Prophet Matching Meets Probing with Commitment

Allan Borodin [*] Calum MacRury [†] Akash Rakheja [‡]

Abstract

We consider the online stochastic matching problem for bipartite graphs where edges adjacent to an online node must be probed to determine if they exist, based on known edge probabilities. Our algorithms respect commitment, in that if a probed edge exists, it must be used in the matching. We study this matching problem subject to a downward-closed constraint on each online node's allowable edge probes. Our setting generalizes the commonly studied patience (or time-out) constraint which limits the number of probes that can be made to an online node's adjacent edges. The generality of our setting leads to some modelling and computational efficiency issues that are not encountered in previous works. To resolve these issues, we introduce a new LP that we prove is a relaxation of an optimal offline probing algorithm (the adaptive benchmark) and which overcomes the limitations of previous LP relaxations. We establish new competitive bounds all of which generalize the classical non-probing setting when edges do not need to be probed (i.e., exist with certainty). Specifically, we establish the following competitive ratio results for a general formulation of edge probing constraints, arbitrary edge weights, and arbitrary edge probabilities:

1. A tight $\frac{1}{2}$ ratio when the stochastic graph is generated from a known stochastic type graph where the t^{th} online node is drawn independently from a known distribution $\mathcal{D}_{\pi(t)}$ and π is chosen adversarially. We refer to this setting as the known i.d. stochastic matching problem with adversarial arrivals.
2. A $1 - 1/e$ ratio when the stochastic graph is generated from a known stochastic type graph where the t^{th} online node is drawn independently from a known distribution $\mathcal{D}_{\pi(t)}$ and π is a random permutation. We refer to this setting as the known i.d. stochastic matching problem with random order arrivals.

Our results improve upon the previous best competitive ratio of 0.46 in the known i.i.d. setting against the standard adaptive benchmark. Moreover, we are the first to study the prophet secretary matching problem in the context of probing, where we match the best known classical result.

1 Introduction

Stochastic probing problems are part of the larger area of decision making under uncertainty and more specifically, stochastic optimization. Unlike more standard forms of stochastic optimization, it is not just that there is some stochastic uncertainty in the set of inputs, stochastic probing problems involve inputs that cannot be determined without probing (at some cost and/or within some constraint). Applications of stochastic probing occur naturally in many settings, such as in matching problems where compatibility cannot be determined without some trial or investigation (for example, in online dating and kidney exchange applications). There is by now an extensive literature for stochastic matching problems.

Although we are only considering "one-sided online bipartite matching", stochastic matching[1] was first considered in the context of a general graph by Chen et al. [18]. In this problem, the algorithm is presented an adversarially generated *stochastic graph* $G = (V, E)$ as input, which has a probability p_e associated with each edge e and a patience (or time-out) parameter ℓ_v associated with each vertex v. An algorithm probes edges in E within the constraint that at most ℓ_v edges are probed incident to any particular vertex v. The

[*]Department of Computer Science, University of Toronto, Toronto, ON, Canada bor@cs.toronto.edu
[†]Department of Computer Science, University of Toronto, Toronto, ON, Canada cmacrury@cs.toronto.edu
[‡]Department of Computer Science, University of Toronto, Toronto, ON, Canada rakhejaakash@gmail.com

[1]Unfortunately, the term "stochastic matching" is also used to refer to more standard optimization where the input (i.e., edges or vertices) are drawn from some known or unknown distributions but no probing is involved.

patience parameter can be viewed as a simple budgetary constraint, where each probe has unit cost and the patience constraint is the budget. When an edge e is probed, it is guaranteed to exist with probability exactly p_e. If an edge (u, v) is found to exist, then the algorithm must *commit* to the edge – that is, it must be added to the current matching (if u and v are currently unmatched). The goal is to maximize the expected size of a matching constructed in this way.

In addition to generalizing the results of Chen et al. to edge weights, Bansal et al. [6] introduced a known i.i.d. bipartite version of the problem where nodes on one side of the partition arrive online and edges adjacent to that node are then probed. In their model, each online vertex is drawn independently and identically from a known distribution. That is, the possible "type" of each online node (i.e., its adjacent edge probabilities, edge weights and patience) is known in advance and the input sequence is then determined i.i.d. from this known distribution, where the instantiation of each node is presented to the algorithm upon arrival. In the Bansal et al. model, each offline node has unbounded patience. The match for an online node must be made before the next online arrival. As in the Chen et al. model, if an edge is probed and confirmed to exist, then it must be included in the current matching (if possible). This problem is referred to as the *online stochastic matching problem* and also referred to as the *stochastic rewards problem*.

We study the online bipartite stochastic matching problem in the more general known i.d. setting. Specifically, each online vertex is drawn from a (potentially) distinct distribution, and these distributions are independent. When online vertices arrive adversarially, we generalize the prophet inequality matching problem of Alaei et al. [4]. When online vertices arrive in random order, we generalize the prophet secretary matching problem of Ehsani et al. [23]. We note that prophet inequalities give rise to (and in some sense are equivalent to) order oblivious posted price mechanisms, as first studied in Hajiaghayi et al. [30] and further developed for multi-parameter settings in Chawla et al. [17] and recently in Correa et al. [19]. Furthermore, we generalize the patience constraint to apply to any downward-closed constraint including budget (equivalently, knapsack) constraints.

Amongst other applications, the online bipartite stochastic matching problem notably models online advertising where the probability of an edge can correspond to the probability of a purchase in online stores or to pay-per-click revenue in online searching. One may also consider a real estate agent (or owner) of several properties working with individual clients each day who first look at properties online. Each buyer has a value for each property but will not purchase until they see the house in person or until the house has been inspected at which time she will commit to buy. An agent can only probabilistically estimate the likelihood of the buyer being satisfied. But the buyer or agent may have limited patience (or time or budget) to visit properties and will usually want to do so in a reasonably efficient way. In these two examples, the objective is clearly (respectively) to maximize the revenue of actual sales or clicks and (respectively) to maximize the value of the properties sold.

2 Preliminaries and Our Results

We define a **(bipartite) stochastic graph** to be a (simple) bipartite graph $G = (U, V, E)$ with edge weights $(w_e)_{e \in E}$ and edge probabilities $(p_e)_{e \in E}$. We refer to U as the **offline** vertices of G and V as its **online** nodes. Each $e \in E$ of $G = (U, V, E)$ is **active** independently with probability p_e, and $\text{st}(e) \sim \text{Ber}(p_e)$ corresponds to the **state** of the edge. Given an arbitrary set S, let $S^{(*)}$ denote the set of all tuples (strings) formed from S, whose entries (characters) are all distinct. Note that we use tuple/string notation and terminology interchangeably. Each $v \in V$ has its own **(online) probing constraint** $\mathcal{C}_v \subseteq \partial(v)^{(*)}$, where $\partial(v)$ is the set of edges adjacent to v. We make the minimal assumption that $\mathcal{C}_v$ is **downward-closed**; that is, if $\boldsymbol{e} \in \mathcal{C}_v$, then any substring or permutation of $\boldsymbol{e}$ is also in $\mathcal{C}_v$. This includes matroid constraints, as well when each $v \in V$ has a **budget** $B_v \geq 0$, and **(edge) probing costs** $(c_e)_{e \in \partial(v)}$, such that $\boldsymbol{e} = (e_1, \ldots, e_k) \in \mathcal{C}_v$ provided $\sum_{i=1}^{k} c_i \leq B_v$. Observe that if each v has uniform probing costs, then this corresponds to the previously discussed case of **one-sided patience values** $(\ell_v)_{v \in V}$. We shall assume w.l.o.g. that each stochastic graph $G = (U, V, E)$ we work with satisfies $E = U \times V$, as we may always exclude an edge e from being active by setting $p_e = 0$.

A solution to the **online stochastic matching problem** with **known i.d. arrivals** is an **online probing algorithm**. An online probing algorithm is first presented a stochastic graph $H_{\text{typ}} = (U, B, F)$ with edge weights $(w_f)_{f \in F}$, edge probabilities $(p_f)_{f \in F}$, and online probing constraints $(\mathcal{C}_b)_{b \in B}$. We refer to

H_{typ} as a **(stochastic) type graph**, and the vertices of B as the online **type nodes** of H_{typ}. The online probing algorithm does *not* execute on H_{typ}. Instead, the adversary fixes a integer $n \geq 1$, and a sequence of distributions $(\mathcal{D}_i)_{i=1}^n$ supported on B, both of which are known to the online probing algorithm. In the **adversarial arrival model**, a permutation π is generated by an **oblivious adversary**, in which case π is a function of H_{typ} and $(\mathcal{D}_i)_{i=1}^n$, and so it *cannot* depend on the instantiation of the online vertices, nor the decisions of the online probing algorithm. In the **random order arrival model**, π is generated uniformly at random (u.a.r.), independently of all other randomization. In either setting, π is unknown to the algorithm. For each $t = 1, \ldots, n$, vertex $v_{\pi(t)}$ is drawn from $\mathcal{D}_{\pi(t)}$ and presented to the algorithm, along with its edge weights, probabilities, and online probing constraint. Note that the algorithm is presented the value $\pi(t)$, and thus learns which distribution $v_{\pi(t)}$ was drawn from. However, the edge states $(\text{st}(e))_{e \in \partial(v_{\pi(t)})}$ initially remain hidden to the algorithm. Instead, using all past available information regarding $v_{\pi(1)}, \ldots, v_{\pi(t-1)}$, the algorithm must **probe** the edges of $\partial(v_{\pi(t)})$ to reveal their states, while adhering to the constraint $\mathcal{C}_{v_{\pi(t)}}$. That is, it may probe a tuple of edges $\boldsymbol{e} \in \partial(v_{\pi(t)})^{(*)}$ which satisfies $\boldsymbol{e} \in \mathcal{C}_{v_{\pi(t)}}$. The algorithm operates in the **probe-commit model**, in which there is a **commitment** requirement upon probing an edge. Specifically, if an edge $e = (u, v)$ is probed and turns out to be active, then the online probing algorithm must make an irrevocable decision as to whether or not to include e in its matching, prior to probing any subsequent edges. The goal of the algorithm is to build a matching of largest possible expected weight. This definition of commitment is the one considered by Gupta et al. [29], and in fact has equivalent power as the previously described Chen et al. [18] model. An online probing algorithm may simply pass on probing an edge if it doesn't intend to add the edge to its matching.

We refer to the randomly generated stochastic graph $G = (U, V, E)$ on which the online probing algorithm executes as the **instantiated stochastic graph**, and denote $G \sim (H_{\text{typ}}, (\mathcal{D}_i)_{i=1}^n)$ to indicate G is drawn from $(H_{\text{typ}}, (\mathcal{D}_i)_{i=1}^n)$. Note that technically V and E are each multi-sets, as multiple copies of a type node $b \in B$ may appear. In general, it is easy to see we cannot hope to obtain a non-trivial competitive ratio against the expected weight of an optimal matching of the instantiated stochastic graph[2]. The standard approach in the literature is to instead consider the **offline stochastic matching problem** and benchmark against an *optimal offline probing algorithm* [6, 2, 15, 16]. An **offline probing algorithm** knows $G = (U, V, E)$, but initially the edge states $(\text{st}(e))_{e \in E}$ are hidden. It can adaptively probe the edges of E in any order, but must satisfy the probing constraints $(\mathcal{C}_v)_{v \in V}$ at each step of its execution[3] while operating in the same probe-commit model as an online probing algorithm. The goal of an offline probing algorithm is to construct a matching with optimal weight in expectation. We define the **adaptive (committal) benchmark** as an optimal offline probing algorithm, and denote $\text{OPT}(G)$ as the expected weight of its matching when executing on G. An alternative weaker benchmark used by Brubach et al. [11, 12] is the **online adaptive benchmark**. This is defined as an optimal offline probing algorithm which executes on G and whose edge probes respect some adaptive vertex ordering on V. Equivalently, the edge probes involving each $v \in V$ occur contiguously: if $e' = (u, v') \in E$ is probed after $e = (u, v)$ for $v' \neq v$, then no edge of $\partial(v)$ is probed following e'. We benchmark against $\mathbb{E}[\text{OPT}(G)]$, where the expectation is over the randomness in $G \sim (H_{\text{typ}}, (\mathcal{D}_i)_{i=1}^n)$. For convenience, we denote $\text{OPT}(H_{\text{typ}}, (\mathcal{D}_i)_{i=1}^n) := \mathbb{E}[\text{OPT}(G)]$, and refer to $(H_{\text{typ}}, (\mathcal{D}_i)_{i=1}^n)$ as a **known i.d. input**.

Theorem 2.1. *Suppose $(H_{typ}, (\mathcal{D}_i)_{i=1}^n)$ is a known i.d. input, to which Algorithm 9 is given full access. If $\mathcal{M}(\pi)$ is the matching returned by the algorithm when presented the online vertices of $G \sim (H_{typ}, (\mathcal{D}_i)_{i=1}^n)$ in an adversarial order $\pi : [n] \to [n]$, then $\mathbb{E}[w(\mathcal{M}(\pi))] \geq \frac{1}{2} OPT(H_{typ}, (\mathcal{D}_i)_{i=1}^n)$.*

Remark 2.2. This is a tight bound since the problem generalizes the classic single item prophet inequality for which $\frac{1}{2}$ is an optimal competitive ratio. Recently, Brubach et al. [11, 12] independently[4] proved the same competitive ratio against the online adaptive benchmark when G has *unknown* patience values. Our results are incomparable, as their results apply to an unknown patience framework, whereas our results apply to downward-closed online probing constraints, and hold against a stronger benchmark.

[2]Consider a single online with a unit patience value, and $k \geq 1$ offline (unweighted) vertices where each edge e has probability $\frac{1}{k}$ of being present. The expectation of an online probing algorithm will be at most $\frac{1}{k}$ while the expected size of an optimal matching will be $1 - (1 - \frac{1}{k})^k \to 1 - \frac{1}{e}$ as $k \to \infty$.

[3]Edges $\boldsymbol{e} \in E^{(*)}$ may be probed in order, provided $\boldsymbol{e}^v \in \mathcal{C}_v$ for each $v \in V$, where $\boldsymbol{e}^v$ is the substring of $\boldsymbol{e}$ restricted to edges of $\partial(v)$.

[4]Our work has been motivated by early results of Brubach et al. Since then, we have been improving and extending results independent of improvements and extensions by Brubach et al.

We say that an online probing algorithm is **non-adaptive**, provided for each $t \in [n]$, the probes of $\partial(v_{\pi(t)})$ are a (randomized) function of H_{typ}, and the type of arrival $v_{\pi(t)}$. In particular, when probing edges adjacent to the online node $v_{\pi(t)}$, the algorithm does not make use of the previously probed edge states of the online nodes $v_{\pi(1)}, \ldots, v_{\pi(t-1)}$, their types, nor the matching decisions made thus far. The algorithm will therefore possibly waste some probes to edges $(u, v_{\pi(t)})$ of which $u \in U$ is unavailable, but it will not violate the matching constraint. Our definition generalizes the notion of non-adaptivity introduced by Manshadi et al. [34] in the classic matching problem with known i.i.d. arrivals.

Theorem 2.3. *Suppose $(H_{typ}, (\mathcal{D}_i)_{i=1}^n)$ is a known i.d. input, to which Algorithm 10 is given full access. If $\mathcal{M}$ is the matching returned by the algorithm when presented the online vertices of $G \sim (H_{typ}, (\mathcal{D}_i)_{i=1}^n)$ in random order, then $\mathbb{E}[w(\mathcal{M})] \geq \left(1 - \frac{1}{e}\right) OPT(H_{typ}, (\mathcal{D}_i)_{i=1}^n)$. Moreover, Algorithm 10 is non-adaptive.*

Remark 2.4. The special case of identical distributions with one-sided patience values has been studied in multiple works [6, 2, 15, 16], beginning with the 0.12 competitive ratio of Bansal et al. [6], and for which the previously best known competitive ratio of 0.46 is due to Brubach et al. [16]. Note that all these previous competitive ratios are proven against the adaptive benchmark. Theorem 2.3 improves on this result, and also is the first to apply to non-identical distributions, as well as to more general probing constraints. We also proved this in an earlier arXiv version of this paper [8]. Recently, Brubach et al. [11] posted an updated version of [14] which independently achieves the same competitive ratio for the case of unknown patience values in the known i.i.d. setting, however again their result is proven against the online adaptive benchmark. Note that $1 - 1/e$ is also the best known competitive ratio in the classic matching problem for known i.d. random order arrivals, due to Ehsani et al. [23].

In order to discuss the efficiency of our algorithms, we work in the **membership oracle model**. An online probing algorithm may make a **membership query** to any string $\boldsymbol{e} \in \partial(b)^{(*)}$ for $b \in B$, thus determining in a single operation whether or not $\boldsymbol{e} \in \partial(b)^{(*)}$ is in $\mathcal{C}_b$.

