ACM, pp. 597–606.

[19] Manshadi, V. H., Gharan, S. O., and Saberi, A. Online stochastic matching: Online actions based on offline statistics. *Mathematics of Operations Research 37*, 4 (2012), 559–573.

[20] Mehta, A. Online matching and ad allocation. *Foundations and Trends in Theoretical Computer Science 8*, 4 (2012), 265–368.

[21] Mehta, A., and Panigrahi, D. Online matching with stochastic rewards. In *Foundations of Computer Science (FOCS)* (2012), IEEE, pp. 728–737.

[22] Mehta, A., Waggoner, B., and Zadimoghaddam, M. Online stochastic matching with unequal probabilities. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms* (2015), SIAM.

A Appendix

A.1 Complementary materials in section 3 (Edge-weighted model).

A.1.1 Proof of Lemma 2

We will prove Lemma 2 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 4. *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 5. *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 6. *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 2.

Proof of Claim 4

Proof. 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 $\mathsf{PM}[\mathbf{F}, 3]$. From the property of $\mathsf{PM}[\mathbf{F}, 3]$, we know 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 1 we have

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

For α_3, we have

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

Hence, we have

$$\Pr[e \text{ is matched}] = \frac{1}{3}\sum_{i=1}^{3} \Pr[e \text{ is matched} \mid \alpha_i] \geq \frac{2}{3}\mathsf{R}[\mathsf{EW}_1, 2/3]$$

where $\mathsf{R}[\mathsf{EW}_1, 2/3] = 0.67529 + (1-h) * 0.00446489$. □

Proof of Claims 5 and 6

Proof. 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 $\psi_i,\ 1 \leq i \leq 3$ is $1/3$.

Similar to the calculations in the proof of Claim 4, we have $\Pr[e \text{ is matched}| \ \psi_1] \geq 0.571861$, $\Pr[e \text{ is matched}| \ \psi_2] \geq 0.144776$ and $\Pr[e \text{ is matched}| \ \psi_3] \geq 0.0344288$. 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 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 3

We will prove Lemma 3 using the following two Claims.

Claim 7. *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 8. *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 3.

Proof of Claim 7

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

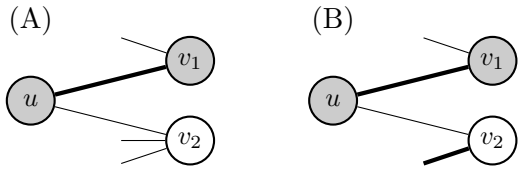


Figure 5: Diagram of configurations for a large edge $e = (u, v_1)$. Thin and Thick lines represent small and large edges respectively.

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 $y_1/3$ and $\Pr[e \text{ is matched} \mid \alpha_1] \geq 0.432332$.

- α_2: $e \in M_1$ and $e' \in M_2$. This happens with probability $y_2/3$ and $\Pr[e \text{ is matched} \mid \alpha_2] \geq 0.580831$.

- α_3: $e \in M_1$ and $e' \notin M_1, e' \notin M_2$. This happens with probability $(1 - y_1/3 - y_2/3)$ and $\Pr[e \text{ is matched}| \ \alpha_1] \geq 0.632121$.

