# When LP is the Cure for Your Matching Woes: Improved Bounds for Stochastic Matchings

## (Extended Abstract)

Nikhil Bansal[1], Anupam Gupta[2⋆], Jian Li[3], Julián Mestre[4], Viswanath Nagarajan[1], and Atri Rudra[5⋆⋆]

[1] IBM T.J. Watson Research Center, Yorktown Heights, NY, USA.
[2] Computer Science Department, Carnegie Mellon University, Pittsburgh, PA, USA.
[3] Computer Science Department, University of Maryland College Park, MD, USA.
[4] Max-Planck-Institut für Informatik, Saarbrücken, Germany.
[5] Computer Science & Engg. Dept., University at Buffalo, SUNY, Buffalo NY, USA.

**Abstract.** Consider a random graph model where each possible edge $e$ is present independently with some probability $p_e$. We are given these numbers $p_e$, and want to build a large/heavy matching in the randomly generated graph. However, the only way we can find out whether an edge is present or not is to query it, and if the edge is indeed present in the graph, we are forced to add it to our matching. Further, each vertex $i$ is allowed to be queried at most $t_i$ times. How should we *adaptively* query the edges to maximize the expected weight of the matching? We consider several matching problems in this general framework (some of which arise in kidney exchanges and online dating, and others arise in modeling online advertisements); we give LP-rounding based constant-factor approximation algorithms for these problems. Our main results are:

- We give a 5.75-approximation for weighted stochastic matching on general graphs, and a 5-approximation on bipartite graphs. This answers an open question from [Chen et al. ICALP 09].
- Combining our LP-rounding algorithm with the natural greedy algorithm, we give an improved 3.88-approximation for unweighted stochastic matching on general graphs and 3.51-approximation on bipartite graphs.
- We introduce a generalization of the stochastic *online* matching problem [Feldman et al. FOCS 09] that also models preference-uncertainty and timeouts of buyers, and give a constant factor approximation algorithm.

## 1 Introduction

Motivated by applications in kidney exchanges and online dating, Chen et al. [4] proposed the following stochastic matching problem: we want to find a maximum

⋆ Supported in part by NSF awards CCF-0448095 and CCF-0729022, and an Alfred P. Sloan Fellowship.

⋆⋆ Supported by NSF CAREER award CCF-0844796.

matching in a random graph $G$ on $n$ nodes, where each edge $(i,j) \in [\binom{n}{2}]$ exists with probability $p_{ij}$, independently of the other edges. However, all we are given are the probability values $\{p_{ij}\}$. To find out whether the random graph $G$ has the edge $(i,j)$ or not, we have to try to add the edge $(i,j)$ to our current matching (assuming that $i$ and $j$ are both unmatched in our current partial matching)—we call this "probing" edge $(i,j)$. As a result of the probe, we also find out if $(i,j)$ exists or not—and if the edge $(i,j)$ indeed exists in the random graph $G$, it gets irrevocably added to the matching. Such policies make sense, e.g., for dating agencies, where the only way to find out if two people are actually compatible is to send them on a date; moreover, if they do turn out to be compatible, then it makes sense to match them to each other. Finally, to model the fact that there might be a limit on the number of unsuccessful dates a person might be willing to participate in, "timeouts" on vertices are also provided. More precisely, valid policies are allowed, for each vertex $i$, to only probe at most $t_i$ edges incident to $i$. Similar considerations arise in kidney exchanges, details of which appear in [4].

Chen et al. asked the question: how can we devise probing policies to maximize the expected cardinality (or weight) of the matching? They showed that the greedy algorithm that probes edges in decreasing order of $p_{ij}$ (as long as their endpoints had not timed out) was a 4-approximation to the cardinality version of the stochastic matching problem. This greedy algorithm (and other simple greedy schemes) can be seen to be arbitrarily bad in the presence of weights, and they left open the question of obtaining good algorithms to *maximize the expected weight* of the matching produced. In addition to being a natural generalization, weights can be used as a proxy for revenue generated in matchmaking services. (The unweighted case can be thought of as maximizing the social welfare.) In this paper, we resolve the main open question from Chen et al.:

**Theorem 1** *There is a* 5.75*-approximation algorithm for the weighted stochastic matching problem. For bipartite graphs, there is a 5-approximation algorithm.*

Our main idea is to use the knowledge of edge probabilities to solve a linear program where each edge $e$ has a variable $0 \leq y_e \leq 1$ corresponding to the probability that a strategy probes $e$ (over all possible realizations of the graph). This is similar to the approach for stochastic packing problems considered by Dean et al. [6, 5]. We then give two different rounding procedures to attain the bounds claimed above. The first algorithm (§2.1) is very simple: it considers edges in a uniformly random order and probes each edge $e$ with probability proportional to $y_e$; the analysis uses Markov's inequality and a Chernoff-type bound (Lemma 2). The second algorithm (§2.2) is more nuanced: we use the $y$-values to define an auxiliary LP that is shown to be integral, and then probe only the edges chosen by this auxiliary LP; the analysis here requires more work and uses certain ideas from the *generalized assignment problem* [18].