Theorem 2.5. *Suppose that $(H_{typ}, (\mathcal{D}_i)_{i=1}^n)$ is a known i.d. input, where $H_{typ} = (U, B, F)$ has downward-closed probing constraints $(\mathcal{C}_b)_{b \in B}$. If $|H_{typ}|$ is the size of H_{typ} (excluding $(\mathcal{C}_b)_{b \in B}$), and $|\mathcal{D}_i|$ is the amount of space needed to encode the distribution $\mathcal{D}_i$, then the following claim holds:*

- *In the membership oracle model, Algorithms 9 and 10 execute in time poly$(|H_{typ}|, (|\mathcal{D}_i|)_{i=1}^n)$, provided for each $b \in B$, $\mathcal{C}_b$ is downward-closed.*

Remark 2.6. $|H_{\text{typ}}|$ may be exponentially large in the size of $G \sim (H_{\text{typ}}, (\mathcal{D}_i)_{i=1}^n)$, however for each $\varepsilon > 0$, our results can be made to run in time poly$(|G|, \log(1/\varepsilon))$ using Monte Carlo simulation (at a loss of $(1 - \varepsilon)$ in performance), assuming we have oracle access to samples drawn from $(\mathcal{D}_i)_{i=1}^n$.

An important special case of the online stochastic matching problem is the case of a **known stochastic graph**. In this setting, the input $H_{\text{typ}} = (V, B, F)$ satisfies $n = |B|$, and the distributions $(\mathcal{D}_i)_{i=1}^n$ are all point-mass on *distinct* vertices of B. Thus, the online vertices of G are not randomly drawn, and G is instead equal to H_{typ}. The online probing algorithm thus knows the stochastic graph G in advance, but remains unaware of the edge states $(\text{st}(e))_{e \in E}$, and so it still must sequentially probe the edges to reveal their states. Again, it must operate in the probe-commit model, and respect the probing constraints $(\mathcal{C}_v)_{v \in V}$ as well as the arrival order π on V. Since all the vertices are known a priori, non-adaptivity reduces to requiring that the probes of the algorithm are a function of G. We define non-adaptivity for *offline* probing algorithms in the same way.

Corollary 2.7. *Suppose $G = (U, V, E)$ is a known stochastic graph to which Algorithm 10 is given full access. If $\mathcal{M}$ is the matching returned by the algorithm when presented the online vertices of G in random order, then $\mathbb{E}[w(\mathcal{M})] \geq \left(1 - \frac{1}{e}\right) OPT(G)$. Moreover, Algorithm 10 is non-adaptive.*

Remark 2.8. Gamlath et al. [26] consider an online probing algorithm in the **unconstrained** or **unbounded patience setting** – i.e., $\mathcal{C}_v = \partial(v)^{(*)}$ for $v \in V$ – when G is known. Both our algorithm and theirs attain a performance guarantee of $1 - 1/e$ against different non-standard LPs. In the special case of unbounded patience, our LPs take on the same value, as we prove in Proposition A.1 of Appendix A. Thus, Theorem 2.3 can be viewed as a generalization of their work to downward-closed online probing constraints and known i.d. random order arrivals.

We complement Corollary 2.7 with a hardness result which applies to *all* non-adaptive probing algorithms (even probing algorithms which execute offline, and thus do not respect the arrival order π of V):

Theorem 2.9. *In the known stochastic graph setting, no non-adaptive offline probing algorithm can attain an approximation ratio against the adaptive benchmark which is greater than* $1 - 1/e$.

2.1 An Overview of Our Techniques

In order to describe the majority of our technical contributions, it suffices to focus on the known stochastic graph setting. Let us suppose we are presented a stochastic graph $G = (U, V, E)$. For the case of patience values $(\ell_v)_{v \in V}$, a natural solution is to solve an LP introduced by Bansal et al. [6] (see LP-std in Appendix A) to obtain fractional values for the edges of G, say $(x_e)_{e \in E}$, such that x_e upper bounds the probability e is probed by the adaptive benchmark. Clearly, $\sum_{e \in \partial(v)} x_e \leq \ell_v$ is a constraint for each $v \in V$, and so by applying a dependent rounding algorithm (such as the GKSP algorithm of Gandhi et al. [27]), one can round the values $(x_e)_{e \in \partial(v)}$ to determine ℓ_v edges of $\partial(v)$ to probe. By probing these edges in a carefully chosen order, and matching v to the first edge revealed to be active, one can guarantee that each $e \in \partial(v)$ is matched with probability reasonably close to $p_e x_e$. This is the high-level approach used in many previous stochastic matching algorithms (for example [6, 2, 7, 16, 13]). However, even for a single online node, LP-std overestimates the value of the adaptive benchmark, and so any algorithm designed in this way will match certain edges with probability strictly less than $p_e x_e$. This is problematic, for the value of the match made to v is ultimately compared to $\sum_{e \in \partial(v)} p_e w_e x_e$, the contribution of the variables $(x_e)_{e \in \partial(v)}$ to the LP solution. In fact, Brubach et al. [14] showed that the ratio between $\text{OPT}(G)$ and an optimum solution to LP-std can be as small as 0.544, so the $1 - 1/e$ competitive ratio of Theorem 2.3 cannot be achieved via a comparison to LP-std, even for the special case of patience values.

Our approach is to introduce a new configuration LP (LP-config) with exponentially many variables which accounts for the many probing strategies available to an arriving vertex v with probing constraint $\mathcal{C}_v$. For each $\boldsymbol{e} = (e_1, \ldots, e_{|\boldsymbol{e}|}) \in E^{(*)}$, define $g(\boldsymbol{e}) := \prod_{i=1}^{|\boldsymbol{e}|}(1 - p_{e_i})$, where $g(\boldsymbol{e})$ corresponds to the probability that all the edges of $\boldsymbol{e}$ are inactive, and $g(\lambda) := 1$ for the empty string/character λ. We also define $\boldsymbol{e}_{<e_i} := (e_1, \ldots, e_{i-1})$ for each $2 \leq i \leq |\boldsymbol{e}|$, which we denote by $\boldsymbol{e}_{<i}$ when clear, where $\boldsymbol{e}_{<1} := \lambda$ by convention. Observe then that $\text{val}(\boldsymbol{e}) := \sum_{i=1}^{|\boldsymbol{e}|} p_{e_i} w_{e_i} g(\boldsymbol{e}_{<i})$ corresponds to the expected weight of the first active edge revealed if $\boldsymbol{e}$ is probed in order of its indices. For each $v \in V$, we introduce a decision variable $x_v(\boldsymbol{e})$:

$$\text{maximize} \quad \sum_{v \in V} \sum_{\boldsymbol{e} \in \mathcal{C}_v} \text{val}(\boldsymbol{e}) \cdot x_v(\boldsymbol{e}) \qquad \text{(LP-config)}$$

$$\text{subject to} \quad \sum_{v \in V} \sum_{\substack{\boldsymbol{e} \in \mathcal{C}_v: \\ (u,v) \in \boldsymbol{e}}} p_{u,v} \cdot g(\boldsymbol{e}_{<(u,v)}) \cdot x_v(\boldsymbol{e}) \leq 1 \qquad \forall u \in U \qquad (2.1)$$

$$\sum_{\boldsymbol{e} \in \mathcal{C}_v} x_v(\boldsymbol{e}) = 1 \qquad \forall v \in V, \qquad (2.2)$$

$$x_v(\boldsymbol{e}) \geq 0 \qquad \forall v \in V, \boldsymbol{e} \in \mathcal{C}_v \qquad (2.3)$$

When each $\mathcal{C}_v$ is downward-closed, LP-config can be solved efficiently by using a deterministic separation oracle for LP-new-dual, the dual of LP-config, in conjunction with the ellipsoid algorithm [36, 28], as we prove in Theorem 5.1 of Section 5. Crucially, LP-config is also a **relaxation** of the adaptive benchmark.

Theorem 2.10. *If* $G = (U, V, E)$ *has downward-closed probing constraints, then* $OPT(G) \leq LPOPT(G)$.

Remark 2.11. For the case of patience values, LP-config was also recently independently introduced by Brubach et al. [11, 12] to design probing algorithms for known i.i.d. arrivals and known i.d. adversarial arrivals. Their competitive ratios are proven against an optimal solution to LP-config, which they argue upper bounds the *online* adaptive benchmark. Theorem 2.10 thus implies that their results in fact hold against the stronger adaptive benchmark.

In order to prove Theorem 2.10, the natural approach is to view $x_v(\boldsymbol{e})$ as the probability that the adaptive benchmark probes the edges of $\boldsymbol{e}$ in order, where $v \in V$ and $\boldsymbol{e} \in \mathcal{C}_v$. Let us suppose that hypothetically we could make the following restrictive assumptions regarding the adaptive benchmark:

(P_1) If $e = (u, v)$ is probed and $\text{st}(e) = 1$, then e is included in the matching, provided v is currently unmatched.

(P_2) For each $v \in V$, the edge probes involving $\partial(v)$ are made independently of the edge states $(\text{st}(e))_{e \in \partial(v)}$.

Observe then that (P_1) and (P_2) would imply that the expected weight of the edge assigned to v is $\sum_{\boldsymbol{e} \in \mathcal{C}_v} \text{val}(\boldsymbol{e}) \cdot x_v(\boldsymbol{e})$. Moreover, the left-hand side of (2.1) would correspond to the probability $u \in U$ is matched, so $(x_v(\boldsymbol{e}))_{v \in V, \boldsymbol{e} \in \mathcal{C}_v}$ would be a feasible solution to LP-config, and so we could upper bound $\text{OPT}(G)$ by $\text{LPOPT}(G)$. Now, if we were working with the *online* adaptive benchmark, then it is clear that we could assume (P_1) and (P_2) simultaneously[5] w.l.o.g. On the other hand, if a probing algorithm does *not* respect an adaptive vertex ordering on V, then the probes involving $v \in V$ will in general depend on $(\text{st}(e))_{e \in \partial(v)}$. For instance, if $e \in \partial(v)$ is probed and inactive, then perhaps the adaptive benchmark probes $e' = (u, v') \in \partial(v')$ for some $v' \neq v$. If e' is active and thus added to the matching by (P_1), then the adaptive benchmark can never subsequently probe (u, v) without violating (P_1), as u is now unavailable to be matched to v. Thus, the natural interpretation of the decision variables of LP-config does not seem to easily lend itself to a proof of Theorem 2.10.

Our solution is to consider a **combinatorial relaxation** of the offline stochastic matching problem, which we define to be a new stochastic probing problem on G whose optimal value $\text{OPT}_{\text{rel}}(G)$ satisfies $\text{OPT}(G) \leq \text{OPT}_{\text{rel}}(G)$. We refer to this problem as the **relaxed stochastic matching problem**, a solution to which is a **relaxed probing algorithm**. Roughly speaking, a relaxed probing algorithm operates in the same framework as an offline probing algorithm, yet it returns a one-sided matching of the online vertices which matches each offline node at most once *in expectation*. We provide a precise definition in Section 3. Crucially, there exists an *optimal* relaxed probing algorithm which satisfies (P_1) and (P_2) simultaneously, which by the above discussion allows us to conclude that $\text{OPT}_{\text{rel}}(G) \leq \text{LPOPT}(G)$. Since $\text{OPT}(G) \leq \text{OPT}_{\text{rel}}(G)$ by construction, this implies Theorem 2.10. Proving the existence of an optimal relaxed probing algorithm with these properties is the most technically challenging part of the paper, and is the content of Lemma 3.1 of Section 3.

After proving that LP-config is a relaxation of the adaptive benchmark, we use it to design online probing algorithms. Suppose that we are presented a feasible solution, say $(x_v(\boldsymbol{e}))_{v \in V, \boldsymbol{e} \in \mathcal{C}_v}$, to LP-config for G. For each $e \in E$, define

$$\widetilde{x}_e := \sum_{\substack{\boldsymbol{e}' \in \mathcal{C}_v: \\ e \in \boldsymbol{e}'}} g(\boldsymbol{e}'_{<e}) \cdot x_v(\boldsymbol{e}'). \tag{2.4}$$

In order to simplify our notation in the later sections, we refer to the values $(\widetilde{x}_e)_{e \in E}$ as the **(induced) edge variables** of the solution $(x_v(\boldsymbol{e}))_{v \in V, \boldsymbol{e} \in \mathcal{C}_v}$. If we now fix $s \in V$, then we can easily leverage constraint (2.2) to design a simple *fixed vertex* probing algorithm which matches each edge of $e \in \partial(s)$ with probability exactly equal to $p_e \widetilde{x}_e$. Specifically, draw $\boldsymbol{e}' \in \mathcal{C}_s$ with probability $x_s(\boldsymbol{e}')$. If $\boldsymbol{e}' = \lambda$, then return the empty set. Otherwise, set $\boldsymbol{e}' = (e'_1, \ldots, e'_k)$ for $k := |\boldsymbol{e}'| \geq 1$, and probe the edges of $\boldsymbol{e}'$ in order. Return the first edge which is revealed to be active, if such an edge exists. Otherwise, return the empty set. We refer to this algorithm as VertexProbe, and denote its output on the input $(s, \partial(s), (x_s(\boldsymbol{e}))_{\boldsymbol{e} \in \mathcal{C}_s})$ by $\text{VertexProbe}(s, \partial(s), (x_s(\boldsymbol{e}))_{\boldsymbol{e} \in \mathcal{C}_s})$. Observe the following claim, which follows immediately from the definition of the edge variables, $(\widetilde{x}_e)_{e \in E}$:

Lemma 2.12. *Let $G = (U, V, E)$ be a stochastic graph with LP-config solution $(x_v(\boldsymbol{e}))_{v \in V, \partial(v)}$, and whose induced edge variables we denote by $(\widetilde{x}_e)_{e \in E}$. If* VertexProbe *is passed $(s, \partial(s), (x_s(\boldsymbol{e}))_{\boldsymbol{e} \in \mathcal{C}_s})$, then each $e \in \partial(s)$ is returned by the algorithm with probability $p_e \widetilde{x}_e$.*

Definition 1. We say that VertexProbe **commits** to the edge $e = (u, s) \in \partial(s)$, or equivalently the vertex $u \in N(s)$, provided the algorithm outputs e when executing on the fixed node $s \in V$. When it is clear that VertexProbe is being executed on s, we say that s commits to e (equivalently the vertex u).

Consider now the simple online probing algorithm, where π is generated either u.a.r. or adversarially.

[5]It is clear that we may assume the adaptive benchmark satisfies (P_1) w.l.o.g., but not (P_2).

Algorithm 1 Known Stochastic Graph

```
Input: a stochastic graph $G = (U, V, E)$.
Output: a matching $\mathcal{M}$ of active edges of $G$.
1: $\mathcal{M} \leftarrow \emptyset$.
2: Compute an optimal solution of LP-config for $G$, say $(x_v(\boldsymbol{e}))_{v \in V, \boldsymbol{e} \in \mathcal{C}_v}$
3: for $s \in V$ in order based on $\pi$ do
4:     Set $e \leftarrow$ VertexProbe$(s, \partial(s), (x_s(\boldsymbol{e}))_{\boldsymbol{e} \in \mathcal{C}_s})$.
5:     if $e = (u, s)$ for some $u \in U$, and $u$ is unmatched then        ▷ this line ensures $e \neq \emptyset$
6:         Add $e$ to $\mathcal{M}$.
7: return $\mathcal{M}$.
```

Remark 2.13. Technically, line (6) should occur within the VertexProbe subroutine to adhere to the probe-commit model, however we express our algorithms in this way for simplicity.

We observe the following claim, which is easily proven so we omit the argument:

Proposition 2.14. *In the adversarial arrival model, Algorithm 1 does not attain a constant competitive ratio. In the random order arrival model, Algorithm 1 attains a competitive ratio of* $1/2$ *and the analysis is asymptotically tight.*

Since the analysis of Algorithm 1 cannot be improved in either arrival model, we must modify the algorithm to prove Theorems 2.1 and 2.3, even in the known stochastic graph setting. Our modification involves concurrently applying an appropriate rank one matroid **contention resolution scheme** to each offline vertex of G, a concept formalized much more generally in the seminal paper by Chekuri, Vondrak, and Zenklusen [38]. Fix $u \in U$, and observe that constraint (2.1) ensures that $\sum_{e \in \partial(u)} p_e \widetilde{x}_e \leq 1$. Moreover, if we set $z_e := p_e \widetilde{x}_e$, then observe that as VertexProbe executes on v, each edge $e = (u, v) \in \partial(u)$ is committed to u independently with probability z_e. On the other hand, there may be many edges which commit to u so we must resolve which one to take. In Algorithm 1, u is matched greedily to the first online vertex which commits to it, regardless of how π is generated. We apply **online** and **random order** contention resolution schemes to ensure that e is matched to u with probability $1/2 \cdot z_e$ when π is generated by an adversary, and $(1 - 1/e) \cdot z_e$ when π is generated u.a.r. This allows us to conclude the desired competitive ratios, as $\sum_{e \in E} w_e p_e \widetilde{x}_e$ upper bounds $\text{OPT}(G)$ by Theorem 2.10. We review the relevant contention resolution schemes in Section 4, and also extend the argument to the case when G is unknown and instead drawn from the known i.d. input $(H_{\text{typ}}, (\mathcal{D}_i)_{i=1}^n)$, thus proving Theorems 2.1 and 2.3.