Therefore we have

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

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 7. □

Proof of Claim 8

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

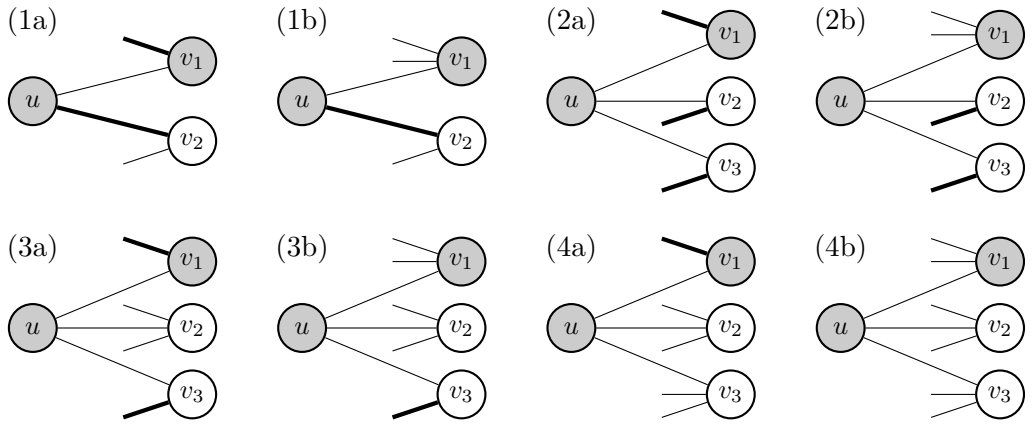


Figure 6: Diagram of configurations for a small edge $e = (u, v_1)$. Thin and Thick lines represent small and large edges respectively.

Similar to the proof of Claim 7, 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 conditional 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

$$\Pr[\mathcal{E}] = y_1/3 * \Pr[\mathcal{E}\,|\alpha_1] + y_2/3 * \Pr[\mathcal{E}\,|\alpha_2] + (1 - y_1/3 - y_2/3) * \Pr[\mathcal{E}\,|\alpha_3]$$

$$\geq \frac{1}{3}(0.13171y_1 + 0.200568y_2 + (3 - y_1 - y_2)0.22933)$$

- $(3b)$. 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

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

- $(4a)$. 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 $\Pr[\mathcal{E}\,|\alpha_3] = 0.2642$.
  - 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 $\Pr[\mathcal{E}\,|\alpha_4] = 0.1317$.
  - 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 $\Pr[\mathcal{E}\,|\alpha_5] = 0.14849$.
  - 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 $\Pr[\mathcal{E}\,|\alpha_6] = 0.22933$.

  Therefore we have

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

- $(4b)$. Observe that in this configuration, we have additional six cases to the ones discussed in $(4a)$. Let α_i be the cases defined in $(4a)$ 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 $\Pr[\mathcal{E}\,|\alpha_1] = 0.3167$.
  - $\beta_2 : e_1 \in M_1, e_2 \in M_2, e_3 \in M_2$. $\Pr[\alpha_2] = y_1y_2^2/27$ and $\Pr[\mathcal{E}\,|\alpha_2] = 0.5374$.
  - $\beta_3 : e_1 \in M_1, e_2 \in M_0, e_3 \in M_0$. $\Pr[\alpha_3] = y_1/3 * (1 - y_1/3 - y_2/3)^2$ and $\Pr[\mathcal{E}\,|\alpha_3] = 0.632$.
  - 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 $\Pr[\mathcal{E}\,|\alpha_4] = 0.4057$.
  - 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 $\Pr[\mathcal{E}\,|\alpha_5] = 0.4323$.
  - 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 $\Pr[\mathcal{E}\,|\alpha_6] = 0.58083$.

Hence, we have

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

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 $(1a)$ and $(1b)$. □

A.2 Supplemental materials in section 4

A.2.1 Proof of Lemma 7 (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.

Claim 9. *If $H_u = 1$ and u is not in C_1, then we have* $\mathsf{R}[\mathsf{RLA}, 1] \geq 0.735622$.

Claim 10. $\mathsf{R}[\mathsf{RLA}, 2/3] \geq 0.7870$.

Claim 11. $\mathsf{R}[\mathsf{RLA}, 1/3] \geq 0.8107$.

Recall that B_u is the event that among the n random lists, there exists a list starting with u and G_u^v 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 P_u is the probability that u gets matched in $\mathsf{RLA}[\mathbf{H}']$. For each u, we compute $\Pr[B_u]$ and $\Pr[G_u^v]$ for all possibilities of $v \sim u$ and using Lemma 6 we get P_u. First, we discuss two different ways to calculate $\Pr[G_u^v]$. For some cases, we use a direct calculation, while for the rest we use the Markov-chain approximation method.

