Random Order Vertex Arrival Contention Resolution Schemes for Matching, with Applications

Hu Fu
ITCS, Shanghai University of Finance and Economics, China

Zhihao Gavin Tang
ITCS, Shanghai University of Finance and Economics, China

Hongxun Wu
IIIS, Tsinghua University, Beijing, China

Jinzhao Wu
Peking University, Beijing, China

Qianfan Zhang
IIIS, Tsinghua University, Beijing, China

Abstract

With a wide range of applications, stochastic matching problems have been studied in different models, including prophet inequality, Query-Commit, and Price-of-Information. While there have been recent breakthroughs in all these settings for bipartite graphs, few non-trivial results are known for general graphs.

In this paper, we study the random order vertex arrival contention resolution scheme for matching in general graphs, which is inspired by the recent work of Ezra et al. (EC 2020). We design an $\frac{8}{15}$-selectable batched RCRS for matching and apply it to achieve $\frac{8}{15}$-competitive/approximate algorithms for all the three models. Our results are the first non-trivial results for random order prophet matching and Price-of-Information matching in general graphs. For the Query-Commit model, our result substantially improves upon the 0.501 approximation ratio by Tang et al. (STOC 2020). We also show that no batched RCRS for matching can be better than $\frac{1}{2} + \frac{1}{2e^2} \approx 0.567$-selectable.

2012 ACM Subject Classification Theory of computation → Online algorithms

Keywords and phrases Matching, Contention Resolution Scheme, Price of Information, Query-Commit

Digital Object Identifier 10.4230/LIPIcs.ICALP.2021.68

Category Track A: Algorithms, Complexity and Games

1 Introduction

Matching is a fundamental object of algorithmic studies, with wide applications. Its recent use in e-commerce often presents two general features: (i) one does not see the entire input graph in the beginning, but there is often prior stochastic information on each edge's value or presence, thanks to accumulated data from the past; depending on the application, the actual value/presence is revealed either according to some order or at the control of the matching algorithm; (ii) in either case, the matching algorithm must be online – that is, once an edge's information is revealed, the algorithm must make an irrevocable decision whether to include the edge in the matching. Examples of such applications include online advertisement [26, 10, 25], kidney exchange [9], and ride-sharing platforms [4, 19, 20]. Mathematically, this type of scenarios has been modeled as prophet inequality (e.g. [24]), Query-Commit [9], and Price of Information [27] problems, according to the way the information is revealed.

The Online Contention Resolution Scheme (OCRS), first proposed by Feldman et al. [14], is a powerful framework that has found applications in various online decision problems such as prophet inequalities [14, 13], oblivious posted pricing [7], and stochastic probing [17]. Ezra et al. [13] recently proposed a generalization of OCRS, that with batched arrivals, and used it to give prophet inequalities for matching in bipartite graphs when vertices on both sides arrive online.

Our main results are two-fold. We follow Ezra et al. and define a batched OCRS for matching in general graphs with vertex arrival, and give the first non-trivial such OCRS for random arrival order. We then give an alternative perspective on Gamlath et al. [15]'s recent progress for the Query-Commit and Price of Information problems on bipartite matching; viewing their algorithms as reductions to batched OCRS with random batched arrivals, we immediately obtain similar reductions and consequently the first non-trivial results for the Query-Commit and Price of Information problems for matching in general graphs. Before elaborating on these results, we give more background on the connection between contention resolution schemes and online decision problems, and motivate the batched OCRSs we study.

Contention Resolution Schemes (CRSs) are first proposed as rounding algorithms in submodular maximization [8]. Such an algorithm treats a fractional solution $\boldsymbol{x}$ as a product distribution, samples elements according to it so that each element i is *active* independently with probability x_i, and selects a feasible subset of active elements, guaranteeing that each element, when active, is selected with probability at least $\alpha > 0$, which in turn guarantees that the selected subset retains performance comparable in expectation to (typically at least α fraction of) that of $\boldsymbol{x}$. Such a CRS is said to be α-selectable.

The Online Contention Resolution Schemes (OCRSs) were introduced by Feldman et al. [14]. Its difference from a CRS is that the algorithm does not see the set of active elements upfront, but observes each element's status (of being active or not) one by one online as each element arrives. Intuitively, in an online stochastic decision problem, the expected performance of the offline optimal can be calculated as a fractional solution: a realization of the stochastic procedure gives rise to an optimal (integral) solution, and taking expectation over the stochastic procedure is equivalent to taking a convex combination of these solutions, yielding a fractional solution. An online algorithm, to be competitive, emulates the performance of the fractional solution but only sees partial realizations of the stochastic procedure in an online fashion. For example, in the *prophet inequality* problem, one has prior knowledge in the form of a distribution over the elements' values, but the online algorithm only sees an element's value when it arrives and has to decide immediately whether to keep the element in the solution.

From this perspective, while Feldman et al. [14]'s setup of OCRS neatly fits the needs of many online decision problems, for other problems it is natural to go beyond and, as Ezra et al. [13] did, to consider settings in which (i) the elements arrive not necessarily one by one but in batches, and (ii) the elements' being active may not be independent events (which treats the fractional solution as a product distribution) but correlated ones. Ezra et al. [13] termed such schemes *batched OCRSs*. The concrete choices in the batched OCRS should adapt to the underlying online decision problem being solved.

For example, in online matching, it is often interesting to consider vertex arrival rather than edge arrival – with vertex arrival, the vertices arrive one by one, and at the arrival of each vertex, all the edges connecting it to the vertices that have arrived are revealed in a batch (i.e., simultaneously), whereas with edge arrival, individual edges arrive one by one. Correlation in elements' status (of being active or not) arises naturally when such batch arrivals are allowed. For example, consider prophet inequalities in bipartite matching with vertex arrival: in a given graph $G = (U, V, E)$, each edge $e \in E$ has a value $v_e \geq 0$ drawn independently from a given distribution F_e, and the goal is to obtain a matching with maximum value. The offline (fractional) solution $\boldsymbol{x} \in [0, 1]^E$ has x_e representing the probability that edge e is in the maximum value matching. In the standard reduction from edge arrival prophet inequality to OCRS, when an element e arrives and reveals its value v_e, one (re)samples the values of all the other edges according to their distributions, and labels e active if v_e is high enough to make e in the maximum value matching among the sampled edges. Now with vertex arrival, when vertex v arrives, the information of all edges connecting v to the vertices that arrived before v is revealed. Let B_v denote this set of edges. It is natural for the reduction to sample values for all the other edges, and labels at most one edge in B_v as active if that edge is in the maximum value matching among the (re)sampled edges. Here, the status of the edges in B_v in the (batched) OCRS are naturally correlated, even though their values in the original prophet inequality problem are independent.[1]

The batched OCRS we study in this work precisely results from this reduction, when the vertices arrive in an order that is uniformly at random. To be precise, we are given a vector $\boldsymbol{x} \in [0, 1]^E$, such that for each vertex v, $\sum_{e\in\delta(v)} x_e \leq 1$, where $\delta(v)$ denotes the set of edges incident to v. Vertices arrive in an order uniformly at random; when vertex v arrives, the status of all edge in B_v are revealed: each edge $e \in B_v$ is active with probability x_e, but at most one of them is active. If an edge is active, we must decide immediately whether to select it into the solution. We must keep the set of selected edges a valid matching and, for as large a constant $\alpha > 0$ as possible, guarantee that any edge is selected with probability at least αx_e. We refer to such an algorithm an *α-selectable batched Random order Contention Resolution Scheme (RCRS)*.[2] The existence of a $\frac{1}{2}$-selectable batched RCRS is known, after all, Ezra et al. [13] constructed a simple $\frac{1}{2}$-selectable batched OCRS for arbitrary vertex arrival order. Their method is very similar to the $\frac{1}{2}$-selectable OCRS for rank 1 matroid due to Alaei [2]. On the contrary, we were unable to generalize existing RCRS (e.g., [23]) to a batched RCRS with a selectability better than $\frac{1}{2}$. Instead, we construct an $\frac{8}{15}$-selectable batched RCRS via a prune-greedy algorithm, which is our main technical result. As with other settings of RCRS, getting beyond the simplest algorithm is technically involved; in contrast with most other RCRS with non-trivial selectability guarantees [23, 1], our batched RCRS is non-adaptive, in the sense that the algorithm's decision to select an edge does not depend on the time when it arrives, besides checking the feasibility of selecting it.

We also show that such batched RCRSs find applications beyond prophet inequalities as in [13]. Recently there is a breakthrough by Gamlath et al. [15] on the Query-Commit and Price of Information problems, where they gave $(1 - \frac{1}{e})$-approximation algorithms for both problems by reducing them to the *prophet secretary* problem in [12, 11]. We notice that the prophet secretary algorithm can be easily replaced by the $(1 - \frac{1}{e})$-selectable RCRS for rank 1 matroid in [23] and still resulting a $(1 - \frac{1}{e})$-approximation algorithm. We find this perspective is not only conceptually simpler, but also powerful in the sense that if we replace RCRS for rank 1 matroid by our batched RCRS, their algorithms (with a slight modification) also apply for general graphs. As a result, we show that our batched RCRSs imply best-known results for Query-Commit and Price of Information for matching in general graphs.