This second rounding algorithm can also be extended to general graphs, but it results in a slightly worse approximation ratio of 7.5. However, this approach has the following two advantages:

- The probing strategy returned by the algorithm is in fact *matching-probing* [4], where we are given an additional parameter $k$ and edges need to be probed in $k$ rounds, each round being a matching. It is clear that this matching-probing model is more restrictive than the usual *edge-probing* model (with timeouts $\min\{t_i, k\}$) where one edge is probed at a time; this algorithm obtains a matching-probing strategy that is only a small constant factor worse than the optimal edge-probing strategy. Hence we also obtain the same constant approximation guarantee for weighted stochastic matching in the matching-probing model; previously only a logarithmic approximation in the unweighted case was known [4].
- We can combine this algorithm with the greedy algorithm [4] to obtain improved bounds for *unweighted* stochastic matching:

**Theorem 2** *There is a* 3.88-*approximation algorithm for the unweighted stochastic matching problem; this improves to a* 3.51-*approximation algorithm in bipartite graphs.*

Apart from solving these open problems and giving improved ratios, our LP-based analysis turns out to be applicable in a wider context:

***Online*** **Stochastic Matching Revisited.** In a bipartite graph $(A, B; E)$ of items $i \in A$ and potential buyer types $j \in B$, $p_{ij}$ denotes the probability that a buyer of type $j$ will buy item $i$. A sequence of $n$ buyers are to arrive online, where the type of each buyer is an i.i.d. sample from $B$ according to some pre-specified distribution—when a buyer of type $j$ appears, he can be shown a list $L$ of up to $t_j$ as-yet-unsold items, and the buyer buys the *first* item on the list according to the given probabilities $p_{\cdot,j}$. (Note that with probability $\prod_{i \in L}(1 - p_{ij})$, the buyer leaves without buying anything.) What items should we show buyers when they arrive online, and in which order, to maximize the expected weight of the matching?

**Theorem 3** *There is a* 7.92-*approximation algorithm for the above online stochastic matching problem.*

This question is an extension of similar online stochastic matching questions considered earlier in [7]—in that paper, $w_{ij}, p_{ij} \in \{0, 1\}$ and $t_j = 1$. Our model tries to capture the facts that buyers may have a limited attention span (using the timeouts), they might have uncertainties in their preferences (using edge probabilities), and that they might buy the first item they like rather than scanning the entire list.

**Other Extensions.** The proof in [4] that the greedy algorithm for stochastic matching was a 4-approximation in the unweighted case was based on a somewhat delicate charging scheme involving the decision trees of the algorithm and the optimal solution. We show that the greedy algorithm, which was defined without reference to any LP, admits a simple LP-based analysis and is a 5 approximation.

We also consider the model from [4] where one can probe as many as $C$ edges in parallel, as long as these $C$ edges form a matching; the goal is to maximize the expected weight of the matched edges after $k$ rounds of such probes. We improve on the $\min\{k, C\}$-approximation offered in [4] (which only works for the unweighted version), and present a constant factor approximation for the weighted cardinality constrained multiple-round stochastic matching.

We also extend our analysis to a much more general situation where we try to pack $k$-hyperedges with random sizes into a $d$-dimensional knapsack of a given size; this is just the stochastic knapsack problem of [5], but where we consider the situation where $k \ll d$. For this setting of parameters, we improve on the $\sqrt{d}$-approximation of [5] and present a $2k$-approximation algorithm.

Due to lack of space, the details on these extensions as well as the omitted proofs in this extended abstract can be found in a full version of the paper [1].

**Related Work.** As mentioned above, perhaps the work most directly related to this work is that on stochastic knapsack problems (Dean et al. [6, 5]) and multi-armed bandits (see [9, 10] and references therein). Also related is some recent work [2] on budget constrained auctions, which uses similar LP rounding ideas.

In recent years stochastic optimization problems have drawn much attention from the theoretical computer science community where stochastic versions of several classical combinatorial optimization problems have been studied. Some general techniques have also been developed [11, 19]. See [20] for a survey.

The online bipartite matching problem was first studied in the seminal paper by Karp *et al.* [13] and an optimal $1 - 1/e$ competitive online algorithm was obtained. Katriel *et al.* [14] considered the two-stage stochastic min-cost matching problem. In their model, we are given in a first stage probabilistic information about the graph and the cost of the edges is low; in a second stage, the actual graph is revealed but the costs are higher. The original online stochastic matching problem was studied recently by Feldman et al. [7]. They gave a 0.67-competitive algorithm, beating the optimal $1 - 1/e$-competitiveness known for worst-case models [13, 12, 16, 3, 8]. Our model differs from that in having a bound on the number of items each incoming buyer sees, that each edge is only present with some probability, and that the buyer scans the list linearly (until she times out) and buys the first item she likes.