3 Relaxing the Adaptive Benchmark via LP-config

Given a stochastic graph $G = (U, V, E)$, we define the **relaxed stochastic matching problem**. A solution to this problem is a **relaxed probing algorithm** $\mathcal{A}$, which operates in the previously described framework of an (offline) probing algorithm. That is, $\mathcal{A}$ is firstly given access to a stochastic graph $G = (U, V, E)$. Initially, the edge states $(\text{st}(e))_{e \in E}$ are unknown to $\mathcal{A}$, and $\mathcal{A}$ must adaptivity probe these edges to reveal their states, while respecting the downward-closed probing constraints $(\mathcal{C}_v)_{v \in V}$. As in the offline problem, $\mathcal{A}$ returns a subset $\mathcal{A}(G)$ of its active probes, and its goal is to maximize $\mathbb{E}[w(\mathcal{A}(G))]$, where $w(\mathcal{A}(G)) := \sum_{e \in \mathcal{A}(G)} w_e$. However, unlike before where the output of the probing algorithm was required to be a matching of G, we relax the required properties of $\mathcal{A}(G)$:

1. Each $v \in V$ appears in at most one edge of $\mathcal{A}(G)$.
2. If N_u counts the number of edges of $\partial(u)$ which are included in $\mathcal{A}(G)$, then $\mathbb{E}[N_u] \leq 1$ for each $u \in U$.

We refer to $\mathcal{A}(G)$ as a **one-sided matching** of the online nodes. In constructing $\mathcal{A}(G)$, $\mathcal{A}$ must operate in the previously described probe-commit model. We define the **relaxed benchmark** as an optimal relaxed probing algorithm, and denote its evaluation on G by $\text{OPT}_{\text{rel}}(G)$. Observe that since any offline probing algorithm is a relaxed probing algorithm, we have that

$$\text{OPT}(G) \leq \text{OPT}_{\text{rel}}(G). \tag{3.1}$$

We say that $\mathcal{A}$ is **non-adaptive**, provided the probes are a (randomized) function of G. Equivalently, $\mathcal{A}$ is non-adaptive if the probes of $\mathcal{A}$ are statistically independent from $(\mathrm{st}(e))_{e \in E}$. Unlike for the offline stochastic matching problem, there exists a relaxed probing algorithm which is both optimal *and* non-adaptive:

Lemma 3.1. *For any stochastic graph $G = (U, V, E)$ with downward-closed probing constraints $(\mathcal{C}_v)_{v \in V}$, there exists an optimum relaxed probing algorithm $\mathcal{B}$ which satisfies the following properties:*

(Q_1) If $e = (u, v)$ is probed and $st(e) = 1$, then e is included in $\mathcal{B}(G)$, provided v is currently unmatched.

(Q_2) $\mathcal{B}$ is non-adaptive on G.

Remark 3.2. Note that (Q_2) implies the hypothetical property (P_2), yet is much stronger.

Let us assume Lemma 3.1 holds for now. Observe that by considering $\mathcal{B}$ of Lemma 3.1, and defining $x_v(\boldsymbol{e})$ as the probability that $\mathcal{B}$ probes the edges of $\boldsymbol{e}$ in order for $v \in V$ and $\boldsymbol{e} \in \mathcal{C}_v$, properties (Q_1) and (Q_2) ensure that $(x_v(\boldsymbol{e}))_{v \in V, \boldsymbol{e} \in \mathcal{C}_v}$ is a feasible solution to LP-config such that

$$\mathbb{E}[w(\mathcal{B}(G))] = \sum_{v \in V} \sum_{\boldsymbol{e} \in \mathcal{C}_v} \mathrm{val}(\boldsymbol{e}) \cdot x_v(\boldsymbol{e}).$$

Thus, the optimality of $\mathcal{B}$ implies that $\mathrm{OPT}_{\mathrm{rel}}(G) \leq \mathrm{LPOPT}(G)$, and so together with (3.1), Theorem 2.10 follows. In fact, LP-config is an exact LP formulation of the relaxed stochastic matching problem:

Theorem 3.3. *$OPT_{rel}(G) = LPOPT(G)$.*

Proof. Clearly, Theorem 2.10 accounts for one side of the inequality, so it suffices to show that $\mathrm{LPOPT}(G) \leq \mathrm{OPT}_{\mathrm{rel}}(G)$. Suppose we are presented a feasible solution $(x_v(\boldsymbol{e}))_{v \in V, \boldsymbol{e} \in \mathcal{C}_v}$ to LP-config. Consider then the following algorithm:

1. $\mathcal{M} \leftarrow \emptyset$.
2. For each $v \in V$, set $e \leftarrow \textsc{VertexProbe}(v, \partial(v), (x_v(\boldsymbol{e}))_{\boldsymbol{e} \in \mathcal{C}_v})$. If $e \neq \emptyset$, then add e to $\mathcal{M}$.
3. Return $\mathcal{M}$.

Using Lemma 2.12, it is clear that

$$\mathbb{E}[w(\mathcal{M})] = \sum_{v \in V} \sum_{\boldsymbol{e} \in \mathcal{C}_v} \mathrm{val}(\boldsymbol{e}) \cdot x_v(\boldsymbol{e}).$$

Moreover, each vertex $u \in U$ is matched by $\mathcal{M}$ at most once in expectation, as a consequence of constraint (2.1) of LP-config, and so the algorithm satisfies the required properties of a relaxed probing algorithm. The proof is therefore complete. ☐

3.1 Proving Lemma 3.1

Let us suppose that $G = (U, V, E)$ is a stochastic graph with downward-closed probing constraints $(\mathcal{C}_v)_{v \in V}$. In order to prove Lemma 3.1, we must show that there exists an optimal relaxed probing algorithm which is non-adaptive and satisfies (Q_1). Our high level approach is to consider an optimal relaxed probing algorithm $\mathcal{A}$ which satisfies (Q_1), and then to construct a new non-adaptive algorithm $\mathcal{B}$ by *stealing* the strategy of $\mathcal{A}$, without any loss in performance. More specifically, we construct $\mathcal{B}$ by writing down for each $v \in V$ and $\boldsymbol{e} \in \mathcal{C}_v$ the probability that $\mathcal{A}$ probes the edges of $\boldsymbol{e}$ in order. These probabilities necessarily satisfy certain inequalities which we make use of in designing $\mathcal{B}$. In order to do so, we need a technical randomized rounding procedure whose precise relevance will become clear in the proof of Lemma 3.1.

Suppose that $\boldsymbol{e} \in E^{(*)}$, and recall that λ is the empty string/character. For each $j \geq 0$, denote $\boldsymbol{e}_j$ as the j^{th} character of $\boldsymbol{e}_j$, where $\boldsymbol{e}_j := \lambda$ when $j = 0$ or $j > |\boldsymbol{e}|$. Let us now assume that $(y_v(\boldsymbol{e}))_{\boldsymbol{e} \in \mathcal{C}_v}$ is a collection of non-negative values which satisfy $y_v(\lambda) = 1$, and

$$\sum_{\substack{e \in \partial(v): \\ (\boldsymbol{e}', e) \in \mathcal{C}_v}} y_v(\boldsymbol{e}', e) \leq y_v(\boldsymbol{e}'), \tag{3.2}$$

for each $\boldsymbol{e}' \in \mathcal{C}_v$.

Proposition 3.4. *Given a collection of values* $(y_v(\boldsymbol{e}))_{\boldsymbol{e} \in \mathcal{C}_v}$ *which satisfy* $y_v(\lambda) = 1$ *and* (3.2)*, there exists a distribution* $\mathcal{D}^v$ *supported on* $\mathcal{C}_v$*, such that if* $\boldsymbol{Y} \sim \mathcal{D}^v$*, then for each* $\boldsymbol{e} \in \mathcal{C}_v$ *with* $k := |\boldsymbol{e}| \geq 1$*, it holds that*

$$\mathbb{P}[(\boldsymbol{Y}_1, \ldots, \boldsymbol{Y}_k) = (\boldsymbol{e}_1, \ldots, \boldsymbol{e}_k)] = y_v(\boldsymbol{e}), \tag{3.3}$$

where $\boldsymbol{Y}_1, \ldots, \boldsymbol{Y}_k$ *are the first* k *characters of* $\boldsymbol{Y}$*.*

Proof. First define $\mathcal{C}_v^{>} := \{\boldsymbol{e}' \in \mathcal{C}_v : y_v(\boldsymbol{e}') > 0\}$, which we observe is downward-closed since by assumption $\mathcal{C}_v$ is downward-closed and (3.2) holds. We prove the proposition for $\mathcal{C}_v^{>}$, which we then argue implies the proposition holds for $\mathcal{C}_v$. Observe now that for each $\boldsymbol{e}' \in \mathcal{C}_v^{>}$, we have that

$$\sum_{\substack{e \in \partial(v): \\ (\boldsymbol{e}', e) \in \mathcal{C}_v^{>}}} \frac{y_v(\boldsymbol{e}', e)}{y_v(\boldsymbol{e}')} \leq 1 \tag{3.4}$$

as a result of (3.2) (recall that $y_v(\lambda) := 1$). We thus define for each $\boldsymbol{e}' \in \mathcal{C}_v^{>}$,

$$z_v(\boldsymbol{e}') := 1 - \sum_{\substack{e \in \partial(v): \\ (\boldsymbol{e}', e) \in \mathcal{C}_v^{>}}} \frac{y_v(\boldsymbol{e}', e)}{y_v(\boldsymbol{e}')}, \tag{3.5}$$

which we observe has the property that $0 \leq z_v(\boldsymbol{e}') < 1$. The strict inequality follows from the definition of $\mathcal{C}_v^{>}$. This leads to the following randomized rounding algorithm, which we claim outputs a random string $\boldsymbol{Y}$ which satisfies the desired properties:

Algorithm 2 VertexRound

```
Input: a collection of values $(y_v(\boldsymbol{e}))_{\boldsymbol{e} \in \mathcal{C}_v^{>}}$ satisfying (3.2) and $y_v(\lambda) = 1$.
Output: a random string $\boldsymbol{Y} = (Y_1, \ldots, Y_{|\partial(v)|})$ supported on $\mathcal{C}_v^{>}$.
1: Set $\boldsymbol{e}' \leftarrow \lambda$.
2: Initialize $Y_i = \lambda$ for each $i = 1, \ldots, |\partial(v)|$.
3: for $i = 1, \ldots, |\partial(v)|$ do
4:     Exit the "for loop" with probability $z_v(\boldsymbol{e}')$.        ▷ pass with a certain probability – see (3.5)
5:     Draw $e \in \partial(v)$ satisfying $(\boldsymbol{e}', e) \in \mathcal{C}_v^{>}$ with probability $y_v(\boldsymbol{e}', e)/(y_v(\boldsymbol{e}')\,(1 - z_v(\boldsymbol{e}')))$.
6:     Set $Y_i = e$.
7:     $\boldsymbol{e}' \leftarrow (\boldsymbol{e}', e)$.
8: return $\boldsymbol{Y} = (Y_1, \ldots, Y_{|\partial(v)|})$.        ▷ concatenate the edges in order and return the resulting string
```

Clearly, the random string $\boldsymbol{Y}$ is supported on $\mathcal{C}_v^{>}$, thanks to line 5 of Algorithm 2. We now show that (3.3) holds. As such, let us first assume $k = 1$, and $e \in \partial(v)$ satisfies $(e) \in \mathcal{C}_v^{>}$. Observe that

$$\mathbb{P}[Y_1 = e] = (1 - z_v(\lambda)) \frac{y_v(e)}{1 - z_v(\lambda)} = y_v(e),$$

as the algorithm does not exit the "for loop" with probability $1 - z_v(\lambda)$, in which case it draws e with probability $y_v(e)/(1 - z_v(\lambda))$. In general, take $k \geq 2$, and assume that for each $\boldsymbol{e}' \in \mathcal{C}_v^{>}$ with $1 \leq |\boldsymbol{e}'| < k$, it holds that

$$\mathbb{P}[(Y_1, \ldots, Y_k) = \boldsymbol{e}'] = y_v(\boldsymbol{e}').$$

If we now fix $\boldsymbol{e} = (e_1, \ldots, e_k) \in \mathcal{C}_v^{>}$ with $|\boldsymbol{e}| = k$, observe that $\boldsymbol{e}_{<k} := (e_1, \ldots, e_{k-1}) \in \mathcal{C}_v^{>}$, as $\mathcal{C}_v^{>}$ is downward-closed. Moreover,

$$\begin{aligned} \mathbb{P}[(Y_1, \ldots, Y_k) = \boldsymbol{e}] &= \mathbb{P}[Y_k = e_k \,|\, (Y_1, \ldots, Y_{k-1}) = \boldsymbol{e}_{<k}] \cdot \mathbb{P}[(Y_1, \ldots, Y_{k-1}) = \boldsymbol{e}_{<k}] \\ &= \mathbb{P}[Y_k = e_k \,|\, (Y_1, \ldots, Y_{k-1}) = \boldsymbol{e}_{<k})] \cdot y_v(\boldsymbol{e}_{<k}), \end{aligned}$$

where the last line follows by the induction hypothesis since $\boldsymbol{e}_{<k} \in \mathcal{C}_v^{>}$ is of length $k - 1$. We know however that

$$\mathbb{P}[Y_k = \boldsymbol{e}_k \,|\, (Y_1, \ldots, Y_{k-1}) = \boldsymbol{e}_{<k}] = (1 - z_v(\boldsymbol{e}_{<k})) \frac{y_v(\boldsymbol{e}_{<k}, e_k)}{y_v(\boldsymbol{e}_{<k})\,(1 - z_v(\boldsymbol{e}_{<k}))} = \frac{y_v(\boldsymbol{e}_{<k}, e_k)}{y_v(\boldsymbol{e}_{<k})}.$$

This is because once we condition on the event $(Y_1, \ldots, Y_{k-1}) = \boldsymbol{e}_{<k}$, we know that the algorithm does not exit the "for loop" with probability $1 - z_v(\boldsymbol{e}_{<k})$, in which case it selects $e_k \in \partial(v)$ with probability $y_v(\boldsymbol{e}_{<k}, e_k)/(y_v(\boldsymbol{e}_{<k})\,(1 - z_v(\boldsymbol{e}_{<k})))$, since $(\boldsymbol{e}_{<k}, e_k) \in \mathcal{C}_v^{>}$ by assumption. As such, we have that

$$\mathbb{P}[(Y_1, \ldots, Y_k) = \boldsymbol{e}] = y_v(\boldsymbol{e}),$$

and so the proposition holds for $\mathcal{C}_v^{>}$. To complete the argument, observe that since $\boldsymbol{Y}$ is supported on $\mathcal{C}_v^{>}$, the substrings of $\boldsymbol{Y}$ are also supported on $\mathcal{C}_v^{>}$, as $\mathcal{C}_v^{>}$ is downward-closed. Thus, $\boldsymbol{Y}$ satisfies (3.3) for the non-empty strings of $\mathcal{C}_v \setminus \mathcal{C}_v^{>}$, in addition to the non-empty strings of $\mathcal{C}_v^{>}$. □

We are now ready to prove Lemma 3.1.

Proof of Lemma 3.1. Suppose that $\mathcal{A}$ is an optimal relaxed probing algorithm which returns the one-sided matching $\mathcal{M}$ after executing on the stochastic graph $G = (U, V, E)$. In a slight abuse of terminology, we say that e is matched by $\mathcal{A}$, provided e is included in $\mathcal{M}$. We shall also make the simplifying assumption that $p_e < 1$ for each $e \in E$, as the proof can be clearly adapted to handle the case when certain edges have $p_e = 1$ by restricting which strings of each $\mathcal{C}_v$ are considered.

Observe that since $\mathcal{A}$ is optimal, it is clear that we may assume the following properties hold w.l.o.g. for each $e \in E$:

1. e is probed only if e can be added to the currently constructed one-sided matching.

