Randomized Online Matching in Regular Graphs

Ilan Reuven Cohen *‡
The Hebrew University of Jerusalem

David Wajc †‡
Carnegie Mellon University

Abstract

In this paper we study the classic online matching problem, introduced in the seminal work of Karp, Vazirani and Vazirani (STOC 1990), in *regular* graphs. For such graphs, an optimal deterministic algorithm as well as efficient algorithms under stochastic input assumptions were known. In this work, we present a novel randomized algorithm with competitive ratio tending to *one* on this family of graphs, under *adversarial* arrival order. Our main contribution is a novel algorithm which achieves competitive ratio $1-O\left(\sqrt{\log d}/\sqrt{d}\right)$ in expectation on d-regular graphs. In contrast, we show that all previously-studied online algorithms have competitive ratio strictly bounded away from one. Moreover, we show the convergence rate of our algorithm's competitive ratio to one is nearly tight, as no algorithm achieves competitive ratio better than $1-O\left(1/\sqrt{d}\right)$. Finally, we show that our algorithm yields a similar competitive ratio with high probability, as well as guaranteeing each vertex a probability of being matched tending to *one*.

1 Introduction

The maximum matching problem constitutes one of the most fundamental problems of computer science, with countless practical and theoretical applications. This problem has proven fertile soil for algorithmic study, giving rise to precursors of key ideas in optimization theory, including linear programming duality and the primal-dual method [25, 46, 47], and even the equation of efficiency with polynomial-time computability [24]. See the books of Lovász and Plummer [48] and Ahuja, Magnanti, and Orlin [4] for an extensive treatise of some of the classic results and techniques pertaining to these problems.

A particularly well-studied instance of the maximum matching problem is maximum matching in d-regular bipartite graphs; i.e., bipartite graphs in which each vertex neighbors d other vertices. The class of bipartite d-regular graphs have been studied in many contexts, including expander graph constructions, scheduling, routing in switch fabrics, and task assignment [3, 17, 55]. In the context of matching theory, a consequence of Hall's Theorem [37] implies that such graphs can be decomposed into d disjoint perfect matchings. Indeed, the existence of a perfect matching in bipartite regular graphs, first proved by König [45], is one of the seminal results in matching theory.

On the algorithmic front, Gabow and Kariv [29] presented an elegant $O(m)$-time algorithm for finding a perfect matching in m-edge d-regular graphs for d an integer power of two. (In contrast, for general bipartite graphs, the best current maximum matching algorithms require $\omega(m)$ time [38, 49].) Subsequent work, including [5, 16, 17, 18, 58], finally culminated in an $O(m)$-time perfect matching algorithm for all d. This linear-time bound was later proven to be optimal for deterministic algorithms by Goel, Kapralov and Khanna [34]. Moreover, those authors [32, 33, 34] presented *sublinear*-time randomized algorithms for this problem, finally presenting an $O(n \log n)$-time algorithm for perfect matching in n-vertex regular bipartite graphs.

Online Matching. Given the importance of maximum matching in the classic offline model of computation, it is no surprise that it was also one of the first problems to be considered in an online setting. In 1990, in a seminal paper, Karp, Vazirani and Vazirani [43] introduced the problem of online matching in general (not necessarily regular) bipartite graphs under vertex arrivals in one side of the graph. Karp et al. showed that any deterministic algorithm, as well as the natural randomized algorithm relying on uncorrelated randomness, are no better than 1/2-competitive. More interestingly, they presented an elegant use of correlated randomness which yields a competitive ratio of $1-1/e$ for this problem, and proved this bound is best possible for any randomized algorithm.

The emergence of Internet advertising proved to be a catalyst for the resurgence of interest in online matching problems and their generalizations. In [54] Mehta et al. presented an (optimal) $1-1/e$-competitive

*email: ilanrcohen@gmail.com. Supported in part by the I-CORE program (center no. 4/11).

†email: dwajc@cs.cmu.edu. Supported in part by NSF grants CCF-1527110 and CCF-1618280.

‡This work conducted in part while the authors were visiting the Simons Institute for the Theory of Computing.

algorithm for the adwords problem; generally, they related online matching and its extensions to Internet advertising, which sparked a flurry of research in this field. See e.g. [8, 19, 27, 28, 35, 36, 39, 42, 50, 52] for a partial list of such work and Mehta [53] for a survey of recent developments in the field. We point out the work of Aggarwal et al. [2], who obtained a similar optimal $1 - 1/e$ bound for the vertex-weighted online matching problem (where offline vertices have some weight and the goal is to maximize the weight of matched offline vertices), a problem which we also consider in this paper.

Breaking the $1-1/e$ Barrier. The $1-1/e$ hardness result of Karp et al. [43] for online matching is prevalent in all this problem's extensions. However, the practical importance of these problems for Internet advertising have motivated researchers to seek a better theoretical understanding of beyond worst-case guarantees.

One such line of research considers stochastic arrival models; in particular, random order arrival and iid arrival models. In the random order model some input graph is fixed ahead of time and the online vertices are randomly permuted. In the iid arrival model, online vertices are drawn from some (known or unknown) distribution. For these arrival models, results beating the $1-1/e$ competitive ratio were shown both for online matching [8, 28, 42, 51, 52] and online vertex-weighted matching [11, 36, 39]. For example, the RANKING algorithm of Karp et al. [43] is the current state-of-the-art for online matching in random order, obtaining competitive ratio of 0.696 [51], breaking the $1 - 1/e \approx 0.632$ barrier in this model. On the other hand, even these stochastic assumptions have their limitations, and no algorithm can achieve competitive ratio better than 0.823 for these arrival models [52].

A different approach considers structural assumptions on the input. For the adwords problem, Buchbinder et al. [12] showed that the natural assumption of online vertices having degree at most some d allows for a competitive ratio of $1 - (1 - 1/d)^d$. Azar et al. [7] later showed this bound to be optimal. For online matching and vertex-weighted matching, Naor and Wajc [56] showed that adding the assumption that offline vertices have degree at least k allows for a deterministic algorithm with competitive ratio $1-(1-1/d)^k$. In particular, for d-regular graphs, they showed the optimal competitive ratio for deterministic algorithms is $1 - (1 - 1/d)^d$. Deterministic algorithms on d-regular graphs therefore outperform the $1 - 1/e$ optimal bound of randomized algorithms in general (non-regular) graphs; however, as d grows, deterministic algorithms' competitive ratio deteriorates back to $1-1/e$.

Online Matching in Regular Graphs. In this work, we study the classic problem of online matching, in the class of d-regular graphs. For this class deterministic algorithms fare better than on general graphs. A natural question thus arises: "what is the optimal competitive ratio for *randomized* online matching algorithms on regular graphs?" We address this question here, showing that while the problem becomes inherently harder for deterministic algorithms as d grows (by [56]), it becomes easier for randomized algorithms.

1.1 Our Results

Our main contribution is a new randomized algorithm, MARKING, which achieves competitive ratio tending to *one* as d grows on d-regular graphs.

Theorem 1.1. *Algorithm* MARKING *achieves competitive ratio* $(1-\frac{2\sqrt{H_d}}{\sqrt{d}}) = 1-O(\frac{\sqrt{\log d}}{\sqrt{d}})$ *on d-regular graphs.*

We show the convergence of our algorithm's competitive ratio to one is near optimal.

Theorem 1.2. *No randomized online matching algorithm is better than* $\left(1-\frac{1}{\sqrt{8\pi d}}\right) = 1-O(\frac{1}{\sqrt{d}})$*-competitive on d-regular graphs.*

We further show our algorithm achieves similar competitive ratio with high probability.

Theorem 1.3. *Algorithm* MARKING *is* $1 - O(\frac{\log n}{\sqrt{d}})$*-competitive w.h.p. on n-vertex d-regular graphs.*

Finally, we show our algorithm matches each vertex with probability tending to one, implying a high competitive ratio for weighted online matching variants (more on this below).

Theorem 1.4. *A modification of Algorithm* MARKING *guarantees each vertex (both offline and online) a probability of* $1 - O\left(\frac{\sqrt[3]{\log d}}{\sqrt[3]{d}}\right)$ *of being matched in d-regular graphs. Algorithm* MARKING *itself matches each vertex with probability at least* $1 - O\left(\frac{\sqrt{\log d}}{\sqrt[4]{d}}\right)$.

Remark 1. We note that Karande et al. [42] and Bahmani and Kapralov [8] showed that algorithms RANKING and RANDOM are $1 - O(\frac{1}{\sqrt{d}})$-competitive on d-regular graphs in the random order arrival model and iid arrival model, respectively.[1] In contrast, we show

[1] More generally, Karande et al. [42] showed that RANKING is $1 - O(\frac{1}{\sqrt{k}})$-competitive on inputs with k edge-disjoint perfect matchings in the random order model. As d-regular graphs can be partitioned into d edge-disjoint perfect matchings, RANKING is therefore $1-O(\frac{1}{\sqrt{d}})$-competitive on d-regular graphs in this model. However, as we show in the full version of the paper, a graph containing many edge-disjoint perfect matchings does not imply even a constant additive improvement over the optimal $1 - 1/e$ competitive ratio in the adversarial arrival model.

that for the stricter adversarial arrival model algorithms RANKING and RANDOM (and many other natural algorithms) do not achieve competitive ratio tending to one as d grows on d-regular graphs. Nonetheless, we present a new online matching algorithm which achieves similar $1 - \tilde{O}(\frac{1}{\sqrt{d}})$ bounds on d-regular graphs in the stricter *adversarial* arrival model.

Remark 2. Previous hardness results for online matching and related problems can all be recast as hardness results for fractional algorithms. This method does not apply to our settings. Instead, our hardness result explicitly considers variance of the integral solution (see Section 8).

Remark 3. To the best of our knowledge, Theorem 1.3 is the first non-trivial (i.e., beating 1/2) high-probability result for online matching in the adversarial arrival model.

Remark 4. Note that this "fairness" property of matching each vertex with probability at least c implies a c-competitive ratio for vertex-weighted online matching, where weights are given to both offline and online vertices. Our algorithm is therefore the first to beat the 1/2-competitive ratio on any non-trivial graph class for the vertex-weighted variant with weights assigned to *online* vertices.

Online Dependent Rounding Throughout this paper, for simplicity of exposition we focus on matching in regular graphs. In the paper's full version we present our general algorithm: an online dependent rounding scheme which yields similar results for graphs admitting an efficient fractional online matching algorithm inducing fractional solutions with low "variance". Dependent rounding schemes have been studied extensively in the offline setting [1, 6, 10, 14, 20, 21, 22, 30] and have found many applications (e.g., [1, 9, 13, 15, 30]). In the full version we present further applications of our online dependent rounding scheme. For example, we obtain a $1 - O(1/\sqrt[3]{d})$-competitive algorithm for graphs with minimum online degree at least d and maximum offline degree at most d, and more generally with each offline vertex having degree at most equal to the harmonic mean of its neighbors' degrees. Similarly, we obtain high competitive ratio if online vertices have low degree and offline vertices have high degree – a scenario relevant to computational advertising. Finally, we use our dependent rounding scheme to obtain sublinear-time $(1 - o(1))$-approximate matching algorithms that are faster than the $O(n \log n)$-time algorithm of Goel et al. [34], in the more restrictive *online* setting.[2]

[2]We thank Noga Alon for pointing out some of these applications and encouraging us to explore applications of our approach to sublinear-time algorithms.

1.2 Our Techniques Our randomized algorithm is guided by the optimal fractional online matching algorithm, which assigns a value of $1/d$ to each edge. Consequently, we strive to give every edge (i, t) a marginal probability of roughly $1/d$ of being included in the matching. This property would imply the probability of each offline vertex i to be matched after arrival time t is equal to $1/d$ times i's *current degree*; that is, the number of previously-arrived neighbors of this offline vertex. While the goal of guaranteeing $1/d$ marginal probability per edge of each online vertex may tempt us to consider choosing matches fairly (uniformly at random) among the unmatched neighbors, this approach does not guarantee the right marginals, as the following example shows.

Example 1. Suppose an online vertex t has $d/2$ neighbors of current degree one (unmatched, as t is their first neighbor) and $d/2$ neighbors of current degree $d/2 + 1$ (each previously matched with probability 1/2, independently). The expected number of non-matched high-degree vertices is $d/4$, and this number is sharply concentrated; thus, the marginal probability of (i, t) for high-degree vertices i is only roughly $2/(3d)$ and for low-degree i, this marginal probability is roughly $4/(3d)$.[3]

This example suggests that in order to obtain near-optimal marginal probabilities, an algorithm must discriminate among offline neighbors based on their current degree. For the above example, the following local allocation rule suffices to closely approximate the proper marginal probabilities. We assign each offline vertex a weight inversely proportional to its *residual degree* (that is, d minus its current degree) and match an arriving online vertex to an unmatched neighbor with probability proportional to its weight. In the previous example, the high-degree vertices would be given a weight of $2/d$ while the low-degree vertices would be assigned a weight of $1/d$. All of t's neighbors will then be matched to t with probability roughly $1/d$.