Our problem is also related to the Adwords problem [16], which has applications to sponsored search auctions. The problem can be modeled as a bipartite matching problem as follows. We want to assign every vertex (a query word) on one side to a vertex (a bidder) on the other side. Each edge has a weight, and there is a budget on each bidder representing the upper bound on the total weight of edges that may be assigned to it. The objective is to maximize the total revenue. The stochastic version in which query words arrive according to some known probability distribution has also been studied [15].

**Preliminaries.** For any integer $m \geq 1$, define $[m]$ to be the set $\{1, \ldots, m\}$. For a maximization problem, an $\alpha$-approximation algorithm is one that computes a solution with expected objective value at least $1/\alpha$ times the expected value of the optimal solution.

We must clarify here the notion of an optimal solution. In standard worst case analysis we would compare our solution against the optimal *offline* solution, e.g. the value of the maximum matching, where the offline knows all the edge instantiations in advance (i.e. which edge will appear when probed, and which will not). However, it can be easily verified that due to the presence of timeouts, this adversary is too strong [4]. Hence, for all problems in this paper we consider the setting where even the optimum does not know the exact instantiation of an edge until it is probed. This gives our algorithms a level playing field. The optimum thus corresponds to a "strategy" of probing the edges, which can be chosen from an exponentially large space of potentially adaptive strategies.

We note that our algorithms in fact yield *non-adaptive* strategies for the corresponding problems, that are only constant factor worse than the adaptive optimum. This is similar to previous results on stochastic packing problems: knapsack (Dean et al. [6, 5]) and multi-armed bandits (Guha-Munagala [9, 10] and references therein).

## 2 Stochastic Matching

We consider the following stochastic matching problem. The input is an undirected graph $G = (V, E)$ with a weight $w_e$ and a probability value $p_e$ on each edge $e \in E$. In addition, there is an integer value $t_v$ for each vertex $v \in V$ (called *patience parameter*). Initially, each vertex $v \in V$ has patience $t_v$. At each step in the algorithm, any edge $e(u, v)$ such that $u$ and $v$ have positive remaining patience can be probed. Upon probing edge $e$, one of the following happens: (1) with probability $p_e$, vertices $u$ and $v$ get *matched* and are removed from the graph (along with all adjacent edges), or (2) with probability $1 - p_e$, the edge $e$ is removed and the remaining patience numbers of $u$ and $v$ get reduced by 1. An algorithm is an adaptive strategy for probing edges: its performance is measured by the expected weight of matched edges. The *unweighted* stochastic matching problem is the special case when all edge-weights are uniform.

Consider the following linear program: as usual, for any vertex $v \in V$, $\partial(v)$ denotes the edges incident to $v$. Variable $y_e$ denotes the probability that edge $e = (u, v)$ gets probed in the adaptive strategy, and $x_e = p_e \cdot y_e$ denotes the probability that $u$ and $v$ get matched in the strategy. (This LP is similar to the LP used for general stochastic packing problems by Dean, Goemans and Vondrák [5].)

$$\text{maximize } \textstyle\sum_{e \in E} w_e \cdot x_e \qquad \text{(LP1)}$$

$$\textstyle\sum_{e \in \partial(v)} x_e \leq 1 \qquad \forall v \in V \qquad (1)$$

$$\textstyle\sum_{e \in \partial(v)} y_e \leq t_i \qquad \forall v \in V \qquad (2)$$

$$x_e = p_e \cdot y_e \qquad \forall e \in E \qquad (3)$$

$$0 \leq y_e \leq 1 \qquad \forall e \in E \qquad (4)$$

It can be shown that the LP above is a valid relaxation for the stochastic matching problem.

## 2.1 Weighted Stochastic Matching: General Graphs

Our algorithm first solves (LP1) to optimality and uses the optimal solution $(x, y)$ to obtain a non-adaptive strategy achieving expected value $\Omega(1) \cdot (w \cdot x)$. Next, we present the algorithm. We note that the optimal solution $(x, y)$ to the above LP gives an upper-bound on any adaptive strategy. Let $\alpha \geq 1$ be a constant to be set later. The algorithm first fixes a uniformly random permutation $\pi$ on edges $E$. It then inspects edges in the order of $\pi$, and *probes* only a subset of the edges. A vertex $v \in V$ is said to have *timed out* if $t_v$ edges incident to $v$ have already been probed (i.e. its remaining patience reduces to 0); and vertex $v$ is said to be *matched* if it has already been matched to another vertex. An edge $(u, v)$ is called *safe* at the time it is considered if (A) neither $u$ nor $v$ is matched, and (B) neither $u$ nor $v$ has timed out. The algorithm is the following:

1. Pick a permutation $\pi$ on edges $E$ uniformly at random
2. For each edge $e$ in the ordering $\pi$, do:
   a. If $e$ is safe then probe it with probability $y_e/\alpha$, else do not probe it.

