Online Bipartite Matching and Adwords

Vijay V. Vazirani*\(^1\)

\(^1\)University of California, Irvine

Abstract

A simple and optimal online algorithm for online bipartite matching, called RANKING, was given in [KVV90]; however, its analysis was difficult to comprehend. Over the years, several researchers contributed valuable ideas to simplifying its proof. We start by observing that the past proofs were incomplete; we have identified the missing piece and named it the no-surpassing property.

The simplicity of the final proof naturally raised the possibility of extending RANKING to the adwords problem. We first managed to extend it to a subcase of the latter, called SINGLE-VALUED. However, a further extension, to the subcase called SMALL, in which bids are small compared to budgets, faced a major hurdle, namely failure of the no-surpassing property. Since SMALL has been of considerable practical significance in ad auctions [MSVV07] and our approach has distinct advantages over [MSVV07], we have stated our result as a Conditional Theorem after assuming the no-surpassing property. We leave the open problem of removing this assumption and salvaging the (substantial) ideas that were needed to analyze the algorithm, modulo the assumption.

\*Supported in part by NSF grant CCF-1815901.
1 Introduction

The online bipartite matching problem (OBM) occupies a central place not only in online algorithms but also in matching-based market design, see details in Sections 1.2 and 1.1. For formal statements of problems studied in this paper, see Section 2. For OBM, a simple, optimal, randomized online algorithm, called RANKING, was given in [KVV90]. Its competitive ratio is \( (1 - \frac{1}{e}) \) and [KVV90] showed that no randomized online algorithm can achieve a better ratio than \( (1 - \frac{1}{e}) + o(1) \).

The analysis of RANKING given in [KVV90] was considered “extremely difficult”. Over the years, several researchers contributed valuable ideas to simplifying its proof; indeed, our work would not have been possible without these ideas, see details in Section 1.2. These proofs involve methodology from two domains, probability theory and combinatorics, with the former playing a dominant role. The simplicity of the final proof [EFFS21] naturally raises the possibility of extending RANKING to generalizations of OBM. Our attempt at this had the effect of applying a stress test to the proof. As a result, the probabilistic part got simplified and the combinatorial part was found to be incomplete, see details in Sections 1.2 and 1.3.

We have identified the missing piece and named it the no-surpassing property, see Property 10. We have also given simple combinatorial facts which complete the argument. Our paper starts by presenting this simple and complete analysis\(^1\) of RANKING. Our goal was to extend RANKING all the way to the adwords problem, called GENERAL in this paper, or even its special case, called SMALL. Informally, GENERAL involves matching keyword queries, as they arrive online, to advertisers having budget limits; SMALL is the special case in which bids are small compared to budgets.

The latter problem captures a key computational issue that arises in the context of ad auctions, for instance in Google’s AdWords marketplace. An optimal algorithm achieving a competitive ratio of \((1 - \frac{1}{e})\) was first given in [MSVV07]; for the impact of this result in the marketplace, see 1.1. We note that previous attempts at extending RANKING to SMALL did not meet with success; the alternative approach followed by [MSVV07] is described in Section 1.2. In contrast, GENERAL is a notoriously difficult problem and has remained largely unresolved; see below for marginal progress made recently.

Considering the difficulty of GENERAL, we decided to first study an intermediate problem, which we call SINGLE-VALUED. Under this problem, each bidder can make bids of one value only, although the value may be different for different bidders. For reasons explained in Section 1.3, we sought an algorithm for this problem from first principles, without reducing it to other problems. As detailed in Section 1.3, new ideas were required on both fronts, probabilistic and combinatorial, to obtain an optimal algorithm for SINGLE-VALUED.

In the final step, we attempted an extension from SINGLE-VALUED to GENERAL. It turns out that GENERAL suffers from an inherent structural difficulty, explained in Section 5.1, which we can circumvent for the special case of SMALL, by using the notion of “fake” money. However, this still does not yield an algorithm for SMALL. The reason is that in going from SINGLE-VALUED to SMALL, the no-surpassing property fails to hold. This came as a surprise, since in

\(^1\)The proof presented in this paper is meant to be a “textbook quality” exposition and will appear in the chapter [EIV22] of an upcoming edited book on matching markets.
the proof of RANKING, this property was a small aspect of the combinatorial part, which was a small part of the entire proof!