[1] See [13] for details of this reduction.
[2] As explained above, batched OCRS as defined by Ezra et al. is more general. Our shorthand term should cause no confusion, as we consider only one OCRS setting in this work.

In the **Query-Commit** matching problem [9], we are given a graph $G = (V, E)$, a probability p_e for each edge e with which e is present, independently of the other edges, and a value v_e of e if e is present. At each step, we may choose to query an edge, but we must take it if it is present. At all times, we must keep the set of edges taken a valid matching, and our goal is to maximize the expected sum of values of the edges taken. Gamlath et al. [15] gave a clever algorithm that is $(1 - \frac{1}{e})$-competitive for bipartite graphs. Their algorithm solves an offline linear program relaxation and rounds the solution through a prophet secretary algorithm. We show that this rounding algorithm can be seen as a reduction to a batched RCRS with random one-sided vertex arrival. This perspective allows us to directly derive an $\frac{8}{15}$-competitive algorithm on general graphs, using our new batched RCRS.

In the same work, Gamlath et al. [15] gave a $(1 - \frac{1}{e})$-approximation algorithm for the closely related **Price of Information** (PoI) problem for matching in bipartite graphs. The PoI problems were introduced by Singla [27] as a generalization of the Pandora's Box problem from the consumer search literature [29, 22], which in turn is closely related to Bayesian bandits. In the PoI problem for matching [15], we are given a graph $G = (V, E)$, a search cost c_e and a value distribution F_e for each edge e. An invisible value v_e is drawn independently from F_e, and is revealed only when the algorithm chooses to *search* the edge e, at the cost of c_e. Our goal is to maximize, in expectation, the maximum value of matching among searched edges, minus the search costs paid along the way. From a new proof for the optimal algorithm for the Pandora's Box problem, given by Kleinberg et al. [22], surfaced a reduction from PoI with Bernoulli distributions to Query-Commit. This reduction was generalized by Singla [27]. Gamlath et al. [15] showed that their algorithm for Query-Commit admits some modifications that allow it to give a $(1 - \frac{1}{e})$-approximation for the general PoI problem for bipartite matching. Our perspective of their algorithm as a reduction to RCRS with random vertex arrival admits the same modifications, and therefore we immediately obtain an $\frac{8}{15}$-approximation for the PoI problem for matching in general graphs.

Other Related Works. Naturally, the most related work is the recent paper by Gamlath et al. [15]. They studied the query-commit matching problem and the price-of-information problem in *bipartite* graphs and achieved $(1 - \frac{1}{e})$ approximation ratios for both settings. Their paper has a comprehensive review of the related literature, and we only discuss briefly some most related works that are not covered there.

A closely related setting is the oblivious matching problem. It is also a query-commit model, while the input instance is adversarial rather than stochastic. In other words, there is no information about the existence probabilities of edges. Obviously, results in this harder model directly apply to our setting. The celebrated Ranking algorithm by Karp et al. [21] gives a $(1 - \frac{1}{e})$-approximation for the unweighted oblivious matching problem in bipartite graphs. Later, the Ranking algorithm is extended to general graphs and shown to achieve an approximation ratio of 0.526 [6]. Another well-studied algorithm in this literature is called modified randomized greedy. It was introduced by Aronson et al. [3] and shown to be $(0.5 + \Omega(1))$-approximate in general graphs. The approximation ratio was recently improved to 0.531 by Tang et al. [28]. For the edge-weighted case, Tang et al. [28] achieved the first non-trivial 0.501-approximation. As we remarked, this result applies to the Query-Commit setting studied in our current work, but our approximation ratio is significantly larger. Moreover, our techniques are entirely different from the previous papers and we believe they provide novel insights on the problem.

Offline contention resolution schemes for matchings also achieved some progress recently. Bruggmann and Zenklusen [5] gave an optimal monotone contention resolution scheme for bipartite matching with its application to submodular maximization, and Guruganesh and Lee [18] studied the connection between correlation gap and contention resolution scheme.

Gupta et al. [16] introduced a Markovian price of information model and use online contention resolution schemes to round the optimum of linear programming. However, as our RCRS is vertex-arrival model and batched, it can not be directly applied to their result.

2 Preliminaries

Price of Information. In a *Price of Information* problem, we are given a universe U and a feasibility system $\mathcal{F} \subseteq 2^U$. For each element $i \in U$, we are also given a *search cost* $c_i \in \mathbb{R}_+$ and a distribution F_i from which an invisible value $v_i \geq 0$ is independently drawn. In a valid algorithm for the problem, at each step we may choose to terminate the search or to inquire a new element i; when we choose to do the latter, we incur a cost c_i, and the value $v_i \sim F_i$ is revealed. At the termination of the algorithm, if S is the set of elements inquired by the algorithm, our utility is $\max_{A \subseteq S : A \in \mathcal{F}} \sum_{i \in A} v_i - \sum_{i \in S} c_i$. Our goal is to maximize the expected utility, where expectation is taken over both the realization of the values and the internal randomness of our algorithm. In this paper, we focus on the problem where the universe is the set of edges E in a given graph $G = (V, E)$, and $\mathcal{F}$ is the set of all matchings in E. An algorithm is said to be α-approximation if it achieves at least α fraction of the optimal utility. That is, we compare to the optimal algorithm.

The Query-Commit Problem for Matching. In a Query-Commit problem, we are given a universe U, a feasibility system $\mathcal{F} \subseteq 2^U$, and, for each element $i \in U$, a probability $p_i \in [0, 1]$ and a value $v_i \geq 0$. Each element $i \in U$ is *active* independently with probability p_i, but this status can be known only via a *query*. For the Query-Commit problem, at each step our algorithm can choose to query the status of an element; if the queried element is active, it must be accepted to the solution. At all time, we need to make sure that the set of accepted elements is feasible (i.e., in $\mathcal{F}$). Let S be the set of elements accepted by an algorithm by the time it terminates, the performance of the algorithm is $\sum_{i \in S} v_i$, the total value of the accepted elements. An algorithm is *α-competitive* if its performance is at least α fraction of the offline optimal, i.e., the expected performance of an algorithm that knows the set of active elements beforehand. In this work, we focus on the case where the universe U is the edge set E of a given graph $G = (V, E)$, and $\mathcal{F}$ is the set of all matchings in G.

Batched RCRS for Matching in General Graphs. We are given a graph $G = (V, E, \boldsymbol{x})$, where $\boldsymbol{x} \in [0, 1]^E$ satisfies $x_u \overset{\text{def}}{=} \sum_{e \in \delta(u)} x_e \leq 1$ for each $u \in V$. All vertices of G arrive online in a uniformly random order. Upon the arrival of a vertex v, the status of the edges connecting v and the vertices that have arrived are sampled and revealed to the algorithm. The sampling is such that, each edge e of these is *active* with probability x_e, and at most one of these edges can be active; that is, these edges' being active are mutually exclusive events. We refer to this sampling scheme as S. A batched random order contention resolution scheme for matching (henceforth batched RCRS) decides, upon the arrival of each vertex, irrevocably whether to select the active edge (if any exists). At any point in time, the selected edges must form a matching. A batched RCRS is *c-selectable* if $\Pr[e \text{ is selected} \mid e \text{ is active}] \geq c$ for every $e \in E$.
Throughout the paper, we use $\delta(u) \subseteq E$ to denote the set of edges incident to a vertex u.

3 Batched RCRS for Matching In General Graphs

In this section we give an $\frac{8}{15}$-selectable batched RCRS for matching in general graphs.

3.1 Batched RCRS and Its Proof Sketch

▶ **Theorem 1.** *There is a polynomial-time computable, $\frac{8}{15}$-selectable batched RCRS for matching in general graphs with random vertex arrival.*

Our proof consists of two steps. Given a graph $G = (V, E, \boldsymbol{x})$. Let S be the corresponding sampling scheme defined in Section 2. An instance is called 1-regular if $x_u = 1$ for all u. We refer to x_u as the degree of u. Our first lemma states that without loss of generality, we can focus on designing batched RCRS for 1-regular instances.

▶ **Lemma 2.** *If c-selectable batched RCRS exists for all 1-regular instances, then c-selectable batched RCRS exists for all instances.*

Proof. Assume c-selectable batched RCRS exists for all 1-regular instances. Consider an arbitrary instance $G = (V, E, \boldsymbol{x})$, we first convert it into a 1-regular instance G' by introducing dummy vertices and edges. Let S be the associated sampling scheme of the instance G. We correspondingly construct a sampling scheme S' on G' from S by simulating the arrival of dummy vertices. Finally, we apply the c-selectable batched RCRS in the converted 1-regular instance to achieve c-selectability in the original instance G.

For the first step, there can be multiple ways of converting a graph into a 1-regular graph. We give a concrete construction below and remark that the reduction does not rely on the details of the construction. Let $n = |V|$ be the number of vertices of G. We introduce n dummy vertices into the graph. For each vertex $u \in V$, we add n dummy edges between u and all dummy vertices, where each edge has active probability $x'_e = \frac{1-x_u}{n}$. Note that now every vertex in V has degree 1 and all dummy vertices have the same degree. Finally, we add a clique to all dummy vertices and set the weight/active probability of those edges appropriately so that all vertices have degree 1.