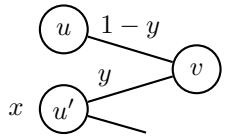


Figure 7: Case 1 in calculation of $\Pr[G_u^v]$

Two ways to compute the value $\Pr[G_u^v]$

1. **Case 1**: Consider the case when v has two neighbors as shown in Figure 7. Assume v has two neighbors u and u' and after modifications, $H'_{(u',v)} = y$, $H'_{(u,v)} = 1 - y$ and $H'_{u'} = x$. Thus, the second certificate event G_u^v 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

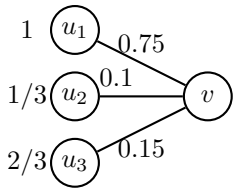


Figure 8: Case 2 in calculation of $\Pr[G_u^v]$

$i < j \leq n$. 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

$$\Pr[G_u^v] = \sum_{i=1}^{n-1} \left(x/n(1-x/n)^{(i-1)}(1-(1-y/n)^{(n-i)} \right) \quad (19)$$

$$\sim \frac{x - e^{-y}x + (-1+e^{-x})y}{x-y} \quad (\text{if } x \neq y) \quad (20)$$

$$\sim 1 - e^{-x}(1+x) \quad (\text{if } x = y) \quad (21)$$

2. **Case 2**: Consider the case when v has three neighbors. The value $\Pr[G_u^v]$ is approximated using the Markov Chain method, similar to [13]. Let us use the following example to illustrate the method.

   Consider the following case as shown in Figure 8 (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 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 $\Pr[G_u^v]$. The one-step transition probability matrix M is shown as follows.

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

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 9, 10 and 11. 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 9

Notice that u is not in the cycle C_1 and thus Lemma 6 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.)