More generally, this weight-based allocation rule maintains roughly $1/d$ marginals, assuming independence. However, our algorithm must further consider correlations, as an offline vertex i's probability of being matched to some online neighbor t depends on the set of unmatched neighbors of t *conditioned on i being unmatched* and may therefore be hard to bound. Crucially, positive correlation between offline vertices' probability of being matched may harm any algorithm's performance, as the following example illustrates.

Example 2. Consider an online vertex t with d neighbors of current degree $d/2 + 1$ which are perfectly

[3]Later we formalize and expand this example to show that the RANDOM algorithm, which picks a match uniformly at random among unmatched neighbors, is $1 - \Omega(1)$-competitive.

positively correlated; specifically, either all of t's neighbors are matched prior to t's arrival, with probability $1/2$, or none are. Then, the marginal probability of each edge (i,t) is only $1/(2d)$.

Unfortunately, every maximally-matching algorithm has d-regular graphs for which it creates positive correlations between some offline vertices. These positive correlations seem difficult to control. With this observation in mind, our algorithm will not be maximal, turning down (some) possible matches.

To avoid positive correlations, our algorithm relies on the more robust notion of **marked** offline vertices. Marked offline vertices will form a superset of the matched offline vertices, and will never be eligible to be matched. The ability to mark more than one offline vertex (or none) following each arrival allows us to ensure each offline vertex a marginal probability of *precisely* $1/d$ of being marked. In particular, for any pair (i,t), our algorithm guarantees a fixed probability of marking the unmarked offline vertex i upon arrival of t *independently* of the realization of t's other unmarked neighbor set.[4] Specifically, if the sum of the weights of t's unmarked neighbors is higher than its mean, each neighbor is also marked, without being matched, independently with some correcting probability (determined by the sum of weights' deviation from its mean). Conversely, if the total weight is smaller than its mean, the algorithm marks no neighbor (and so does not match t) with probability proportional to the deviation.

For the above marking procedure, the indicators for pair (i,t) incurring a mark satisfy both previously-discussed positive properties: $1/d$ marginal probabilities and *negative correlation*. The latter property allows us to show the total weight of the unmarked neighbors is concentrated, and as such allows us to bound the deviation, which corresponds to the expected loss of the algorithm.

Moreover, the negative correlation of these indicators allows us to appeal to high concentration bounds, in the form of Chernoff's and Bernstein's inequality on (weighted) sums of these variables. This allows us to bound the sum of weights of neighbors of online vertices, showing it does not deviate much from its expectation. As such deviation determines our algorithm's probability of matching edges, the concentration of this sum yields both our high probability and per-vertex guarantees.

[4] Of course, marking each offline neighbor independently would satisfy the above properties. However, this approach would cause each online vertex to not mark *any* of its neighbors, and in particular would cause it to not be matched, with probability $(1-1/d)^d \to 1/e$. Our algorithm must therefore mark its neighbors using *correlated* randomness.

1.3 Paper Outline We start by formally defining the problems we consider and outlining some negative dependence properties we use in our analysis, in Section 2. We then analyze several natural randomized online matching algorithms in the context of regular graphs, including the well-studied RANKING and RANDOM algorithms and show that these algorithms are highly suboptimal on this family of inputs, in Section 3. We then introduce our randomized algorithm and prove some useful properties of this algorithm used for its analysis, including its aforementioned negative dependence properties, in Section 4. Using these properties, we then proceed to bound our algorithm's competitive ratio in expectation and with high probability in Sections 5 and 6 respectively, as well as bounding its per-vertex guarantees in Section 7. Finally, we complement our positive results with a nearly-matching lower bound in Section 8.

2 Problem Definitions and Preliminaries

An instance $\mathcal{I}$ of the online matching problem consists of a bipartite graph $G=(L,R,E)$. The left hand side vertices, which we refer to as *offline* vertices, are known a priori; the right hand side vertices, which we refer to as *online* vertices, are revealed over time. WLOG, we assume the online vertices are numbered $1,2,\ldots,|R|$. The t-th online vertex is revealed at time step t together with all of its edges. At this point, an online matching algorithm immediately and irrevocably choose which of t's unmatched neighbors to match t to (if any). An instance of the online vertex-weighted matching problem consists similarly of a bipartite graph with similar arrival dynamics, with the vertices associated with a weight function $w: V \to \mathbb{R}^+$. Online matching is the special case of online vertex-weighted matching where the weights are all one.

For an online vertex-weighted matching algorithm $\mathcal{A}$ and instance $\mathcal{I}$, denote by $M_{\mathcal{A}}(\mathcal{I})$ the matching output by $\mathcal{A}$ on $\mathcal{I}$ and by $OPT(\mathcal{I})$ a maximum-weight matching on $\mathcal{I}$, where the weight of a matching M is simply the sum of its matched vertices' weights; that is, $w(M) \triangleq \sum_{v \in V(M)} w(v)$. Algorithm $\mathcal{A}$ is *c-competitive* on instance $\mathcal{I}$ if $\mathbb{E}[w(M_{\mathcal{A}}(\mathcal{I}))] \geq c \cdot w(OPT(\mathcal{I}))$. Moreover, algorithm $\mathcal{A}$ is said to be *c-competitive* on a family of instances $\mathcal{F}$ if it is c-competitive on every instance $\mathcal{I} \in \mathcal{F}$. Naturally, one seeks an algorithm of highest possible competitive ratio on families of interest.

2.1 Negative Dependence Properties. In our analysis we rely on several notions of negative dependence between random variables. In particular, at the heart of our analysis is the notion of *negative associa-*

tion, introduced by Khursheed and Lai Saxena [44] and Joag-Dev and Proschan [40].

Definition 2.1 (Negative Association [40, 44]). *A joint distribution $X_1, X_2, \ldots, X_n$ is said to be* negatively associated (NA) *if for any two functions f, g both monotone increasing or both monotone decreasing, with $f(\vec{X})$ and $g(\vec{X})$ depending on disjoint subsets of the X_i, $f(\vec{X})$ and $g(\vec{X})$ are negatively correlated; i.e.,*

$$\mathbb{E}[f(\vec{X}) \cdot g(\vec{X})] \leq \mathbb{E}[f(\vec{X})] \cdot \mathbb{E}[g(\vec{X})].$$

Clearly, independent random variables are NA. Another class of NA distributions is captured by the *zero-one rule*. This rule asserts that if $X_1, X_2, \ldots, X_n$ are zero-one random variables whose sum is always at most one, $\sum_i X_i \leq 1$, then $X_1, X_2, \ldots, X_n$ are NA (see [23]). Additional, more complex, NA distributions can be "built" from simpler NA distributions using the following closure properties.

(P1) Independent Union.
If $X_1, X_2, \ldots, X_n$ are NA, $Y_1, Y_2, \ldots, Y_m$ are NA, and $\{X_i\}_i$ are independent of $\{Y_j\}_j$, then $X_1, X_2, \ldots, X_n, Y_1, Y_2, \ldots, Y_m$ are NA.

(P2) Concordant monotone functions.
Let $f_1, f_2, \ldots, f_k : \mathbb{R}^n \to \mathbb{R}$ be all monotone increasing or all monotone decreasing, with the $f_i(\vec{X})$ depending on disjoint subsets of $X_1, X_2, \ldots, X_n$. Then, if $X_1, X_2, \ldots, X_n$ are NA, so are $f_1(\vec{X}), f_2(\vec{X}), f_k(\vec{X})$.

Negative association implies several useful properties, including the applicability of Chernoff-Hoeffding type bounds [23]. In addition, NA clearly implies pairwise negative correlation. More generally, NA implies the stronger notion of *negative orthant dependence*.

Definition 2.2. *A joint distribution $X_1, X_2, \ldots, X_n$ is said to be* Negative Upper Orthant Dependent (NUOD), *if for all $\vec{x} \in \mathbb{R}^n$ it holds that*

$$\Pr[\bigwedge_{i \in [n]} X_i \geq x_i] \leq \prod_{i \in [n]} \Pr[X_i \geq x_i],$$

and Negative Lower Orthant Dependent (NLOD) *if for all $\vec{x} \in \mathbb{R}^n$ it holds that*

$$\Pr[\bigwedge_{i \in [n]} X_i \leq x_i] \leq \prod_{i \in [n]} \Pr[X_i \leq x_i].$$

A joint distribution is said to be Negative Orthant Dependent (NOD) *if it is both NUOD and NLOD.*

Lemma 2.3 (NA variables are NOD ([23, 40])). *If a joint distribution $X_1, \ldots, X_n$ is NA, then it is NOD.*

In our analysis we will prove some scaled Bernoulli random variables are NUOD. To motivate our interest in this form of negative dependence, we note that for binary NUOD variables $X_1, X_2, \ldots, X_n$, we have that for each set $I \subseteq [n]$, $\Pr[\bigwedge_{i \in I} X_i = 1] \leq \prod_{i \in I} \Pr[X_i = 1]$. As shown by Panconesi and Srinivasan [57, proof of Theorem 3.2, with $\lambda = 1$], this property implies that the moment generating function of the sum of the X_i is upper bounded by the moment generating function of the sum of *independent* copies of the X_i variables. A simple extension of their argument shows the same holds if the X_i are NUOD *scaled* Bernoulli variables. As in [57], following the standard proofs of Chernoff-Hoeffding type bounds, this upper bound on the moment generating function implies the applicability of the following upper tail bounds to the sum of NUOD scaled Bernoulli variables "as though these variables were independent".

Lemma 2.4 (Chernoff Bound for NUOD Bernoulli Variables, [57]). *Let X be the sum of binary NUOD random variables $X_1, X_2, ..., X_n$. Then, for any $\delta > 1$,*

$$\Pr[X > (1+\delta) \cdot \mathbb{E}[X]] \leq \exp\left(\frac{-\delta \cdot \mathbb{E}[X]}{3}\right).$$

Lemma 2.5 (Bernstein's Inequality for NUOD Scaled Bernoulli Variables). *Let X be the sum of NUOD random variables $X_1, X_2, ..., X_n$ with $X_i \in \{0, M_i\}$ and $M_i \leq M$ for each $i \in [n]$. Then, if $\sigma^2 = \sum_{i=1}^n Var(X_i)$, we have for all $a > 0$,*

$$\Pr[X > \mathbb{E}[X] + a] \leq \exp\left(\frac{-a^2}{2(\sigma^2 + aM/3)}\right).$$

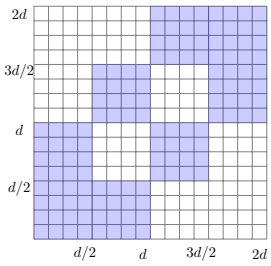
3 Natural Approaches and Their Limitations

First, we consider several natural randomized online matching algorithms and prove bounds on their competitive ratio, showing that none of these algorithms achieves competitive ratio tending to one as d grows. As these results are somewhat tangential to our main result, we defer the proofs of this section to Appendix A, where we also discuss several other natural randomized algorithms. A particular family of d-regular instances of interest in our proofs of this section is given in Figure 1 (here $d = 2k$).

RANDOM. The simplest and arguably the most natural randomized algorithm to consider for online matching is algorithm RANDOM, which matches an online vertex to some unmatched neighbor chosen uniformly at random. While this algorithm has competitive ratio tending to $\frac{1}{2}$ for general graphs (see [43]), it can be shown to be at least $1 - (1 - 1/d)^d > 1 - \frac{1}{e}$ competitive for regular graphs (see [56]). This algorithm does not, however, achieve competitive ratio tending to one as d grows.

$$N(t) = \begin{cases} [2k] & t \in [k] \\ [k] \cup (2k, 3k] & t \in (k, 2k] \\ (k, 2k] \cup (3k, 4k] & t \in (2k, 3k] \\ (2k, 4k] & t \in (3k, 4k] \end{cases}$$

(a)



(b)

Figure 1: Instance Family. Subfigure 1a shows the online vertices' neighborhoods. Subfigure 1b shows the bipartite adjacency matrix of the input. Blue entries correspond to edges ("1" entries).

Lemma 3.1. *Algorithm* RANDOM *is at most* 11/12*-competitive on d-regular graphs.*

RANKING. Another natural algorithm to consider is the optimal randomized algorithm for general bipartite graphs, namely Algorithm RANKING of Karp et al. [43]. This algorithm starts by choosing a random permutation $\sigma \in S_n$ of the n offline vertices, following which each online vertex is matched upon arrival to its unmatched neighbor of lowest value according to the permutation σ. While this algorithm achieves the optimal $1 - 1/e$ competitive ratio for general bipartite graphs, and gets a competitive ratio of $1 - O(\frac{1}{\sqrt{d}})$ on d-regular graphs under random arrival order [42], this algorithm's competitive ratio on d-regular graphs under adversarial arrival order does not tend to one as d grows.