Recall G' is the converted instance and let $\boldsymbol{x}'$ be the corresponding vector of active probability. Next, we show how to construct the corresponding sampling scheme S' from S. There are $2n$ time slots in total for vertex arrivals. We select uniformly at random n of them to pair with n dummy vertices uniformly at random. Upon the arrival of a dummy vertex, we sample at most one of the edges connecting it to previously arrived vertices with probability consistent with $\boldsymbol{x}'$. Upon the arrival of an empty slot, we pair it with the next vertex arrival in the original instance and call S. If there exists an active edge realized by S, let it also be active in the new instance. Otherwise, we further sample at most one of the dummy edges connecting the current vertex to previously arrived dummy vertices with appropriate probability, so that the overall active probability is consistent with $\boldsymbol{x}'$. This is implementable since the degree of each vertex is exactly 1.

Now, we have defined a sampling scheme S'. Together with G' and $\boldsymbol{x}'$, it is a 1-regular instance and we can apply the c-selectable batched RCRS for 1-regular instances. Whenever an edge of G (i.e., edges between real vertices) is selected by the batched RCRS, we also select it in the real run of the instance. It is straightforward to verify that this is a c-selectable batched RCRS for the original instance. ◀

Next, we design an $\frac{8}{15}$-selectable batched RCRS for 1-regular instances.

▶ **Lemma 3.** *For every 1-regular instance, there exists an $\frac{8}{15}$-selectable batched RCRS.*

Before going into the details of our analysis, we present our algorithm and provide a proof sketch to highlight the essential ideas. The actual proof of Lemma 3 is deferred to Section 3.3.

Proof Sketch. Consider a 1-regular instance $G = (V, E, \boldsymbol{x})$. We apply the following prune-greedy algorithm: 1) Prune the active probability of each edge from x_e to $f(x_e) \overset{\text{def}}{=} \frac{3x_e}{3+2x_e}$; 2) Run greedy on the pruned instance.

The first step is a preprocessing step and we refer to it as the pruning step. Before the instance starts, we independently flip a coin for each edge e and decide whether to consider it later. In particular, an edge e survives with probability $\frac{f(x_e)}{x_e}$ and otherwise we would ignore it regardless being active or not in the future. The second step is just a greedy algorithm, that always select an active edge if adding it does not violate the matching constraint. This procedure is equivalent to assuming that each edge is active with probability $f(x_e)$ and we run greedy. For the ease of presentation, in the rest of the proof, we assume that each edge e is active with probability $f(x_e)$. Our specific choice of the function f is to benefit the analysis, and it be explained in the full proof.

For analysis purpose, we use the following interpretation of the random arrival order assumption. Let t_u be the arrival time of each vertex $u \in V$, that is drawn independently from $U[0, 1]$. All vertices arrive in the ascending order of their arrival times.

Fixing an arbitrary edge $(u, v) \in E$, we use $v \to u$ to denote the event that edge (u, v) is selected by our algorithm upon the arrival of vertex v. To prove the selectability, we need a lower bound on $\Pr[v \to u]$. Fixing the arrival time $t_v = t$, since we use a greedy algorithm, this event happens if 1) u arrives before t; 2) u remains unmatched at t; 3) (u, v) is active. Thus,

$$\begin{aligned}\Pr[v \to u \mid t_v = t] &= \Pr[t_u \le t, u \text{ unmatched } @t, (u, v) \text{ active} \mid t_v = t] \\ &= f(x_{uv}) \cdot \Pr[t_u \le t, u \text{ unmatched } @t \mid t_v = t] \\ &= f(x_{uv}) \cdot (t - \Pr[u \text{ matched } @t \mid t_v = t]) . \qquad (*)\end{aligned}$$

Consider the following warm-up analysis,

$$\Pr[u \text{ matched } @t \mid t_v = t] = \sum_{z \neq u,v} \Pr[(u, z) \text{ matched before } t \mid t_v = t] \le \sum_{z \neq u,v} t^2 f(x_{uz}) \le t^2,$$

where the inequality follows since the probability that both u, z arrives before t is t^2 and the edge is matched only if it is active, that happens with probability $f(x_{uz})$. Applying this relaxation to equation $(*)$ and integrating over t gives a simple but weak bound of the selectablity of our algorithm, where the ratio is smaller than $\frac{1}{2}$. Observe that the warm-up analysis does not make use of the pruning step, as well as the 1-regularity of the instance.

Instead, we do recursive type of analysis for $\Pr[(u, z) \text{ matched before } t \mid t_v = t]$:

$$\begin{aligned}&\Pr[(u, z) \text{ matched before } t \mid t_v = t] \\ &\qquad = \int_0^t \Big(\Pr[u \to z \mid t_u = s, t_v = t] + \Pr[z \to u \mid t_z = s, t_v = t]\Big) \mathrm{d}s. \qquad (**)\end{aligned}$$

We can thus expand the term $\Pr[u \to z \mid t_v = t, t_u = s]$ as in $(*)$, and we would have a term $\Pr[z \text{ matched before } s \mid t_u = s, t_v = t]$ therein. However, since our final goal is to derive a lower bound of $(*)$, we need a lower bound of this term rather than an upper bound as we derived in the warm-up analysis. This is where we use the 1-regularity of the instance.

Intuitively, the larger the degree of a vertex, the larger the probability it is matched. It is not difficult to construct a counterexample showing that our prune-greedy algorithm is indeed worse than $\frac{1}{2}$-selectable for non 1-regular instances. For technical reasons, we also want to avoid edges with large active probabilities and our pruning step is designed to take care of such edges.

Roughly speaking, $(*)$ together with $(**)$ expands $\Pr[v \to u \mid t_v = t]$ to terms of form $\Pr[u \to z \mid t_u = s, t_v = t]$. The formal analysis recursively applies such expansion for two steps and uses the simple bound in the warm-up analysis in the last step to bound $\Pr[j \text{ matched } @r \mid t_v = t, t_u = s, t_i = r]$. We defer the proof to Section 3.3.

3.2 Hardness

We complement our positive result with a hardness result, showing that no algorithm can be better than $\frac{1}{2} + \frac{1}{2e^2} \approx 0.567$-selectable.

▶ **Theorem 4.** *No batched RCRS for matching is better than* $(\frac{1}{2} + \frac{1}{2e^2})$*-selectable.*

Proof. Consider a complete graph G with n vertices. Each edge e has $x_e = \frac{1}{n-1}$. Fix an algorithm and let $y_i \cdot n$ be the number of matched vertices after the arrival of the i-th vertex, where $y_i \in [0,1]$ is increasing and $y_1 = 0$. Notice that all edges are symmetric and every vertex has degree 1, the selectability of the algorithm is thus upper bounded by $\mathbf{E}[y_n]$. Moreover, we have that

$$\mathbf{E}\left[y_{i+1} \mid y_i\right] \le y_i + 2 \cdot \left(\frac{i}{n-1} - y_i\right),$$

since we can select an edge only if 1) there exists an active edge, which happens with probability $\frac{i}{n-1}$ and 2) the corresponding vertex is unmatched. Taking expectation over y_i, we have that

$$\mathbf{E}\left[y_{i+1}\right] - \mathbf{E}\left[y_i\right] \le 2 \cdot \left(\frac{i}{n-1} - \mathbf{E}\left[y_i\right]\right).$$

When n goes to infinity, the above family of inequalities converges to $\frac{dz_t}{dt} \le 2 \cdot (t - z_t)$, where z_t corresponds to $\mathbf{E}[y_{\lfloor t \cdot n \rfloor}]$. Solving the differential equation gives us

$$z_t \le t - \frac{1}{2} + \frac{1}{2} e^{-2t}, \forall t \in [0,1] .$$

Hence, the total portion of matched vertices is upper bounded by $z_1 \le \frac{1}{2} + \frac{1}{2e^2}$ when n goes to infinity, that implies no RCRS can be better than $\left(\frac{1}{2} + \frac{1}{2e^2}\right) \approx 0.567$-selectable. ◀

3.3 Proof of Lemma 3

Proof of Lemma 3. Fix an edge (u, v), our goal is to lower bound the probability of

$$\Pr[v \to u] = \int_0^1 \Pr[v \to u \mid t_v = r_v] dr_v. \tag{1}$$

As we discussed in the proof sketch, the probability term on the right side can be expanded as following

$$\begin{aligned}
&\Pr[v \to u \mid t_v = r_v] \\
&= f(x_{uv}) \cdot (\Pr[u \text{ arrives before } t_v] - \Pr[u \text{ is matched at } t_v]) \\
&= f(x_{uv}) \cdot \left(t_v - \sum_{\substack{(u,i)\in E \\ i \neq u,v}} \Pr[(u,i) \text{ is selected before } t_v \mid t_v = r_v] \right) \\
&= f(x_{uv}) \cdot \left(t_v - \sum_{\substack{(u,i)\in E \\ i \neq u,v}} (\Pr[u \to i \text{ and } t_u \le t_v \mid t_v = r_v] + \Pr[i \to u \text{ and } t_i \le t_v \mid t_v = r_v]) \right). \qquad (2)
\end{aligned}$$