In the rest of this section, we prove that this algorithm achieves a 5.75-approximation. We begin with the following property:

**Lemma 1** *For any edge $(u, v) \in E$, when $(u, v)$ is considered under $\pi$,*
*(a) the probability that vertex $u$ has timed out is at most $\frac{1}{2\alpha}$, and*
*(b) the probability that vertex $u$ is matched is at most $\frac{1}{2\alpha}$.*

**Proof:** We begin with the proof of part (a). Let random variable $U$ denote the number of probes incident to vertex $u$ by the time edge $(u, v)$ is considered in $\pi$.

$$\begin{aligned}\mathbb{E}[U] &= \textstyle\sum_{e\in\partial(u)} \Pr[\text{edge } e \text{ appears before } (u,v) \text{ in } \pi \text{ AND } e \text{ is probed}] \\ &\leq \textstyle\sum_{e\in\partial(u)} \Pr[\text{edge } e \text{ appears before } (u,v) \text{ in } \pi] \cdot \frac{y_e}{\alpha} = \textstyle\sum_{e\in\partial(u)} \frac{y_e}{2\alpha} \quad \leq \quad \frac{t_u}{2\alpha}.\end{aligned}$$

The first inequality above follows from the fact that the probability that edge $e$ is probed (conditioned on $\pi$) is at most $y_e/\alpha$. The second equality follows since $\pi$ is a u.a.r. permutation on $E$. The last inequality is by the LP constraint (2). The probability that vertex $u$ has timed out when $(u, v)$ is considered equals $\Pr[U \geq t_u] \leq \frac{\mathbb{E}[U]}{t_u} \leq \frac{1}{2\alpha}$, by the Markov inequality. This proves part (a). The proof of part (b) is identical (where we consider the event that an edge is matched instead of being probed and replace $y_e$ and $t_u$ by $x_e$ and 1 respectively and use the LP constraint (1)) and is omitted. ■

Now, a vertex $u \in V$ is called *low-timeout* if $t_u = 1$, else $u$ is called a *high-timeout* vertex if $t_u \geq 2$. We next prove the following bound for high-timeout vertices that is stronger than the one from Lemma 1(a).

**Lemma 2** *Suppose $\alpha \geq$ e. For a high-timeout vertex $u \in V$, and any edge $f$ incident to $u$, the probability that $u$ has timed out when $f$ is considered in $\pi$ is at most $\frac{2}{3\alpha^2}$.*

Using this, we can analyze the probability that an edge is safe. (The proof is a case analysis on whether the end-points are low-timeout or high-timeout.)

**Lemma 3** *For* $\alpha \geq \mathrm{e}$*, an edge* $f = (u, v)$ *is safe with probability at least* $(1 - \frac{1}{\alpha} - \frac{4}{3\alpha^2})$ *when* $f$ *is considered under a random permutation* $\pi$*.*

Theorem 1 follows from the definition of the algorithm, the LP formulation and using Lemma 3 (with $\alpha = 1 + \sqrt{5}$).

### 2.2 Weighted Stochastic Matching: Bipartite Graphs

In this section, we obtain an improved bound for stochastic matching on bipartite graphs, via a different rounding procedure. In fact, the algorithm produces a *matching-probing strategy* whose expected value is a constant fraction of the optimal value of (LP1) (which was for edge-probing). A similar rounding algorithm also works for non-bipartite graphs, achieving a slightly weaker bound. Furthermore, we show in the next subsection that this LP-rounding algorithm can be combined with the greedy algorithm of [4] to get improved bounds for *unweighted* stochastic matching.

**Algorithm** ROUND-COLOR-PROBE. First, we find an optimal fractional solution $(x, y)$ to (LP1) and round $x$ to identify a set of interesting edges $\widehat{E}$. Then we use edge coloring to partition $\widehat{E}$ into a small collection of matchings $M_1, \ldots, M_h$, which are then probed in a random order. If we are only interested in edge-probing strategies, probing the edges in $\widehat{E}$ in random order would suffice. We denote this edge-probing strategy by EDGE-PROBE. The key difference from the rounding algorithm of the previous subsection is in the choice of $\widehat{E}$, which we describe next.