2. If e is probed and $\mathrm{st}(e) = 1$, then e is included in $\mathcal{M}$.

Thus, in order to prove the lemma, we must find an alternative algorithm $\mathcal{B}$ which is non-adaptive, yet continues to be optimal. To this end, we shall first express $\mathbb{E}[w(\mathcal{M}(v))]$ in a convenient form for each $v \in V$, where $w(\mathcal{M}(v))$ is the weight of the edge matched to v (which is 0 if no match occurs).

Given $v \in V$ and $1 \leq i \leq |U|$, we define X_i^v to be the i^{th} edge adjacent to v that is probed by $\mathcal{A}$. This is set equal to λ by convention, provided no such edge exists. We may then define $\boldsymbol{X}^v := (X_1^v, \ldots, X_{|U|}^v)$, and $\boldsymbol{X}_{\leq k}^v := (X_1^v, \ldots, X_k^v)$ for each $1 \leq k \leq |U|$. Moreover, given $\boldsymbol{e} = (e_1, \ldots, e_k) \in E^{(*)}$ with $k \geq 1$, define $S(\boldsymbol{e})$ to be the event in which e_k is the only active edge amongst $e_1, \ldots, e_k$. Observe then that

$$\mathbb{E}[w(\mathcal{M}(v))] = \sum_{\substack{\boldsymbol{e}=(e_1,\ldots,e_k)\in\mathcal{C}_v: \\ k\geq 1}} w_{e_k} \mathbb{P}[S(\boldsymbol{e}) \cap \{\boldsymbol{X}_{\leq k}^v = \boldsymbol{e}\}],$$

as (1) and (2) ensure v is matched to the first probed edge which is revealed to be active. Moreover, if $\boldsymbol{e} = (e_1, \ldots, e_k) \in \mathcal{C}_v$ for $k \geq 2$, then

$$\mathbb{P}[S(\boldsymbol{e}) \cap \{\boldsymbol{X}_{\leq k}^v = \boldsymbol{e}\}] = \mathbb{P}[\{\mathrm{st}(e_k) = 1\} \cap \{\boldsymbol{X}_{\leq k}^v = \boldsymbol{e}\}], \tag{3.6}$$

as (1) and (2) ensure $\boldsymbol{X}_{\leq k}^v = \boldsymbol{e}$ only if $e_1, \ldots, e_{k-1}$ are inactive. Thus,

$$\begin{aligned}
\mathbb{E}[w(\mathcal{M}(v))] &= \sum_{\substack{\boldsymbol{e}=(e_1,\ldots,e_k)\in\mathcal{C}_v: \\ k\geq 1}} w_{e_k} \mathbb{P}[S(\boldsymbol{e}) \cap \{\boldsymbol{X}_{\leq k}^v = \boldsymbol{e}\}] \\
&= \sum_{\substack{\boldsymbol{e}=(e_1,\ldots,e_k)\in\mathcal{C}_v: \\ k\geq 1}} w_{e_k} \mathbb{P}[\{\mathrm{st}(e_k) = 1\} \cap \{\boldsymbol{X}_{\leq k}^v = \boldsymbol{e}\}] \\
&= \sum_{\substack{\boldsymbol{e}=(e_1,\ldots,e_k)\in\mathcal{C}_v: \\ k\geq 1}} w_{e_k} p_{e_k} \mathbb{P}[\boldsymbol{X}_{\leq k}^v = \boldsymbol{e}],
\end{aligned}$$

where the final equality holds since $\mathcal{A}$ must decide on whether to probe e_k prior to revealing $\mathrm{st}(e_k)$. As a result, after summing over $v \in V$,

$$\mathbb{E}[w(\mathcal{M})] = \sum_{v \in V} \sum_{\substack{\boldsymbol{e}=(e_1,\ldots,e_k)\in\mathcal{C}_v: \\ k\geq 1}} w_{e_k} p_{e_k} \mathbb{P}[\boldsymbol{X}_{\leq k}^v = \boldsymbol{e}]. \tag{3.7}$$

Our goal is to find a non-adaptive relaxed probing algorithm which matches the value of (3.7). Thus, for each $v \in V$ and $\boldsymbol{e} = (e_1, \ldots, e_k) \in \mathcal{C}_v$ with $k \geq 1$, define

$$x_v(\boldsymbol{e}) := \mathbb{P}[\boldsymbol{X}^v_{\leq k} = \boldsymbol{e}],$$

where $x_v(\lambda) := 1$. Observe now that for each $\boldsymbol{e}' = (e'_1, \ldots, e'_k) \in \mathcal{C}_v$,

$$\sum_{\substack{e \in \partial(v): \\ (\boldsymbol{e}', e) \in \mathcal{C}_v}} \mathbb{P}[\boldsymbol{X}^v_{\leq k+1} = (\boldsymbol{e}', e) \,|\, \boldsymbol{X}^v_{\leq k} = \boldsymbol{e}'] \leq 1 - p_{e'_k}. \tag{3.8}$$

To see (3.8), observe that the the left-hand side corresponds to the probability $\mathcal{A}$ probes some edge $e \in \partial(v)$, given it already probed $\boldsymbol{e}'$ in order. On the other hand, if a subsequent edge is probed, then (1) and (2) imply that e'_k must have been inactive, which occurs independently of the event $\boldsymbol{X}^v_{\leq k} = \boldsymbol{e}'$. This explains the right-hand side of (3.8). Using (3.8), the values $(x_v(\boldsymbol{e}))_{\boldsymbol{e} \in \mathcal{C}_v}$ satisfy

$$\sum_{\substack{e \in \partial(v): \\ (\boldsymbol{e}', e) \in \mathcal{C}_v}} x_v(\boldsymbol{e}', e) \leq (1 - p_{e'_k}) \cdot x_v(\boldsymbol{e}'), \tag{3.9}$$

for each $\boldsymbol{e}' = (e'_1, \ldots, e'_k) \in \mathcal{C}_v$ with $k \geq 1$. Moreover, clearly $\sum_{e \in \partial(v)} x_v(e) \leq 1$.

Given $\boldsymbol{e} = (e_1, \ldots, e_k) \in \mathcal{C}_v$ for $k \geq 1$, recall that $\boldsymbol{e}_{<k} := (e_1, \ldots, e_{k-1})$ where $\boldsymbol{e}_{<1} := \lambda$ if $k = 1$. Moreover, $g(\boldsymbol{e}_{<k}) := \prod_{i=1}^{k-1} (1 - p_{e_i})$, where $g(\lambda) := 1$. Using this notation, define for each $\boldsymbol{e} \in \mathcal{C}_v$

$$y_v(\boldsymbol{e}) := \begin{cases} x_v(\boldsymbol{e}) / g(\boldsymbol{e}_{<|\boldsymbol{e}|}) & \text{if } |\boldsymbol{e}| \geq 1, \\ 1 & \text{otherwise.} \end{cases} \tag{3.10}$$

Observe that (3.9) ensures that for each $\boldsymbol{e}' \in \mathcal{C}_v$,

$$\sum_{\substack{e \in \partial(v): \\ (\boldsymbol{e}', e) \in \mathcal{C}_v}} y_v(\boldsymbol{e}', e) \leq y_v(\boldsymbol{e}'), \tag{3.11}$$

and $y_v(\lambda) := 1$. As a result, Proposition 3.4 implies that for each $v \in V$, there exists a distribution $\mathcal{D}^v$ such that if $\boldsymbol{Y}^v \sim \mathcal{D}^v$, then for each $\boldsymbol{e} \in \mathcal{C}_v$ with $|\boldsymbol{e}| = k \geq 1$,

$$\mathbb{P}[\boldsymbol{Y}^v_{\leq k} = \boldsymbol{e}] = y_v(\boldsymbol{e}). \tag{3.12}$$

Moreover, $\boldsymbol{Y}^v$ is drawn independently from the edge states, $(\mathrm{st}(e))_{e \in E}$. Consider now the following algorithm $\mathcal{B}$, which satisfies the desired properties (Q_1) and (Q_2) of Lemma 3.1:

Algorithm 3 Algorithm $\mathcal{B}$

```
Input: a stochastic graph $G = (U, V, E)$.
Output: a one-sided matching $\mathcal{N}$ of $G$ of active edges.
1: Set $\mathcal{N} \leftarrow \emptyset$.
2: Draw $(\boldsymbol{Y}^v)_{v \in V}$ according to the product distribution $\prod_{v \in V} \mathcal{D}^v$.
3: for $v \in V$ do
4:     for $i = 1, \ldots, |\boldsymbol{Y}^v|$ do
5:         Set $e \leftarrow \boldsymbol{Y}^v_i$.                ▷ $\boldsymbol{Y}^v_i$ is the $i^{th}$ edge of $\boldsymbol{Y}^v$
6:         Probe the edge $e$, revealing $\mathrm{st}(e)$.
7:         if $\mathrm{st}(e) = 1$ and $v$ is unmatched by $\mathcal{N}$ then
8:             Add $e$ to $\mathcal{N}$.
9: return $\mathcal{N}$.
```

Using (3.12) and the non-adaptivity of $\mathcal{B}$, it is clear that for each $v \in V$,

$$\begin{aligned}\mathbb{E}[w(\mathcal{N}(v))] &= \sum_{\substack{\boldsymbol{e}=(e_1,\ldots,e_k)\in\mathcal{C}_v: \\ k\geq 1}} w_{e_k}\mathbb{P}[S(\boldsymbol{e})]\cdot\mathbb{P}[\boldsymbol{Y}^v_{\leq k}=\boldsymbol{e}] \\ &= \sum_{\substack{\boldsymbol{e}=(e_1,\ldots,e_k)\in\mathcal{C}_v: \\ k\geq 1}} w_{e_k}p_{e_k}g(\boldsymbol{e}_{<k})y_v(\boldsymbol{e}) \\ &= \sum_{\substack{\boldsymbol{e}=(e_1,\ldots,e_k)\in\mathcal{C}_v: \\ k\geq 1}} w_{e_k}p_{e_k}x_v(\boldsymbol{e}) \\ &= \mathbb{E}[w(\mathcal{M}(v))].\end{aligned}$$

Thus, after summing over $v \in V$, it holds that $\mathbb{E}[w(\mathcal{N})] = \mathbb{E}[w(\mathcal{M})] = \mathrm{OPT}_{\mathrm{rel}}(G)$, and so in addition to satisfying (Q_1) and (Q_2), $\mathcal{B}$ is optimal. Finally, it is easy to show that each $u \in U$ is matched by $\mathcal{N}$ at most once in expectation since $\mathcal{M}$ has this property. Thus, $\mathcal{B}$ is a relaxed probing algorithm which is optimal and satisfies the required properties of Lemma 3.1.

□

4 Proving Theorems 2.1 and 2.3

In this section, we first review rank one contention resolution schemes. We then prove Theorems 2.1 and 2.3 for the case of a known stochastic graph. Finally, in Subsection 4.1, we generalize to the case of an arbitrary known i.d. input.

Given $k \geq 1$, consider the ground set $[k] := \{1,\ldots,k\}$. Fix $\boldsymbol{z} \in [0,1]^k$, and let $R(\boldsymbol{z}) \subseteq [k]$ denote the random set where each $i \in [k]$ is included in $R(\boldsymbol{z})$ independently with probability z_i. Let us denote $\mathcal{P} := \{\boldsymbol{z} \in [0,1]^k : \sum_{i=1}^k z_i \leq 1\}$. Note that $\mathcal{P}$ is the convex relaxation of the constraint imposed by the rank one matroid on $[k]$ (i.e., at most one element of $[k]$ may be selected).

Definition 2 (Contention Resolution Scheme – Rank One Matroid – [38])**.** A **contention resolution scheme** (CRS) for the rank one matroid on $[k]$ is a (randomized) algorithm ψ, which given $\boldsymbol{z} \in \mathcal{P}$ and $S \subseteq [k]$ as inputs, returns a single element $\psi_{\boldsymbol{z}}(S)$ of S. Given $c \in [0,1]$, ψ is said to be c**-selectable**, provided for all $i \in [k]$ and $\boldsymbol{z} \in \mathcal{P}$,

$$\mathbb{P}[i \in \psi_{\boldsymbol{z}}(R(\boldsymbol{z}))\,|\, i \in R(\boldsymbol{z})] \geq c, \tag{4.1}$$

where the probability is over the generation of $R(\boldsymbol{z})$, and the potential randomness used by ψ.

Feldman et al. [25] considered a more restricted class of contention resolution schemes, called **online contention resolution schemes** (OCRS). These are schemes in which $R(\boldsymbol{z})$ is *not* known to the scheme ahead of time. Instead, the elements of $[k]$ are presented to the scheme ψ in adversarial order, where in each step, an arriving $i \in [k]$ reveals if it is in $R(\boldsymbol{z})$, at which point ψ must make an irrevocable decision as to whether it wishes to return i as its output.

In the adversarial arrival model, we make use of the OCRS recently introduced by Ezra et al. [24], restricted to the case of a rank one matroid. Note that this scheme is similar to the OCRS considered by Lee and Singla [33], however it has the benefit of not requiring the adversary to present the arrival order of $[k]$ to the algorithm upfront. Given the ground set $[k] = \{1,\ldots k\}$, suppose the elements of $[k]$ arrive according to some permutation $\sigma : [k] \to [k]$ (i.e., $\sigma(1),\ldots,\sigma(k)$), and $\boldsymbol{z} \in [0,1]^k$ satisfies $\sum_{i=1}^k z_i \leq 1$. Upon the arrival of element $\sigma(t) \in [k]$, compute

$$q_t := \frac{1}{2 - \sum_{i=1}^{t-1} z_{\sigma(i)}}.$$

Observe that $1/2 \leq q_t \leq 1$, as $0 \leq \sum_{i=1}^k z_i \leq 1$, and so the following OCRS is well-defined:

Algorithm 4 OCRS – Ezra et al. [24]

```
Input: $\boldsymbol{z} \in \mathcal{P}$, where $\mathcal{P} \subseteq [0,1]^k$.        ▷ $\mathcal{P}$ is the convex relaxation of the rank one matroid
Output: at most one element of $[k]$.
1: for $t = 1, \ldots, k$ do
2:     if $\sigma(t) \in R(\boldsymbol{z})$ then        ▷ $\sigma(t)$ is in $R(\boldsymbol{z})$ with probability $z_{\sigma(t)}$
3:         Compute $q_t$ based on the arrivals $\sigma(1), \ldots, \sigma(t-1)$.
4:         return $\sigma(t)$ independently with probability $q_t$.
5: return $\emptyset$.        ▷ pass on returning an element of $[k]$
```

Theorem 4.1 (Ezra et al. [33])**.** *Algorithm 4 is an OCRS for a rank one matroid which is* $1/2$*-selectable.*

Suppose now we are presented a known stochastic graph $G = (U, V, E)$, whose online vertices $v_1, \ldots, v_n$ are presented to the online probing algorithm according to an adversarially chosen permutation $\pi : [n] \rightarrow [n]$ (i.e., $v_{\pi(1)}, \ldots, v_{\pi(n)}$). Let $(x_v(\boldsymbol{e}))_{v \in V, \boldsymbol{e} \in \mathcal{C}_v}$ be an optimum solution to LP-config for G with induced edge variables $(\widetilde{x}_e)_{e \in E}$. For each $t \in [n]$ and $u \in U$, define

$$q_{u,t} := \frac{1}{2 - \sum_{i=1}^{t-1} z_{u, v_{\pi(i)}}}, \tag{4.2}$$

where $z_e := p_e \widetilde{x}_e$ for $e \in E$, and $q_{u,1} := 1/2$. Clearly, $\sum_{v \in V} z_{u,v} \leq 1$, by constraint (2.1) of LP-config, and so $1/2 \leq q_{u,t} \leq 1$. We consider the following algorithm, is presented V in order π:

Algorithm 5 Known Stochastic Graph – AOM – Modified

```
Input: a stochastic graph $G = (U, V, E)$.
Output: a matching $\mathcal{M}$ of $G$ of active edges.
1: $\mathcal{M} \leftarrow \emptyset$.
2: Compute an optimum solution of LP-config for $G$, say $(x_v(\boldsymbol{e}))_{v \in V, \boldsymbol{e} \in \mathcal{C}_v}$.
3: for $t = 1, \ldots, n$ do
4:     Based on the previous arrivals $v_{\pi(1)}, \ldots, v_{\pi(t-1)}$ before $v_{\pi(t)}$, compute values $(q_{u,t})_{u \in U}$.
5:     Set $e \leftarrow \text{VertexProbe}\left(v_{\pi(t)}, \partial(v_{\pi(t)}), (x_{v_{\pi(t)}}(\boldsymbol{e}))_{\boldsymbol{e} \in \mathcal{C}_{v_{\pi(t)}}}\right)$.
6:     if $e = (u, v_{\pi(t)})$ for some $u \in U$, and $u$ is unmatched then
7:         Add $e$ to $\mathcal{M}$ independently with probability $q_{u,t}$.        ▷ OCRS is used here
8: return $\mathcal{M}$.
```

Proposition 4.2. *Algorithm 5 attains a competitive ratio of* $1/2$.