Lemma 3.2. *Algorithm* RANKING *is at most* $(7/8 + o(1))$*-competitive on d-regular graphs.*

Randomized Multiplicative Weights. In [56] Naor and Wajc proved that matching each online vertex to an unmatched neighbor of maximum (current) degree is the optimal deterministic algorithm for d-regular graphs. The analysis of [56], which interprets the choice of a highest-degree unmatched neighbor to match as a choice of an unmatched neighbor i maximizing $(d/(d-1))^{d_t(i)}$ (where $d_t(i)$ is the degree of i at time t), suggests the following randomized matching algorithm: for each online vertex t, match t to its unmatched neighbor i with probability proportional to $(d/(d-1))^{d_t(i)}$ this driving exponential weight. We call this matching algorithm Random Multiplicative Weights, or RMWM for short. This algorithm can be shown to be at least $1-(1-1/d)^d$-competitive (see Appendix A) but its competitive ratio too does not converge to one as d increases.

Lemma 3.3. *Algorithm* RMWM *is at most* $\frac{3}{4} + \frac{\mu}{2} \approx 0.946$*-competitive on d-regular graphs. Here* $\mu \approx 0.393$ *is the solution to* $2\mu + \mu^{\sqrt{e}} = 1$.

Random Among Highest-Degree Neighbors Another natural randomized algorithm works as follows: whenever an online vertex arrives, match it uniformly at random to one of its unmatched neighbors of highest (current) degree. (This algorithm can be verified to be optimal for $d = 2$. We omit the proof for the sake of brevity.) By the result of [56], as this algorithm matches each online vertex to an unmatched neighbor of highest degree, this algorithm's competitive ratio on d-regular graphs is at least $1 - (1 - 1/d)^d > 1 - \frac{1}{e}$. However, by the next lemma, this bound is effectively tight for RANDOM-AMONG-HIGHEST, which is therefore asymptotically no better than the best deterministic online matching algorithm for d-regular graphs.

Lemma 3.4. *Algorithm* RANDOM-AMONG-HIGHEST *run on d-regular graphs has competitive ratio at most*

$$1 - \left(1 - \frac{1}{d}\right)^d + O\left(\frac{1}{d}\right).$$

In particular, for $d \to \infty$*, this algorithm's competitive ratio tends to* $1 - \frac{1}{e}$.

4 The MARKING Algorithm

In this section we present our algorithm for online matching in regular graphs, Algorithm MARKING, which is parameterized by some $\epsilon \in [0, 1]$, though generally we will consider $\epsilon = 1/\sqrt{d}$. (Unless otherwise stated, we implicitly assume $\epsilon = 1/\sqrt{d}$.) Algorithm MARKING guarantees each offline vertex a probability of exactly $(1-\epsilon)/d$ of being marked following each of its neighbors' arrivals. (This ϵ helps us control variance, key to our per-vertex and high probability guarantees.) In order to do so, at time t the algorithm associates a weight to each unmarked offline neighbor i of t inversely proportional to i's probability of not being marked prior (for which, by construction, we have a closed form). We call unmarked neighbors of t *candidates*, and only they are eligible to be matched to t. The algorithm then, with probability $1 - \epsilon$, chooses a single candidate vertex to match t to (and mark), chosen with probability proportional to its weight. This step alone may give t's low-degree neighbors a probability of being marked greater or less than $\frac{1-\epsilon}{d}$ if the overall weight of t's candidates is too low or too high, respectively.

Accordingly, if the weight of t's candidate set is smaller than its expectation, the algorithm chooses not to match t to any neighbor with probability $1 - P_1$, proportional to this deviation. If the converse holds, the algorithm marks each candidate neighbor independently with the correcting probability P_2. These steps guarantee both that the marginal probability of an edge causing a mark is equal to the fractional value $(1-\epsilon)/d$, together with some strong negative correlation properties, which imply the algorithm's guarantees. Our algorithm, given below, requires the following notation.

- d_i^t the degree of i at time t.
- For a set S with weights $w : S \to \mathbb{R}^+$ we denote by $w(S) = \sum_{i \in S} w(i)$. Moreover, we have $P_1(S, w) \triangleq \min\{\frac{w(S)}{d}, 1\} \cdot (1-\epsilon)$ and $P_2(S, i, w) \triangleq \max\left\{\frac{\left(\frac{w(S)}{d}-1\right)\cdot w(i)\cdot(1-\epsilon)}{w(S)-w(i)\cdot(1-\epsilon)}, 0\right\}$.
- F^t, the set of free (unmarked) offline vertices until time t (inclusive).

Algorithm 1 MARKING

1: **for all** online vertices t with free neighbor **do**
2: $c^t \leftarrow N(t) \cap F^{t-1}$.
3: **for all** $i \in c^t$ **do**
4: set $w^t(i) \leftarrow \frac{d}{d - d_i^t \cdot (1-\epsilon)}$.
5: w.p. $P_1(c^t, w^t)$ match t to some $i^* \in c^t$ chosen w.p. $w^t(i^*)/w^t(c^t)$ and mark i^*.
6: **for all** $j \in c^t \setminus \{i^*\}$ **do**
7: w.p. $P_2(c^t, j, w^t)$ mark j.

By definition, we have $w^t(i) \in (0, d]$ for all $i \in c^t$, and so $P_1, P_2 \in [0, 1]$ are both well-defined probabilities. Now, let $F_i^t \triangleq \mathbb{1}[i \in F^t]$ be an indicator for whether offline vertex i is free after time t (inclusive), and $M_i^t \triangleq \mathbb{1}[i \in F^{t-1} \setminus F^t]$ be an indicator for whether offline vertex i was marked at time t. Finally, let C^t be the random set corresponding to c^t. In the following claims, we show that $\Pr[M_i^t] = (1-\epsilon)/d$ for all $(i, t) \in E$, yielding a closed-form expression for $\Pr[F_i^t]$.

Claim 4.1. *For any $c^t \ni i$, we have* $\Pr[M_i^t \mid C^t = c^t] = w^t(i) \cdot \frac{1-\epsilon}{d}$.

Proof. If $w^t(c^t) \le d$ then $P_1(c^t, w^t) = \frac{w^t(c^t)}{d} \cdot (1-\epsilon)$, while $P_2(c^t, i, w^t) = 0$, and so

$$\Pr[M_i^t \mid C^t = c^t] = P_1(c^t, w^t) \cdot \frac{w^t(i)}{w^t(c^t)} = w^t(i) \cdot \frac{1-\epsilon}{d}.$$

Conversely, if $w^t(c^t) > d$ then $P_1(c^t, w^t) = 1 - \epsilon$, while $P_2(c^t, i, w^t) = \frac{\left(\frac{w^t(c^t)}{d}-1\right)\cdot w^t(i)\cdot(1-\epsilon)}{w^t(c^t)-w^t(i)\cdot(1-\epsilon)}$. A simple calculation verifies that these choices of P_1 and P_2 imply $\Pr[M_i^t \mid C^t = c^t] = w^t(i) \cdot \frac{1-\epsilon}{d}$. (Indeed, P_2 was chosen precisely with this goal in mind.) □

Taking total probability over all $c^t \ni i$, we obtain the following.

Claim 4.2. *For any offline vertex i and time t with $(i, t) \in E$, we have* $\Pr[M_i^t \mid F_i^{t-1}] = w^t(i) \cdot \frac{1-\epsilon}{d}$.

The above claim yields a simple closed-form expression for the probability of a vertex to be free, given in the following lemma.

Lemma 4.3. *For any offline vertex i and time t, we have* $\Pr[M_i^t] = \frac{(1-\epsilon)\cdot\mathbb{1}[(i,t)\in E]}{d}$, *and consequently*

$$\Pr[F_i^t] = 1 - \frac{d_i^t \cdot (1-\epsilon)}{d} = \frac{d - d_i^t \cdot (1-\epsilon)}{d}.$$

Proof. We prove by induction on t that $\Pr[M_i^t] = \frac{(1-\epsilon)\cdot\mathbb{1}[(i,t)\in E]}{d}$. Together with the obvious observation that $\Pr[i \in F^0] = 1$, this claim implies the lemma. Clearly $\Pr[M_i^t] = 0$ if $(i, t) \notin E$, so we now consider $(i, t) \in E$. Indeed, by Claim 4.2 and the inductive hypothesis, we have

$$\begin{aligned}\Pr[M_i^t] &= \Pr[M_i^t \mid F_i^{t-1}] \cdot \Pr[F_i^{t-1}] \\ &= w^t(i) \cdot \frac{1-\epsilon}{d} \cdot \frac{d - d_i^{t-1} \cdot (1-\epsilon)}{d},\end{aligned}$$

which by our choice of $w^t(i) = \frac{d}{d - d_i^t \cdot (1-\epsilon)}$ is just $\frac{1-\epsilon}{d}$. □

The above lemma immediately gives us the expected weight of C^t.

Corollary 4.4. *For each online vertex t, we have that* $\mathbb{E}[w^t(C^t)] = d$.

Proof. By Lemma 4.3, every neighbor i of t has weight $w^t(i) = F_i^{t-1} / \Pr[F_i^{t-1}]$ and thus expected weight precisely $\mathbb{E}[w^t(i)] = 1$. Thus, by linearity of expectation we have that indeed $\mathbb{E}[w^t(C^t)] = \sum_{i \in N(t)} 1 = d$. □

We will later wish to show the weight of the set C^t is concentrated around its mean. In the next section we set the stage for such claims, proving negative dependence between the F_i^t.

4.1 Negative Correlation of the F_i^t.

Here we show that for any time t, the random indicator variables F_i^t are negative upper orthant dependent (NUOD). That is, for any set of offline vertices $I \subseteq L$ we have

$$\Pr[\bigwedge_{i \in I} F_i^t] \le \prod_{i \in I} \Pr[F_i^t]. \tag{4.1}$$

Essential to our proof of the above is the negative association of the variables M_i^t conditioned on the set C^t of candidates of online vertex t.

Lemma 4.5. *For any time t and set $I \subseteq N(t)$, for any $c^t \supseteq I$, the variables $\{M_i^t \mid i \in I\}$ are negatively associated conditioned on $C^t = c^t$.*

Proof. For all $k \in c^t$, let A_k be an indicator for whether k was marked in Line 5 and let $\{B_k \mid k \in c^t\}$ be independent Bernoulli trials with success probability $P_2(c^t, i, w^t)$. Clearly the A_k are zero-one variables with $\sum_{k \in c^t} A_k \leq 1$, and so by the zero-one rule, the variables $\{A_k \mid k \in c^t\}$ are NA. Furthermore, the variables $\{B_k \mid k \in c^t\}$ are independent, and as such are NA. Moreover, the joint distributions $\{A_k \mid k \in c^t\}$ and $\{B_k \mid k \in c^t\}$ are independent of each other and so, by Property (P1), the joint distribution $\mathcal{D} = \{A_k, B_k \mid k \in c^t\}$ is NA. Finally, the set $\{M_k^t\}$ are the output of monotone increasing functions defined by disjoint subsets of a set of NA variables (as $M_k^t = A_k \vee B_k$, since k is marked either in Line 5 or in Line 7), and so the variables M_k^t are NA, by Property (P2). □

Corollary 4.6. *For any time t and sets $I, J \subseteq L$, with $I \subseteq N(t)$, we have*

$$\Pr[\bigwedge_{i \in I} \overline{M_i^t} \mid \bigwedge_{i \in I \cup J} F_i^{t-1}] \leq \prod_{i \in I} \left(1 - w^t(i) \cdot \frac{1-\epsilon}{d}\right).$$

Proof. By Lemma 4.5, for all $c^t \supseteq I$ the variables $\{M_i^t \mid i \in I\}$ conditioned on $C^t = c^t$ are NA, and so by Lemma 2.3 are NLOD. Thus, by Claim 4.1, we obtain

$$\begin{aligned}\Pr[\bigwedge_{i \in I} \overline{M_i^t} \mid C^t = c^t] &\leq \prod_{i \in I} \Pr[\bigwedge_{i \in I} \overline{M_i^t} \mid C^t = c^t] \\ &= \prod_{i \in I} \left(1 - w^t(i) \cdot \frac{1-\epsilon}{d}\right).\end{aligned}$$

Taking total probability over the events where $\bigwedge_{i \in I \cup J} F_i^{t-1}$ then yields the desired result. □

Having proved Claim 4.2 and Corollary 4.6, we are now ready to prove that the F_i^t satisfy 4.1.