Since the ideas underlying the algorithm and the rest of the proof of correctness are quite substantial, it was not appropriate to simply abandon them, especially considering the potential gains if the hurdle can be overcome; these gains are specified below. For these reasons, in Section 5.2, we have assumed that the property of no-surpassing and stated our result for SMALL as Conditional Theorem 39. We leave the open problem of filling the gap.

The previous algorithms for SMALL [MSVV07, BJN07] were obtained by viewing SMALL as a generalization of \( b \)-matching, for which an optimal (deterministic) algorithm, called BALANCE, was given in [KP00]. In turn, the algorithms of [MSVV07, BJN07] can we viewed as extensions of BALANCE; both are LP-duality-based and are deterministic, see Section 1.2 for further details. In contrast, our approach for SMALL builds directly on OBM and RANKING, a more basic approach. In turn, it has distinct advantages in the ad auctions marketplace, as detailed below.

Our algorithm for SMALL, with its conditional proof, is more elementary than the previous algorithm [MSVV07]. The effective bid of each bidder \( j \) for a query is simply its bid multiplied by its price \( p_j = e^{w_j} - 1 \), where \( w_j \), called the rank of \( j \), is picked at random from \([0,1]\). On the other hand, the effective bid in [MSVV07] is the bid multiplied by \( (1 - e^{L_j/B_j}) \), where \( B_j \) and \( L_j \) are the total budget and the leftover budget of bidder \( j \), respectively. As a result, whereas the algorithm of [MSVV07] needs to know the total budget of each bidder, our algorithm does not. During its run, our algorithm only needs to know whether the budget of a bidder has been exhausted. Yet, its revenue is compared to the optimal revenue generated by an offline algorithm with full knowledge of the budget. This budget-obliviousness gives our approach a distinct advantage, since it can be used in autobidding platforms [ABM19, DM22], which dynamically adjust the bids and budgets of advertisers over multiple search engines to improve performance.

For GENERAL, the greedy algorithm, which matches each query to the highest bidder, achieves a competitive ratio of 1/2. Until recently, that was the best possible. In [HZZZ20] a marginally improved algorithm, with a ratio of 0.5016, was given. It is important to point out that this 60-page paper was a tour-de-force, drawing on a diverse collection of ideas — a testament to the difficulty of this problem.

**Remark 1.** The objective of all adwords problems studied in this paper is to maximize the total revenue accrued by the online algorithm. In economics, such a solution is referred to as efficient, since the amount bid by an advertiser is indicative of how useful the query is to it, and hence to the economy.

### 1.1 Significance and Practical Impact

Google’s AdWords marketplace generates multi-billion dollar revenues annually and the current annual worldwide spending on digital advertising is almost half a trillion dollars. These revenues of Google and other Internet services companies enable them to offer crucial services, such as search, email, videos, news, apps, maps etc. for free — services that have virtually transformed our lives.
We note that SMALL is the most relevant case of adwords for the search ads marketplace e.g., see [DM22]. A remarkable feature of Google, and other search engines, is the speed with which they are able to show search results, often in milliseconds. In order to show ads at the same speed, together with search results, the solution for SMALL needed to be minimalistic in its use of computing power, memory and communication.

The online algorithm of [MSVV07] satisfied these criteria and therefore had a substantial impact in this marketplace. Furthermore, the idea underlying their algorithm was extracted into a simple heuristic, called bid scaling, which uses even less computation and is widely used by search engine companies today. Our randomized algorithm for SMALL is even more elementary as mentioned above. Unlike [MSVV07], it does not need to maintain the leftover budget of an advertiser. In fact, it is budget-oblivious — it does not even need to know the budgets of advertisers.

It will be useful to view the AdWords marketplace in the context of a bigger revolution, namely the advent of the Internet and mobile computing, and the consequent resurgence of the area of matching-based market design. The birth of this area goes back to the seminal 1962 paper of Gale and Shapley on stable matching [GS62]. Over the decades, this area became known for its highly successful applications, having economic as well as sociological impact. These included matching medical interns to hospitals, students to schools in large cities, and kidney exchange.

The resurgence led to a host of highly innovative and impactful applications. Besides the AdWords marketplace, which matches queries to advertisers, these include Uber, matching drivers to riders; Upwork, matching employers to workers; and Tinder, matching people to each other, see [Ins19] for more details.