**Computing $\widehat{E}$.** Our scheme is based on the rounding procedure of Shmoys and Tardos for the generalized assignment problem [18]. Let $q^*$ denote the values of $x$-variables in an optimal solution to (LP1). For each vertex $u$, sort the edges incident on $u$ in non-increasing values of their probabilities $e_1^u, e_2^u, \ldots, e_{\mathsf{deg}(u)}^u$, and write a new LP:

$$\text{maximize } \textstyle\sum_{e \in E} w_e\, p_e \cdot z_e \qquad \text{(LP2)}$$

$$\textstyle\sum_{e \in \partial(u)} z_e \leq t_u \qquad \forall\, i \in V \qquad (5)$$

$$\textstyle\sum_{j=1}^{i} z_{e_j^u} \leq \left\lceil \sum_{j=1}^{i} q^*_{e_j^u} \right\rceil \qquad \forall u \in V, i = 1, \ldots, \mathsf{deg}(u) \qquad (6)$$

$$z_e \in [0, 1] \qquad \forall e \in E$$

Notice that $q^*$ is a feasible solution of this new program. Thus, the optimal value of (LP2) is at least that of (LP1). As shown in the next lemma, this new linear program has the nice property of being integral.

**Lemma 4** *All basic solutions of* (LP2) *are integral.*

Let $\widehat{q}$ be an optimal basic (and therefore integral) solution of (LP2) and $\widehat{E}$ be the set of edges in the support of $\widehat{q}$, i.e., $\widehat{E} = \{e \mid \widehat{q}_e = 1\}$. Let $h = \max_{v \in V} \mathsf{deg}_{\widehat{E}}(v)$. Using König's Theorem [17, Ch. 20], we can decompose $\widehat{E}$ into $h$ matchings in polynomial time. Notice that each vertex $u \in V$ will be matched in at most $t_u$ of these matchings.

**Analysis.** We now analyze the performance guarantee. First, we notice that the downside of exchanging LPs is that the "expected number of successful probes" incident on a vertex can be larger than 1. However, the excess can be bounded by the next lemma.

**Lemma 5** *For any feasible (integral or fractional) solution $\widehat{q}$ of* (LP2) *we have*

$$\sum_{e \in \partial(u)} p_e \widehat{q}_e \ \leq \ 1 + p_{\max} \qquad \forall u \in V, \ \textit{where } p_{max} = \max_{e \in E} p_e.$$

It only remains to bound the probability that a given edge $e = (u, v) \in \widehat{E}$ is in fact probed by our probing strategy. Consider a random permutation of the $h$ matchings used by the edge coloring. Let $\pi$ be the edge ordering induced by this permutation where edges within a matching are listed in some arbitrary but fixed order. Let us denote by $B(e, \pi)$ the set of edges incident on $u$ or $v$ that appear before $e$ in $\pi$. It is not hard to see that

$$\Pr[\, e \text{ was probed} \,] \geq \mathbb{E}_\pi \left[ \prod_{f \in B(e,\pi)} (1 - p_f) \right]; \tag{7}$$

here we assume that $\prod_{f \in B(e,\pi)} (1 - p_f) = 1$ when $B(e, \pi) = \emptyset$.

Notice that in (7) we only care about the order of edges incident on $u$ and $v$. Furthermore, the expectation does not range over all possible orderings of these edges, but only those that are consistent with some matching permutation. We call this type of restricted ordering *random matching ordering* and we denote it by $\pi$; similarly, we call an unrestricted ordering *random edge ordering* and we denote it by $\sigma$. Our plan to lower bound the probability of $e$ being probed is to study first the expectation in (7) over random edge orderings and then to show that the expectation can only increase when restricted to range over random matching orderings.

The following simple lemma is useful in several places.

**Lemma 6** *Let $r$ and $p_{max}$ be positive real values. Consider the problem of minimizing $\prod_{i=1}^t (1 - p_i)$ subject to the constraints $\sum_{i=1}^t p_i \leq r$ and $0 \leq p_i \leq p_{\max}$ for $i = 1, \ldots, t$. Denote the minimum value by $\eta(r, p_{\max})$. Then,*

$$\eta(r, p_{\max}) = (1 - p_{\max})^{\lfloor \frac{r}{p_{\max}} \rfloor} \left(1 - (r - \left\lfloor \frac{r}{p_{\max}} \right\rfloor p_{\max})\right) \geq (1 - p_{\max})^{r/p_{\max}}.$$

Let $\partial_{\widehat{E}}(e)$ be the set of edges in $\widehat{E} \setminus \{e\}$ incident on either endpoint of $e$.