Since both terms in the right side of (2) are symmetric, we will only show how to upper bound the first term $\Pr[u \to i \text{ and } t_u \le t_v \mid t_v = r_v]$. As explained in the sketch, we further recursively expand it in essentially the same way as $\Pr[v \to u \mid t_v = r_v]$.

$$\begin{aligned}
&\Pr[u \to i \text{ and } t_u \le t_v \mid t_v = r_v] \\
&= \int_0^{t_v} \Pr[u \to i \mid t_v = r_v, t_u = r_u] dr_u \\
&= \int_0^{t_v} f(x_{ui})(t_u - \Pr[i \text{ is matched at } t_u \mid t_v = r_v, t_u = r_u]) dr_u \\
&= f(x_{ui}) \cdot \int_0^{t_v} \left(t_u - \sum_{\substack{(i,j)\in E \\ j \neq i,u,v}} \Pr[(i,j) \text{ is selected before } t_u \mid t_u = r_u, t_v = r_v] \right) dr_u \\
&= f(x_{ui}) \cdot \int_0^{t_v} \Bigg(t_u - \sum_{\substack{(i,j)\in E \\ j \neq i,u,v}} (\Pr[i \to j, t_i \le t_u \mid t_u = r_u, t_v = r_v] \\
&\qquad\qquad + \Pr[j \to i, t_j \le t_u \mid t_u = r_u, t_v = r_v]) \Bigg) dr_u. \qquad (3)
\end{aligned}$$

Now we further expand the last equation and apply similar argument as the warm up analysis in the proof sketch.

$$\begin{aligned}
&\Pr[i \to j, t_i \le t_u \mid t_u = r_u, t_v = r_v] \\
&= \int_0^{t_u} \Pr[i \to j \mid t_i = r_i, t_u = r_u, t_v = r_v] dr_i \\
&= \int_0^{t_u} f(x_{ij}) \left(t_i - \Pr[j \text{ is matched at } t_i \mid t_i = r_i, t_u = r_u, t_v = r_v]\right) dr_i. \qquad (4)
\end{aligned}$$

As in the proof sketch, we enumerate the vertex k which j is matched to.

$$\begin{aligned}
&\Pr[j \text{ is matched at } t_i \mid t_i = r_i, t_u = r_u, t_v = r_v] \\
&\qquad = \sum_{\substack{(j,k)\in E \\ k \neq i,j,u,v}} \Pr[(j,k) \text{ is selected before } t_i \mid t_i = r_i, t_u = r_u, t_v = r_v]. \qquad (5)
\end{aligned}$$

Notice that if an edge (j,k) is selected by the algorithm R before time t_i, it has to satisfy at least three conditions:

1. j arrives before t_i;
2. k arrives before t_i;
3. the edge is sampled by the sampling scheme S and selected by the algorithm.

One can observe that these three events are independent. In order to bound (4), we can apply the trivial upper bound to (5)

$$\Pr[(j,k) \text{ is selected before } t_i \mid t_i = r_i, t_u = r_u, t_v = r_v] \le t_i^2 f(x_{jk})$$

and we obtain

$$\begin{aligned}
&\Pr[i \to j, t_i \le t_u \mid t_u = r_u, t_v = r_v] \\
&\ge f(x_{ij}) \cdot \int_0^{t_u} t_i \left(1 - t_i \sum_{\substack{(j,k)\in E\\ k\neq i,j,u,v}} f(x_{jk})\right) dt_i \\
&\ge f(x_{ij}) \cdot \int_0^{t_u} t_i \cdot (1 - t_i \cdot (1 - x_{ij}))\, dt_i = f(x_{ij}) \cdot \left(\frac{1}{2}t_u^2 - \frac{1}{3}t_u^3(1 - x_{ij})\right).
\end{aligned}$$

Similarly, it also holds that

$$\Pr[j \to i, t_j \le t_u \mid t_u = r_u, t_v = r_v]) \ge f(x_{ij}) \cdot \left(\frac{1}{2}t_u^2 - \frac{1}{3}t_u^3(1 - x_{ij})\right).$$

Now we start to plug them back to the two-level recursive analysis. First, plugging these two inequalities into (3), we can bound the first term in the right side of (2).

$$\begin{aligned}
&\Pr[u \to i \text{ and } t_u \le t_v \mid t_v = r_v] \\
&\le f(x_{ui}) \cdot \int_0^{t_v} \left(t_u - \sum_{\substack{(i,j)\in E\\ j\neq i,u,v}} f(x_{ij}) \cdot \left(t_u^2 - \frac{2}{3}t_u^3(1 - x_{ij})\right)\right) dt_u \\
&= f(x_{ui}) \cdot \left(\frac{1}{2}t_v^2 - \sum_{\substack{(i,j)\in E\\ j\neq i,u,v}} f(x_{ij}) \cdot \left(\frac{1}{3}t_v^3 - \frac{1}{6}t_v^4(1 - x_{ij})\right)\right).
\end{aligned}$$

By symmetry,

$$\Pr[i \to u \text{ and } t_i \le t_v \mid t_v = r_v] \le f(x_{ui}) \cdot \left(\frac{1}{2}t_v^2 - \sum_{\substack{(u,j)\in E\\ j\neq i,u,v}} f(x_{uj}) \cdot \left(\frac{1}{3}t_v^3 - \frac{1}{6}t_v^4(1 - x_{uj})\right)\right).$$

Then, by plugging in both terms back to (2), we have

$$\begin{aligned}
(2) \ge f(x_{uv}) \cdot \Bigg(t_v &- \sum_{\substack{(u,i)\in E\\ i\neq u,v}} f(x_{ui}) \cdot \left(\frac{1}{2}t_v^2 - \sum_{\substack{(i,j)\in E\\ j\neq i,u,v}} f(x_{ij}) \cdot \left(\frac{1}{3}t_v^3 - \frac{1}{6}t_v^4(1 - x_{ij})\right)\right) \\
&- \sum_{\substack{(u,i)\in E\\ i\neq u,v}} f(x_{ui}) \cdot \left(\frac{1}{2}t_v^2 - \sum_{\substack{(u,j)\in E\\ j\neq i,u,v}} f(x_{uj}) \cdot \left(\frac{1}{3}t_v^3 - \frac{1}{6}t_v^4(1 - x_{uj})\right)\right)\Bigg).
\end{aligned}$$

Therefore, (1) can be further bounded as follows:

$$
\begin{aligned}
(1) \geq f(x_{uv}) \cdot \Bigg(\frac{1}{2} - \sum_{\substack{(u,i)\in E\\ i\neq u,v}} f(x_{ui}) \cdot \Bigg(\frac{1}{6} - \sum_{\substack{(i,j)\in E\\ j\neq i,u,v}} f(x_{ij}) \cdot \left(\frac{1}{20} + \frac{1}{30}x_{ij}\right)\Bigg) \\
- \sum_{\substack{(u,i)\in E\\ i\neq u,v}} f(x_{ui}) \cdot \Bigg(\frac{1}{6} - \sum_{\substack{(u,j)\in E\\ j\neq i,u,v}} f(x_{uj}) \cdot \left(\frac{1}{20} + \frac{1}{30}x_{uj}\right)\Bigg)\Bigg) \\
= f(x_{uv}) \cdot \Bigg(\frac{1}{2} - \sum_{\substack{(u,i)\in E\\ i\neq u,v}} f(x_{ui}) \cdot \Bigg(\frac{1}{6} - \sum_{\substack{(i,j)\in E\\ j\neq i,u,v}} \frac{1}{20}x_{ij}\Bigg) - \sum_{\substack{(u,i)\in E\\ i\neq u,v}} f(x_{ui}) \cdot \Bigg(\frac{1}{6} - \sum_{\substack{(u,j)\in E\\ j\neq i,u,v}} \frac{1}{20}x_{uj}\Bigg)\Bigg) \\
= f(x_{uv}) \cdot \Bigg(\frac{1}{2} - \sum_{\substack{(u,i)\in E\\ i\neq u,v}} f(x_{ui}) \cdot \left(\frac{1}{3} - \frac{1}{20}(1 - x_{ui} - x_{vi}) - \frac{1}{20}(1 - x_{ui} - x_{uv})\right)\Bigg) \\
= f(x_{uv}) \cdot \Bigg(\frac{1}{2} - \sum_{\substack{(u,i)\in E\\ i\neq u,v}} f(x_{ui}) \cdot \left(\frac{7}{30} + \frac{1}{20}(2x_{ui} + x_{vi} + x_{uv})\right)\Bigg).
\end{aligned}
$$

▶ Remark. Note here for the first equality, we use the fact that $f(x)\left(\frac{1}{20} + \frac{1}{30}x\right) = cx$ for $c = \frac{1}{20}$. (Actually, we only need one side that it is larger than cx.) The choice of $f(x) = \frac{3x}{3+2x}$ is to maximize c while keeping $f(x) \leq x$ for all $x \in [0,1]$ so that it is a valid pruning. For the second equality (where again we only need one side), we use the fact that $\sum_{\substack{(i,j)\in E\\ j\neq i,u,v}} x_{i,j} \geq 1 - x_{u,i} - x_{v,i}$ which follows from the 1-regularity of our graph.