A successful launch of such markets calls for economic and game-theoretic insights, together with algorithmic ideas. The Gale-Shapley deferred acceptance algorithm and its follow-up works provided the algorithmic backbone for the “first life” of matching-based market design. The algorithm RANKING has become the paradigm-setting algorithmic idea in the “second life” of this area. Interestingly enough, this result was obtained in the pre-Internet days, over thirty years ago.

1.2 Related Works

We start by describing simplifications to the proof of RANKING for OBM. The first simplifications, in [GM08, BM08], got the ball rolling, setting the stage for the substantial simplification given in [DJK13], using a randomized primal-dual approach. [DJK13] introduced the idea of splitting the contribution of each matched edge into primal and dual contributions and lower-bounding each part separately. Their method for defining prices $p_j$ of goods, using randomization, was used by subsequent papers, including this one$^2$.

Interestingly enough, the next simplification involved removing the scaffolding of LP-duality and casting the proof in purely probabilistic terms$^3$, using notions from economics to split the contribution of each matched edge into the contributions of the buyer and the seller. This elegant

$^2$For a succinct proof of optimality of the underlying function, $e^{x-1}$, see Section 2.1.1 in [HT22].

$^3$Even though there is no overt use of LP-duality in the proof of [EFFS21], it is unclear if this proof could have been obtained directly, without going the LP-duality-route.
analysis was given by [EFFS21]. We note that when we move to generalizations of OBM, even this economic interpretation needs to be dropped, see Remark 17.

We now point out the exact places in the last two papers in which the proofs were incomplete. In Lemma 1, [DJK13] state “Since \( Y_i < y' \), \( j \) is matched to \( i' \); note that in this paper, goods are indexed by \( i \) and buyers by \( j \). At this place, the proof needs to argue that there is no other good \( i' \), which is available to \( j \) and satisfies \( Y_{i'} < Y_i < y' \). Next, in Observation 1 in the proof of Lemma 2.3, [EFFS21] state, “This follows since buyer \( l \) derives higher utility from item \( r_j \) than from its match in \( M_{-l} \).” Once again, the proof needs to show that in the run with all goods, an even better match than \( r_j \) does not show up for buyer \( l \).

An important generalization of OBM is online \( b \)-matching. This problem is a special case of GENERAL in which the budget of each advertiser is \( b \) and the bids are 0/1. [KP00] gave a simple optimal online algorithm, called BALANCE, for this problem. BALANCE awards the next query to the interested bidder who has been matched least number of times so far. [KP00] showed that as \( b \) tends to infinity, the competitive ratio of BALANCE tends to \((1 - \frac{1}{e})\).

The importance of online \( b \)-matching arises from the fact that it is a special case of SMALL, if \( b \) is large. Indeed, the first online algorithm [MSVV07] for SMALL was obtained by extending BALANCE as follows: [MSVV07] first gave a simpler proof of the competitive ratio of BALANCE using the notion of a factor-revealing LP [JMM+03]. Then they gave the notion of a tradeoff-revealing LP, which yielded an algorithm achieving a competitive ratio of \((1 - \frac{1}{e})\). [MSVV07] also proved that this is optimal for \( b \)-matching, and hence SMALL, by proving that no randomized algorithm can achieve a better ratio for online \( b \)-matching; previously, [KP00] had shown a similar result for deterministic algorithms. Following [MSVV07], a second optimal online algorithm for SMALL was given in [BJN07], using a primal-dual approach.

Another relevant generalization of OBM is online vertex weighted matching, in which the offline vertices have weights and the objective is to maximize the weight of the matched vertices. [AGKM11] extended RANKING to obtain an optimal online algorithm for this problem. Clearly, SINGLE-VALUED is intermediate between GENERAL and online vertex weighted matching; moreover, it can be reduced to the latter by creating \( k_j \) copies of each advertiser \( j \). As observed by [AGKM11], via this reduction, their algorithm for online vertex weighted matching yields an optimal online algorithm for SINGLE-VALUED, see Section 1.3 for additional comments on this.

In the decade following the conference version (FOCS 2005) of [MSVV07], search engine companies generously invested in research on models derived from OBM and adwords. Their motivation was two-fold: the substantial impact of [MSVV07] and the emergence of a rich collection of digital ad tools. It will be impossible to do justice to this substantial body of work, involving both algorithmic and game-theoretic ideas; for a start, see the surveys [Meh13, HT22].