**Lemma 7** *Let $e$ be an edge in $\widehat{E}$ and let $\sigma$ be a random edge ordering. Let $p_{\max} = \max_{f \in \widehat{E}} p_f$. Assume that $\sum_{f \in \partial_{\widehat{E}}(e)} p_f \leq r$ for all $u \in V$. Then,*

$$\mathbb{E}_\sigma \left[ \prod_{f \in B(e,\sigma)} (1 - p_f) \right] \geq \int_0^1 \eta(xr, x p_{\max}) \, \mathrm{d}x.$$

**Corollary 1.** *Let* $\rho(r, p_{\max}) = \int_0^1 \eta(xr, xp_{\max})\,\mathrm{d}x$. *For any* $r, p_{\max} > 0$, *we have*

1. $\rho(r, p_{\max})$ *is convex and decreasing on* $r$.
2. $\rho(r, p_{\max}) \geq \frac{1}{r+p_{\max}} \cdot \left(1 - (1-p_{\max})^{1+\frac{r}{p_{\max}}}\right) > \frac{1}{r+p_{\max}} \cdot \left(1 - e^{-r}\right)$

**Lemma 8** *Let* $e = (u, v) \in \widehat{E}$. *Let* $\pi$ *be a random matching ordering and* $\sigma$ *be a random edge ordering of the edges adjacent to* $u$ *and* $v$. *Then*

$$\mathbb{E}_\pi \left[\prod_{f \in B(e,\pi)} (1-p_f)\right] \geq \mathbb{E}_\sigma \left[\prod_{f \in B(e,\sigma)} (1-p_f)\right].$$

Everything is in place to derive a bound on the expected weight of the matching found by our algorithm.

**Theorem 4** *If* $G$ *is bipartite then there is a* $1/\rho(2+2p_{\max}, p_{\max})$ *approximation with* $\rho$ *as in Corollary 1. The worst ratio is attained at* $p_{\max} = 1$ *and is* 5.

**Proof:** Recall that the optimal value of (LP2) is exactly $\sum_{e \in \widehat{E}} w_e\, p_e$. On the other hand, the expected size of the matching found by the algorithm is

$$\begin{aligned}\mathbb{E}\left[\,\text{our solution}\,\right] &= \sum_{e \in \widehat{E}} w_e\, p_e \,\Pr\left[\, e \text{ was probed}\,\right] \geq \sum_{e \in \widehat{E}} w_e\, p_e\, \mathbb{E}_\pi \Big[ \prod_{f \in B(e,\pi)} (1-p_f) \Big] \\ &\geq \sum_{e \in \widehat{E}} w_e\, p_e\, \mathbb{E}_\sigma \Big[ \prod_{f \in B(e,\sigma)} (1-p_f) \Big] \geq \rho(2+2p_{\max}, p_{\max}) \text{ value}(\widehat{q})\end{aligned}$$

where the first inequality follows from (7) and the second from Lemma 8—here $\pi$ is a random matching ordering and $\sigma$ is a random edge ordering. The third inequality follows from Lemma 7 and setting $r = 2 + 2p_{max}$ (using Lemma 5 on endpoints of $e$). Recall that the value of $\widehat{q}$ is at least the value of $q^*$ and this, in turn, is an upper bound on the cost of an optimal probing strategy. ■

In the full version of the paper, we present the final version of ROUND-COLOR-PROBE which obtains a a slightly weaker bound of $\frac{k+1}{k} \cdot \frac{3}{2} \cdot \frac{1}{\rho\,(2+2p_{\max}, p_{\max})}$ for the matching-probing model on general graphs, and EDGE-PROBE which is a $\frac{3}{2} \cdot \frac{1}{\rho\,(2+2p_{\max}, p_{\max})}$-approximation for the edge-probing model on general graphs.

### 2.3 Improved Bounds for Unweighted Stochastic Matching

In this subsection, we consider the unweighted stochastic matching problem, and show that our algorithm from §2.2 can be combined with the natural greedy algorithm [4] to obtain a better approximation guarantee than either algorithm can achieve on its own. Basically, our algorithm attains its worst ratio when $p_{\max}$ is large and greedy attains its worst ratio when $p_{\max}$ is small. Therefore, we can combine the two algorithms as follows: We probe edges using the greedy heuristic until the maximum edge probability in the remaining graph is less than a critical value $p_c$, at which point we switch to algorithm EDGE-PROBE.

**Theorem 5** *Suppose we use the greedy rule until all remaining edges have probability less than $p_c$, at which point we switch to an algorithm with approximation ratio $\gamma(p_c)$. Then the approximation ratio of the overall scheme is $\alpha(p_c) = \max\{4 - p_c, \gamma(p_c)\}$.*

The proof follows by an induction on the size of the problem instance (and we use existing bounds on the optimum from Chen *et al.* [4]).

The proof of Theorem 2 follows by setting the cut-off point $p_c = 0.49$ for bipartite graphs and $p_c = 0.12$ for general graphs and using the EDGE-PROBE algorithm.

We remark that the approximation ratio of the algorithm in §2.1 does not depend on $p_{\max}$, thus we can not combine that algorithm with the greedy algorithm to get a better bound.