Finally we are ready to bound $\Pr[(u,v) \text{ is selected}]$. Without loss of generality, we can assume that $x_{uv} = 0$ for those $(u,v) \notin E$. Recall that $\Pr[(u,v) \text{ is selected}] = \Pr[u \to v] + \Pr[v \to u]$ by definition and we have

$$
\begin{aligned}
&\Pr[(u,v) \text{ is selected}] \\
&\geq f(x_{uv}) \cdot \Bigg(1 - \sum_{i\neq u,v} \Bigg(f(x_{ui}) \cdot \left(\frac{7}{30} + \frac{1}{20}(2x_{ui} + x_{vi} + x_{uv})\right) + \\
&\qquad f(x_{vi}) \cdot \left(\frac{7}{30} + \frac{1}{20}(2x_{vi} + x_{ui} + x_{uv})\right)\Bigg)\Bigg) \\
&\geq f(x_{uv}) \cdot \Bigg(1 - \frac{1}{10}x_{uv} - \sum_{i\neq u,v} \Bigg(f(x_{ui}) \cdot \left(\frac{7}{30} + \frac{1}{20}(2x_{ui} + x_{vi})\right) \\
&\qquad + f(x_{vi}) \cdot \left(\frac{7}{30} + \frac{1}{20}(2x_{vi} + x_{ui})\right)\Bigg)\Bigg) \\
&\geq f(x_{uv}) \cdot \Bigg(1 - \frac{1}{10}x_{uv} - \sum_{i\neq u,v} \frac{7}{30} \cdot (x_{ui} + x_{vi})\Bigg) \\
&= f(x_{uv}) \cdot \left(1 - \frac{1}{10}x_{uv} - \frac{7}{30} \cdot (2 - 2x_{uv})\right) \\
&= \frac{3x_{uv}}{3 + 2x_{uv}} \cdot \left(\frac{8}{15} + \frac{11}{30}x_{uv}\right) \geq \frac{8}{15}x_{uv}.
\end{aligned}
$$

Here the second inequality uses the fact that $\sum_{i\neq u,v} f(x_{ui}) \leq \sum_{i\neq u,v} x_{ui} \leq 1$ and $\sum_{i\neq u,v} f(x_{vi}) \leq \sum_{i\neq u,v} x_{vi} \leq 1$. We are left to prove the third inequality, which we state as the following lemma.

▶ **Lemma 5.** *The inequality*

$$f(x) \cdot \left(\frac{7}{30} + \frac{1}{20}(2x+y)\right) + f(y) \cdot \left(\frac{7}{30} + \frac{1}{20}(2y+x)\right) \leq \frac{7}{30} \cdot (x+y)$$

holds for every $x, y \geq 0$.

Proof. It is equivalent to prove that

$$\frac{1}{20}f(x)(2x+y) + \frac{1}{20}f(y)(2y+x) \leq \frac{7}{30}(x - f(x) + y - f(y)).$$

Expand $f(x)$ and we get

$$\frac{1}{10}\left(\frac{3x^2}{3+2x} + \frac{3y^2}{3+2y}\right) + \frac{1}{20}\left(\frac{3xy}{3+2x} + \frac{3xy}{3+2y}\right) \leq \frac{7}{15}\left(\frac{3x^2}{3+2x} + \frac{3y^2}{3+2y}\right).$$

Rewriting the formula, we find that it's equivalent to prove

$$30x^2 + 30y^2 - 54xy + 2x^2y + 2xy^2 \geq 0.$$

Notice that $x, y \geq 0$, thus $2x^2y + 2xy^2 \geq 0$. Also, $30x^2 + 30y^2 - 54xy \geq 30x^2 + 30y^2 - 60xy \geq 30(x-y)^2 \geq 0$, which completes our proof. ◀

This finishes the proof of Lemma 3. ◀

4 Applications

Our definition of the batched RCRS for matching with random vertex arrival is inspired by the standard reduction from prophet matching to online contention resolution schemes. Therefore, Theorem 1 immediately implies the first nontrivial result for prophet matching in general graphs with random vertex arrival.

▶ **Theorem 6.** *There is an $\frac{8}{15}$-competitive algorithm for prophet matching in general graphs with random vertex arrival.*

In this section, we mainly discuss the application of our batched RCRS in Query-Commit and Price-of-Information problems for matching. The algorithms below can be seen as reinterpretations of Gamlath et al. [15]'s algorithms for the problems on bipartite matching. For Query-Commit, their algorithm first solves a linear program relaxation of the corresponding problem, then, using characterization of polymatroids, interprets the fractional solution as follows: each vertex u on the left samples a permutation over the edges in δ_u, and *proposes* to query these edges in that order; literally following these proposals for every vertex obviously leads to collisions, and so vertices on the right need to resolve potential collisions by turning down some of the proposals. The algorithm lets the vertices on the left arrive in a uniformly random order to propose their permutations. Each vertex on the right then runs a *prophet secretary* algorithm, which guarantees that, in expectation, the proposed queries turned down by the nodes on the right do not carry too much value. We suggest that, one may bypass the more complex prophet secretary setup, by seeing the last step as an online contention resolution step with random vertex arrival: when vertex u arrives and proposes its permutation, if we let edge $e \in \delta_u$ be *active* if it is the first edge present when we query the edges following the proposed order, then a c-selectable batched RCRS with vertex arrival keeps each active edge with probability at least c, which leads to a c-approximation algorithm. The description at this level omits many details and twists, and the Price-of-Information algorithm has even more of those. Nonetheless, our perspective easily generalizes to the non-bipartite cases, and allows us to use the batched RCRS in Section 3 to these problems. We give the details below.

4.1 The (Weighted) Query-Commit Problem for Matching

We now present an $\frac{8}{15}$-competitive algorithm for the Query-Commit problem by a reduction to batched RCRS.

▶ **Theorem 7.** *There is a polynomial-time computable, $\frac{8}{15}$-competitive algorithm for the Query-Commit problem for weighted matching in graphs not necessarily bipartite.*

4.1.1 Bounding the optimal utility

First we construct a linear program, where for every edge $e \in E$, x_e represents the probability that e is present and included in the solution. We show that the value of the linear program is an upper bound on even the offline optimal. Therefore, if we implement an algorithm so that each edge e is selected with probability at least $\alpha \cdot x_e^*$, then the competitive ratio of the algorithm will be at least α.

Think of x_e as the probability with which edge e is present and included in the offline solution. For a subset of edges $F \subseteq E$, let $f(F)$ be the probability that at least one edge in F exists, i.e., $f(F) = 1 - \prod_{e \in F}(1 - p_e)$. For any $F \subseteq \delta_u$, since in a matching no more than one edge in F can be in the solution, $\sum_{e \in F} x_e$ is the probability any edge in F is in the offline solution, and this should be upper bounded by the probability any edge in F is present. The following linear program $\mathrm{LP_{QC}}$ therefore upper bounds the offline optimal:

$$\begin{aligned}
\text{max:} \quad & \sum_{e\in E} x_e \cdot v_e \\
\text{s.t.} \quad & \sum_{e\in F} x_e \le f(F), && \forall u \in V, F \subseteq \delta_u; \\
& x_e \ge 0, && \forall e \in E.
\end{aligned}$$

▶ **Lemma 8** ([15])**.** *In the edge-weighted Query-Commit problem for matching, the offline optimal is upper bounded by the value of* $\mathrm{LP_{QC}}$*. Furthermore,* $\mathrm{LP_{QC}}$ *is polynomial-time solvable.*

Lemma 8 is a straightforward generalization of Lemma 2.1 and Lemma 2.2 in [15]. In the following, we need to assume $0 < p_e < 1$ for every $e \in E$ since some proofs requires strict monotonicity and strict submodularity of f. It turns out that we can ignore those edges with zero probabilities and scale down probabilities by $1 - \gamma$ for other edges due to the following lemma.

▶ **Lemma 9** (Lemma 2.3 of [15])**.** *For $0 < \gamma < 1$, let $\tilde{p}_e = (1-\gamma)p_e$ for every $e \in E$. Define $\tilde{f}$ using $\tilde{p}$ instead of p. Similarly, define* $\tilde{\mathrm{LP}}_{\mathrm{QC}}$ *use $\tilde{p}, \tilde{f}$ instead of of p, f. Then the value of* $\tilde{\mathrm{LP}}_{\mathrm{QC}}$ *is at least $(1-\gamma)$ times the value of* $\mathrm{LP_{QC}}$*.*

For any solution $\boldsymbol{x}$ to $\mathrm{LP_{QC}}$, the following lemma defines a "decomposition into permutations" over (subsets of) δ_u for each $u \in V$. It is essentially Lemma 2.6 in [15] restated on general graphs.

▶ **Lemma 10** ([15])**.** *Suppose $0 < p_e < 1$ for every $e \in E$. Let $\boldsymbol{x}^*$ be an optimal solution to* $\mathrm{LP_{QC}}$*. For every $u \in V$, fix any subset $\delta'_u \subseteq \delta_u$. Then there exists a polynomial-time samplable distribution $\mathcal{D}_u^{\mathrm{QC}}$ over the permutations on subsets of δ'_u, so that, if one queries according to the permutation sampled from $\mathcal{D}_u^{\mathrm{QC}}$, each edge e is the first present one with probability x_e^*.*

The meaning that the permutations are over *subsets of δ'_u* is that some edges may not be present in the permutation (which indicates that they should not be queried).