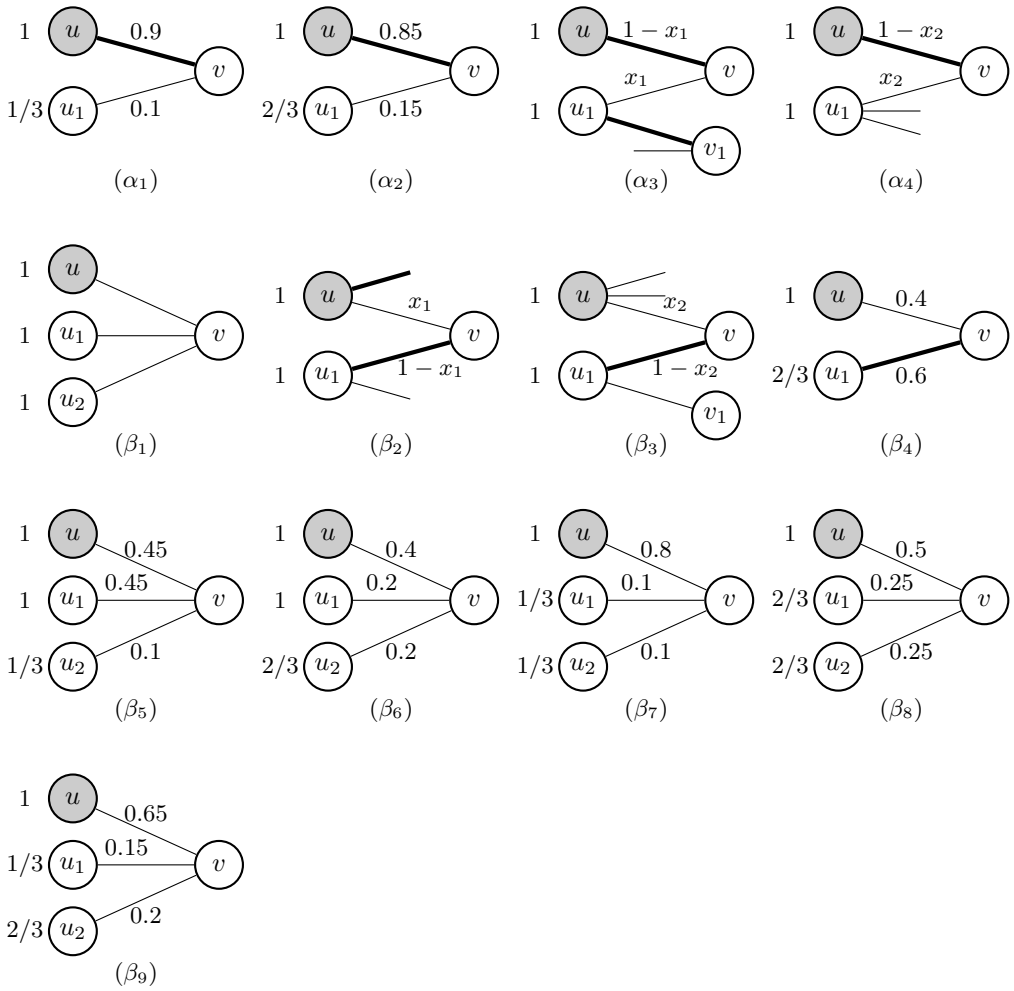


Figure 9: Vertex-weighted $H_u = 1$ cases. The value assigned to each edge represents the value after the second modification. No value indicates no modification. Here, $x_1 = 0.2744$ and $x_2 = 0.15877$.

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 $\alpha_i, 1 \leq i \leq 4$ and v_2 is chosen from some $\beta_i, 1 \leq i \leq 9$. For $H_u = 1$, we need to find the worst combination among these such that the value

P_u is minimized. We can find this WS using the Lemma 6.

For each type of α_i, β_j, we compute the values it will contribute to the term $(1 - B_u)\prod_{v\sim u}(1 - \Pr[G_u^v])$. For example, assume v_1 is of type α_1, denoted by $v_1(\alpha_1)$. It contributes $e^{-0.9}$ to the term $(1 - B_u)$ and $(1 - \Pr[G_u^{v_1}])$ to $\prod_{v\sim u}(1 - \Pr[G_u^v])$, thus the total value it contributes is $\gamma(v_1, \alpha_1) = e^{-0.9}(1 - \Pr[G_u^{v_1}])$. 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 P_u and $\mathsf{R}[\mathsf{RLA}, 1]$ is as follows:

$$\mathsf{P}_u = 1 - \gamma(v_1, \alpha_{i^*})\gamma(v_2, \beta_{j^*})$$

$$\mathsf{R}[\mathsf{RLA}, 1] = \mathsf{P}_u / H_u = \mathsf{P}_u$$

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 $\Pr[G_u^v] = 1 - e^{-0.1} * 1.1$ and $\gamma(v, \alpha_1) = e^{-0.1} * 1.1 * e^{-0.9} = 0.404667$.

- α_2: $\Pr[G_u^v] \ge 1 - e^{-0.15} * 1.15$ and $\gamma(v, \alpha_2) \le 0.423$.

  Notice that after modifications, $H'_{u_1} \ge 0.15$. Hence, we use this and Equation 19 to compute the lower bound of $\Pr[G_u^v]$.

- α_3: $\Pr[G_u^v] \ge 0.0916792$ and $\gamma(v, \alpha_3) \le 0.439667$.

  Notice that for any large edge e incident to a node u with $H_u = 1$ (before modification), we have after modification, $H'_e \ge 1 - 0.2744 = 0.7256$. Thus we have $H'_{(u_1,v_1)} \ge 0.7256$ and $H'_{u_1} \ge 1$. From Equation 19, we get $\Pr[G_u^v] \ge 0.0916792$.

- α_4: $\Pr[G_u^v] \ge 0.0307466$ and $\gamma(v, \alpha_4) \le 0.417923$.

  Notice that for any small edge e incident to a node u with $H_u = 1$ (before modification), we have after modification, $H'_e \ge 0.15877$. Thus, we have $H'_{u_1} \ge 3 * 0.15877$.

- β_1: $\Pr[G_u^v] = 0.1608$ and $\gamma(v, \beta_1) = 0.601313$.

- β_2: $\Pr[G_u^v] \ge 0.208812$ and $\gamma(v, \beta_2) \le 0.601313$.

  After modifications, we have $H'_{(u_1,v_1)} \ge 0.2744$ and thus we get $H'_{u_1} \ge 1$.