Lemma 4.7. *For any set of offline vertices $I \subseteq L$ and time t, we have*

$$\Pr[\bigwedge_{i \in I} F_i^t] \leq \prod_{i \in I} \Pr[F_i^t].$$

Proof. We prove this lemma by induction on t. First, clearly $\Pr[\bigwedge_{i \in I} F_i^0] = 1 = \prod_{i \in I} \Pr[F_i^0]$. Next, we assume for $t-1$ and prove for t. In our proof, for the step of time t, for any vertex $i \in I$ we let $X_i = \Pr[F_i^{t-1}]$. By the inductive hypothesis, $\Pr[\bigwedge_{i \in I} F_i^{t-1}] \leq \prod_{i \in I} \Pr[F_i^{t-1}] = \prod_{i \in I} X_i$. We now address the inductive step. Let $J = I \cap N(t)$. Then for each $i \in I \setminus J$ we clearly have $\Pr[F_i^t] = \Pr[F_i^{t-1}] = X_i$. On the other hand, by Claim 4.2, for any $i \in J$ we have $\Pr[F_i^t] = \Pr[\overline{M_i^t} \mid F_i^{t-1}] \cdot \Pr[F_i^{t-1}] = \left(1 - w^t(i) \cdot \frac{1-\epsilon}{d}\right) \cdot X_i$. Combining these bounds with the inductive step and Corollary 4.6, we obtain

$$\begin{aligned}\Pr[\bigwedge_{i \in I} F_i^t] &= \Pr[\bigwedge_{i \in J} \overline{M_i^t} \mid \bigwedge_{i \in I} F_i^{t-1}] \cdot \Pr[\bigwedge_{i \in I} F_i^{t-1}] \\ &\leq \prod_{i \in J} \left(1 - w^t(i) \cdot \frac{1-\epsilon}{d}\right) \cdot \prod_{i \in I} X_i \\ &= \prod_{i \in I} \Pr[F_i^t].\end{aligned}$$ □

In particular, Lemma 4.7 implies that the variables F_i^t are pairwise negatively correlated.

Corollary 4.8. *For any two offline vertices i, j and for any time t, we have $Cov(F_i^t, F_j^t) \leq 0$.*

For the next corollary, let $\Delta_i^t \triangleq d - d_i^t + 1$ denote the "residual" degree of i before time t.

Corollary 4.9. *For each online vertex t, we have that*

$$Var(w^t(C^t)) \leq \sum_{i \in N(t)} Var(w_i^t) \leq \min\left\{\sum_{i \in N(t)} \frac{d}{\Delta_i^t}, \frac{d}{\epsilon}\right\}.$$

Proof. As $w^t(i) = \frac{F_i^{t-1}}{\Pr[F_i^{t-1}]}$ by Lemma 4.3, the variance of this weight is at most $Var(w^t(i)) \leq \frac{1}{\Pr[F_i^{t-1}]} = \frac{d}{d - d_i^{t-1}(1-\epsilon)} \leq \min\{\frac{d}{\Delta_i^t}, \frac{1}{\epsilon}\}$. The corollary then follows by sub-additivity of variance of pairwise negatively-correlated variables, together with Corollary 4.8. □

Lemma 4.10. *Taking the expectation over a uniformly-chosen online vertex t, we have*

$$\mathbb{E}_t[Var(w^t(C^t))] \leq d \cdot H_d.$$
$$\mathbb{E}_t[\sqrt{Var(w^t(C^t))}] \leq \sqrt{d \cdot H_d}.$$

Proof. By Corollary 4.9, for each time t, we have that $Var(w^t(C^t)) \leq \sum_{i \in N(t)} \frac{d}{\Delta_i^t}$. Consequently,

$$\begin{aligned}\mathbb{E}_t[Var(w^t(C^t))] &\leq \frac{\sum_t \sum_{i:\, i \in N(t)} \frac{d}{\Delta_i^t}}{n} \\ &= \frac{\sum_i \sum_{t:\, i \in N(t)} \frac{d}{\Delta_i^t}}{n} \\ &= \frac{n \cdot \sum_{\Delta=1}^{d} \frac{d}{\Delta}}{n} \\ &= d \cdot H_d,\end{aligned}$$

implying the bound on the expected variance of $w^t(C^t)$. The bound on the expected standard deviation of $w^t(C^t)$ then follows by concavity of $\sqrt{x}$, which implies

$$\mathbb{E}_t[\sqrt{Var(w^t(C^t))}] \leq \sqrt{\mathbb{E}_t[Var(w^t(C^t))]} \leq \sqrt{d \cdot H_d}.$$

□

5 Expected Competitive Ratio

We now proceed to bound the competitive ratio of algorithm MARKING, restated below.

Theorem 1.1. *Algorithm* MARKING *achieves competitive ratio* $(1-\frac{2\sqrt{H_d}}{\sqrt{d}}) = 1-O(\frac{\sqrt{\log d}}{\sqrt{d}})$ *on d-regular graphs.*

Proof. The probability of an online vertex t to be matched by Algorithm MARKING is precisely

$$\begin{aligned}\Pr[t \text{ matched}] &= \mathbb{E}[P_1(C^t, w^t)] \\ &= \mathbb{E}\left[\min\left\{\frac{w^t(C^t)}{d}, 1\right\} \cdot (1-\epsilon)\right].\end{aligned}$$

Now, if $w^t(C^t)$ were always precisely equal to its expectation, which is $\mathbb{E}[w^t(C^t)] = d$ by Corollary 4.4, we would be done, as the probability of t being matched would be precisely $1-\epsilon = 1-1/\sqrt{d}$, implying (an even better bound than) the claimed bound. However, as $w^t(C^t)$ is not always equal to its expectation, we have still to account for deviation from the mean.

Let $Y^t = \max\{\mathbb{E}[w^t(C^t)] - w^t(C^t), 0\}$. Using this notation, the probability of t not being matched by the algorithm is at most $\Pr[t \text{ unmatched}] = 1 - \mathbb{E}[P_1(C^t, w^t)] = \frac{\mathbb{E}[Y^t]}{d} \cdot (1-\epsilon) + \epsilon \leq \frac{\mathbb{E}[Y^t]}{d} + \epsilon$. Now, for a fixed t, the expectation of Y^t can be bounded by appealing to Jensen's inequality and using the concavity of $f(x) = \sqrt{x}$, yielding

$$\begin{aligned}\mathbb{E}[Y^t] &= \mathbb{E}[\max\{\mathbb{E}[w^t(C^t)] - w^t(C^t), 0\}] \\ &\leq \mathbb{E}[\,|\, w^t(C^t) - \mathbb{E}[w^t(C^t)]\,|\,] \\ &= \mathbb{E}[\sqrt{(w^t(C^t) - \mathbb{E}[w^t(C^t)])^2}] \\ &\leq \sqrt{\mathbb{E}[(w^t(C^t) - \mathbb{E}[w^t(C^t)])^2]} \\ &= \sqrt{Var(w^t(C^t))}.\end{aligned}$$

By the above, the probability of a randomly-chosen online vertex t to not be matched is at most $\mathbb{E}_t[\Pr[t \text{ unmatched}]] \leq \frac{\mathbb{E}_t[\sqrt{Var(w^t(C^t))}]}{d} + \epsilon$. By Lemma 4.10, the first summand of this upper bound is at most

$$\frac{\mathbb{E}_t[\sqrt{Var(w^t(C^t))}]}{d} \leq \frac{\sqrt{\mathbb{E}_t[Var(w^t(C^t))]}}{d} \leq \frac{\sqrt{H_d}}{\sqrt{d}}.$$

We conclude that the expected number of online vertices left unmatched by algorithm MARKING is

$$\begin{aligned}\sum_t \Pr[t \text{ unmatched}] &= n \cdot \left(\frac{\sqrt{H_d}}{\sqrt{d}} + \epsilon\right) \\ &= n \cdot \left(\frac{\sqrt{H_d}}{\sqrt{d}} + \frac{1}{\sqrt{d}}\right) \\ &\leq n \cdot \frac{2\sqrt{H_d}}{\sqrt{d}}.\end{aligned}$$

As the number of online vertices and OPT are both n, the theorem follows. □

6 High Probability Guarantees

For our high probability result, we turn the tables around, and analyze the loss of the algorithm from the point of view of the offline vertices. The main property we rely on for this proof (in several ways) is Lemma 4.7; i.e., the fact that for all time t the variables F_i^t are NUOD. This property allows us to appeal to Chernoff bounds, as well as Bernstein's inequality for weighted versions of these variables, as discussed in Section 2.1 As a first such application, we bound the number of unmarked offline vertices.

Lemma 6.1. *With high probability, after running* MARKING *with any* $\epsilon \geq \frac{\log n}{n}$ *on a d-regular graph, the number of unmarked offline vertices,* $\sum_i F_i^n$*, satisfies*

$$\sum_i F_i^n = O\left(n \cdot \epsilon\right).$$

Proof. By Lemma 4.3, $\mathbb{E}[F_i^n] = \frac{\epsilon d}{d} = \epsilon$. Therefore $\mathbb{E}[\sum_i F_i^n] = n \cdot \epsilon \geq \log n$. The lemma then follows by applying the multiplicative upper tail Chernoff bound to the sum of the NUOD variables F_i^n, namely $\Pr[\sum_i F_i^n \geq (1+\delta)\cdot\mathbb{E}[\sum_i F_i^n]] \leq \exp\left(\frac{-\delta\cdot\mathbb{E}[\sum_i F_i^n]}{3}\right)$ (Lemma 2.4). □

Having bounded the number of unmarked offline vertices, we now turn to bounding the number of superfluously marked vertices. For this, we will require the following lemma, bounding the probability of an online vertex having an unexpectedly heavy candidate set.

Lemma 6.2. *Let* $k > 0$*. For any online vertex* t*, let*

$$a(t,k) \triangleq \sqrt{Var(w^t(C^t)) \cdot \log k} + (1/3\epsilon) \cdot \log k. \tag{6.2}$$

Then, for any $c \geq \frac{1}{4}$ *we have*

$$\Pr[w^t(C^t) > d + 4c \cdot a(t,k)] \leq 1/k^c.$$

Proof. By Lemma 4.7, the F_i^{t-1} are NUOD; consequently, so are $w^t(i) = F_i^{t-1}/Pr[F_i^{t-1}]$ for $(i,t) \in E$. We may therefore apply Bernstein's Inequality, Lemma 2.5, to the sum of these w_i^t. To upper bound the maximum absolute value of any w_i^t, note that $|w^t(i)| = F_i^{t-1} \cdot \frac{d}{d - d_i^{t-1}\cdot(1-\epsilon)} \leq \frac{d}{d-d\cdot(1-\epsilon)} = \frac{1}{\epsilon}$. But by Corollary 4.4, $\mathbb{E}[w^t(C^t)] = d$. Plugging these values into Bernstein's Inequality, we have that for all $a > 0$,

$$\begin{aligned}Pr[w^t(C^t) > d + a] &= Pr[w^t(C^t) > \mathbb{E}[w^t(C^t)] + a] \\ &\leq \exp\left(\frac{-a^2}{2(Var(w^t(C^t)) + a/3\epsilon)}\right).\end{aligned} \tag{6.3}$$

Consider $a = 4c \cdot a(t,k)$. For this a we have both

$$Var(w^t(C^t)) \leq \frac{a(t,k)^2}{\log k} = \frac{a^2}{16c^2 \log k} \leq \frac{a^2}{4c \log k}, \tag{6.4}$$

as well as

$$\frac{1}{3\epsilon} \leq \frac{a(t,k)}{\log k} \leq \frac{a}{4c \log k}. \tag{6.5}$$

Plugging Inequalities 6.4 and 6.5 into Inequality 6.3, we conclude that for the above choice of a, the probability $Pr[w^t(C^t) > d + a]$ is at most

$$\exp\left(\frac{-a^2}{2(a^2/4c\log k + a^2/4c\log k)}\right) = \frac{1}{k^c}. \qquad \square$$

Bounding the Number of Superfluous Marks. We now turn our attention to upper bounding the number of offline vertices which are marked but not matched by algorithm MARKING, which we refer to as *superfluous* marks.

Lemma 6.3. *With high probability, after running* MARKING *on a d-regular graph, the number of superfluously marked offline vertices is at most $O\left(n \cdot \frac{\log n}{\sqrt{d}}\right)$.*

For our proof of Lemma 6.3 we will use the following standard result, easily obtained by a simple coupling argument. We omit the proof for the sake of brevity.

Lemma 6.4. *Let $X_1, X_2, \ldots, X_m$ be random variables and $Y_1, Y_2, \ldots, Y_m$ be binary random variables such that $Y_i = f_i(X_1, X_2, \ldots, X_i)$ for all i and for all $\vec{x} \in \mathbb{R}^m$,*

$$\Pr[Y_i = 1 \mid \bigwedge_{\ell \in [i]} X_\ell = x_\ell] \leq p_i.$$

Then, if $Z_1, Z_2, \ldots, Z_m$ are independent random variables with $Z_i \sim Bernoulli(p_i)$ for each i, we have

$$\Pr[\sum_i Y_i \geq k] \leq \Pr[\sum_i Z_i \geq k].$$

We now turn to proving Lemma 6.3.

Proof of Lemma 6.3. For our proof, for all $(i,t) \in E$, we let S_i^t be an indicator variable for offline vertex i being superfluously marked at time t and let X_i^t be an indicator random variable for $w^t(C^t) \leq d + O(a(t))$, where $a(t) = a(t,n)$. By Lemma 6.2 and union bound, all X_i^t are one with high probability. Next, we let $Y_i^t = S_i^t \cdot X_i^t$ be an indicator random variable for i being superfluously marked at time t and candidate set C^t not being unexpectedly large; i.e., $w^t(C^t) \leq d + O(a(t))$. We now proceed to upper bound the conditional probabilities

$$\Pr[Y_i^t = 1 \mid \bigwedge_{(i',t') \leq (i,t)} X_{i'}^{t'} = x_{i'}^{t'}] = \Pr[Y_i^t = 1 \mid X_i^t = x_i^t].$$

If $X_i^t = 0$ clearly $Y_i^t = 0$. It remains to consider the case of $X_i^t = 1$.