4.1.2 Rounding with batched RCRS

The game plan becomes clearer with the decomposition from Lemma 10. Ideally, if one may naïvely follow the permutation sampled from $\mathcal{D}_u^{\mathrm{QC}}$ for each u, and take the first present edge, then one may recover the offline optimal. This is of course infeasible, because matching constraints may be violated by collisions caused by doing this for different vertices. Gamlath et al. [15] handled this via a clever application of prophet secretary algorithms. Instead, we replace this with a reduction to the batched RCRS we developed in Section 3.

We may view the naïve algorithm which always commits to the first present edge in $\sigma_u \sim \mathcal{D}_u^{\mathrm{QC}}$ as a sampling scheme S_{QC} that indicates the first present edge as *active*, and apply our batched RCRS. The matching constraint is then respected by the feasibility of RCRS solutions. However, there is one more subtlety: in the Query-Commit problem, we must commit to the edge we just queried, whereas a batched RCRS assumes that it can see an active edge and discards it. Therefore S_{QC} should not query the presence of an edge if the batched RCRS algorithm R would not accept it *even if it is present.* To handle this issue, we need to change the order of the events by modifying the sampling scheme S_{QC}: upon the arrival of each vertex u, S_{QC} first obtains an indicator vector $\boldsymbol{I}$ from the batched RCRS algorithm R, where $I_e = 1$ if and only if R is willing to accept edge e if S_{QC} indicates that e is active. If $I_e = 0$, instead of actual querying the presence of edge e, S_{QC} simply tosses a coin to simulate a query. We remark that Gamlath et al.'s algorithm has a similar argument. Following is the formal description for our reduction from a Query-Commit problem to batched RCRS.

Our algorithm. Our algorithm first solves $\mathrm{LP}_{\mathrm{QC}}$ to get an optimal solution $\boldsymbol{x}^*$. Let R be a batched RCRS instance corresponding to graph $\langle V, E, \boldsymbol{x}^* \rangle$. Let S_{QC} be the sampling scheme to be defined later. Additionally, for each vertex $u \in V$ we sample $t_u \sim U[0,1]$ as its arrival time. We define $\delta'_u = \{(u,w) \in \delta_u \mid t_w < t_u\}$ to be the edges batch that arrives with u.

Our algorithm iterates over all vertices $u \in V$ in the increasing order of t_u. For each vertex u, it first obtains the indicator vector $\boldsymbol{I}$ where I_e indicates whether R is willing to accept e if it is active. Passing $\boldsymbol{I}$ to the sampling scheme S_{QC}, it obtains the active edge $e \in \delta'_u$ for RCRS algorithm R. We commit to e if R accepts it.

The sampling scheme S_{QC}. Let u be the vertex just arrived in batched RCRS. First sample a permutation σ_u from $\mathcal{D}_u^{\mathrm{QC}}$ by Lemma 10, which is a permutation containing a subset of δ'_u. Consider each edge $e = (u,w)$ in the order of σ_u. There are two cases:

- If $I_e = 1$: query if e is active in the query-commit instance. If it is active, report e as the active edge to R and exit; otherwise continue to the next edge.
- If $I_e = 0$: with probability p_e, report e as the active edge to R and exit; otherwise continue to the next edge.

For the analysis, we should first verify that S_{QC} is a valid sampling scheme. First, it will sample at most one active edge for a vertex, and the sampling result is independent from the indicator vector $\boldsymbol{I}$ since when considering an edge e, in both cases it essentially do a coin flip which heads up with probability p_e and use the result to determine whether e is active. Having observed the independency, we can show that x^*_e is the probability of edge e being active in S_{QC} for any arrival times by using Lemma 10.

▶ **Lemma 11.** *Fix arrival times $\{t_u\}_{u \in V}$. For every vertex $u \in V$ and every edge $e \in \delta'_u$, the probability of e being active in S_{QC} upon the arrival of u equals x^*_e.*

Proof. Fixing a vertex $u \in V$, we have that the probability of every edge $e \in \delta'_u$ being the first active edge in the random permutation σ_u equals x^*_e, according to Lemma 10. The problem is that in the second case, the algorithm does not even check whether an edge is active in the query-commit instance.

Nevertheless, in both cases each edge e will be active independently with probability p_e, which is exactly the probability of e being active in the query-commit instance. Therefore, the random process for determining the active edge is identical to finding the first active element in σ_u. And we have the probability of e being active is exactly x^*_e. ◀

At last, $\sum_{e \in \delta_u} x^*_e \leq 1$ hold for every $u \in V$ trivially by constraints in $\mathrm{LP}_{\mathrm{QC}}$. We conclude the validity of S_{QC} with Lemma 12.

▶ **Lemma 12.** *S_{QC} is a valid sampling scheme for batched RCRS. In particular, the following three conditions hold:*
1. *each time S_{QC} will sample at most one edge and the result is independent from $\boldsymbol{I}$;*
2. *each edge $e \in E$ will be active with probability x^*_e for any arrival times $\{t_u\}_{u \in V}$;*
3. *$\sum_{e \in \delta_u} x^*_e \leq 1$ holds for every $u \in V$.*

We now are ready to prove the competitive ratio for the algorithm by the selectability of the batched RCRS.

Proof of Theorem 7. The validity of batched RCRS is already proven in Lemma 12. We start the rest of the proof by showing the correctness of the reduction. First, those committed edges must form a matching by the correctness of R. And once a query of e has succeed, R will always decide to take e because we only query e when the indicator vector $I_e = 1$. Finally, no edge will be committed without a query since every edge e that has not been queried will be active to R only when $I_e = 0$.

Then we can easily show the ratio of the reduction equals the selectability of the batched RCRS. By Lemma 11 and the fact that $\cup_{u \in V} \delta'_u = E$, the probability of every edge $e \in E$ being active in R is x^*_e. By Theorem 1, the expected utility for our algorithm is at least $\frac{8}{15} \cdot \sum_{e \in E} x^*_e \cdot v_e$. Further by Lemma 8, we conclude it is an $\frac{8}{15}$-competitive algorithm for the query-commit problem on general graphs. ◀

4.2 The Price of Information Problem for Matching

The Price of Information (PoI) problem has search costs but imposes no obligation on an algorithm to immediately accept a queried element regardless of what is revealed. Kleinberg et al. [22] upper bounded the optimal utility using a variant of query-commit, by giving a new proof for the optimal algorithm in the special case where $\mathcal{F}$ is the set of singleton sets (known as the Pandora's Box problem [29]); Singla [27] generalized the bound and proposed approximation algorithms using the bound. Gamlath et al. [15] studied the Price of Information problem for bipartite matching and made use of the upper bound.

Their method converts the original "price-of-information" world to "free-information" world by setting a threshold value τ_i for each element. In "free-information" world, there is no search cost c_i, but the algorithm gets a lower utility $\kappa_i = \min\{v_i, \tau_i\}$ (instead of v_i) for accepting an element i. Intuitively, the $(v_i - \tau_i)_+$ part[3] of the utility pays for the search cost. However, this is only the case if the algorithm accepts element i in the end. If the algorithm queried element i without accepting it, the utility of the algorithm in "free-information" world

[3] $(z)_+$ denotes $\max\{z, 0\}$

will only be an upper bound of that in the "price-of-information" world. In Section 4.2.1, we upper bound the optimal utility in "free-information" world. Therefore, to match this upper bound, the algorithm has to always accept element i whenever element i is queried and $v_i > \tau_i$.

The intuition above is summarized by the following lemma from [27].

▶ **Lemma 13** ([27]). *In any instance of Price of Information problem, for each element $i \in U$, let τ_i be the unique solution to the equation $\mathbf{E}_{v_i \sim F_i}[(v_i - \tau_i)_+] = c_i$, where $(z)_+$ denotes $\max\{z, 0\}$. Let κ_i be $\min\{v_i, \tau_i\}$. Then no algorithm's utility exceeds $\mathbf{E}[\max_{S \in \mathcal{F}} \sum_{i \in S} \kappa_i]$. In particular, in order for algorithm $\mathcal{A}$ to match this upper bound, $\mathcal{A}$ must accept each element i such that i is queried by $\mathcal{A}$ and $v_i > \tau_i$.*

Similar as query-commit setting, our algorithm is essentially the same as that of [15], but we provide the perspective that the use of prophet secretary in [15] can be replaced by batched RCRS and extend their results to general graphs.

▶ **Theorem 14.** *There is a polynomial time, $\frac{8}{15}$-approximate algorithm for the price of information problem for weighted matching in graphs not necessarily bipartite.*

4.2.1 Bounding the optimal utility

We start by presenting the LP which upper bounds the optimal utility in the "free-information" world. Therefore by Lemma 13, it is also an upper bound for the optimal utility in the "price-of-information" world. The LP and the proofs are essentially the same as that in [15] with the only difference that this is for general graph.

Recall in the PoI problem for matching, there is an undirected graph $G = (V, E)$. For each edge $e \in E$, its value is a random variable $v_e \sim F_e$. Then we set τ_e and κ_e as in Lemma 13.