1.3 Technical Ideas

Our proof of RANKING involves two new ideas, one in each aspect of the proof, probabilistic and combinatorial. The first is a new random variable, \( u_e \), called threshold, corresponding to each edge \( e = (i, j) \) in the underlying graph, see Definition 9. The key fact needed in the analysis of RANKING is that for any edge \( (i, j) \), its expected contribution is at least \((1 - 1/e)\), and our proof of this fact crucially uses the threshold random variable for edge \( (i, j) \).
The second is Lemma 7 and Corollary 8. The proof of the no-surpassing property becomes quite transparent with these facts. Moreover, these facts also extend in a seamless manner to Lemma 20 and Corollary 21, which are required for an analogous purpose in the analysis of SINGLE-VALUED.

As noted in Section 1.2, RANKING has been extended all the way to SINGLE-VALUED. Our goal is to extend it to GENERAL, and thereby address SMALL and k-TYPICAL. However, GENERAL is very different from SINGLE-VALUED in the following sense. Whereas the latter can be reduced to online vertex weighted matching, the former cannot. The reason is that the manner in which budget $B_j$ of bidder $j$ gets partitioned into bids is not predictable; it depending on the queries, their order of arrival and the randomization executed in a run of the algorithm. Therefore, in order to solve GENERAL, we will first need to solve SINGLE-VALUED without reducing it to online vertex weighted matching. An immediate advantage is that such an algorithm for SINGLE-VALUED will require fewer random bits — only one random rank for each bidder $j$, as opposed one rank for each of the $k_j$ copies of $j$.

This is done in Algorithm 19. Its analysis requires several new ideas. First, since vertex $j$ is not split into $k_j$ copies, we cannot talk about the contribution of edges anymore. Even worse, we don’t have individual vertices for keeping track of the revenue accrued from each match, as per the scheme of [EFFS21].

Our algorithm gets around this difficulty by accumulating revenue in the same “account” each time bidder $j$ gets matched. The corresponding random variable, $r_j$, is called the total revenue of bidder $j$, for want of a better name, see Remark 17. Lower bounding $\mathbb{E}[r_j]$ is much more tricky than lower bounding the revenue of a good in OBM, since it involves “teasing apart” the $k_j$ accumulations made into this account.

A replacement is also needed for the key lemma in the analysis of RANKING, namely Lemma 13, which lower bounds the contribution of each edge. For this purpose, we give the notion of a $j$-star, denoted $X_j$, which consists of bidder $j$ together with edges to $k_j$ of its neighbors in $G$, see Definition 23. The contribution of $j$-star $X_j$, is denoted by $\mathbb{E}[X_j]$, which is also defined in Definition 23. Finally, using the lower bound on $\mathbb{E}[r_j]$, Lemma 26 gives a lower $\mathbb{E}[X_j]$ for every $j$-star, $X_j$. This lemma crucially uses a new random variable, called truncated threshold, see Definition 22.

Next, we explain the reason for truncation in the definition of this random variable. Consider bidder $j$ and a query $i_l$ that is desired by $j$. Observe that in run $R_j$, query $i_l$ can get a bid as large as $B \cdot (1 - \frac{1}{\varepsilon})$, where $B = \max_{k \in A} \{ b_k \}$, whereas the largest bid that $j$ can make to $i_l$ is $b_j \cdot (1 - \frac{1}{\varepsilon})$; in general, $b_j$ may be smaller than $B$. Now, $i_l$ contributes revenue to $r_j$ only if $i_l$ is matched to $j$ in run $R_j$, an event which will definitely not happen if $u_{e_l} > b_j \cdot (1 - \frac{1}{\varepsilon})$. Therefore, whenever $u_{e_l} \in [b_j \cdot (1 - \frac{1}{\varepsilon}), B \cdot (1 - \frac{1}{\varepsilon})]$, the contribution to $r_j$ is zero. By truncating $u_{e_l}$ to $b_j \cdot (1 - \frac{1}{\varepsilon})$, we have effectively changed the probability density function of $u_{e_l}$ so that the probability of the event $u_{e_l} \in [b_j \cdot (1 - \frac{1}{\varepsilon}), B \cdot (1 - \frac{1}{\varepsilon})]$ is now concentrated at the event $u_{e_l} = b_j \cdot (1 - \frac{1}{\varepsilon})$. From the viewpoint of lower bounding the revenue accrued in $r_j$, the two probability density functions are equivalent since the revenue accrued is zero under both these events. On the other hand, the truncated random variable enables us to apply the law of total expectation, in the proof of Lemma 26, in the same way as it was done in the proof of lemma 11, without introducing more
difficulties.