## 3 Stochastic *Online* Matching (Revisited)

As mentioned in the introduction, the stochastic online matching problem is best imagined as selling a finite set of goods to buyers that arrive over time. The input to the problem consists of a bipartite graph $G = (A, B, A \times B)$, where $A$ is the set of *items* that the seller has to offer, with exactly one copy of each item, and $B$ is a set of *buyer types/profiles*. For each buyer type $b \in B$ and item $a \in A$, $p_{ab}$ denotes the probability that a buyer of type $b$ will like item $a$, and $w_{ab}$ denotes the revenue obtained if item $a$ is sold to a buyer of type $b$. Each buyer of type $b \in B$ also has a patience parameter $t_b \in \mathbb{Z}_+$. There are $n$ buyers arriving online, with $e_b \in \mathbb{Z}$ denoting the expected number of buyers of type $b$, with $\sum e_b = n$. Let $\mathcal{D}$ denote the induced probability distribution on $B$ by defining $\Pr_{\mathcal{D}}[b] = e_b/n$. All the above information is given as input.

The stochastic online model is the following: At each point in time, a buyer arrives, where her type $b \in_{\mathcal{D}} B$ is an i.i.d. draw from $\mathcal{D}$. The algorithm now shows her *up to $t_b$ distinct items one-by-one*: the buyer likes each item $a \in A$ shown to her independently with probability $p_{ab}$. The buyer purchases the first item that she is *offered and likes*; if she buys item $a$, the revenue accrued is $w_{ab}$. If she does not like any of the items shown, she leaves without buying. The objective is to maximize the expected revenue.

We get the stochastic online matching problem of Feldman et al. [7] if we have $w_{ab} = p_{ab} \in \{0, 1\}$, in which case we need only consider $t_b = 1$. Their focus was on beating the $1 - 1/e$-competitiveness known for worst-case models [13, 12, 16, 3, 8]; they gave a 0.67-competitive algorithm that works for the unweighted case whp; whereas our results are for the weighted case (with preference-uncertainty and timeouts), but only in expectation.

*By making copies of buyer types, we may assume that $e_b = 1$ for all $b \in B$, and $\mathcal{D}$ is uniform over $B$.* For a particular run of the algorithm, let $\hat{B}$ denote the actual set of buyers that arrive during that run. Let $\hat{G} = (A, \hat{B}, A \times \hat{B})$, where for each $a \in A$ and $\hat{b} \in \hat{B}$ (and suppose its type is some $b \in B$), the probability associated with edge $(a, \hat{b})$ is $p_{ab}$ and its weight is $w_{ab}$. Moreover, for each $\hat{b} \in \hat{B}$

(with type, say, $b \in B$), set its patience parameter to $t_{\hat{b}} = t_b$. We will call this the *instance graph*; the algorithm sees the vertices of $\hat{B}$ in random order, and has to adaptively find a large matching in $\hat{G}$.

It now seems reasonable that the algorithm of §2.1 should work here. But the algorithm does not know $\hat{G}$ (the actual instantiation of the buyers) up front, it only knows $G$, and hence some more work is required to obtain an algorithm. Further, as was mentioned in the preliminaries, we use $\mathsf{OPT}$ to denote the optimal adaptive strategy (instead of the optimal offline matching in $\hat{G}$ as was done in [7]), and compare our algorithm's performance with this $\mathsf{OPT}$.

**The Linear Program.** For a graph $H = (A, C, A \times C)$ with each edge $(a, c)$ having a probability $p_{ac}$ and weight $w_{ac}$, and vertices in $C$ having patience parameters $t_j$, consider the LP$(H)$:

$$\text{maximize } \textstyle\sum_{a \in A,\, c \in C} \; w_{ac} \cdot x_{ac} \tag{LP3}$$

$$\textstyle\sum_{c \in C} x_{ac} \leq 1 \qquad \forall a \in A \tag{8}$$

$$\textstyle\sum_{a \in A} x_{ac} \leq 1 \qquad \forall c \in C \tag{9}$$

$$\textstyle\sum_{a \in A} y_{ac} \leq t_c \qquad \forall c \in C \tag{10}$$

$$x_{ac} = p_{ac} \cdot y_{ac} \qquad \forall a \in A,\, c \in C \tag{11}$$

$$y_{ac} \in [0, 1] \qquad \forall a \in A,\, c \in C \tag{12}$$

Note that this LP is very similar to the one in §2, but the vertices on the left do not have timeout values. Let $\mathsf{LP}(H)$ denote the optimal value of this LP.

**The algorithm:**

1. Before buyers arrive, solve the LP on the expected graph $G$ to get values $y^*$.
2. When any buyer $\hat{b}$ (of type $b$) arrives online:
   a. If $\hat{b}$ is the first buyer of type $b$, consider the items $a \in A$ in u.a.r. order. One by one, offer each unsold item $a$ to $\hat{b}$ independently with probability $y^*_{ab}/\alpha$; stop if either $t_b$ offers are made or $\hat{b}$ purchases any item.
   b. If $\hat{b}$ is not the first arrival of type $b$, do not offer any items to $\hat{b}$.