- β_3: $\Pr[G_u^v] \ge 0.251611$ and $\gamma(v, \beta_2) \le 0.63852$.

  After modifications, we have $H'_{(u_1,v_1)} \ge 0.2744$ and thus we get $H'_{u_1} \ge 1 - 0.15877 + 0.2744$.

- β_4: $\Pr[G_u^v] = 0.121901$ and $\gamma(v, \beta_4) = 0.588607$.

- β_5: $\Pr[G_u^v] = 0.1346$ and $\gamma(v, \beta_5) = 0.551803$.

- β_6: $\Pr[G_u^v] \geq 0.1140$ and $\gamma(v, \beta_6) \leq 0.593904$.

- β_7: $\Pr[G_u^v] = 0.0084$ and $\gamma(v, \beta_7) = 0.4455$.

- β_8: $\Pr[G_u^v] \geq 0.0397$ and $\gamma(v, \beta_8) \leq 0.582451$.

- β_9: $\Pr[G_u^v] \geq 0.0230$ 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 10

The proof is similar to that of Claim 9. The Figure 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 $\alpha_i, 1 \leq i \leq 3$; if $H_{(u,v)} = 1/3$, then v is in one of $\beta_i, 1 \leq i \leq 8$. 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 $\Pr[G_u^v] = 1 - e^{-0.25} * 1.25$ and $\gamma(v, \alpha_1) = e^{-0.25} * 1.25 * e^{-0.75} = 0.459849$.

- α_2: We have $\Pr[G_u^v] \geq 0.0528016$ and $\gamma(v, \alpha_1) \leq 0.470365$.

- α_3. We have $\Pr[G_u^v] \geq 0.13398$ and $\gamma(v, \alpha_3) \leq 0.475282$.

- β_1: We have $\Pr[G_u^v] = 1 - e^{-0.7} * 1.7$ and $\gamma(v, \beta_1) = 0.625395$.

- β_2: We have $\Pr[G_u^v] \geq 0.226356$ and $\gamma(v, \beta_2) \leq 0.665882$.

- β_3: We have $\Pr[G_u^v] \geq 0.1819$ and $\gamma(v, \beta_3) \leq 0.669804$.

- β_4: We have $\Pr[G_u^v] \geq 0.1130$ and $\gamma(v, \beta_4) \leq 0.635563$.