Proof. Given $u \in U$, let $\mathcal{M}(u)$ denote the edge matched to u by $\mathcal{M}$, where $\mathcal{M}(u) := \emptyset$ if no such edge exists. Observe now that if $C(e)$ corresponds to the event in which VertexProbe commits to $e \in \partial(u)$, then $\mathbb{P}[C(e)] = p_e \widetilde{x}_e$ by Lemma 2.12. Moreover, the events $(C(e))_{e \in \partial(u)}$ are independent, and satisfy

$$\sum_{e \in \partial(u)} \mathbb{P}[C(e)] = \sum_{e \in \partial(u)} p_e \widetilde{x}_e \leq 1, \tag{4.3}$$

by constraint (2.1) of LP-config. As such, denote $\boldsymbol{z} := (z_e)_{e \in \partial(u)}$ where $z_e = p_e \widetilde{x}_e$, and observe that (4.3) ensures that $\boldsymbol{z} \in \mathcal{P}$, where $\mathcal{P}$ is the convex relaxation of the rank one matroid on $\partial(u)$. Let us denote $R(\boldsymbol{z})$ as those those $e \in \partial(u)$ for which $C(e)$ occurs.

If ψ is the OCRS defined in Algorithm 4, then we may pass $\boldsymbol{z}$ to ψ, and process the edges of $\partial(u)$ in the order induced by π. Denote the resulting output by $\psi_{\boldsymbol{z}}(R(\boldsymbol{z}))$. By coupling the random draws of lines (4) and (7) of Algorithms 4 and 5, respectively, we get that

$$w(\mathcal{M}(u)) = \sum_{e \in \partial(u)} w_e \cdot \mathbf{1}_{[e \in R(\boldsymbol{z})]} \cdot \mathbf{1}_{[e \in \psi_{\boldsymbol{z}}(R(\boldsymbol{z}))]}$$

Thus, after taking expectations,

$$\mathbb{E}[w(\mathcal{M}(u))] = \sum_{e \in \partial(u)} w_e \cdot \mathbb{P}[e \in \psi_{\boldsymbol{z}}(R(\boldsymbol{z})) \,|\, e \in R(\boldsymbol{z})] \cdot \mathbb{P}[e \in R(\boldsymbol{z})].$$

Now, Theorem 4.1 ensures that for each $e \in \partial(u)$, $\mathbb{P}[e \in \psi_{\boldsymbol{z}}(R(\boldsymbol{z})) \,|\, e \in R(\boldsymbol{z})] \geq 1/2$. It follows that $\mathbb{E}[w(\mathcal{M}(u))] \geq \frac{1}{2} \sum_{e \in \partial(u)} w_e p_e \widetilde{x}_e$, for each $u \in U$. Thus,

$$\begin{aligned}\mathbb{E}[w(\mathcal{M})] &= \sum_{u \in U} \mathbb{E}[w(\mathcal{M}(u))] \\ &\geq \frac{1}{2} \sum_{e \in E} w_e p_e \widetilde{x}_e = \frac{\text{LPOPT}(G)}{2},\end{aligned}$$

where the equality follows since $(x_v(\boldsymbol{e}))_{v \in V, \boldsymbol{e} \in \mathcal{C}_v}$ is an optimum solution to LP-config. On the other hand, $\text{LPOPT}(G) \geq \text{OPT}(G)$ by Theorem 2.10, and so the proof is complete. □

Both Lee and Singla [33], as well as Adamczyk and Wlodarczyk [3], defined a **random order contention resolution scheme** (RCRS), which is an OCRS where the elements of $[k]$ arrive in random order. In this definition, the random order is incorporated into the probabilistic computation of selectibility (4.1). We improve the competitive ratio of Algorithm 1 by applying a specific RCRS introduced by Lee and Singla [33]. Given the ground set $[k] = \{1, \dots k\}$, draw $Y_i \sim [0,1]$ u.a.r. and independently for $i = 1, \dots, k$.

Algorithm 6 RCRS – Lee and Singla [33]

```
Input: $\boldsymbol{z} \in \mathcal{P}$, where $\mathcal{P} \subseteq [0,1]^k$.
Output: at most one element of $[k]$.
1: for $i \in [k]$ in increasing order of $Y_i$ do
2:     if $i \in R(\boldsymbol{z})$ then
3:         return $i$ independently with probability $\exp(-Y_i \cdot z_i)$
4: return $\emptyset$.                    ▷ pass on returning an element of $[k]$.
```

Theorem 4.3 (Lee and Singla [33])**.** *Algorithm 6 is a $1 - 1/e$-selectable RCRS for the case of a rank one matroid.*

For each $v \in V$, draw $\widetilde{Y}_v \in [0,1]$ independently and uniformly at random. We assume the vertices of V are presented to the below algorithm in non-decreasing order, based upon the values $(\widetilde{Y}_v)_{v \in V}$.

Algorithm 7 Known Stochastic Graph – ROM– Modified

```
Input: a stochastic graph $G = (U, V, E)$.
Output: a matching $\mathcal{M}$ of $G$ of active edges.
1: $\mathcal{M} \leftarrow \emptyset$.
2: Compute an optimum solution of LP-config for $G$, say $(x_v(\boldsymbol{e}))_{v \in V, \boldsymbol{e} \in \mathcal{C}_v}$.
3: for $s \in V$ in increasing order of $\widetilde{Y}_s$ do
4:     Set $e \leftarrow \textsc{VertexProbe}(s, \partial(s), (x_s(\boldsymbol{e}))_{\boldsymbol{e} \in \mathcal{C}_s})$.
5:     if $e = (u, s)$ for some $u \in U$, and $u$ is unmatched then
6:         Add $e$ to $\mathcal{M}$ independently with probability $\exp(-\widetilde{Y}_s \cdot p_{u,s} \cdot \widetilde{x}_{u,s})$.
7: return $\mathcal{M}$.
```

Proposition 4.4. *Algorithm 7 attains a competitive ratio of* $1 - 1/e$.

The proof follows almost identically to the proof of Proposition 4.2, and so we defer it to Appendix B

4.1 Extending to Known I.D. Arrivals

Suppose that $(H_{\text{typ}}, (\mathcal{D}_i)_{i=1}^n)$ is a known *i.d.* input, where $H_{\text{typ}} = (U, B, F)$ has downward-closed online probing constraints $(\mathcal{C}_b)_{b \in B}$. If $G \sim (H_{\text{typ}}, (\mathcal{D}_i)_{i=1}^n)$, where $G = (U, V, E)$ has vertices $V = \{v_1, \ldots, v_n\}$, then define $r_i(b) := \mathbb{P}[v_i = b]$ for each $i \in [n]$ and $b \in B$, where we hereby assume that $r_i(b) > 0$. We generalize LP-config to account for the distributions $(\mathcal{D}_i)_{i=1}^n$. For each $i \in [n], b \in B$ and $\boldsymbol{e} \in \mathcal{C}_b$, we introduce a decision variable $x_i(\boldsymbol{e} \,||\, b)$ to encode the probability that v_i has type b *and* $\boldsymbol{e}$ is the sequence of edges of $\partial(v_i)$ probed by the *relaxed* benchmark.

$$\text{maximize} \quad \sum_{i \in [n], b \in B, \boldsymbol{e} \in \mathcal{C}_b} \text{val}(\boldsymbol{e}) \cdot x_i(\boldsymbol{e} \,||\, b) \qquad \text{(LP-config-id)}$$

$$\text{subject to} \quad \sum_{i \in [n], b \in B} \sum_{\substack{\boldsymbol{e} \in \mathcal{C}_b: \\ (u,b) \in \boldsymbol{e}}} p_{u,b} \cdot g(\boldsymbol{e}_{<(u,b)}) \cdot x_i(\boldsymbol{e} \,||\, b) \leq 1 \qquad \forall u \in U \qquad (4.4)$$

$$\sum_{\boldsymbol{e} \in \mathcal{C}_b} x_i(\boldsymbol{e} \,||\, b) = r_i(b) \qquad \forall b \in B, i \in [n] \qquad (4.5)$$

$$x_i(\boldsymbol{e} \,||\, b) \geq 0 \qquad \forall b \in B, \boldsymbol{e} \in \mathcal{C}_b, i \in [n] \qquad (4.6)$$

Let us denote $\text{LPOPT}(H_{\text{typ}}, (\mathcal{D}_i)_{i=1}^n)$ as the value of an optimum solution to LP-config-id.

Theorem 4.5. *$OPT(H_{typ}, (\mathcal{D}_i)_{i=1}^n) \leq LPOPT(H_{typ}, (\mathcal{D}_i)_{i=1}^n)$.*

One way to prove Theorem 4.5 is to use the properties of the relaxed benchmark on G guaranteed by Lemma 3.1, and the above interpretation of the decision variables to argue that

$$\mathbb{E}[\text{OPT}_{\text{rel}}(G)] \leq \text{LPOPT}(H_{\text{typ}}, (\mathcal{D}_i)_{i=1}^n),$$

where $\text{OPT}_{\text{rel}}(G)$ is the value of the relaxed benchmark on G. Specifically, we can interpret (4.4) as saying that the relaxed benchmark matches each offline vertex at most once in expectation. Moreover, (4.5) holds by observing that if v_i is of type b, then the relaxed benchmark selects some $\boldsymbol{e} \in \mathcal{C}_b$ to probe (note $\boldsymbol{e}$ could be the empty-string). We provide a morally equivalent proof of Theorem 4.5 in Appendix B. Specifically, we consider an optimum solution of LP-config with respect to G, and apply a conditioning argument in conjunction with Theorem 2.10.

Given a feasible solution to LP-config-id, say $(x_i(\boldsymbol{e} \,||\, b))_{i \in [n], b \in B, \boldsymbol{e} \in \mathcal{C}_b}$, for each $u \in U, i \in [n]$ and $b \in B$ define

$$\widetilde{x}_{u,i}(b) := \sum_{\substack{\boldsymbol{e} \in \mathcal{C}_b: \\ (u,b) \in \boldsymbol{e}}} g(\boldsymbol{e}_{<(u,b)}) \cdot x_i(\boldsymbol{e} \,||\, b). \qquad (4.7)$$

We refer to $\widetilde{x}_{u,i}(b)$ as an **(induced) edge variable**, thus extending the definition from the known stochastic graph setting. Suppose now that we fix $i \in [n]$ and $b \in B$, and consider the variables, $(x_i(\boldsymbol{e} \,||\, b))_{\boldsymbol{e} \in \mathcal{C}_b}$. Observe that (4.5) ensures that

$$\frac{\sum_{\boldsymbol{e} \in \mathcal{C}_b} x_i(\boldsymbol{e} \,||\, b)}{r_i(b)} = 1.$$

Hence, regardless of which type node v_i is drawn as,

$$\frac{\sum_{\boldsymbol{e} \in \mathcal{C}_{v_i}} x_i(\boldsymbol{e} \,||\, v_i)}{r_i(v_i)} = 1.$$

We can therefore generalize VertexProbe as follows. Given vertex v_i, draw $\boldsymbol{e}' \in \mathcal{C}_{v_i}$ with probability $x_i(\boldsymbol{e}' \,||\, v_i)/r_i(v_i)$. If $\boldsymbol{e}' = \lambda$, then return the empty-set. Otherwise, set $\boldsymbol{e}' = (e_1', \ldots, e_k')$ for $k := |\boldsymbol{e}'| \geq 1$, and probe the edges of $\boldsymbol{e}'$ in order. Return the first edge which is revealed to be active, if such an edge exists. Otherwise, return the empty-set. We denote the output of VertexProbe on the input $(v_i, \partial(v_i), (x_i(\boldsymbol{e} \,||\, v_i)/r_i(v_i))_{\boldsymbol{e} \in \mathcal{C}_{v_i}})$ by $\text{VertexProbe}(v_i, \partial(v_i), (x_i(\boldsymbol{e} \,||\, v_i)/r_i(v_i))_{\boldsymbol{e} \in \mathcal{C}_{v_i}})$. Define $C(u, v_i)$ as the event in which VertexProbe outputs the edge (u, v_i), and observe the following extension of Lemma 2.12:

Lemma 4.6. *If* VERTEXPROBE *is passed* $\left(v_i, \partial(v_i), (x_i(\boldsymbol{e} \,||\, v_i)/r_i(v_i))_{\boldsymbol{e} \in \mathcal{C}_{v_i}}\right)$*, then for any* $b \in B$ *and* $u \in U$*,*

$$\mathbb{P}[C(u, v_i) \,|\, v_i = b] = \frac{p_{u,b} \cdot \widetilde{x}_{u,i}(b)}{r_i(b)}.$$

Remark 4.7. As in Definition 1, if $C(u, v_i)$ occurs, then we say that u commits to (u, v_i) or v_i.

Consider now the generalization of Algorithm 1 where π is generated either u.a.r. or adversarially.

Algorithm 8 Known I.D

```
Input: a known i.d. input $(H_{\text{typ}}, (\mathcal{D}_i)_{i=1}^n)$.
Output: a matching $\mathcal{M}$ of active edges of $G \sim (H_{\text{typ}}, (\mathcal{D}_i)_{i=1}^n)$.
1: $\mathcal{M} \leftarrow \emptyset$.
2: Compute an optimum solution of LP-config-id for $(H_{\text{typ}}, (\mathcal{D}_i)_{i=1}^n)$, say $(x_i(\boldsymbol{e} \,||\, b))_{i \in [n], b \in B, \boldsymbol{e} \in \mathcal{C}_b}$.
3: for $t = 1, \ldots, n$ do
4:     Let $a \in B$ be the type of the current arrival $v_{\pi(t)}$.          ▷ to simplify notation
5:     Set $e \leftarrow$ VERTEXPROBE $\left(v_{\pi(t)}, \partial(v_{\pi(t)}), \left(x_{\pi(t)}(\boldsymbol{e} \,||\, a) \cdot r_{\pi(t)}^{-1}(a)\right)_{\boldsymbol{e} \in \mathcal{C}_a}\right)$.
6:     if $e = (u, v_{\pi(t)})$ for some $u \in U$, and $u$ is unmatched then
7:         Add $e$ to $\mathcal{M}$.
8: return $\mathcal{M}$.
```

Similarly, to Algorithm 1 of Proposition 2.14, one can show that Algorithm 8 attains a competitive ratio of 1/2 for random order arrivals. Interestingly, if the distributions $(\mathcal{D}_i)_{i=1}^n$ are identical – that is, we work with known i.i.d. arrivals – then it is relatively easy to show that this algorithm's competitive ratio improves to $1 - 1/e$.

Proposition 4.8. *If Algorithm 8 is presented a known i.i.d. input, say the type graph* H_{typ} *together with the distribution* $\mathcal{D}$*, then* $\mathbb{E}[w(\mathcal{M})] \geq (1 - 1/e)$ $OPT(H_{typ}, \mathcal{D})$*.*

Remark 4.9. Proposition 4.8 is proven explicitly in an earlier arXiv version of this paper for the case of patience values.

Returning to the case of non-identical distributions, observe that in the execution of Algorithm 8 the probability that v_i commits to the edge (u, v_i) for $u \in U$ is precisely

$$z_{u,i} := \sum_{b \in B} p_{u,b} \cdot \widetilde{x}_{u,i}(b) = \sum_{b \in B} \sum_{\substack{\boldsymbol{e} \in \mathcal{C}_b: \\ (u,b) \in \boldsymbol{e}}} p_{u,b} \cdot g(\boldsymbol{e}_{<(u,b)}) \cdot x_i(\boldsymbol{e} \,||\, b). \tag{4.8}$$

Moreover, the events $(C(u, v_i))_{i=1}^n$ are independent, so this suggests applying the same contention resolutions schemes as in the known stochastic graph setting. We first focus on the adversarial arrival model, where we assume the vertices $v_1, \ldots, v_n$ are presented in some unknown order $\pi : [n] \to [n]$. We make use of the OCRS from before (Algorithm 4). For each $t \in [n]$ and $u \in U$, define

$$q_{u,t} := \frac{1}{2 - \sum_{i=1}^{t-1} z_{u,\pi(i)}}, \tag{4.9}$$

where $q_{u,1} := 1/2$. Note that $1/2 \leq q_{u,t} \leq 1$ as $\sum_{j \in [n]} z_{u,j} \leq 1$ by constraint (4.4) of LP-config-id. We define Algorithm 9 by modifying Algorithm 8 using the OCRS to ensure that each $i \in [n]$ is matched to $u \in U$ with probability $z_{u,i}/2$. However, to achieve a competitive ratio of 1/2, we require the stronger claim that for each type node $a \in B$, the probability (u, v_i) is added to the matching *and* v_i is of type a is lower bounded by $p_{u,a}\widetilde{x}_{u,i}(a)/2$. Crucially, if we condition on $u \in U$ being unmatched when v_i is processed, v_i having type a, and $C(u, v_i)$, then the probability the OCRS matches u to v_i does *not* depend on a. As we show below in the proof of Theorem 2.1, this implies the desired lower bound of $p_{u,a}\widetilde{x}_{u,i}(a)/2$, and so Algorithm 9 attains a competitive ratio of 1/2 by (4.7) and Theorem 4.5.