Finally, Algorithm 33 for GENERAL needs to get around the structural difficulties mentioned in Section 5.1. The idea of “fake” money helps partially finesse this problem, since for SMALL, the fake money can be upper-bounded in the worst case.

2 Preliminaries

Online Bipartite Matching (OBM): Let \( B \) be a set of \( n \) buyers and \( S \) a set of \( n \) goods. A bipartite graph \( G = (B, S, E) \) is specified on vertex sets \( B \) and \( S \), and edge set \( E \), where for \( i \in B \), \( j \in S \), \((i, j) \in E \) if and only if buyer \( i \) likes good \( j \). \( G \) is assumed to have a perfect matching and therefore each buyer can be given a unique good she likes. Graph \( G \) is revealed in the following manner. The \( n \) goods are known up-front. On the other hand, the buyers arrive one at a time, and when buyer \( i \) arrives, the edges incident at \( i \) are revealed. We are required to design an online algorithm \( A \) in the following sense. At the moment a buyer \( i \) arrives, the algorithm needs to match \( i \) to one of its unmatched neighbors, if any; if all of \( i \)'s neighbors are matched, \( i \) remains unmatched. The difficulty is that the algorithm does not “know” the edges incident at buyers which will arrive in the future and yet the size of the matching produced by the algorithm will be compared to the best off-line matching; the latter of course is a perfect matching. The formal measure for the algorithm is defined in Section 2.1.

General Adwords Problem (GENERAL): Let \( A \) be a set of \( m \) advertisers, also called bidders, and \( Q \) be a set of \( n \) queries. A bipartite graph \( G = (Q, A, E) \) is specified on vertex sets \( Q \) and \( A \), and edge set \( E \), where for \( i \in Q \) and \( j \in A \), \((i, j) \in E \) if and only if bidder \( j \) is interested in query \( i \). Each query \( i \) needs to be matched to at most one bidder who is interested in it. For each edge \((i, j)\), bidder \( j \) knows his bid for \( i \), denoted by \( \text{bid}(i, j) \in \mathbb{Z}_+ \). Each bidder also has a budget \( B_j \in \mathbb{Z}_+ \) which satisfies \( B_j \geq \text{bid}(i, j) \), for each edge \((i, j)\) incident at \( j \).

Graph \( G \) is revealed in the following manner. The \( m \) bidders are known up-front and the queries arrive one at a time. When query \( i \) arrives, the edges incident at \( i \) are revealed, together with the bids associated with these edges. If \( i \) gets matched to \( j \), then the matched edge \((i, j)\) is assigned a weight of \( \text{bid}(i, j) \). The constraint on \( j \) is that the total weight of matched edges incident at \( j \) be at most \( B_j \). The objective is to maximize the total weight of all matched edges at all bidders.

Adwords under Single-Valued Bidders (SINGLE-VALUED): SINGLE-VALUED is a special case of GENERAL in which each bidder \( j \) will make bids of a single value, \( b_j \in \mathbb{Z}_+ \), for the queries he is interested in. If \( i \) accepts \( j \)'s bid, then \( i \) will be matched to \( j \) and the weight of this matched edge will be \( b_j \). Corresponding to each bidder \( j \), we are also given \( k_j \in \mathbb{Z}_+ \), the maximum number of times \( j \) can be matched to queries. The objective is to maximize the total weight of matched edges. Observe that the matching \( M \) found in \( G \) is a \( b \)-matching with the \( b \)-value of each query \( i \) being 1 and of advertiser \( j \) being \( k_j \).

Adwords under Small Bids (SMALL): SMALL is a special case of GENERAL in which for each bidder \( j \), each bid of \( j \) is small compared to its budget. Formally, we will capture this condition

\[ 4 \text{Clearly, this is not a matching in the usual sense, since a bidder may be matched to several queries.} \]
by imposing the following constraint. For a valid instance $I$ of SMALL, define

$$\mu(I) = \max_{j \in A} \left\{ \frac{\max_{(i,j) \in E} \{\text{bid}(i,j) - 1\}}{B_j} \right\}.$$ 

Then we require that

$$\lim_{n(I) \to \infty} \mu(I) = 0,$$

where $n(I)$ denotes the number of queries in instance $I$.