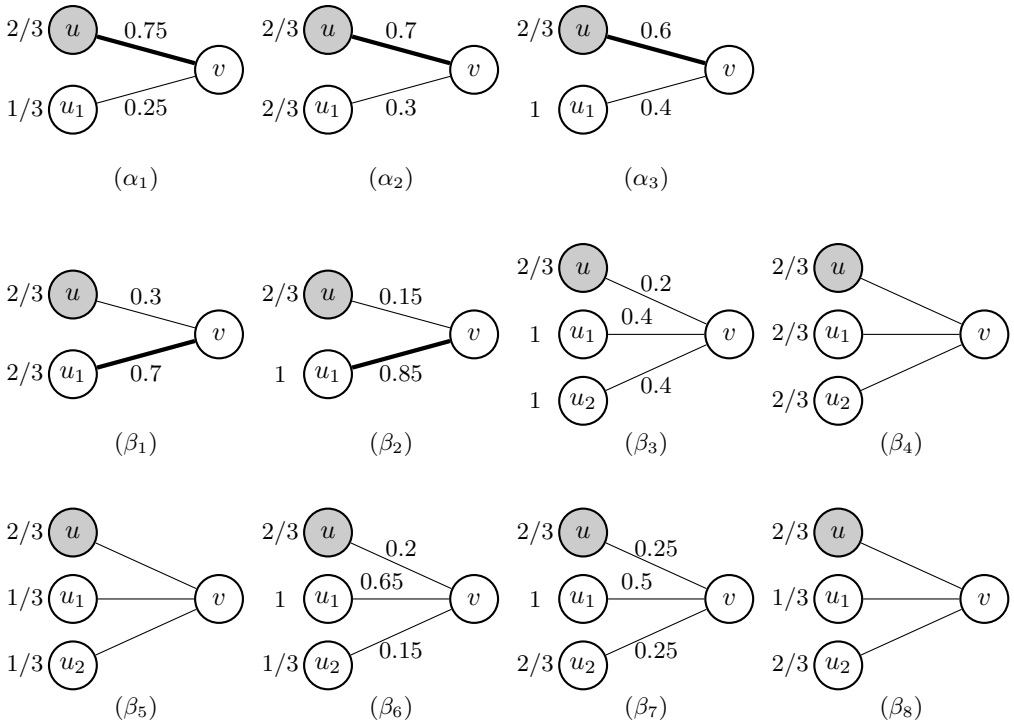


Figure 10: Vertex-weighted $H_u = 2/3$ cases. The value assigned to each edge represents the value after the second modification. No value indicates no modification.

- β_5: We have $\Pr[G_u^v] \geq 0.0587$ and $\gamma(v, \beta_5) \leq 0.674471$.

- β_6: We have $\Pr[G_u^v] \geq 0.1688$ and $\gamma(v, \beta_6) \leq 0.680529$.

- β_7: We have $\Pr[G_u^v] \geq 0.1318$ and $\gamma(v, \beta_7) \leq 0.676155$.

- β_8: We have $\Pr[G_u^v] \geq 0.0587$ 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 11

The Figure 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 $\Pr[G_u^v] = 1 - e^{-0.75} * 1.75$ and $\gamma(v, \alpha_1) = 0.643789$.

- α_2: We have $\Pr[G_u^v] \geq 0.282256$ and $\gamma(v, \alpha_2) \leq 0.649443$.

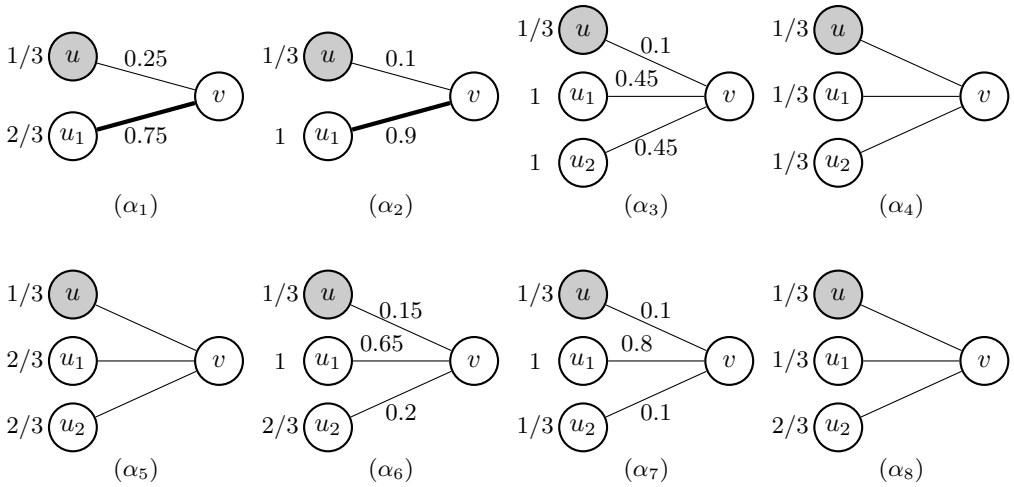


Figure 11: Vertex-weighted $H_u = 1/3$ cases. The value assigned to each edge represents the value after the second modification. No value indicates no modification.

- α_3: We have $\Pr[G_u^v] \geq 0.1935$ and $\gamma(v, \alpha_3) \leq 0.729751$.
- α_4: We have $\Pr[G_u^v] \geq 0.0587$ and $\gamma(v, \alpha_4) \leq 0.674471$.
- α_5: $\gamma(v, \alpha_5) \leq 0.674471$.
- α_6: We have $\Pr[G_u^v] \geq 0.1546$ and $\gamma(v, \alpha_6) \leq 0.727643$.
- α_7: We have $\Pr[G_u^v] \geq 0.1938$ 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.