First, the probability of i being unmarked before time t conditioned on $X_i^t = 1$ is at most

$$\begin{aligned} \Pr[F_i^{t-1} \mid X_i^t] &= \frac{\Pr[F_i^{t-1}, X_i^t]}{\Pr[X_i^t]} \\ &\leq \frac{\Pr[F_i^{t-1}]}{\Pr[X_i^t]} \\ &\leq \Pr[F_i^{t-1}] \cdot (1 + o(1)). \end{aligned} \tag{6.6}$$

Next, the probability of i being superfluously marked at time t conditioned on i not being marked before time t, with t's candidate set being some $c^t \ni i$, that is $\Pr[S_i^t \mid C^t = c^t]$, is precisely

$$\begin{aligned} &\left(1 - P_1(c^t, w^t) \cdot \frac{w^t(i)}{w^t(c^t)}\right) \cdot P_2(c^t, i, w^t) \\ &= \max\left\{\frac{\left(\frac{w^t(c^t)}{d} - 1\right) \cdot w^t(i) \cdot (1-\epsilon)}{w^t(c^t)}, 0\right\}. \end{aligned}$$

As this expression is monotone increasing in $w^t(c^t)$, ans since by Lemma 4.3 we have $w^t(i) = 1/\Pr[F_i^{t-1}]$, the probability of i being superfluously matched by t, conditioned on i being free before time t as well as $w^t(c^t) \leq d + O(a(t))$, is at most

$$\begin{aligned} \Pr[Y_i^t \mid F_i^{t-1}, X_i^t] &\leq \frac{\frac{O(a(t))}{d} \cdot w^t(i) \cdot (1-\epsilon)}{d + O(a(t))} \\ &\leq \frac{\frac{O(a(t))}{d} \cdot \frac{1}{\Pr[F_i^{t-1}]}}{d}. \end{aligned} \tag{6.7}$$

Combining Inequalities 6.6 and 6.7 we obtain

$$\Pr[Y_i^t \mid X_i^t] \leq \frac{O(a(t))}{d^2}.$$

In order to bound the expected sum $\sum_{i,t} \mathbb{E}[Y_i^t]$, we must first upper bound the sum of the different $a(t)$. By Lemma 4.10, we have $\mathbb{E}_t[\sqrt{Var(w^t(C^t))}] \leq \sqrt{d \cdot H_d}$. Consequently, plugging this bound into 6.2, we obtain

$$\begin{aligned} \sum_t O(a(t)) &= \sum_t O(\sqrt{Var(w^t(C^t)) \cdot \log n} + \sqrt{d} \cdot \log n) \\ &\leq O\left(n \cdot \sqrt{d \cdot H_d \cdot \log n} + \sqrt{d} \cdot \log n\right) \\ &= O\left(n \cdot \sqrt{d} \cdot \log n\right). \end{aligned}$$

Therefore, summing over all i and t, we have that

$$\begin{aligned} \sum_t \sum_{i \in N(t)} \mathbb{E}[Y_i^t] &\leq \sum_t \sum_{i \in N(t)} \frac{O(a(t))}{d^2} \\ &= \sum_t \frac{O(a(t))}{d} \\ &= O\left(n \cdot \frac{\log n}{\sqrt{d}}\right). \end{aligned}$$

Appealing to Lemma 6.4 and the Chernoff bound for the coupled independent random variables, we find that $\sum_{i,t} Y_i^t = O\left(n \cdot \frac{\log n}{\sqrt{d}}\right)$ with high probability. But, with high probability $X_i^t = 1$ for all pairs $(i,t) \in E$, and so with high probability the number of superfluously marked offline vertices is at most

$$\sum_{i,t} S_i^t = \sum_{i,t} Y_i^t \cdot X_i^t = \sum_{i,t} Y_i^t = O\left(n \cdot \frac{\log n}{\sqrt{d}}\right). \quad \square$$

We can now prove our algorithm's high probability guarantees, restated below.

Theorem 1.3. *Algorithm* MARKING *is* $1 - O(\frac{\log n}{\sqrt{d}})$*-competitive w.h.p. on* n*-vertex* d*-regular graphs.*

Proof. The number of matched offline vertices is precisely the number of marked offline vertices minus the number of superfluously marked offline vertices. That is, if we denote by $|M|$ the size of the matching, by $|F|$ and $|S|$ the number of non-marked and superfluously marked offline vertices, we have $|M| = n - |F| - |S|$. The theorem then follows directly from Lemmas 6.1 and 6.3. $\square$

The results of this section imply that each offline vertex is matched with probability at least $1 - O(\frac{\log n}{\sqrt[4]{d}})$. In the next section we give tighter bounds on this probability, as well as bounds for the corresponding probability for online vertices.

7 Per-Vertex Guarantees

In this section we show our algorithm guarantees each vertex a probability of being matched tending to one as d increases. As a byproduct, this implies that our algorithm is a vertex-weight oblivious algorithm achieving competitive ratio tending to one for all weight functions, for weights on both offline and even on online vertices.

Theorem 7.1. *Algorithm* MARKING *run with a given value of* $\epsilon \geq \frac{\log d}{3d}$ *matches each vertex with probability at least* $1 - \epsilon - \frac{16\sqrt{\log d}}{\sqrt{\epsilon d}} - \frac{1}{d}$ *on* d*-regular graphs.*

The following two corollaries of Theorem 7.1 imply Theorem 1.4.

Corollary 7.2. *Algorithm* MARKING *(with* $\epsilon = 1/\sqrt{d}$*) on* d*-regular graphs matches each vertex with probability at least* $1 - O\left(\frac{\sqrt{\log d}}{\sqrt[4]{d}}\right)$.

Corollary 7.3. *Algorithm* MARKING *with* $\epsilon = \sqrt[3]{\log d}/\sqrt[3]{d}$ *on* d*-regular graphs matches each vertex with probability at least* $1 - O\left(\frac{\sqrt[3]{\log d}}{\sqrt[3]{d}}\right)$*, respectively.*

To relate the above results to online weighted matching, we note that a vertex-weight oblivious algorithm $\mathcal{A}$ is α-competitive if and only if it matches every vertex with probability at least α. This observation implies that Algorithm MARKING with the above values of ϵ is a $1 - O\left(\frac{\sqrt{\log d}}{\sqrt[4]{d}}\right)$- and $1 - O\left(\frac{\sqrt[3]{\log d}}{\sqrt[3]{d}}\right)$-competitive vertex-oblivious online vertex-weighted matching algorithm, respectively. We now turn to proving this section's main result.

Proof of Theorem 7.1. Consider an edge $(i,t) \in E$. By Corollary 4.9 the online vertex t has variance of weights $Var(w^t(C^t)) \leq \frac{d}{\epsilon}$. Combined with the lower bound on ϵ, we find that $a(t,d) \leq 2 \cdot \sqrt{\frac{d}{\epsilon} \cdot \log d}$. So, if we denote by A^t the event that $w^t(C^t) > d + a$, for $a = 16 \cdot \sqrt{\frac{d}{\epsilon} \cdot \log d}$, we have $\Pr[A^t] \leq 1/d^2$, by Lemma 6.2. But by Lemma 4.3, as $(i,t) \in E$ we have $\Pr[F_i^{t-1}] = \frac{d - d_i^{t-1} \cdot (1-\epsilon)}{d} \geq \frac{d - d_i^{t-1}}{d} \geq \frac{1}{d}$. Consequently, we obtain the following lower bound on i's conditional probability of being free by time t.

$$\begin{aligned}
\Pr[F_i^{t-1}, \overline{A^t}] &= \Pr[F_i^{t-1}] - \Pr[F_i^{t-1}, A^t] \\
&\geq \Pr[F_i^{t-1}] - \Pr[A^t] \\
&\geq \Pr[F_i^{t-1}] - 1/d^2 \\
&\geq \Pr[F_i^{t-1}] \cdot (1 - 1/d).
\end{aligned}$$

On the other hand, the probability of i being matched to t conditioned on i being free and t's candidate set having sum of weights at most $w^t(C^t) \leq d + a$ is at least

$$\begin{aligned}
\Pr[(i,t) \in M \mid F_i^{t-1}, \overline{A^t}] &= P_1(w^t(C^t)) \cdot \frac{w^t(i)}{w^t(C^t)} \\
&\geq \frac{w^t(i) \cdot (1-\epsilon)}{d + a} \\
&\geq \frac{w^t(i)}{d} \cdot \left(1 - \epsilon - \frac{a}{d}\right).
\end{aligned}$$

Combining the above lower bounds and recalling that by Lemma 4.3 $w^t(i) = 1/\Pr[F_i^{t-1}]$, we find that for every edge $(i,t) \in E$ the probability of (i,t) being matched is at least

$$\begin{aligned}
\Pr[(i,t) \in M] &\geq \Pr[(i,t) \in M, F_i^{t-1}, \overline{A^t}] \\
&\geq \Pr[(i,t) \in M \mid F_i^{t-1}, \overline{A^t}] \cdot \Pr[F_i^{t-1}, \overline{A^t}] \\
&\geq \frac{1}{d} \cdot \left(1 - \epsilon - \frac{a}{d} - \frac{1}{d}\right).
\end{aligned}$$

Consequently, for any (online or offline) vertex v, summing up over v's d edges, we find that indeed, by our

choice of a, v's probability of being matched is at least

$$\begin{aligned}\Pr[v \text{ matched}] &= \sum_{e \ni v} \Pr[e \in M] \\ &\geq d \cdot \frac{1}{d} \cdot \left(1 - \epsilon - \frac{a}{d} - \frac{1}{d}\right) \\ &= 1 - \epsilon - \frac{16\sqrt{\log d}}{\sqrt{d\epsilon}} - \frac{1}{d}. \end{aligned}$$ □

Better bounds for online vertices. The following lemma shows a refined bound for online vertices.

Lemma 7.4. *Algorithm* MARKING *run with a given value of ϵ matches each online vertex with probability at least $1 - \epsilon - \frac{1}{\sqrt{\epsilon d}}$ on d-regular graphs.*

Proof. As observed in the proof of Theorem 1.1, the probability of an online vertex t being left unmatched by Algorithm MARKING is at most $\Pr[t \text{ unmatched}] = 1 - \mathbb{E}[P_1(C^t, w^t)] = \frac{\mathbb{E}[Y^t]}{d} \cdot (1-\epsilon) + \epsilon \leq \frac{\mathbb{E}[Y^t]}{d} + \epsilon$, where $Y^t = \max\{\mathbb{E}[w^t(C^t)] - w^t(C^t), 0\}$. Moreover, as noted in said proof, $\mathbb{E}[Y^t] \leq \sqrt{Var(w^t(C^t))}$. But, by Corollary 4.9, t satisfies $Var(w^t(C^t)) \leq \frac{d}{\epsilon}$. Consequently, we have, as claimed

$$\Pr[t \text{ unmatched}] \leq \epsilon + \frac{1}{\sqrt{\epsilon d}}.$$ □

8 Hardness Result

In this section we present our hardness result for online matching in d-regular graphs.

We note that previous hardness results for online matching and its variants [7, 26, 41, 43] were stated, or can be recast, as hardness results for fractional algorithms. This approach, standard in the online algorithms literature, relies on the simple observation that any randomized algorithm $\mathcal{A}$ naturally induces a fractional algorithm $\mathcal{A}'$ with the same competitive ratio, by assigning each edge (i, t) a value equal to its expected value according to $\mathcal{A}$. While this "first-moment method" for proving hardness results is simple and powerful, it fails miserably for online matching on regular graphs. To see this, observe that an algorithm assigning a value of $1/d$ to each edge yields an *optimal* solution on d-regular graphs, or competitive ratio *one*. Consequently, this approach can yield no non-trivial hardness result for randomized algorithms on regular graphs. Moreover, one cannot hope to obtain a competitive ratio of one for randomized integral matching (for example, a simple distribution over 8-cycles proves no algorithm is better than 7/8-competitive for 2-regular graphs). Given the failing of such first-moment methods (i.e., considering solely the marginal probabilities of each edge being matched), our hardness result must deviate from previous hardness results and explicitly consider higher moments; specifically, variance.