Algorithm 9 Known I.D. – AOM – Modified

```
Input: a known i.d. input $(H_{\text{typ}}, (\mathcal{D}_i)_{i=1}^n)$.
Output: a matching $\mathcal{M}$ of active edges of $G \sim (H_{\text{typ}}, (\mathcal{D}_t)_{t=1}^n)$.
1: $\mathcal{M} \leftarrow \emptyset$.
2: Compute an optimum solution of LP-config-id for $(H_{\text{typ}}, (\mathcal{D}_i)_{i=1}^n)$, say $(x_i(\boldsymbol{e} \,||\, b))_{i \in [n], b \in B, \boldsymbol{e} \in \mathcal{C}_b}$.
3: for $t = 1, \ldots, n$ do
4:     Let $a \in B$ be the type of the current arrival $v_{\pi(t)}$.
5:     Based on the previous arrivals $v_{\pi(1)}, \ldots, v_{\pi(t-1)}$ before $v_{\pi(t)}$, compute values $(q_{u,t})_{u \in U}$.
6:     Set $e \leftarrow \text{VertexProbe}\left(v_{\pi(t)}, \partial(v_{\pi(t)}), \left(x_{\pi(t)}(\boldsymbol{e} \,||\, a) \cdot r^{-1}_{\pi(t)}(a)\right)_{\boldsymbol{e} \in \mathcal{C}_a}\right)$.
7:     if $e = (u, v_t)$ for some $u \in U$, and $u$ is unmatched then
8:         Add $e$ to $\mathcal{M}$ independently with probability $q_{u,t}$.
9: return $\mathcal{M}$.
```

Proof of Theorem 2.1. For notational simplicity, let us assume that $\pi(t) = t$ for each $t \in [n]$, so that the online vertices arrive in order $v_1, \ldots, v_n$. Now, the edge variables $(\widetilde{x}_{u,t}(b))_{u \in U, t \in [n], b \in B}$ satisfy

$$\text{LPOPT}(H_{\text{typ}}, (\mathcal{D}_i)_{i=1}^n) = \sum_{u \in U, t \in [n], b \in B} p_{u,b} w_{u,b} \widetilde{x}_{u,t}(b).$$

Thus, to complete the proof it suffices to show that

$$\mathbb{P}[(u, v_t) \in \mathcal{M} \text{ and } v_t = b] \geq \frac{\widetilde{x}_{u,t}(b)}{2} \tag{4.10}$$

for each $u \in U, t \in [n]$ and $b \in B$, where we hereby assume w.l.o.g. that $\widetilde{x}_{u,t}(b) > 0$. In order to prove this, we first observe that by the same coupling argument used in the proof of Proposition 4.2,

$$\mathbb{P}[(u, v_t) \in \mathcal{M}] \geq \frac{z_{u,t}}{2} = \frac{1}{2} \sum_{b \in B} p_{u,b} \widetilde{x}_{u,t}(b) \tag{4.11}$$

as a result of the 1/2-selectability of Algorithm 4. Let us now define R_t as the unmatched vertices of U when v_t arrives. Observe then that

$$\mathbb{P}[(u, v_t) \in \mathcal{M} \,|\, v_t = b, C(u, v_t) \text{ and } u \in R_t] = q_{u,t}. \tag{4.12}$$

Now, $\mathbb{P}[v_t = b, C(u, v_t) \text{ and } u \in R_t] = p_{u,b} \cdot \widetilde{x}_{u,t}(b) \cdot \mathbb{P}[u \in R_t]$, by Lemma 4.6 and the independence of the events $\{v_t = b\} \cap \{C(u, v_t)\}$ and $\{u \in R_t\}$. Thus, by the law of total probability,

$$\begin{aligned}\sum_{b \in B} p_{u,b} \widetilde{x}_{u,t} q_{u,t} \cdot \mathbb{P}[u \in R_t] &= \mathbb{P}[(u, v_t) \in \mathcal{M}] \\ &\geq \frac{z_{u,t}}{2} \\ &= \frac{1}{2} \sum_{b \in B} p_{u,b} \widetilde{x}_{u,t}(b)\end{aligned}$$

where the second inequality follows from (4.11). Thus, $q_{u,t} \cdot \mathbb{P}[u \in R_t] \geq 1/2$, and so combined with (4.12), (4.10) follows, thus completing the proof. ☐

Suppose now that each vertex v_t has an arrival time, say $\widetilde{Y}_t \in [0, 1]$, drawn u.a.r. and independently for $t \in [n]$. The values $(\widetilde{Y}_t)_{t=1}^n$ indicate the increasing order in which the vertices $v_1, \ldots, v_n$ arrive.

Algorithm 10 Known I.D. – ROM – Modified

```
Input: a known i.d. input $(H_{\text{typ}}, (\mathcal{D}_t)_{t=1}^n)$.
Output: a matching $\mathcal{M}$ of active edges of $G \sim (H_{\text{typ}}, (\mathcal{D}_t)_{t=1}^n)$.
1: $\mathcal{M} \leftarrow \emptyset$.
2: Compute an optimum solution of LP-config-id for $(H_{\text{typ}}, (\mathcal{D}_t)_{t=1}^n)$, say $(x_t(\boldsymbol{e} \,||\, b))_{t \in [n], b \in B, \boldsymbol{e} \in \mathcal{C}_b}$.
3: for $t \in [n]$ in increasing order of $\widetilde{Y}_t$ do
4:     Set $e \leftarrow \text{VertexProbe}\left(v_t, \partial(v_t), (x_t(\boldsymbol{e} \,||\, v_t)/r_t(v_t))_{\boldsymbol{e} \in \mathcal{C}_{v_t}}\right)$.
5:     if $e = (u, v_t)$ for some $u \in U$, and $u$ is unmatched then
6:         Add $e$ to $\mathcal{M}$ independently with probability $\exp(-\widetilde{Y}_t \cdot z_{u,t})$.
7: return $\mathcal{M}$.
```

Proof of Theorem 2.3. Clearly, Algorithm 10 is non-adaptive. The competitive ratio of $1 - 1/e$ follows by the same coupling argument as in Proposition 4.4, together with the same observations used in the proof of Theorem 2.1, and so we omit the argument. □

5 Efficiency of Our Algorithms

In this section, we prove Theorem 2.5, thus confirming the efficiency of the online probing algorithms of Theorems 2.1 and 2.3. We show how LP-config can be solved efficiently, as the extension to LP-config-id follows identically.

Theorem 5.1. *Suppose that $G = (U, V, E)$ is a stochastic graph with downward-closed probing constraints $(\mathcal{C}_v)_{v \in V}$. Given access to a membership oracle, LP-config is efficiently solvable in the size of G (excluding the constraints $(\mathcal{C}_v)_{v \in V}$).*

We prove Theorem 5.1 by first considering the dual of LP-config. Note, that in the below LP formulation, if $\boldsymbol{e} = (e_1, \ldots, e_k) \in \mathcal{C}_v$, then we set $e_i = (u_i, v)$ for $i = 1, \ldots, k$ for convenience.

$$\begin{aligned}
&\text{minimize} && \sum_{u \in U} \alpha_u + \sum_{v \in V} \beta_v && && \text{(LP-new-dual)} \\
&\text{subject to} && \beta_v + \sum_{j=1}^{|\boldsymbol{e}|} p_{e_j} \cdot g(\boldsymbol{e}_{<j}) \cdot \alpha_{u_j} \geq \sum_{j=1}^{|\boldsymbol{e}|} p_{e_j} \cdot w_{e_j} \cdot g(\boldsymbol{e}_{<j}) && \forall v \in V, \boldsymbol{e} \in \mathcal{C}_v && (5.1) \\
& && \alpha_u \geq 0 && \forall u \in U && (5.2) \\
& && \beta_v \in \mathbb{R} && \forall v \in V && (5.3)
\end{aligned}$$

Observe that to prove Theorem 5.1, it suffices to show that LP-new-dual has a deterministic polynomial time separation oracle, as a consequence of how the ellipsoid algorithm [36, 28] executes (see [39, 38, 1, 33] for more detail).

Suppose that we are presented a particular selection of dual variables, say $(\alpha_u)_{u \in U}$ and $(\beta_v)_{v \in V}$, which may or may not be a feasible solution to LP-new-dual. Our separation oracle must determine efficiently whether these variables satisfy all the constraints of LP-new-dual. In the case in which the solution is *infeasible*, the oracle must additionally return a constraint which is violated. It is clear that we can accomplish this for the non-negativity constraints, so let us fix a particular $v \in V$ in what follows. We wish to determine whether there exists some $\boldsymbol{e} = (e_1, \ldots, e_k) \in \mathcal{C}_v$, such that if $e_i = (u_i, v)$ for $i = 1, \ldots, k$, then

$$\sum_{j=1}^{|\boldsymbol{e}|} (w_{e_j} - \alpha_{u_j}) \cdot p_{e_j} \cdot g(\boldsymbol{e}_{<j}) > \beta_v, \tag{5.4}$$

where the left-hand side of (5.4) is 0 if $\boldsymbol{e} = \lambda$. In order to make this determination, it suffices to solve the following maximization problem. Given any selection of real values, $(\alpha_u)_{u \in U}$,

$$\text{maximize} \quad \sum_{i=1}^{|\boldsymbol{e}|} (w_{e_i} - \alpha_{u_i}) \cdot p_{e_i} \cdot \prod_{j=1}^{i-1} (1 - p_{e_j}) \tag{5.5}$$

$$\text{subject to} \quad \boldsymbol{e} \in \mathcal{C}_v \tag{5.6}$$

Before we show how (5.5) can be solved, we provide a buyer/seller interpretation of the optimization problem. Assuming first that the edges exist with certainty (i.e. $p_e \in \{0, 1\}$ for all $e \in \partial(v)$), let us suppose a seller is trying to allocate the items of U to a number of buyers. We view the vertex v as a *buyer* who wishes to purchase a subset of items $S \subseteq U$, based on their valuation function $f(S)$. Assume that v has **unit demand**, that is $f(S) := \max_{s \in S} p_{s,v} w_{s,v}$. The values $(\alpha_s)_{s \in U}$ are viewed as prices the buyer must pay[6], and the demand oracle returns a solution to $\max_{S \subseteq U}(f(S) - \sum_{s \in S} \alpha_s)$, thereby maximizing the utility of v. Clearly, for the simple case of a unit-demand buyer, an optimum assignment is the item $u \in U$ for which $p_{u,v} w_{u,v} - \alpha_u$ is maximized.

Returning the setting of arbitrary edge probabilities, even the case of a unit-demand buyer is a non-trivial optimization problem in the stochastic probing framework. Observe that we may view the edge probabilities $(p_e)_{e \in \partial(v)}$ as modelling the setting when there is uncertainty in whether or not the purchase proposals will succeed; that is, $\text{st}(u, v) = 1$, provided the seller agrees to sell item u to buyer v. In this interpretation, (5.5) is the expected utility of the unit-demand buyer v which purchases the first item $u \in U$ such that $\text{st}(u, v) = 1$, at which point v gains utility $w_{u,v} - \alpha_u$. In [9, 10] [7], we show how a buyer can solve (5.5) and use it to design a greedy online probing algorithm. We include the proof here for completeness.

Proposition 5.2 ([9, 10])**.** *If $\mathcal{C}_v$ is downward-closed, then for any selection of values $(\alpha_u)_{u \in U}$, (5.5) can be solved efficiently, assuming access to a membership query oracle for $\mathcal{C}_v$.*

Proof. Compute $\widetilde{w}_e := w_e - \alpha_u$ for each $e = (u, v) \in \partial(v)$, and define $P := \{e \in \partial(v) : \widetilde{w}_e \geq 0\}$. First observe that if $P = \emptyset$, then (5.5) is maximized by the empty-string λ. Thus, for now on assume that $P \neq \emptyset$. Since $\mathcal{C}_v$ is downward-closed, it suffices to consider those $\boldsymbol{e} \in \mathcal{C}_v$ whose edges all lie in P. As such, for notational convenience, let us hereby assume that $\partial(v) = P$.

For any $\boldsymbol{e} \in \mathcal{C}_v$, let $\boldsymbol{e}^r$ be the rearrangement of $\boldsymbol{e}$, based on the non-increasing order of the weights $(w_e)_{e \in \boldsymbol{e}}$. Since $\mathcal{C}_v$ is downward-closed, we know that $\boldsymbol{e}^r$ is also in $\mathcal{C}_v$. Moreover, $\text{val}(\boldsymbol{e}^r) \geq \text{val}(\boldsymbol{e})$ (following observations in [35, 14]). Hence, let us order the edges of $\partial(v)$ as $e_1, \ldots, e_m$, such that $w_{e_1} \geq \ldots \geq w_{e_m}$, where $m := |\partial(s)|$. Observe then that it suffices to maximize (5.5) over those strings within $\mathcal{C}_v$ which respect this ordering on $\partial(s)$. Stated differently, let us denote $\mathcal{I}_v$ as the family of subsets of $\partial(v)$ induced by $\mathcal{C}_v$, and define the set function $f : 2^{\partial(v)} \rightarrow [0, \infty)$, where $f(B) := \text{val}(\boldsymbol{b})$ for $B = \{b_1, \ldots, b_{|B|}\} \subseteq \partial(v)$, such that $\boldsymbol{b} = (b_1, \ldots, b_{|B|})$ and $w_{b_1} \geq \ldots \geq w_{b_{|B|}}$. Our goal is then to efficiently maximize f over the set-system $(\partial(v), \mathcal{I}_v)$. Observe that $\mathcal{I}_v$ is downward-closed and that we can simulate oracle access to $\mathcal{I}_v$, based on our oracle access to $\mathcal{C}_v$.

For each $i = 0, \ldots, m-1$, denote $\partial(v)^{>i} := \{e_{i+1}, \ldots, e_m\}$, and $\partial(v)^{>m} := \emptyset$. Moreover, define the family of subsets $\mathcal{I}_v^{>i} := \{B \subseteq \partial(v)^{>i} : B \cup \{e_i\} \in \mathcal{I}_v\}$ for each $1 \leq i \leq m$, and $\mathcal{I}_v^{>0} := \mathcal{I}_v$. Observe then that $(\partial(v)^{>i}, \mathcal{I}_v^{>i})$ is a downward-closed set system, as $\mathcal{I}_v$ is downward-closed. Moreover, we may simulate oracle access to $\mathcal{I}_v^{>i}$ based on our oracle access to $\mathcal{I}_v$.

Denote $\text{OPT}(\mathcal{I}_v^{>i})$ as the maximum value of f over constraints $\mathcal{I}_v^{>i}$. Observe then that for each $0 \leq i \leq m-1$, the following recursion holds:

$$\text{OPT}(\mathcal{I}_v^{>i}) := \max_{j \in \{i+1, \ldots, m\}} (p_{e_j} \cdot w_{e_j} + (1 - p_{e_j}) \cdot \text{OPT}(\mathcal{I}_v^{>j})) \tag{5.7}$$

Hence, given access to the values $\text{OPT}(\mathcal{I}_v^{>i+1}), \ldots, \text{OPT}(\mathcal{I}_v^{>m})$, we can compute $\text{OPT}(\mathcal{I}_v^{>i})$ efficiently. Moreover, $\text{OPT}(\mathcal{I}_v^{>m}) = 0$ by definition. Thus, it is clear that we can use (5.7) to recover an optimal solution to f, and so the proof is complete. □

We conclude the section by noting that if we are instead given a known i.d. input $(H_{\text{typ}}, (\mathcal{D}_i)_{i=1}^n)$, then LP-config-id can be solved in time $\text{poly}(|H_{\text{typ}}|, (|\mathcal{D}_i|)_{i=1}^n)$ using the same strategy, as the same maximization

[6]See Eden et al. [22] for a buyer/seller interpretation of the classical Ranking algorithm [31] for bipartite matching.

[7]The title of the conference version [10] differs from that of the arXiv version [9].

problem (5.5) is needed to separate the dual of LP-config-id. The efficiency of Algorithms 9 and 10 thereby follows, as claimed in Theorem 2.5.