Without loss of generality, we assume that the distributions of κ_e are discrete.[4] Let K_e be the set of possible values of κ_e. For all $u \in V$, let $E_u = \{(e, \kappa) : e \in \delta_u, \kappa \in K_e\}$ be the edge-value pairs incidents to u and $E_{\text{all}} = \cup_{u \in U} E_u$ be the set of all edge-value pairs in the graph. For each $F \subseteq E_u$, we define $f(F)$ be the probability that $\kappa_e = \kappa$ for at least one of the edge-value pair $(e, \kappa) \in F$. Namely $f(F) = \prod_{v \in V}(1 - \sum_{e:(e,\kappa) \in F} p_{e,\kappa})$ where $p_{e,\kappa} = \Pr[\kappa_e = \kappa]$.

For any algorithm $\mathcal{A}$ for the PoI problem for matching, let $x_{e,\kappa}$ be the probability that $\mathcal{A}$ accepts edge e and $\kappa_e = \kappa$. For any $F \subseteq E_u$, similar with Section 4.1, we know that $\sum_{(e,\kappa) \in F} x_{e,\kappa} \le f(F)$. Therefore it is natural to consider the following LP, which is called LP_{PoI}.

$$\begin{aligned}
\text{max:} \quad & \sum_{(e,\kappa) \in E_{\text{all}}} x_{e,\kappa} \cdot \kappa & \\
\text{s.t.} \quad & \sum_{(e,\kappa) \in F} x_{e,\kappa} \le f(F), & \forall u \in V, F \subseteq E_u; \\
& x_e \ge 0, & \forall e \in E_{\text{all}}.
\end{aligned}$$

We restate Lemma 3.2 and Lemma 3.3 of [15] below for general graphs.

[4] In the case where the distributions are continuous. We can discretize them by geometric grouping edge weights into classes.

▶ **Lemma 15** ([15]). *In the price of information problem for matching, the optimal expected utility* $\mathrm{OPT}_{\mathrm{PoI}}$ *is upper bounded by the value of* $\mathrm{LP}_{\mathrm{PoI}}$*. Furthermore,* $\mathrm{LP}_{\mathrm{PoI}}$ *is polynomial-time solvable.*

4.2.2 Rounding with batched RCRS

Let $\boldsymbol{x}^*$ be an optimal solution of $\mathrm{LP}_{\mathrm{PoI}}$. We proceed to "round" it to an algorithm for the PoI problem. Let $\langle V, E, \boldsymbol{x}_{\mathrm{PoI}}\rangle$ be the batched RCRS instance. Note the graph of the RCRS instance is the same as the original graph. Before the formal description of S_{PoI}, we first sketch it here:

From the LP solution $\boldsymbol{x}^*$, we first obtain a polynomial-time samplable distribution $\mathcal{D}_u^{\mathrm{PoI}}$ over permutations of edge-value pairs (e, κ). S_{PoI} first draw a permutation σ from $\mathcal{D}_u^{\mathrm{PoI}}$. An edge-value pair (e, κ) is said to be active if $\kappa_e = \kappa$. Roughly speaking, S_{PoI} returns the first active edge-value pair in σ as the active edge. However, as Lemma 15 suggests, our algorithm must accept edge e if e is queried and $v_e = \tau_e$. Therefore, there is an additional requirement that in each σ drawn from $\mathcal{D}_u^{\mathrm{PoI}}$, the edge-value pairs of the same edge should be in decreasing order of their values. Hence (e, τ_e) is always the first pair associated with edge e in σ.

The following lemma formally defines the distribution $\mathcal{D}_u^{\mathrm{PoI}}$. It is a restatement of Lemma 3.6 in [15].

▶ **Lemma 16** ([15]). *Suppose* $0 < p_{e,\kappa} < 1$ *for every* $(e, \kappa) \in E_{\mathrm{all}}$*. Let* $\boldsymbol{x}^*$ *be an optimal solution to* $\mathrm{LP}_{\mathrm{PoI}}$*. For every* $u \in V$*, fix any subset* $\delta'_u \subseteq \delta_u$*, and let* $E'_u = \{(e, \kappa) \in E_u, e \in \delta'_u\}$ *be the corresponding edge-value pairs.*

We call an edge-value pair (e, κ) *active if* $\kappa_e = \kappa$*. Let*

$$y_{e,\kappa} = \Pr_{\sigma\sim\mathcal{D}_u^{\mathrm{PoI}}, \{\kappa_e\}}[(e, \kappa) \text{ is the first active pair in } \sigma].$$

Then there exists a polynomial-time samplable distribution $\mathcal{D}_u^{\mathrm{PoI}}$ *over the permutations on (subsets of)* E'_u *such that the following holds:*

1. *For all permutation* σ *drawn from* $\mathcal{D}_u^{\mathrm{PoI}}$*, if edge-value pair* (e, κ) *appears in* σ*, then the edge-value pair* (e, w) *appears before* (e, κ) *in* σ *for all* $w \in K_e$ *such that* $w \geq \kappa$*.*
2. $\sum_{\kappa\in K_e} y_{e,\kappa} = \sum_{\kappa\in K_e} x^*_{e,\kappa}$ *for all* $e \in \delta'_u$*.*
3. $\sum_{\kappa\in K_e} y_{e,\kappa} \cdot \kappa \geq \sum_{\kappa\in K_e} x^*_{e,\kappa} \cdot \kappa$ *for all* $e \in \delta'_u$*.*

Now we formally define the sampling scheme S_{PoI}. Initially, each vertex $u \in V$ has its arrival time $t_u \sim U[0, 1]$. We define $\delta'_u = \{(u, w) \in \delta_u \mid t_w < t_u\}$ to be the edges batch that arrives with u, and let $E'_u = \{(e, \kappa) \mid e \in \delta'_u, \kappa \in K_e\}$ be the corresponding edge-value pairs. When u arrives, using Lemma 16, S_{PoI} draw a permutation σ from $\mathcal{D}_u^{\mathrm{PoI}}$ over (subsets of) E'_u.

The subtlety in S_{QC} still exists in S_{PoI}, i.e., S_{PoI} should not query the value of an edge if batched RCRS algorithm R do not accept it. Therefore, S_{PoI} first obtains an indicator vector $\boldsymbol{I}$ from RCRS algorithm R. $I_e = 1$ if and only if R would accept edge e if S_{PoI} choose it to be active. S_{PoI} draws variables κ'_e from the same distribution of κ_e for all $e \in \delta'_u$. If $I_e = 0$, namely R will not take the edge, instead of query the true κ_e, S_{PoI} simply use κ'_e to replace it. In this way, the exact behavior of S_{PoI} depends on $\boldsymbol{I}$. Nevertheless, the returned edge of S_{PoI} is still independent of $\boldsymbol{I}$.

S_{PoI} handles each pair $\sigma_i = (e_i, \kappa_i)$ in order as follows:

- If $I_{e_i} = 1$: S_{PoI} queries the value of κ_{e_i} if it is not queried before. If $\kappa_{e_i} = \kappa_i$, S_{PoI} returns e_i as active edges. Otherwise, it continues to the next edge.
- If $I_{e_i} = 0$: S_{PoI} looks at κ'_{e_i} instead since R will not accept the edge anyway. If $\kappa'_{e_i} = \kappa_i$, S_{PoI} returns e_i as the active edges. Otherwise, it continues to the next edge.

The proof of validity for S_{PoI} is similar to the proof for S_{QC}.

▶ **Lemma 17.** *S_{PoI} is a valid sampling scheme for batched RCRS. In particular, the following three conditions hold:*

1. *each time S_{PoI} will sample at most one edge and the result is independent from $\boldsymbol{I}$.*
2. *Each edge $e \in E$ is active with probability $x_e = \sum_{\kappa \in K_e} x^*_{e,\kappa}$ for any arrival times $\{t_u\}_{u \in V}$. Specifically, the probability that S_{PoI} returns at edge-value pair (e, κ) is exactly $y_{e,\kappa}$ as defined in Lemma 16.*
3. *$\sum_{e \in \delta_u} x_e \le 1$ for all $u \in V$.*

Proof. Firstly, by the procedure of S_{PoI}, it returns only one edge. Since κ' and κ has the same distribution, the edge returned by S_{PoI} is independent of $\boldsymbol{I}$. Secondly, because of this independence, if we define $y_{e,\kappa}$ as in Lemma 16, the probability of S_{PoI} to return at edge-value pair (e, κ) is exactly $y_{e,\kappa}$. By the definition of x_e and part 2 of Lemma 16, we know that $x_e = \sum_{\kappa \in K_e} y_{e,\kappa} = \sum_{\kappa \in K_e} x^*_{e,\kappa}$. Thirdly, from the definition of LP_{PoI}, when $F = E_u$, we have $\sum_{(e,\kappa) \in E_u} x^*_{e,\kappa} \le 1$. Namely, $\sum_{e \in \delta_u} x_e \le 1$. ◀

In order to let the $(v_e - \tau_e)_+$ part of the utility pays for the search cost, the algorithm must accept an edge e if its v_e is larger than τ_e (which only happens when $\kappa_e = \tau_e$). The definition of S_{PoI} guarantees such property, and it will be useful in the proof of Theorem 14.