Theorem 1.2. *No randomized online matching algorithm is better than $\left(1 - \frac{1}{\sqrt{8\pi d}}\right) = 1 - O(\frac{1}{\sqrt{d}})$-competitive on d-regular graphs.*

Proof. We appeal to Yao's Lemma [60], giving a distribution over inputs for which no deterministic algorithm achieves competitive ratio better than the above bound, implying our claimed result. Without loss of generality, we may assume that the deterministic algorithm is maximal; i.e., the algorithm always matches when possible. The input consists of $n = d^2$ offline vertices, partitioned into d many d-tuples of offline vertices. During the first phase, each of these d-tuples' vertices all neighbor $d/2$ common online neighbors.[5] Following the first phase, the d offline vertices of each of the offline d-tuples are randomly permuted and correspondingly numbered 1 through d. Next, a second phase begins, during which, for each $i \in [d]$, all the d offline vertices numbered i neighbor $d/2$ common online vertices. By the maximality of the algorithm, each offline vertex is matched with probability $1/2$ during the first phase. Therefore, for each $i \in [d]$, if we denote by X_i the number of vertices of the i-th tuple which are not matched during the first phase, we find that X_i is distributed binomially, $X_i \sim Bin(d, 1/2)$. In particular, X_i's expectation is $\mathbb{E}[X_i] = d/2$. On the other hand, at most $d/2$ vertices numbered i can be matched during the second phase, and so the algorithm leaves $U_i = \max\{0, X_i - d/2\}$ unmatched vertices among these d vertices. But as $X_i \sim Bin(d, 1/2)$ is binomially distributed, then by the normal approximation of the binomial distribution, we find that for large d, X_i is approximately distributed $N(d/2, \sqrt{d}/2)$ and so the expectation of $|X_i - \mathbb{E}[X_i]|$ is $\mathbb{E}[|X_i - \mathbb{E}[X_i]|] \approx \sqrt{d}/2 \cdot \sqrt{2/\pi} = \sqrt{\frac{d}{2\pi}}$ ([31]). But as X_i is symmetric around its expectation, the expected number of i-numbered vertices left unmatched after the second phase is

$$\mathbb{E}[U_i] = \mathbb{E}[\max\{0, X_i - d/2\}] = \sqrt{\frac{d}{2\pi}} \cdot \frac{1}{2} = \frac{\sqrt{d}}{\sqrt{8\pi}}.$$

That is, for any $i \in [d]$, the expected fraction of the d offline vertices numbered i and left unmatched is at least $(\sqrt{d/8\pi})/d = 1/\sqrt{8\pi d}$. Consequently, the competitive ratio of any deterministic algorithm on this distribution of inputs is at most $1 - 1/\sqrt{8\pi d}$. □

[5] Note that if we want to let n increase arbitrarily compared to d, this example can be extended to contain any number of vertices n which is an integer product of d^2, by taking multiple disjoint and independently drawn copies of this graph and arrival order.

Appendix

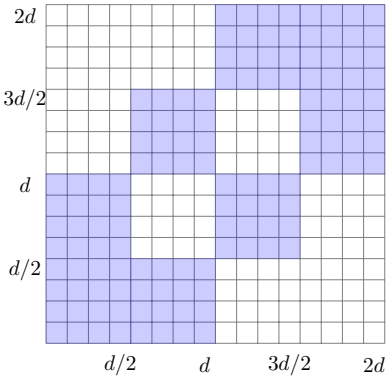
A Omitted Proofs of Section 3

In this section we prove the bounds on the competitive ratios of some natural randomized algorithms stated in Section 3.

A particular family of instances of interest in our proofs of upper bounds for the competitive ratio of these algorithms is as follows. Let $d = 2k$ be an even integer. The input is a d-regular input, consisting of $2d$ offline and online vertices, with the neighborhood of online vertex t given in Figure 2a. See also the bipartite adjacency matrix of this input in Figure 2b. We note that on these instances the generally optimal fractional algorithm, WATER-FILLING of Pruhs & Kalyanasundaram [41], attains a competitive ratio of 7/8. We now turn our attention to analyzing the competitive ratio of some simple randomized algorithms on these instances.

$$N(t) = \begin{cases} [2k] & t \in [k] \\ [k] \cup (2k, 3k] & t \in (k, 2k] \\ (k, 2k] \cup (3k, 4k] & t \in (2k, 3k] \\ (2k, 4k] & t \in (3k, 4k] \end{cases}$$

(a) Online vertices' neighborhood



(b) Bipartite adjacency matrix of the input. Blue entries correspond to edges ("1" entries).

Figure 2: Bad Instance

A particular distribution which will appear often in our analysis of prior algorithms is the hypergeometric distribution, which corresponds to the number of red balls drawn among $k \leq n$ draws without replacement from an urn containing $n = r+b$ balls, r of which red and b of which blue, where balls are picked without replacement (sometimes referred to as the 'urn problem'). By linearity of expectation, the number of red balls drawn is precisely $k \cdot \frac{r}{r+b}$.

A.1 Algorithm RANDOM

In this section we analyze the simplest randomized online matching algorithm, RANDOM given in Algorithm 2.

Algorithm 2 RANDOM

```
for all online vertices t do
    if t has an unmatched neighbor i then
        match t to an unmatched neighbor i chosen uniformly at random.
```

Lemma 3.1. *Algorithm* RANDOM *is at most* 11/12*-competitive on d-regular graphs.*

Proof. Consider the $2k$-regular input of Figure 2. As the k online vertices $t \in (k, 2k]$ neighbor k common unmatched offline vertices – the offline vertices $(2k, 3k]$ – these k online vertices are all matched by RANDOM. Similarly, the k online vertices $t \in (2k, 3k]$ are all matched, too. Suppose that after time k exactly x offline vertices in $[k]$ are unmatched (and therefore $k-x$ offline vertices in $(k, 2k]$ are unmatched, as exactly k of the offline vertices in $[2k]$ are matched by time k). Then, by a standard urn problem argument, Algorithm RANDOM will match $k \cdot \left(\frac{x}{k+x}\right) = k \cdot \left(1 - \frac{k}{k+x}\right)$ and $k \cdot \left(\frac{k-x}{k+k-x}\right) = k \cdot \left(1 - \frac{k}{k+k-x}\right)$ offline vertices in $[2k]$ to online vertices in $(k, 2k]$ and $(2k, 3k]$, respectively. Consequently, the number of matched offline vertices in $[2k]$ matched by Algorithm RANDOM is at most

$$\begin{aligned} &\leq k + k \cdot \left(1 - \frac{k}{k+x} + 1 - \frac{k}{k+k-x}\right) \\ &\leq 3k - k^2 \cdot \left(\frac{2}{3k/2}\right) = \frac{5k}{3}, \end{aligned}$$

where the inequality follows by convexity of $\frac{1}{k+x}$. We conclude that Algorithm RANDOM achieves a gain of at most $\frac{11k}{3}$ out of an optimum of $4k$. The lemma follows. □

A.2 Algorithm RANKING

In this section we analyze the classic algorithm of Karp et al. [43], which draws a random permutation σ of the offline vertices upon initialization and greedily matches each online vertex to its unmatched neighbor of highest priority according to σ. In our analysis we will use an equivalent formulation of RANKING, introduced by Devanur et al. [19]; in their terminology, each offline vertex i samples independently a random value Y_i distributed uniformly in $[0, 1]$, and each offline vertex is matched to its unmatched neighbor i minimizing Y_i. The algorithm is stated in this terminology in Algorithm 3.

While this algorithm achieves the optimal $1 - 1/e$ competitive ratio for general bipartite graphs, its

Algorithm 3 RANKING

1: **for all** offline vertices i **do**
2: sample $Y_i \sim U[0,1]$ (independently).
3: **for all** online vertices t **do**
4: **if** t has an unmatched neighbor i **then**
5: match t to $\arg\min\{Y_i \mid i \in N(t)$ unmacthed$\}$.

competitive ratio on d-regular graphs does not tend to one as d grows, as the next lemma asserts.

Lemma 3.2. *Algorithm* RANKING *is at most* $(7/8 + o(1))$*-competitive on* d*-regular graphs.*

Proof. Consider the $2k$-regular input of Figure 2. We will consider the expected number of offline vertices in $[k]$ matched by Algorithm RANKING, noting that the exact same bound holds for the vertices in $(k, 2k]$, by symmetry. Let M be the median Y-value among the Y-values of the offline vertices in $[2k]$. That is, M is the k-th order statistic among $2k$ independent uniform variables in $(0, 1)$. In that case, its expectation can be verified to be $\mathbb{E}[M] = \frac{k}{2k+1}$. Using this terminology, we note that all online vertices in $[k]$ are matched to (all the) offline vertices in $[2k]$ of Y-value at most M and online vertices in $(k, 2k]$ are first matched to offline vertices with Y-value at most M before being matched to any other vertex. We first study these early matches. Denote by O and N ("old" and "new", respectively) the number of offline vertices in $[k]$ and $(2k, 3k]$ with Y-value at most M. Clearly, as exactly k of the $2k$ vertices in $[2k]$ have Y-value at most M and these vertices have i.i.d Y-values, then by symmetry we have $\mathbb{E}[O] = k/2$. On the other hand, for a given value of $M = x$, the expected number of offline vertices in $(2k, 3k]$ with Y-value at most $M = x$ is precisely $k \cdot x$ by linearity and the definition of the uniform distribution. Therefore, conditioning on M we find that $\mathbb{E}[N] = \int_0^1 k \cdot x \cdot f_M(x)\, dx = k \cdot \mathbb{E}[M] = \frac{k^2}{2k+1} = k/2 - 1/4 + 1/(8k+4)$. We now turn to analyzing the number of offline vertices in $[k]$ matched during time range $(k, 2k]$.

Conditioning on O and N as above, we find that by time $k + N + 1$, the unmatched offline vertices in the neighborhood of the online vertices $(k, 2k]$ have i.i.d Y-values, as these are all i.i.d uniform $(0, 1)$ values greater than M (lit. "conditioned on being greater than M"). As such, by a standard urn problem argument, Algorithm RANKING will match an expected $(k - N) \cdot \left(\frac{k-O}{2k-O-N}\right)$ additional offline vertices in $[k]$ during time range $(k, 2k]$. Overall, if we denote by M_O the number of offline vertices in $[k]$ matched by the algorithm, we have, after rearranging terms, that

$$\mathbb{E}[M_O] = \mathbb{E}\left[k - \frac{(k-O)^2}{2k-O-N}\right].$$

Now, considering the expression $k - \frac{(k-O)^2}{2k-O-N}$ as a function of O and N, $f(O, N) = k - \frac{(k-O)^2}{2k-O-N}$, taking the second derivative of f with respect to N we find that this function is concave in N for $N \in [0, k]$. Consequently, by Jensen's Inequality, for any fixed value of $O = x$, $\mathbb{E}[f(O, N) \mid O = x] \leq \mathbb{E}[f(O, \mathbb{E}[N]) \mid O = x]$. Plugging in $\mathbb{E}[N] = k/2 - 1/4 + 1/(8k+4)$ into the above, we obtain a new function $g(x) = \mathbb{E}[f(O, \mathbb{E}[N]) \mid O = x] = k - \frac{(k-O)^2}{3k/2+1/4-1/(8k+4)-x}$. This function, in turn, can be verified to be concave in $x \in [0, k]$, and therefore

$$\mathbb{E}[g(O)] \leq g(\mathbb{E}[O]) = g(k/2) \leq \frac{3k}{4} + O(1).$$

To conclude, the number of offline vertices in $[2k]$ matched by Algorithm RANKING on the instance of Figure 2 is $2 \cdot \mathbb{E}[f(O, N)] \leq 2 \cdot \mathbb{E}[g(O)] \leq \frac{3k}{2} + O(1)$, and the overall number of matched offline vertices is therefore at most $\frac{7k}{2} + O(1)$, out of an optimum of $4k$. The theorem follows. □

Comparing algorithms RANDOM and RANKING We note that the proofs of Lemmas 3.1 and 3.2 rely on the same family of instances, for which these analyses are tight up to $o(1)$ terms; that is, algorithm RANDOM achieves competitive ratio $11/12 - o(1)$ on these instances, while algorithm RANKING achieves a strictly lower competitive ratio, of at most $7/8 + o(1)$. This is, to the best of our knowledge, the first family of instances for which algorithm RANDOM was shown to outperform the worst-case optimal algorithm RANKING.

Observation A.1. *There exists a family of* d*-regular graphs on which* RANKING *is* $(7/8 + o(1))$*-competitive while* RANDOM *is* $(11/12 - o(1))$*-competitive.*

We next consider these algorithms from the point of view of vertex-weighted online matching. These algorithms are clearly vertex-weight-oblivious algorithms. By [56], Algorithm RANDOM is $1 - (1 - 1/d)^d$-competitive for vertex-weighted online matching on d-regular graphs. An alternative proof of this fact is readily obtained by observing that any unmatched offline vertex has probability of at least $1/d$ of being matched to its next online neighbor, regardless of prior random choices; as such, each offline vertex is matched to one of its d neighbors with probability at least $1 - (1 - 1/d)^d$. Here too algorithm RANKING is outperformed by algorithm RANDOM, as RANKING does not yield such guarantees.