6 A Tight Adaptivity Gap

Similar to the definition of the adaptive benchmark, we define the **non-adaptive benchmark** as the optimum performance of a non-adaptive probing algorithm on G. That is, $\text{OPT}_{\text{n-adap}}(G) := \sup_{\mathcal{B}} \mathbb{E}[w(\mathcal{B}(G))]$, where the supremum is over all offline non-adaptive probing algorithms. The upper bound (negative result) of Theorem 2.9 can thus be viewed a statement regarding the power of adaptivity. More precisely, we define the **adaptivity gap** of the **stochastic matching problem with one-sided probing constraints**, as the ratio

$$\inf_{G} \frac{\text{OPT}_{\text{n-adap}}(G)}{\text{OPT}(G)}, \tag{6.1}$$

where the infimum is over all (bipartite) stochastic graphs $G = (U, V, E)$ with **substring-closed** probing constraints $(\mathcal{C}_v)_{v \in V}$. Here $\mathcal{C}_v$ is closed under substrings if any substring of $\boldsymbol{e} \in \mathcal{C}_v$ is also in $\mathcal{C}_v$. This is a less restrictive definition than imposing $\mathcal{C}_v$ must be downward-closed, and is the minimal assumption one needs to ensure stochastic matching with commitment is well-defined.

We can therefore restate Theorem 2.9 in the following terminology:

Theorem 6.1. *The adaptivity gap of the stochastic matching problem with one-sided probing constraints is no smaller than* $1 - 1/e$.

Theorem 6.1 follows by considering a sequence of stochastic graphs. In particular, given $n \geq 1$, consider functions $p = p(n)$ and $s = s(n)$ which satisfy the following:

1. $p \ll 1/\sqrt{n}$ and $s \to \infty$ as $n \to \infty$.
2. $s \leq pn$ and $s = (1 - o(1))pn$.

Consider now an unweighted stochastic graph $G_n = (U, V, E)$ with unit patience values, and which satisfies $|U| = s$ and $|V| = n$. Moreover, assume that $p_{u,v} = p$ for all $u \in U$ and $v \in V$. Observe that G_n corresponds to the bipartite Erdős–Rényi random graph $\mathbb{G}(s, n, p)$.

Lemma 6.2. *The adaptive benchmark returns a matching of size asymptotically equal to s when executing on G_n; that is, $OPT(G_n) = (1 + o(1))s$.*

We omit the proof of Lemma 6.2, as it is routine analysis of the Erdős–Rényi random graph $\mathbb{G}(s, n, p)$. Instead, we focus on proving the following lemma, which together with Lemma 6.2 implies the upper bound of Theorem 6.1:

Lemma 6.3. *The non-adaptive benchmark returns in expectation a matching of size at most* $(1+o(1))\left(1 - \frac{1}{e}\right) s$ *when executing on G_n. That is,*

$$OPT_{n\text{-}adp}(G) \leq (1 + o(1)) \left(1 - \frac{1}{e}\right) s.$$

Proof. Let $\mathcal{A}$ be a non-adaptive probing algorithm, which we may assume is deterministic without loss of generality. As the probes of $\mathcal{A}$ are determined independently of the random variables $(\text{st}(e))_{e \in E}$, we can define $x_e \in \{0, 1\}$ for each $e \in E$ to indicate whether or not $\mathcal{A}$ probes the edge e.

Now, if $\mathcal{A}(G)$ is the matching returned by $\mathcal{A}$, then using the independence of the edge states $(\text{st}(e))_{e \in E}$, we get that

$$\mathbb{P}[u \text{ matched by } \mathcal{A}(G)] = \mathbb{P}\left[\cup_{\substack{v \in V: \\ x_{u,v} = 1}} \text{st}(u, v) = 1\right] \tag{6.2}$$

$$\geq 1 - \prod_{v \in V} (1 - p x_{u,v}) \tag{6.3}$$

and so,

$$\mathbb{E}[|\mathcal{A}(G))|] \leq s - \sum_{u \in U} \prod_{v \in V} (1 - p x_{u,v}).$$

As such, if we can show that

$$\sum_{u \in U} \prod_{v \in V} (1 - p x_{u,v}) \geq (1 - o(1)) \frac{s}{e},$$

then this will imply that

$$\mathbb{E}[|\mathcal{A}(G)|] \leq (1 + o(1)) \left(1 - \frac{1}{e}\right) s.$$

To see this, first observe that since $p(n) \to 0$ as $n \to \infty$, we know that

$$1 - p x_{u,v} = (1 + o(1)) \exp(-p x_{u,v})$$

for each $v \in V$. In fact, since $p x_{u,v} \leq p$ for all $v \in V$, the asymptotics are uniform across V. More precisely, there exists $C > 0$, such that for n sufficiently large,

$$1 - p x_{u,v} \geq (1 - C p^2) \exp(-p x_{u,v})$$

for all $v \in V$. As a result,

$$\begin{aligned} \prod_{v \in V} (1 - p x_{u,v}) &\geq (1 - C p^2)^n \exp\left(-\sum_{v \in V} p x_{u,v}\right) \\ &= (1 + o(1)) \exp\left(-\sum_{v \in V} p x_{u,v}\right), \end{aligned}$$

where the second line follows since $p \ll 1/\sqrt{n}$ by assumption. On the other hand, Jensen's inequality ensures that

$$\sum_{u \in U} \frac{\exp\left(-\sum_{v \in V} p x_{u,v}\right)}{s} \geq \exp\left(-\frac{\sum_{u \in U, v \in V} p x_{u,v}}{n}\right).$$

However, $\sum_{u \in U} x_{u,v} \leq 1$ for each $v \in V$. Thus, $\sum_{u \in U, v \in V} p x_{u,v} \leq pn$, and so

$$\exp\left(-\frac{\sum_{u \in U, v \in V} p x_{u,v}}{s}\right) \geq \exp\left(-\frac{pn}{s}\right) \geq \frac{1}{e},$$

where the last line follows since $pn \leq s$. It follows that

$$\sum_{u \in U} \prod_{v \in V} (1 - p x_{u,v}) \geq (1 + o(1)) \frac{s}{e},$$

and so

$$\mathbb{E}[|\mathcal{A}(G)|] \leq (1 + o(1)) \left(1 - \frac{1}{e}\right) s.$$

As the asymptotics hold uniformly across each deterministic non-adaptive algorithm $\mathcal{A}$, this completes the proof.

□

Note that the competitive ratio of Corollary 2.7 in fact holds whenever the stochastic graph has substring-closed probing constraints. The stronger downward closed condition is only needed to ensure the efficiency of Algorithm 7. Thus, Corollary 2.7 and Theorem 6.1 exactly characterize the adaptivity gap of the stochastic matching problem with one-sided probing constraints:

Corollary 6.4. *The adaptivity gap of the stochastic matching problem with one-sided probing constraints is* $1 - 1/e$.

7 Conclusion and open problems

We have considered the stochastic bipartite matching problem (with probing constraints) in a few settings. As discussed, our results generalize the prophet inequality and prophet secretary matching problems. Our algorithms are polynomial time assuming a mild assumption on the probing constraints which, in particular, generalizes the standard patience constraints.

There are some basic questions that are unresolved. Perhaps the most basic question which is also unresolved in the classical setting is to bridge the gap between the positive $1 - 1/e$ competitive ratio and inapproximations in the context of known i.d. random order arrivals. In terms of the single item prophet secretary problem (without probing), Correa et al. [20] obtain a 0.669 competitive ratio following Azar et al. [5] who were the first to surpass the $1-1/e$ "barrier". Correa et al. [20] also establish a .732 inapproximation for the i.d. setting. Our adaptivity gap proves the optimality of the $1-1/e$ competitive ratio for non-adaptive algorithms. Can we surpass $1 - 1/e$ in the probing setting for i.d. input arrivals or for the the special case of i.i.d. input arrivals? Is there a provable difference between stochastic bipartite matching (with probing constraints) and the classical online settings? Can we obtain the same competitive results against an optimal offline *non-committal* benchmark which respects the probing constraints but not the commitment constraint.

One interesting extension of the probing model is to allow non-Bernoulli edge random variables to describe edge uncertainty. Even for a single online vertex with full patience, this problem is interesting and has been studied significantly less (see, ProblemMax in Segev and Singla [37]). A general understanding of edge uncertainty suggests a possible relation between stochastic probing and online algorithms with ML (untrusted) advice (see, for example, Lavastida et al. [32]).

References

[1] Marek Adamczyk, Fabrizio Grandoni, Stefano Leonardi, and Michal Wlodarczyk. When the optimum is also blind: a new perspective on universal optimization. In *ICALP*, 2017.

[2] Marek Adamczyk, Fabrizio Grandoni, and Joydeep Mukherjee. Improved approximation algorithms for stochastic matching. In Nikhil Bansal and Irene Finocchi, editors, *Algorithms - ESA 2015 - 23rd Annual European Symposium, Patras, Greece, September 14-16, 2015, Proceedings*, volume 9294 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2015.

[3] Marek Adamczyk and Michał Włodarczyk. Random order contention resolution schemes. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 790–801. IEEE, 2018.

[4] Saeed Alaei, MohammadTaghi Hajiaghayi, and Vahid Liaghat. Online prophet-inequality matching with applications to ad allocation. In *Proceedings of the 13th ACM Conference on Electronic Commerce*, EC '12, page 18–35, New York, NY, USA, 2012. Association for Computing Machinery.

[5] Yossi Azar, Ashish Chiplunkar, and Haim Kaplan. Prophet secretary: Surpassing the 1-1/e barrier. In Éva Tardos, Edith Elkind, and Rakesh Vohra, editors, *Proceedings of the 2018 ACM Conference on Economics and Computation, Ithaca, NY, USA, June 18-22, 2018*, pages 303–318. ACM, 2018.

[6] Nikhil Bansal, Anupam Gupta, Jian Li, Julián Mestre, Viswanath Nagarajan, and Atri Rudra. When LP is the cure for your matching woes: Improved bounds for stochastic matchings. *Algorithmica*, 63(4):733–762, 2012.

[7] Alok Baveja, Amit Chavan, Andrei Nikiforov, Aravind Srinivasan, and Pan Xu. Improved bounds in stochastic matching and optimization. *Algorithmica*, 80(11):3225–3252, Nov 2018.

[8] Allan Borodin, Calum MacRury, and Akash Rakheja. Bipartite stochastic matching: Online, random order, and i.i.d. models. *CoRR*, abs/2004.14304, 2020.

[9] Allan Borodin, Calum MacRury, and Akash Rakheja. Greedy approaches to online stochastic matching. *CoRR*, abs/2008.09260, 2021.

[10] Allan Borodin, Calum MacRury, and Akash Rakheja. Secretary matching meets probing with commitment. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2021, August 16-18, 2021, Virtual Conference*, 2021.

[11] Brian Brubach, Nathaniel Grammel, Will Ma, and Aravind Srinivasan. Follow your star: New frameworks for online stochastic matching with known and unknown patience. *CoRR*, abs/1907.03963, 2021.

[12] Brian Brubach, Nathaniel Grammel, Will Ma, and Aravind Srinivasan. Follow your star: New frameworks for online stochastic matching with known and unknown patience. In Arindam Banerjee and Kenji Fukumizu, editors, *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pages 2872–2880. PMLR, 13–15 Apr 2021.

[13] Brian Brubach, Nathaniel Grammel, Will Ma, and Aravind Srinivasan. Improved guarantees for offline stochastic matching via new ordered contention resolution schemes. *CoRR*, abs/2106.06892, 2021.

[14] Brian Brubach, Nathaniel Grammel, and Aravind Srinivasan. Vertex-weighted online stochastic matching with patience constraints. *CoRR*, abs/1907.03963, 2019.

[15] Brian Brubach, Karthik Abinav Sankararaman, Aravind Srinivasan, and Pan Xu. New algorithms, better bounds, and a novel model for online stochastic matching. In *24th Annual European Symposium on Algorithms, ESA 2016, August 22-24, 2016, Aarhus, Denmark*, pages 24:1–24:16, 2016.

[16] Brian Brubach, Karthik Abinav Sankararaman, Aravind Srinivasan, and Pan Xu. Attenuate locally, win globally: Attenuation-based frameworks for online stochastic matching with timeouts. *Algorithmica*, 82(1):64–87, 2020.

[17] Shuchi Chawla, Jason D. Hartline, David L. Malec, and Balasubramanian Sivan. Multi-parameter mechanism design and sequential posted pricing. In Leonard J. Schulman, editor, *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*, pages 311–320. ACM, 2010.

[18] Ning Chen, Nicole Immorlica, Anna R. Karlin, Mohammad Mahdian, and Atri Rudra. Approximating matches made in heaven. In *Proceedings of the 36th International Colloquium on Automata, Languages and Programming: Part I*, ICALP '09, pages 266–278, 2009.

[19] José R. Correa, Patricio Foncea, Dana Pizarro, and Victor Verdugo. From pricing to prophets, and back! *Oper. Res. Lett.*, 47(1):25–29, 2019.

[20] José R. Correa, Raimundo Saona, and Bruno Ziliotto. Prophet secretary through blind strategies. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 1946–1961, 2019.

[21] Kevin P. Costello, Prasad Tetali, and Pushkar Tripathi. Stochastic matching with commitment. In Artur Czumaj, Kurt Mehlhorn, Andrew Pitts, and Roger Wattenhofer, editors, *Automata, Languages, and Programming*, pages 822–833, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[22] Alon Eden, Michal Feldman, Amos Fiat, and Kineret Segal. An economic-based analysis of RANKING for online bipartite matching. *CoRR*, abs/1804.06637, 2018.

[23] Soheil Ehsani, MohammadTaghi Hajiaghayi, Thomas Kesselheim, and Sahil Singla. Prophet secretary for combinatorial auctions and matroids. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '18, page 700–714, USA, 2018. Society for Industrial and Applied Mathematics.

[24] Tomer Ezra, Michal Feldman, Nick Gravin, and Zhihao Gavin Tang. Online stochastic max-weight matching: Prophet inequality for vertex and edge arrival models. In *Proceedings of the 21st ACM Conference on Economics and Computation*, EC '20, page 769–787, New York, NY, USA, 2020. Association for Computing Machinery.

[25] Moran Feldman, Ola Svensson, and Rico Zenklusen. Online contention resolution schemes. In Robert Krauthgamer, editor, *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 1014–1033. SIAM, 2016.

[26] Buddhima Gamlath, Sagar Kale, and Ola Svensson. Beating greedy for stochastic bipartite matching. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '19, page 2841–2854, USA, 2019. Society for Industrial and Applied Mathematics.

[27] Rajiv Gandhi, Samir Khuller, Srinivasan Parthasarathy, and Aravind Srinivasan. Dependent rounding and its applications to approximation algorithms. *J. ACM*, 53(3):324–360, May 2006.

[28] Bernd Gärtner and Jirí Matousek. *Understanding and using linear programming.* Universitext. Springer, 2007.

[29] Anupam Gupta, Viswanath Nagarajan, and Sahil Singla. Algorithms and adaptivity gaps for stochastic probing. In Robert Krauthgamer, editor, *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 1731–1747. SIAM, 2016.

[30] Mohammad Taghi Hajiaghayi, Robert D. Kleinberg, and Tuomas Sandholm. Automated online mechanism design and prophet inequalities. In *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence, July 22-26, 2007, Vancouver, British Columbia, Canada*, pages 58–65. AAAI Press, 2007.

[31] Richard M. Karp, Umesh V. Vazirani, and Vijay V. Vazirani. An optimal algorithm for on-line bipartite matching. In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, May 13-17, 1990, Baltimore, Maryland, USA*, pages 352–358, 1990.

[32] Thomas Lavastida, Benjamin Moseley, R. Ravi, and Chenyang Xu. Learnable and instance-robust predictions for online matching, flows and load balancing. *CoRR*, abs/2011.11743, 2020.

[33] Euiwoong Lee and Sahil Singla. Optimal Online Contention Resolution Schemes via Ex-Ante Prophet Inequalities. In Yossi Azar, Hannah Bast, and Grzegorz Herman, editors, *26th Annual European Symposium on Algorithms (ESA 2018)*, volume 112 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 57:1–57:14, Dagstuhl, Germany, 2018. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[34] Vahideh H. Manshadi, Shayan Oveis Gharan, and Amin Saberi. Online stochastic matching: Online actions based on offline statistics. *Math. Oper. Res.*, 37(4):559–573, 2012.

[35] Manish Purohit, Sreenivas Gollapudi, and Manish Raghavan. Hiring under uncertainty. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5181–5189. PMLR, 09–15 Jun 2019.

[36] D. Seese. Groetschel, m., l. lovasz, a. schrijver: Geometric algorithms and combinatorial optimization. (algorithms and combinatorics. eds.: R. l. graham, b. korte, l. lovasz. vol. 2), springer-verlag 1988, xii, 362 pp., 23 figs., dm 148,-. isbn 3–540–13624-x. *Biometrical Journal*, 32(8):930–930, 1990.