In the following, we prove that our algorithm achieves a constant approximation to the stochastic online matching problem. The first lemma show that the expected value obtained by the best online adaptive algorithm is bounded above by $\mathbb{E}[\mathsf{LP}(\hat{G})]$.

**Lemma 9** *The optimal value* $\mathsf{OPT}$ *of the given instance is at most* $\mathbb{E}[\mathsf{LP}(\hat{G})]$*, where the expectation is over the random draws to create* $\hat{G}$*.*

The proof of the next lemma is similar to the analysis of Theorem 1 for weighted stochastic matching.

**Lemma 10** *Our expected revenue is at least* $\left(1 - \frac{1}{e}\right) \frac{1}{\alpha} \left(1 - \frac{1}{\alpha} - \frac{2}{3\alpha^2}\right) \cdot \mathsf{LP}(G)$.

Note that we have shown that $\mathbb{E}[\mathsf{LP}(\hat{G})]$ is an upper bound on $\mathsf{OPT}$, and that we can get a constant fraction of $\mathsf{LP}(G)$. The final lemma relates these two, namely the LP-value of the expected graph $G$ (computed in Step 1) to the expected LP-value of the instantiation $\hat{G}$; the proof uses a simple but subtle duality-based argument.

**Lemma 11** $\mathsf{LP}(G) \geq \mathbb{E}[\mathsf{LP}(\hat{G})]$.

Lemmas 9, 10 and 11, with $\alpha = \frac{2}{\sqrt{3}-1}$, prove Theorem 3.

## References


1. Bansal, N., Gupta, A., Li, J., Mestre, J., Nagarajan, V., Rudra, A.: When LP is the Cure for Your Matching Woes: Improved Bounds for Stochastic Matchings. arXiv (2010)
2. Bhattacharya, S., Goel, G., Gollapudi, S., Munagala, K.: Budget constrained auctions with heterogeneous items. In: STOC (2009), arxiv:abs/0907.4166
3. Birnbaum, B.E., Mathieu, C.: On-line bipartite matching made simple. SIGACT News 39(1), 80–87 (2008)
4. Chen, N., Immorlica, N., Karlin, A.R., Mahdian, M., Rudra, A.: Approximating matches made in heaven. In: ICALP (1). pp. 266–278 (2009)
5. Dean, B.C., Goemans, M.X., Vondrák, J.: Adaptivity and approximation for stochastic packing problems. In: SODA. pp. 395–404 (2005)
6. Dean, B.C., Goemans, M.X., Vondrák, J.: Approximating the stochastic knapsack problem: the benefit of adaptivity. Math. Oper. Res. 33(4), 945–964 (2008), http://dx.doi.org/10.1287/moor.1080.0330
7. Feldman, J., Mehta, A., Mirrokni, V.S., Muthukrishnan, S.: Online stochastic matching: Beating $1 - 1/e$. In: FOCS (2009), *Arxiv:abs/0905.4100*
8. Goel, G., Mehta, A.: Online budgeted matching in random input models with applications to adwords. In: SODA. pp. 982–991 (2008)
9. Guha, S., Munagala, K.: Approximation algorithms for partial-information based stochastic control with markovian rewards. In: FOCS. pp. 483–493 (2007)
10. Guha, S., Munagala, K.: Multi-armed bandits with metric switching costs. In: ICALP. pp. 496–507 (2009)
11. Gupta, A., Pál, M., Ravi, R., Sinha, A.: Boosted sampling: approximation algorithms for stochastic optimization. In: STOC. pp. 417–426. ACM (2004)
12. Kalyanasundaram, B., Pruhs, K.: Online weighted matching. J. Algorithms 14(3), 478–488 (1993)
13. Karp, R.M., Vazirani, U.V., Vazirani, V.V.: An optimal algorithm for on-line bipartite matching. In: STOC. pp. 352–358 (1990)
14. Katriel, I., Kenyon-Mathieu, C., Upfal, E.: Commitment under uncertainty: Two-stage stochastic matching problems. Theoretical Computer Science 408(2-3), 213–223 (2008)
15. Mahdian, M., Nazerzadeh, H., Saberi, A.: Allocating online advertisement space with unreliable estimates. In: EC. p. 294 (2007)
16. Mehta, A., Saberi, A., Vazirani, U.V., Vazirani, V.V.: Adwords and generalized on-line matching. In: FOCS. pp. 264–273 (2005)
17. Schrijver, A.: Combinatorial Optimization. Springer-Verlag (2003)
18. Shmoys, D.B., Tardos, É.: An approximation algorithm for the generalized assignment problem. Math. Program. 62, 461–474 (1993)
19. Shmoys, D., Swamy, C.: An approximation scheme for stochastic linear programming and its application to stochastic integer programs. J. ACM 53(6), 1012 (2006)
20. Swamy, C., Shmoys, D.: Approximation algorithms for 2-stage stochastic optimization problems. ACM SIGACT News 37(1), 46 (2006)