Lemma A.2. *For any d, there exist d-regular online matching instances and offline vertices of these instances which Algorithm* RANKING *matches with probability at most $\frac{1}{d} + \frac{8}{d(d+1)}$.*

Corollary A.3. RANKING *is $o(1)$-competitive for vertex-weighted matching on regular graphs.*

Proof of Lemma A.2. Let i be some offline vertex in a d-regular input, with i neighboring $[d]$. For each online vertex $t = 1, 2, \ldots, d$, let t neighbor i and $d-1$ other offline vertices with no previous neighbors. In order to bound the probability of i being matched, we wish to bound the following conditional probability.

$$\begin{aligned} &\Pr[(i,t) \in M \mid \bigwedge_{t'<t} (i,t') \notin M] \\ &= \frac{\Pr[(i,t) \in M \wedge \bigwedge_{t'<t}(i,t') \notin M]}{\Pr[\bigwedge_{t'<t}(i,t') \notin M]}. \end{aligned} \tag{A.1}$$

By the input's construction, for i to not be matched to its first $t-1$ online neighbors, i must have lower priority than $t-1$ other offline vertices, which happens with probability $\Pr[\bigwedge_{t'<t}(i,t') \notin M] = 1/t$. For i to be matched to its t-th online neighbor, i must have lower priority than $t-1$ other offline vertices and higher priority than $d-1$ other offline vertices, and so this event happens with probability $\Pr[(i,t) \in M \wedge \bigwedge_{t'<t}(i,t') \notin M] = (d-1)!(t-1)!/(t+d-1)!$. Putting these two together, we find that the conditional probability in A.1 is precisely

$$\frac{(d-1)!(t-1)!/(t+d-1)!}{1/t} = \frac{1}{\binom{t+d-1}{t}}.$$

So, by the law of total probability we find that i's probability of being matched, $\Pr[i \in M]$, is at most

$$\begin{aligned} &= \sum_{t=1}^{d} \Pr[(i,t) \in M \mid \bigwedge_{t'<t} (i,t') \notin M] \cdot \Pr[\bigwedge_{t'<t} (i,t') \notin M] \\ &\leq \sum_{t=1}^{d} \Pr[(i,t) \in M \mid \bigwedge_{t'<t} (i,t') \notin M] \\ &= \sum_{t=1}^{d} \frac{1}{\binom{t+d-1}{t}} \\ &\leq \frac{1}{d} + \frac{2}{d(d+1)} + (d-2) \cdot \frac{2 \cdot 3}{d(d+1)(d+2)} \\ &\leq \frac{1}{d} + \frac{8}{d(d+1)}. \end{aligned}$$

□

Algorithm 4 RMWM

1: **for all** online vertices t **do**
2: **if** t has an unmatched neighbor i **then**
3: let $\phi^{t-1}(i) = \left(\frac{d}{d-1}\right)^{d_i^{t-1}}$ for all unmatched $i \in N(t)$, and $\Phi_t = \sum_{i \in N(t)} \phi^{t-1}(i)$.
4: match t to an unmatched neighbor i chosen with probability $\frac{\phi^{t-1}(i)}{\Phi_t}$.

A.3 Randomized Multiplicative Weights Method In this section we discuss Algorithm RMWM, given in Algorithm 4.

We start by proving that Algorithm RMWM is at least $1-(1-1/d)^k$-competitive on (k,d)-bounded graphs, introduced by Naor and Wajc in [56]. These are graphs for which online vertices have degree at most d and offline vertices have degree at least k. Note that d-regular graphs are a special case of (d,d)-bounded graphs.

Lemma A.4. *Algorithm* RMWM *on is $1-(1-1/d)^k$-competitive on (k,d)-bounded graphs.*

Corollary A.5. *Algorithm* RMWM *is $1-(1-1/d)^d$-competitive on d-regular graphs.*

Our proof follows the potential-based analysis of [56]. The same theorem can alternatively be proven using the primal-dual method, as in [56].

Proof of Lemma A.4. Let $U \subseteq L$ be the set of unmatched vertices on the left hand (offline) side of the graph. We consider the potential $\Phi^t = \sum_{i \in U} \phi^t(i)$, which we shall show to be non-increasing over time (in expectation). Let t be some online vertex and $\Delta\Phi_t$ the change to the potential Φ incurred by t's arrival. We condition on the set of unmatched neighbors of t upon its arrival, F_t. If $F_t = \emptyset$, then t is not matched and therefore $\Delta\Phi_t = 0$. If, however, $F_t \neq \emptyset$, then each unmatched neighbor of t which is matched has its contribution to the potential, $\phi^{t-1}(i)$, increase by a $\left(\frac{d}{d-1} - 1\right) \cdot \phi^{t-1}(i) = \frac{1}{d-1} \cdot \phi^{t-1}(i)$. Therefore, if we denote by $\Phi_t = \sum_{i \in F_t} \phi^{t-1}(i)$ the contribution of j's neighborhood to the potential, and denote for the sake of brevity $\phi_i = \phi^{t-1}(i)$, we have

$$\begin{aligned} \mathbb{E}[\Delta\Phi_j \mid F_j] &= \sum_i \frac{-\phi_i^2}{\Phi_t} + \sum_i \sum_{i' \neq i} \frac{\phi_i \cdot \phi_{i'}}{\Phi_t} \cdot \frac{1}{d-1} \\ &= \frac{1}{d-1} \cdot \sum_{i \neq i'} -\frac{\left(\phi_i - \phi i'\right)^2}{\Phi_t} \\ &\leq 0, \end{aligned}$$

where the first inequality follows from $|F_t| \leq |N(t)| \leq d$. Given the above, conditioning on the possible F_t for each t, we deduce that the the expected initial and final potentials, $\Phi_{initial}$ and Φ_{final}, satisfy

$$\mathbb{E}[\Phi_{final}] = \Phi_{initial} + \sum_t \mathbb{E}[\Delta\Phi_t] \leq |L|.$$

But on the other hand, the final potential is at least

$$\mathbb{E}[\Phi_{final}] \geq \mathbb{E}[|U|] \cdot \left(\frac{d}{d-1}\right)^k.$$

Concatenating the above inequalities, we obtain

$$\mathbb{E}[|U|] \leq |L| \cdot \left(1 - \frac{1}{d}\right)^k.$$

and consequently the output matching M has expected size at least

$$\mathbb{E}[|M|] \geq |L| \cdot \left(1 - \left(1 - \frac{1}{d}\right)^k\right). \qquad \square$$

The above analysis implies that Algorithm RMWM achieves a non-trivial competitive ratio on d-regular graphs. However, as the next theorem asserts, this algorithm's competitive ratio when run on d-regular graphs is still bounded away from one.

Lemma 3.3. *Algorithm* RMWM *is at most* $\frac{3}{4} + \frac{\mu}{2} \approx 0.946$*-competitive on* d*-regular graphs. Here* $\mu \approx 0.393$ *is the solution to* $2\mu + \mu^{\sqrt{e}} = 1$.

Proof. Consider the $2k$-regular input of Figure 2. We will consider the expected number of offline vertices in $[k]$ matched by Algorithm RMWM, noting that the exact same bound holds for the vertices in $(k, 2k]$, by symmetry. First, following the first k online arrivals, some $x \in [0, k]$ offline vertices in $[k]$ are matched. By symmetry, this value x is drawn from a hypergeometric distribution with k draws from k white and k black balls with equal probability. By concentration of this distribution (which follows, for example, by negative association of this distribution), we can apply chernoff bounds and show that with high probability this x is $\frac{k}{2} \pm O(\sqrt{k \log k}) = \frac{k}{2} \cdot (1 \pm o(1))$.

The number of offline vertices in $[k]$ matched to any of the online vertices in $(k, 2k]$ is drawn according to Wallenius' noncentral hypergeometric distribution [59] with k draws, $N = x + k$ balls of either color; x white balls of weight $\omega = (2k/(2k-1))^k$ and k black balls of weight one.[6] As $\lim_{k\to\infty} \omega = \lim_{k\to\infty} (2k/(2k-1))^k = \sqrt{e}$, the expected number of white balls drawn (that is, the expected number of vertices in $[k]$ matched) is approximated by a solution μ' to

$$\frac{\mu'}{x} + \left(1 - \frac{k - \mu'}{k}\right)^{\sqrt{e}} = 1.$$

That is, as $N = k + x$, this is

$$\frac{\mu'}{x} + \left(\frac{\mu'}{k}\right)^{\sqrt{e}} = 1. \tag{A.2}$$

Now, as $x = \frac{k}{2} \cdot (1 \pm o(1))$ with high probability, the solution to A.2 is approximately $\mu' \approx 0.393 \cdot k$, which is $\mu' = \mu \cdot k$ for $\mu \approx 0.393$ the solution to $2\mu + \mu^{\sqrt{e}} = 1$. Consequently, an expected $k/2 + 0.393 \cdot k(1 \pm o(1))$ offline vertices in $[k]$ (and likewise, in $(k, 2k]$) are matched by RMWM. Accounting for the $2k$ offline vertices in $(2k, 4k]$, we find that algorithm RMWM matches at most $2k + 2 \cdot (1/2 + \mu) \cdot (1 \pm o(1)) \cdot k$ offline vertices, out of an optimum of $4k$. The theorem follows. $\square$

A.4 Random Among High-Degree Neighbors

In this section we analyze the natural generalization to the optimal algorithm for 2-regular graphs, given in Algorithm 5.

Algorithm 5 RANDOM-AMONG-HIGHEST

```
1: for all online vertices t do
2:     if t has an unmatched neighbor i then
3:         match t to an unmatched neighbor i of highest
           current degree d_i^t chosen u.a.r.
```

Lemma 3.4. *Algorithm* RANDOM-AMONG-HIGHEST *run on* d*-regular graphs has competitive ratio at most*

$$1 - \left(1 - \frac{1}{d}\right)^d + O\left(\frac{1}{d}\right).$$

In particular, for $d \to \infty$*, this algorithm's competitive ratio tends to* $1 - \frac{1}{e}$.

Proof. Consider the following adversarial d-regular input sequence on $n = d^{d+1}$ offline and online vertices, with online vertices arriving over d phases. We say an offline vertex is *active* at online arrival t if it has a non-zero probability of being unmatched prior to this arrival, and *inactive* otherwise. Clearly, all d^{d+1} offline vertices are active at first. The input maintains the invariant that (under Algorithm RANDOM-AMONG-HIGHEST) exactly $(d-1)^i d^{d-i+1}$ offline vertices are active by the end of phase i, each of degree i. During phase $i \leq d-1$, a $\frac{1}{d}$-fraction of the active offline vertices (that

[6] Note that in our problem the weights of white and black balls grow, but at the same speed. Thus, their ratio – which determines the probability of a particular color being drawn – is unchanged.

is, $(d-1)^{i-1}d^{d-i+1}$ offline vertices) are divided into d-tuples, each neighboring a new online vertex. These offline vertices now reach degree $i+1$. Each of these degree-$(i+1)$ offline vertices together with $d-1$ active degree-i offline vertices neighbor a new online vertex. By virtue of the algorithm's choice of matches, the $(d-1)^{i-1}d^{d-i+1}$ degree-$(i+1)$ vertices all become inactive. The phase ends with $(d-i-1)\cdot(d-1)^{i-1}d^{d-i}$ online vertices neighboring these new inactive vertices (with d inactive neighbors per such online vertex), bringing these newly inactive vertices to degree d. We note that the algorithm accrues no gain from these latter online vertices, as all of their neighbors are inactive, and thus matched. We lower bound the number of these online neighbors in order to obtained the desired upper bound on the algorithm's competitive ratio on this input.