[37] Danny Segev and Sahil Singla. Efficient approximation schemes for stochastic probing and prophet problems. In *Proceedings of the 22nd ACM Conference on Economics and Computation*, EC '21, page 793–794, New York, NY, USA, 2021. Association for Computing Machinery.

[38] Jan Vondrák, Chandra Chekuri, and Rico Zenklusen. Submodular function maximization via the multilinear relaxation and contention resolution schemes. In *Proceedings of the Forty-Third Annual ACM Symposium on Theory of Computing*, STOC '11, page 783–792, New York, NY, USA, 2011. Association for Computing Machinery.

[39] David P. Williamson and David B. Shmoys. *The Design of Approximation Algorithms.* Cambridge University Press, USA, 1st edition, 2011.

A LP Relations

Suppose that we are given an arbitrary stochastic graph $G = (U, V, E)$. In this section, we state LP-std, the standard LP in the stochastic matching literature, as introduced by Bansal et al. [6], as well as LP-QC, the LP introduced by Gamlath et al. [26]. We then show that LP-QC and LP-config have the same optimum value when G has unbounded patience.

Consider LP-std, which is defined only when G has patience values $(\ell_v)_{v \in V}$. Here each $e \in E$ has a variable x_e corresponding to the probability that the adaptive benchmark probes e.

$$\begin{aligned}
\text{maximize} \quad & \sum_{e \in E} w_e \cdot p_e \cdot x_e && && \text{(LP-std)}\\
\text{subject to} \quad & \sum_{e\partial(u)} p_e \cdot x_e \leq 1 && \forall u \in U && \text{(A.1)}\\
& \sum_{e \in \partial(v)} p_e \cdot x_e \leq 1 && \forall v \in V && \text{(A.2)}\\
& \sum_{e \in \partial(v)} x_e \leq \ell_v && \forall v \in V && \text{(A.3)}\\
& 0 \leq x_e \leq 1 && \forall e \in E. && \text{(A.4)}
\end{aligned}$$

Gamlath et al. modified LP-std for the special case of unbounded patience by adding in exponentially many extra constraints. Specifically, for each $v \in V$ and $S \subseteq \partial(v)$, they ensure that

$$\sum_{e \in S} p_e \cdot x_e \leq 1 - \prod_{e \in S} (1 - p_e), \tag{A.5}$$

In the same variable interpretation as LP-std, the left-hand side of (A.5) corresponds to the probability the adaptive benchmark matches an edge of $S \subseteq \partial(v)$, and the right-hand side corresponds to the probability an edge of S is active[8].

$$\begin{aligned}
\text{maximize} \quad & \sum_{e \in E} w_e \cdot p_e \cdot x_e && && \text{(LP-QC)}\\
\text{subject to} \quad & \sum_{e \in S} p_e \cdot x_e \leq 1 - \prod_{e \in S} (1 - p_e) && \forall v \in V, S \subseteq \partial(v) && \text{(A.6)}\\
& \sum_{e \in \partial(u)} p_e \cdot x_e \leq 1 && \forall u \in U && \text{(A.7)}\\
& x_e \geq 0 && \forall e \in E. && \text{(A.8)}
\end{aligned}$$

Let us denote $\text{LPOPT}_{\text{QC}}(G)$ as the optimum value of LP-QC.

Proposition A.1. *If G has unbounded patience, then $LPOPT_{QC}(G) = LPOPT(G)$.*

In order to prove Proposition A.1, we make use of a result of Gamlath et al. We mention that an almost identical result is also proven by Costello et al. [21] using different techniques.

Theorem A.2 ([26])**.** *Suppose that $G = (U, V, E)$ is a stochastic graph with unbounded patience, and $(x_e)_{e \in E}$ is a solution to LP-QC. For each $v \in V$, there exists an online probing algorithm $\mathcal{B}_v$ whose input is $(v, \partial(v), (x_e)_{e \in \partial(v)})$, and which satisfies $\mathbb{P}[\mathcal{B}_v$ matches v to $e] = p_e x_e$ for each $e \in \partial(v)$.*

Proof of Proposition A.1. Observe that by Theorem 3.3, in order to prove the claim it suffices to show that $\text{LPOPT}_{\text{QC}}(G) = \text{OPT}_{\text{rel}}(G)$. Clearly, $\text{OPT}_{\text{rel}}(G) \leq \text{LPOPT}_{\text{QC}}(G)$, as can be seen by defining x_e as the probability that the relaxed benchmark probes the edge $e \in E$. Thus, we focus on showing that $\text{LPOPT}_{\text{QC}}(G) \leq \text{OPT}_{\text{rel}}(G)$.

Suppose that $(x_e)_{e \in E}$ is an optimum solution to $\text{LPOPT}_{\text{QC}}(G)$. We design the following algorithm, which we denote by $\mathcal{B}$:

[8]The LP considered by Gamlath et al. in [26] also places the analogous constraints of (A.5) on the vertices of U. That being said, these additional constraints are not used anywhere in the work of Gamlath et al., so we omit them.

1. $\mathcal{M} \leftarrow \emptyset$.
2. For each $v \in V$, execute $\mathcal{B}_v$ on $(v, \partial(v), (x_e)_{e \in \partial(v)})$, where $\mathcal{B}_v$ is the online probing algorithm of Theorem A.2. If $\mathcal{B}_v$ matches v, then let e' be this edge, and add e' to $\mathcal{M}$
3. Return $\mathcal{M}$.

Using Theorem A.2, it is clear that
$$\mathbb{E}[w(\mathcal{M})] = \sum_{e \in E} w_e p_e x_e.$$
Moreover, each vertex $u \in U$ is matched by $\mathcal{M}$ at most once in expectation, as a consequence of constraint (A.8). As a result, $\mathcal{B}$ is a relaxed probing algorithm. Thus, $\text{LPOPT}_{\text{QC}}(G) = \sum_{e \in E} w_e p_e x_e \leq \text{OPT}_{\text{rel}}(G)$, and so the proof is complete. $\square$

B Section 4 Additions

Proof of Proposition 4.4. Given $u \in U$, let $\mathcal{M}(u)$ denote the edge matched to u by $\mathcal{M}$, where $\mathcal{M}(u) := \emptyset$ if no such edge exists. Observe now that if $C(e)$ corresponds to the event in which VertexProbe commits to $e \in \partial(u)$, then $\mathbb{P}[C(e)] = p_e \widetilde{x}_e$ by Lemma 2.12. Moreover, the events $(C(e))_{e \in \partial(u)}$ are independent, and satisfy
$$\sum_{e \in \partial(u)} \mathbb{P}[C(e)] = \sum_{e \in \partial(u)} p_e \widetilde{x}_e \leq 1, \tag{B.1}$$
by constraint (2.1) of LP-config. As such, denote $\boldsymbol{z} := (z_e)_{e \in \partial(u)}$ where $z_e := p_e \widetilde{x}_e$, and observe that (B.1) ensures that $\boldsymbol{z} \in \mathcal{P}$, where $\mathcal{P}$ is the convex relaxation of the rank 1 matroid on $\partial(u)$. Let us denote $R(\boldsymbol{z})$ as those those $e \in \partial(u)$ for which $C(e)$ occurs.

For each $e = (u, v) \in \partial(u)$, define $Y_{u,v} := \widetilde{Y}_v$. Observe then that the random variables $(Y_e)_{e \in \partial(u)}$ are independent and drawn *u.a.r.* from $[0, 1]$. Thus, if ψ is the RCRS defined in Algorithm 6, then we may pass $\boldsymbol{z}$ to ψ, and process the edges of $\partial(u)$ in non-increasing order based on $(Y_e)_{e \in \partial(u)}$. Denote the resulting output by $\psi_{\boldsymbol{z}}(R(\boldsymbol{z}))$. By coupling the random draws of lines (3) and (6) of Algorithms 6 and 7, respectively, we get that
$$w(\mathcal{M}(u)) = \sum_{e \in \partial(u)} w_e \cdot \mathbf{1}_{[e \in R(\boldsymbol{z})]} \cdot \mathbf{1}_{[e \in \psi_{\boldsymbol{z}}(R(\boldsymbol{z}))]}$$
Thus, after taking expectations,
$$\mathbb{E}[w(\mathcal{M}(u))] = \sum_{e \in \partial(u)} w_e \cdot \mathbb{P}[e \in \psi_{\boldsymbol{z}}(R(\boldsymbol{z})) \,|\, e \in R(\boldsymbol{z})] \cdot \mathbb{P}[e \in R(\boldsymbol{z})].$$
Now, Theorem 4.3 ensures that for each $e \in \partial(u)$, $\mathbb{P}[e \in \psi_{\boldsymbol{z}}(R(\boldsymbol{z})) \,|\, e \in R(\boldsymbol{z})] \geq \left(1 - \frac{1}{e}\right)$. It follows that $\mathbb{E}[w(\mathcal{M}(u))] \geq \left(1 - \frac{1}{e}\right) \sum_{e \in \partial(u)} w_e p_e \widetilde{x}_e$, for each $u \in U$. Thus,
$$\begin{aligned}\mathbb{E}[w(\mathcal{M})] &= \sum_{u \in U} \mathbb{E}[w(\mathcal{M}(u))] \\ &\geq \left(1 - \frac{1}{e}\right) \sum_{e \in E} w_e p_e \widetilde{x}_e = \left(1 - \frac{1}{e}\right) \text{LPOPT}(G),\end{aligned}$$
where the equality follows since $(x_v(\boldsymbol{e}))_{v \in V, \boldsymbol{e} \in \mathcal{C}_v}$ is an optimum solution to LP-config. On the other hand, $\text{LPOPT}(G) \geq \text{OPT}(G)$ by Theorem 2.10, and so the proof is complete.

$\square$

Proof of Theorem 4.5. Suppose that $(H_{\text{typ}}, (\mathcal{D}_t)_{t=1}^n)$ is a known i.d. instance, where $H_{\text{typ}} = (U, B, F)$. Recall that $\mathcal{C}_b$ corresponds to the online probing constraint of each type node $b \in B$. For convenience, we denote $\mathcal{I} := \sqcup_{b \in B} \mathcal{C}_b$. We can then define the following collection of random variables, denoted $(X_t(\boldsymbol{e}))_{t \in [n], \boldsymbol{e} \in \mathcal{I}}$, based on the following randomized procedure:

- Draw the instantiated graph $G \sim (H_{\text{typ}}, (\mathcal{D}_t)_{t=1}^n)$, whose vertex arrivals we denote by $v_1, \ldots, v_n$.
- Compute an optimum solution of LP-config for G, which we denote by $(x_{v_t}(\boldsymbol{e}))_{t \in [n], \boldsymbol{e} \in \mathcal{C}_{v_t}}$.
- For each $t = 1, \ldots, n$ and $\boldsymbol{e} \in \mathcal{I}$, set $X_t(\boldsymbol{e}) = x_{v_t}(\boldsymbol{e})$ if $\boldsymbol{e} \in \mathcal{C}_{v_t}$, otherwise set $X_t(\boldsymbol{e}) = 0$.

Observe then that since by definition $(X_{v_t}(\boldsymbol{e}))_{t \in [n], \boldsymbol{e} \in \mathcal{C}_{v_t}}$ is a feasible solution to LP-config for G, it holds that for each $t = 1, \ldots, n$

$$\sum_{\boldsymbol{e} \in \mathcal{I}} X_t(\boldsymbol{e}) = 1, \tag{B.2}$$

and

$$\sum_{t \in [n], b \in B} \sum_{\substack{\boldsymbol{e} \in \mathcal{I}: \\ (u,b) \in \boldsymbol{e}}} p_{u,b} \cdot g(\boldsymbol{e}_{<(u,b)}) \cdot X_t(\boldsymbol{e}) \leq 1, \tag{B.3}$$

for each $u \in U$. Moreover, $(X_t(\boldsymbol{e}))_{t \in [n], \boldsymbol{e} \in \mathcal{C}_{v_t}}$ is a optimum solution to LP-config for G, so Theorem 2.10 implies that

$$\text{OPT}(G) \leq \text{LPOPT}(G) = \sum_{t=1}^{n} \sum_{\boldsymbol{e} \in \mathcal{I}} \text{val}(\boldsymbol{e}) \cdot X_t(\boldsymbol{e}). \tag{B.4}$$

In order to make use of these inequalities in the context of the type graph H_{typ}, let us first fix a type node $b \in B$ and a string $\boldsymbol{e} \in \mathcal{C}_b$. For each $t \in [n]$, we can then define

$$x_t(\boldsymbol{e} \,||\, b) := \mathbb{E}[X_t(\boldsymbol{e}) \cdot \mathbf{1}_{[v_t = b]}], \tag{B.5}$$

where the randomness is over the generation of G. Observe that by definition of the $(X_t(\boldsymbol{e}))_{t \in [n], \boldsymbol{e} \in \mathcal{I}}$ values,

$$x_t(\boldsymbol{e} \,||\, b) = 0,$$

provided $\boldsymbol{e} \notin \mathcal{C}_b$. We claim that $(x_t(\boldsymbol{e} \,||\, b))_{b \in B, t \in [n], \boldsymbol{e} \in \mathcal{C}_b}$ is a feasible solution to LP-config-id. To see this, first observe that if we multiply (B.2) by the indicator random variable $\mathbf{1}_{[b_t = v]}$, then we get that

$$\sum_{\boldsymbol{e} \in \mathcal{I}} X_t(\boldsymbol{e}) \cdot \mathbf{1}_{[v_t = b]} = \mathbf{1}_{[v_t = b]}.$$

As a result, if we take expectations over this equality,

$$\begin{aligned}\sum_{\boldsymbol{e} \in \mathcal{I}} x_t(\boldsymbol{e} \,||\, b) &= \sum_{\boldsymbol{e} \in \mathcal{I}} \mathbb{E}\left[X_t(\boldsymbol{e}) \cdot \mathbf{1}_{[v_t = b]}\right] \\ &= \mathbb{P}[v_t = b] \\ &=: r_t(b),\end{aligned}$$

for each $b \in B$ and $t \in [n]$. Let us now fix $u \in U$. Observe that since $X_t(\boldsymbol{e}) \cdot \mathbf{1}_{[v_t = b]} = X_t(\boldsymbol{e})$ for each $\boldsymbol{e} \in \mathcal{C}_b$, (B.3) ensures that

$$\sum_{t \in [n], b \in B} \sum_{\substack{\boldsymbol{e} \in \mathcal{C}_b: \\ (u,b) \in \boldsymbol{e}}} p_{u,b} \cdot g(\boldsymbol{e}_{<(u,b)}) \cdot X_t(\boldsymbol{e}) \cdot \mathbf{1}_{[v_t = b]} = \sum_{t \in [n], b \in B} \sum_{\substack{\boldsymbol{e} \in \mathcal{C}_b: \\ (u,b) \in \boldsymbol{e}}} p_{u,b} \cdot g(\boldsymbol{e}_{<(u,b)}) \cdot X_t(\boldsymbol{e}) \leq 1 \tag{B.6}$$

Thus, after taking expectations over (B.6),

$$\sum_{t \in [n], b \in B} \sum_{\substack{\boldsymbol{e} \in \mathcal{C}_b: \\ (u,b) \in \boldsymbol{e}}} p_{u,b} \cdot g(\boldsymbol{e}_{<(u,b)}) \cdot x_t(\boldsymbol{e} \,||\, b) \leq 1,$$

for each $u \in U$. Since $(x_t(\boldsymbol{e} \,||\, b))_{t \in [n], b \in B, \boldsymbol{e} \in \mathcal{C}_b}$ satisfies these inequalities, and the variables are clearly all non-negative, it follows that $(x_t(\boldsymbol{e} \,||\, b))_{t \in [n], b \in B, \boldsymbol{e} \in \mathcal{C}_b}$ is a feasible solution to LP-config-id. Let us now express the right-hand side of (B.4) as in (B.6) and take expectations. We then get that

$$\mathbb{E}[\text{OPT}(G)] \leq \sum_{b \in B, t \in [n]} \sum_{\boldsymbol{e} \in \mathcal{I}} \text{val}(\boldsymbol{e}) \cdot x_t(\boldsymbol{e} \,||\, b).$$

Now, $\mathrm{OPT}(H_{\mathrm{typ}}, (\mathcal{D}_i)_{i=1}^n = \mathbb{E}[\mathrm{OPT}(G)]$ by definition, so since $(x_t(\boldsymbol{e} \,||\, b))_{b \in B, t \in [n], \boldsymbol{e} \in \mathcal{C}_b}$ is feasible, it holds that

$$\mathrm{OPT}(H_{\mathrm{typ}}, (\mathcal{D}_i)_{i=1}^n) \leq \mathrm{LPOPT}_{new-id}(H_{\mathrm{typ}}, (\mathcal{D}_i)_{i=1}^n),$$

thus completing the proof.

□