▶ **Lemma 18.** *Suppose an edge e is queried by the reduction and $\kappa_e = \tau_e$. Then the edge e is always accepted in the end.*

Proof. Since e is queried by the reduction, we know that $I_e = 1$, namely the RCRS algorithm R is willing to take this edge. By part 1 of Lemma 16, if (e, τ_e) is the first edge-value pair associated with e in σ. Therefore, as e is queried by S_{PoI}, it returns e as the active edge when handling (e, τ_e). Since $I_e = 1$, R accepts edge e as well. ◀

Now we are ready to prove Theorem 14, i.e. the approximation ratio of the algorithm.

Proof of Theorem 14. By Lemma 15, we know the optimum of LP_{PoI} is an upper bound of the optimal utility for the price of information problem for weighted matching. Namely, $\sum_{(e,\kappa) \in E_{\text{all}}} x^*_{e,\kappa} \cdot \kappa \ge \text{OPT}_{\text{PoI}}$. Recall K_e is the set of possible values of κ_e, and let V_e be the set of possible values of v_e.

On the other hand, by Lemma 18 we know when $v_e \ge \tau_e$, the edge is always accepted when queried. Together with the definition of τ_e, we know that the cost of query is

$$\begin{aligned}\sum_{e \in E} \Pr[e \text{ is queried}] \cdot c_e &= \sum_{e \in E} \Pr[e \text{ is queried}] \cdot \mathbf{E}_{v_i \sim F_i}\left[(v_e - \tau_e)_+\right] \\ &= \sum_{e \in E} \Pr[e \text{ is accepted}] \cdot \mathbf{E}_{v_i \sim F_i}\left[(v_e - \tau_e)_+\right].\end{aligned}$$

The utility of our algorithm is therefore

$$\begin{aligned}&\sum_{e \in E} \sum_{v \in V_e} \Pr[e \text{ is accepted and } v_e = v] \cdot v - \sum_{e \in E} \Pr[e \text{ is accepted}] \cdot \mathbf{E}_{v_i \sim F_i}\left[(v_e - \tau_e)_+\right] \\ =&\sum_{e \in E} \sum_{\kappa \in K_e} \Pr[e \text{ is accepted and } \kappa_e = \kappa] \cdot \kappa.\end{aligned}$$

The equality follows from $\kappa_e = v_e - (v_e - \tau_e)_+$. Let $y_{e,\kappa}$ be defined as Lemma 16. By Lemma 17, S_{PoI} is a valid sampling scheme. So we can apply the batched RCRS in Theorem 1. Since the only use of κ_e is to determine the active edge in S_{PoI}, conditioning on e is active, whether e is accepted is independent of κ_e.

Hence

$$\begin{aligned}
&\Pr[e \text{ is accepted and } \kappa_e = \kappa] \\
=\ &\Pr[e \text{ is accepted} \mid e \text{ is active and } \kappa_e = \kappa] \cdot \Pr[e \text{ is active and } \kappa_e = \kappa] \\
=\ &\Pr[e \text{ is accepted} \mid e \text{ is active}] \cdot \Pr[e \text{ is active and } \kappa_e = \kappa] \\
\geq\ &\frac{8}{15} y_{e,\kappa}.
\end{aligned}$$

Then by part 2 of Lemma 16, the utility of our algorithm is

$$\sum_{e\in E}\sum_{\kappa\in K_e}\frac{8}{15}y_{e,\kappa_e}\cdot\kappa \geq \frac{8}{15}\sum_{e\in E}\sum_{\kappa\in K_e}x^*_{e,\kappa}\cdot v = \frac{8}{15}\mathrm{OPT}_{\mathrm{PoI}},$$

where the inequality follows from part 3 of Lemma 16. ◀

References

1 Marek Adamczyk and Michal Wlodarczyk. Random order contention resolution schemes. In Mikkel Thorup, editor, *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 790–801. IEEE Computer Society, 2018.

2 Saeed Alaei. Bayesian combinatorial auctions: Expanding single buyer mechanisms to many buyers. In *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*, pages 512–521, 2011.

3 Jonathan Aronson, Martin Dyer, Alan Frieze, and Stephen Suen. Randomized greedy matching. ii. *Random Structures & Algorithms*, 6(1):55–73, 1995.

4 Itai Ashlagi, Maximilien Burq, Chinmoy Dutta, Patrick Jaillet, Amin Saberi, and Chris Sholley. Edge weighted online windowed matching. In *EC*, pages 729–742. ACM, 2019.

5 Simon Bruggmann and Rico Zenklusen. An optimal monotone contention resolution scheme for bipartite matchings via a polyhedral viewpoint. *Mathematical Programming*, pages 1–51, 2020.

6 T.-H. Hubert Chan, Fei Chen, Xiaowei Wu, and Zhichao Zhao. Ranking on arbitrary graphs: Rematch via continuous linear programming. *SIAM J. Comput.*, 47(4):1529–1546, 2018.

7 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.

8 Chandra Chekuri, Jan Vondrák, and Rico Zenklusen. Submodular function maximization via the multilinear relaxation and contention resolution schemes. *SIAM J. Comput.*, 43(6):1831–1879, 2014.

9 Ning Chen, Nicole Immorlica, Anna R. Karlin, Mohammad Mahdian, and Atri Rudra. Approximating matches made in heaven. In *Automata, languages and programming. Part I*, volume 5555 of *Lecture Notes in Comput. Sci.*, pages 266–278. Springer, Berlin, 2009.

10 Nikhil R. Devanur and Kamal Jain. Online matching with concave returns [extended abstract]. In *STOC'12—Proceedings of the 2012 ACM Symposium on Theory of Computing*, pages 137–143. ACM, New York, 2012.

11 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*, pages 700–714. SIAM, 2018.

12 Hossein Esfandiari, MohammadTaghi Hajiaghayi, Vahid Liaghat, and Morteza Monemizadeh. Prophet secretary. *SIAM Journal on Discrete Mathematics*, 31(3):1685–1701, 2017.

13 Tomer Ezra, Michal Feldman, Nick Gravin, and Zhihao Gavin Tang. Online stochastic max-weight matching: Prophet inequality for vertex and edge arrival models. In Péter Biró, Jason Hartline, Michael Ostrovsky, and Ariel D. Procaccia, editors, *EC '20: The 21st ACM Conference on Economics and Computation, Virtual Event, Hungary, July 13-17, 2020*, pages 769–787. ACM, 2020.

14 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.

15 Buddhima Gamlath, Sagar Kale, and Ola Svensson. Beating greedy for stochastic bipartite matching. In Timothy M. Chan, editor, *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 2841–2854. SIAM, 2019.

16 Anupam Gupta, Haotian Jiang, Ziv Scully, and Sahil Singla. The markovian price of information. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 233–246. Springer, 2019.

17 Anupam Gupta and Viswanath Nagarajan. A stochastic probing problem with applications. In Michel X. Goemans and José R. Correa, editors, *Integer Programming and Combinatorial Optimization - 16th International Conference, IPCO 2013, Valparaíso, Chile, March 18-20, 2013. Proceedings*, volume 7801 of *Lecture Notes in Computer Science*, pages 205–216. Springer, 2013.

18 Guru Guruganesh and Euiwoong Lee. Understanding the correlation gap for matchings. In *37th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science*, 2018.

19 Zhiyi Huang, Ning Kang, Zhihao Gavin Tang, Xiaowei Wu, Yuhao Zhang, and Xue Zhu. Fully online matching. *J. ACM*, 67(3):17:1–17:25, 2020.

20 Zhiyi Huang, Zhihao Gavin Tang, Xiaowei Wu, and Yuhao Zhang. Fully online matching II: beating ranking and water-filling. In *FOCS*, pages 1380–1391. IEEE, 2020.

21 Richard M. Karp, Umesh V. Vazirani, and Vijay V. Vazirani. An optimal algorithm for on-line bipartite matching. In *STOC*, pages 352–358. ACM, 1990.

22 Robert D. Kleinberg, Bo Waggoner, and E. Glen Weyl. Descending price optimally coordinates search. In Vincent Conitzer, Dirk Bergemann, and Yiling Chen, editors, *Proceedings of the 2016 ACM Conference on Economics and Computation, EC '16, Maastricht, The Netherlands, July 24-28, 2016*, pages 23–24. ACM, 2016.

23 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, August 20-22, 2018, Helsinki, Finland*, volume 112 of *LIPIcs*, pages 57:1–57:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

24 Brendan Lucier. An economic view of prophet inequalities. *SIGecom Exch.*, 16(1):24–47, 2017.

25 Aranyak Mehta. Online matching and ad allocation. *Found. Trends Theor. Comput. Sci.*, 8(4):265–368, 2012.

26 Aranyak Mehta, Amin Saberi, Umesh Vazirani, and Vijay Vazirani. AdWords and generalized online matching. *J. ACM*, 54(5):Art. 22, 19, 2007.

27 Sahil Singla. The price of information in combinatorial optimization. In Artur Czumaj, editor, *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 2523–2532. SIAM, 2018.

28 Zhihao Gavin Tang, Xiaowei Wu, and Yuhao Zhang. Towards a better understanding of randomized greedy matching. In *STOC*, pages 1097–1110. ACM, 2020.

29 Martin L Weitzman. Optimal search for the best alternative. *Econometrica: Journal of the Econometric Society*, pages 641–654, 1979.