Summarizing the above, we have that the number of unmatched online vertices is at least $\sum_{i=1}^{d-1}(d-i-1)\cdot(d-1)^{i-1}d^{d-i}$. Normalizing by $OPT = n = d^{d+1}$, we find that on this input sequence Algorithm RANDOM-AMONG-HIGHEST suffers a loss of at least

$$\begin{aligned}
&\geq \frac{1}{d}\cdot\sum_{i=1}^{d-1}\left(1-\frac{1}{d}\right)^{i-1}\cdot\left(\frac{d-i-1}{d}\right)\\
&= \frac{1}{d}\cdot\sum_{i=1}^{d-1}\left(1-\frac{1}{d}\right)^{i} - \frac{1}{d}\cdot\sum_{i=1}^{d-1}\frac{1}{d}\left(1-\frac{1}{d}\right)^{i-1}\cdot i\\
&= \frac{1}{d}\cdot\frac{\left(1-\frac{1}{d}\right)^{d}-\left(1-\frac{1}{d}\right)}{\left(1-\frac{1}{d}\right)-1} - \frac{1}{d}\cdot\sum_{i=1}^{d-1}\frac{1}{d}\left(1-\frac{1}{d}\right)^{i-1}\cdot i\\
&= \left(1-\frac{1}{d}\right)-\left(1-\frac{1}{d}\right)^{d} - \frac{1}{d}\cdot\sum_{i=1}^{d-1}\frac{1}{d}\left(1-\frac{1}{d}\right)^{i-1}\cdot i.
\end{aligned}$$

Now, the term $\sum_{i=1}^{d-1}\frac{1}{d}\cdot\left(1-\frac{1}{d}\right)^{i-1}\cdot i$ above should be familiar. Indeed, this is the contribution of the first $f=d-1$ possible values of a geometric random variable $X\sim Geo(p)$ with success probability $p=\frac{1}{d}$ to the expectation of X; that is,

$$\sum_{i=1}^{f}p(1-p)^{i-1}\cdot i = \sum_{i=1}^{\infty}p(1-p)^{i-1}\cdot i - \sum_{i=f+1}^{\infty}p(1-p)^{i-1}\cdot i.$$

Now, the former term in the right hand side of the above equation is just $\mathbb{E}[X]=\frac{1}{p}$, whereas the latter term is simply $\Pr[X>f]\cdot\mathbb{E}[X|X>f]$, which, by memorylessness of the geometric distribution, is precisely $(1-p)^f\cdot\left(\frac{1}{p}+f\right)$. Evaluating this expression with $p=\frac{1}{d}$ and $f=d-1$ and plugging the result into our lower bound on the loss of Algorithm RANDOM-AMONG-HIGHEST, namely $\left(1-\frac{1}{d}\right)-\left(1-\frac{1}{d}\right)^{d}-\frac{1}{d}\cdot\sum_{i=1}^{d-1}\frac{1}{d}\cdot\left(1-\frac{1}{d}\right)^{i-1}\cdot i = \left(1-\frac{1}{d}\right)-\left(1-\frac{1}{d}\right)^{d}-\frac{1}{d}\cdot\left(d-\left(1-\frac{1}{d}\right)^{d-1}\cdot(2d-1)\right)$, we find that the loss of this algorithm is at least

$$\begin{aligned}
\mathbb{E}[Loss_{\text{RMWM}}] &= -\frac{1}{d}+\left(1-\frac{1}{d}\right)^{d}\cdot\left(-1+\frac{2d-1}{d-1}\right)\\
&= -\frac{1}{d}+\left(1-\frac{1}{d}\right)^{d}\cdot\left(1+\frac{1}{d-1}\right)\\
&\geq -\frac{3}{4}\cdot\frac{1}{d}+\left(1-\frac{1}{d}\right)^{d},
\end{aligned}$$

where the inequality follows from $\left(1-\frac{1}{d}\right)\geq 4^{-1/d}$ for all $d\geq 2$. We conclude that the competitive ratio of Algorithm RANDOM-AMONG-HIGHEST is at most $1-\left(1-\frac{1}{d}\right)^{d}+\frac{3}{4}\cdot\frac{1}{d} = 1-\left(1-\frac{1}{d}\right)^{d}+O\left(\frac{1}{d}\right)$, as claimed. □

References

[1] Ageev, A. A. and Sviridenko, M. I. 2004. Pipage rounding: A new method of constructing algorithms with proven performance guarantee. *Journal of Combinatorial Optimization 8,* 3, 307–328.

[2] Aggarwal, G., Goel, G., Karande, C., and Mehta, A. 2011. Online vertex-weighted bipartite matching and single-bid budgeted allocations. In *Proceedings of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA).* 1253–1264.

[3] Aggarwal, G., Motwani, R., Shah, D., and Zhu, A. 2003. Switch scheduling via randomized edge coloring. In *Proceedings of the 44th Symposium on Foundations of Computer Science (FOCS).* 502–512.

[4] Ahuja, R. K., Magnanti, T. L., and Orlin, J. B. 1993. *Network flows - theory, algorithms, and applications.* Prentice hall.

[5] Alon, N. 2003. A simple algorithm for edge-coloring bipartite multigraphs. *Information Processing Letters 85,* 6, 301–302.

[6] Arora, S., Frieze, A., and Kaplan, H. 2002. A new rounding procedure for the assignment problem with applications to dense graph arrangement problems. *Mathematical programming 92,* 1, 1–36.

[7] Azar, Y., Cohen, I. R., and Roytman, A. 2017. Online lower bounds via duality. In *Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA).* 1038–1050.

[8] Bahmani, B. and Kapralov, M. 2010. Improved bounds for online stochastic matching. In *Proceedings of the 18th Annual European Symposium on Algorithms (ESA).* 170–181.

[9] Bansal, N., Gupta, A., Li, J., Mestre, J., Nagarajan, V., and Rudra, A. 2012. When lp is the cure for your matching woes: Improved bounds for stochastic matchings. *Algorithmica 63,* 4, 733–762.

[10] Bertsimas, D., Teo, C., and Vohra, R. 1999. On dependent randomized rounding algorithms. *Operations Research Letters 24,* 3, 105–114.

[11] Brubach, B., Sankararaman, K. A., Srinivasan, A., and Xu, P. 2016. New algorithms, better bounds, and a novel model for online stochastic matching. In *Proceedings of the 24th Annual European Symposium on Algorithms (ESA)*. 24:1–24:16.

[12] Buchbinder, N., Jain, K., and Naor, J. S. 2007. Online primal-dual algorithms for maximizing ad-auctions revenue. In *Proceedings of the 15th Annual European Symposium on Algorithms (ESA)*. 253–264.

[13] Calinescu, G., Chekuri, C., Pál, M., and Vondrák, J. 2011. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM Journal on Computing (SICOMP) 40,* 6, 1740–1766.

[14] Chekuri, C., Vondrak, J., and Zenklusen, R. 2010. Dependent randomized rounding via exchange properties of combinatorial structures. In *Proceedings of the 51st Symposium on Foundations of Computer Science (FOCS)*. 575–584.

[15] Chekuri, C., Vondrák, J., and Zenklusen, R. 2011. Multi-budgeted matchings and matroid intersection via dependent rounding. In *Proceedings of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. 1080–1097.

[16] Cole, R. and Hopcroft, J. 1982. On edge coloring bipartite graphs. *SIAM Journal on Computing (SICOMP) 11,* 3, 540–546.

[17] Cole, R., Ost, K., and Schirra, S. 2001. Edge-coloring bipartite multigraphs in $O(E \log D)$ time. *Combinatorica 21,* 1, 5–12.

[18] Csima, J. and Lovász, L. 1992. A matching algorithm for regular bipartite graphs. *Discrete Applied Mathematics 35,* 3, 197–203.

[19] Devanur, N. R., Jain, K., and Kleinberg, R. D. 2013. Randomized primal-dual analysis of ranking for online bipartite matching. In *Proceedings of the 24th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. 101–107.

[20] Doerr, B. 2003. Non-independent randomized rounding. In *Proceedings of the 14th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. 506–507.

[21] Doerr, B. 2005. Nonindependent randomized rounding and an application to digital halftoning. *SIAM Journal on Computing (SICOMP) 34,* 2, 299–317.

[22] Doerr, B. 2006. Generating randomized roundings with cardinality constraints and derandomizations. In *Proceedings of the 23rd International Symposium on Theoretical Aspects of Computer Science (STACS)*. 571–583.

[23] Dubhashi, D. and Ranjan, D. 1996. Balls and bins: A study in negative dependence. *BRICS Report Series 3,* 25.

[24] Edmonds, J. 1965. Paths, trees, and flowers. *Canadian Journal of mathematics 17,* 3, 449–467.

[25] Egerváry, J. 1931. Matrixok kombinatorius tulajdonságairól. *Matematikai és Fizikai Lapok 38,* 1931, 16–28.

[26] Epstein, L., Levin, A., Segev, D., and Weimann, O. 2013. Improved bounds for online preemptive matching. In *Proceedings of the 30th International Symposium on Theoretical Aspects of Computer Science (STACS)*. 389.

[27] Feldman, J., Korula, N., Mirrokni, V., Muthukrishnan, S., and Pál, M. 2009a. Online ad assignment with free disposal. In *Proceedings of the 5th Conference on Web and Internet Economics (WINE)*. 374–385.

[28] Feldman, J., Mehta, A., Mirrokni, V., and Muthukrishnan, S. 2009b. Online stochastic matching: Beating 1-1/e. In *Proceedings of the 50th Symposium on Foundations of Computer Science (FOCS)*. 117–126.

[29] Gabow, H. N. and Kariv, O. 1982. Algorithms for edge coloring bipartite graphs and multigraphs. *SIAM Journal on Computing (SICOMP) 11,* 1, 117–129.

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

[31] Geary, R. 1935. The ratio of the mean deviation to the standard deviation as a test of normality. *Biometrika 27,* 3/4, 310–332.

[32] Goel, A., Kapralov, M., and Khanna, S. 2009. Perfect matchings in $O(n^{1.5})$ time in regular bipartite graphs. *arXiv preprint arXiv:0902.1617*.

[33] Goel, A., Kapralov, M., and Khanna, S. 2010. Perfect matchings via uniform sampling in regular bipartite graphs. *ACM Transactions on Algorithms (TALG) 6,* 2, 27.

[34] Goel, A., Kapralov, M., and Khanna, S. 2013. Perfect matchings in $O(n \log n)$ time in regular bipartite graphs. *SIAM Journal on Computing (SICOMP) 42,* 3, 1392–1404.

[35] Goel, G. and Mehta, A. 2008. Online budgeted matching in random input models with applications to adwords. In *Proceedings of the 19th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. 982–991.

[36] Haeupler, B., Mirrokni, V. S., and Zadimoghaddam, M. 2011. Online stochastic weighted matching: Improved approximation algorithms. In *Proceedings of the 7th Conference on Web and Internet Economics (WINE)*. 170–181.

[37] Hall, P. 1935. On representatives of subsets. *Journal of the London Mathematical Society 1,* 1, 26–30.

[38] Hopcroft, J. E. and Karp, R. M. 1971. An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. In *Proceedings of the 12th Annual Symposium on Switching and Automata Theory (SWAT)*. 122–125.

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

[40] Joag-Dev, K. and Proschan, F. 1983. Negative association of random variables with applications. *The Annals of Statistics*, 286–295.

[41] Kalyanasundaram, B. and Pruhs, K. R. 2000. An optimal deterministic algorithm for online b-matching. *Theoretical Computer Science 233,* 1, 319–325.

[42] Karande, C., Mehta, A., and Tripathi, P. 2011. Online bipartite matching with unknown distributions. In *Proceedings of the 43rd Annual ACM Symposium on Theory of Computing (STOC)*. 587–596.

[43] Karp, R. M., Vazirani, U. V., and Vazirani, V. V. 1990. An optimal algorithm for on-line bipartite matching. In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing (STOC)*. 352–358.

[44] Khursheed, A. and Lai Saxena, K. 1981. Positive dependence in multivariate distributions. *Communications in Statistics - Theory and Methods 10,* 12, 1183–1196.

[45] König, D. 1916. Über graphen und ihre anwendung auf determinantentheorie und mengenlehre. *Mathematische Annalen 77,* 4, 453–465.

[46] König, D. 1931. Gráfok és mátrixok. *Matematikai és Fizikai Lapok 38,* 1931, 116–119.

[47] Kuhn, H. W. 1955. The hungarian method for the assignment problem. *Naval research logistics quarterly 2,* 1-2, 83–97.

[48] Lovász, L. and Plummer, M. D. 2009. *Matching theory*. Vol. 367. American Mathematical Society.

[49] Madry, A. 2013. Navigating central path with electrical flows: From flows to matchings, and back. In *Proceedings of the 54th Symposium on Foundations of Computer Science (FOCS)*. 253–262.

[50] Mahdian, M., Nazerzadeh, H., and Saberi, A. 2007. Allocating online advertisement space with unreliable estimates. In *Proceedings of the 8th ACM Conference on Electronic Commerce (EC)*. 288–294.

[51] Mahdian, M. and Yan, Q. 2011. Online bipartite matching with random arrivals: an approach based on strongly factor-revealing lps. In *Proceedings of the 43rd Annual ACM Symposium on Theory of Computing (STOC)*. 597–606.

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

[53] Mehta, A. 2012. Online matching and ad allocation. *Theoretical Computer Science 8,* 4, 265–368.

[54] Mehta, A., Saberi, A., Vazirani, U., and Vazirani, V. 2007. Adwords and generalized online matching. *Journal of the ACM (JACM) 54,* 5, 22.

[55] Motwani, R. and Raghavan, P. 2010. *Randomized algorithms*. Cambridge University Press.

[56] Naor, J. S. and Wajc, D. 2015. Near-optimum online ad allocation for targeted advertising. In *Proceedings of the 16th ACM Conference on Economics and Computation (EC)*. 131–148.

[57] Panconesi, A. and Srinivasan, A. 1997. Randomized distributed edge coloring via an extension of the chernoff–hoeffding bounds. *SIAM Journal on Computing (SICOMP) 26,* 2, 350–368.

[58] Schrijver, A. 1998. Bipartite edge coloring in O(Δm) time. *SIAM Journal on Computing (SICOMP) 28,* 3, 841–846.

[59] Wallenius, K. T. Biased sampling; the noncentral hypergeometric probability distribution. Ph.D. thesis, Stanford University.

[60] Yao, A. C.-C. 1977. Probabilistic computations: Toward a unified measure of complexity. In *Proceedings of the 18th Symposium on Foundations of Computer Science (FOCS).* 222–227.