### 2.1 The competitive ratio of online algorithms

We will define the notion of competitive ratio of a randomized online algorithm in the context of OBM.

**Definition 2.** Let $G = (B, S, E)$ be a bipartite graph as specified above. The competitive ratio of a randomized algorithm $A$ for OBM is defined to be:

$$c(A) = \min_{G=(B,S,E)} \min_{\rho(B)} \frac{\mathbb{E}[A(G,\rho(B))]}{n},$$

where $\mathbb{E}[A(G,\rho(B))]$ is the expected size of matching produced by $A$; the expectation is over the random bits used by $A$. We may assume that the worst case graph and the order of arrival of buyers, given by $\rho(B)$, are chosen by an adversary who knows the algorithm. It is important to note that the algorithm is provided random bits after the adversary makes its choices.

**Remark 3.** For each problem studied in this paper, we will assume that the offline matching is complete. It is easy to extend the arguments, without changing the competitive ratio, in case the offline matching is not complete. As an example, this is done for OBM in Remark 16.

### 3 Online Bipartite Matching: RANKING

Algorithm 4 presents an optimal algorithm for OBM. Note that this algorithm picks a random permutation of goods only once. Its competitive ratio is $(1 - \frac{1}{e})$, as shown in Theorem 15. Furthermore, as shown in [KVV90], it is an optimal online bipartite matching algorithm: no randomized algorithm can do better, up to an $o(1)$ term.

We will analyze Algorithm 6 which is equivalent to Algorithm 4 and operates as follows. Before the execution of Step (1), the adversary determines the order in which buyers will arrive, say $\rho(B)$. In Step (1), each good $j$ is assigned a price $p_j = e^{w_j - 1}$, where $w_j$, called the rank of $j$, is picked at random from $[0, 1]$; observe that $p_j \in [\frac{1}{e}, 1]$. In Step (2), buyers will arrive in the order $\rho(B)$, picked by the adversary, and will be matched to the cheapest available good. With probability 1 all $n$ prices are distinct and sorting the goods by increasing prices results in a random permutation. Furthermore, since Algorithm 6 uses this sorted order only and is oblivious of the actual prices, it is equivalent to Algorithm 4. As we will see, the random variables representing actual prices are crucially important as well – in the analysis. We remark that for the generalizations of OBM studied in this paper, the prices are used not only in the analysis, but also by the algorithms.
Algorithm 4. (Algorithm RANKING)

1. **Initialization:** Pick a random permutation, $\pi$, of the goods in $S$.

2. **Online buyer arrival:** When a buyer, say $i$, arrives, match her to the first unmatched good she likes in the order $\pi$; if none, leave $i$ unmatched.

Output the matching, $M$, found.

3.1 **Analysis of RANKING**

We will use an economic setting for analyzing Algorithm 6 as follows. Each buyer $i$ has unit-demand and 0/1 valuations over the goods she likes, i.e., she accrues unit utility from each good she likes, and she wishes to get at most one of them. The latter set is precisely the set of neighbors of $i$ in $G$. If on arrival of $i$ there are several of these which are still unmatched, $i$ will pick one having the smallest price\(^5\). Therefore the buyers will maximize their utility as defined below.

For analyzing this algorithm, we will define two sets of random variables, $u_i$ for $i \in B$ and $r_j$, for $j \in S$. These will be called utility of buyer $i$ and revenue of good $j$, respectively. Each run of RANKING defines these random variables as follows. If RANKING matches buyer $i$ to good $j$, then define $u_i = 1 - p_j$ and $r_j = p_j$, where $p_j$ is the price of good $j$ in this run of RANKING. Clearly, $p_j$ is also a random variable, which is defined by Step (1) of the algorithm. If $i$ remains unmatched, define $u_i = 0$, and if $j$ remains unmatched, define $r_j = 0$. Observe that for each good $j$, $p_j \in [\frac{1}{e}, 1]$ and for each buyer $i$, $u_i \in [0, 1 - \frac{1}{e}]$. Let $M$ be the matching produced by RANKING and let random variable $|M|$ denote its size.

Lemma 5 pulls apart the contribution of each matched edge $(i, j)$ into $u_i$ and $r_j$. Next, we established in Lemma 13 that for each edge $(i, j)$ in the graph, the total expected contribution of $u_i$ and $r_j$ is at least $1 - \frac{1}{2}$. Then, linearity of expectation allows us to reassemble the $2n$ terms in the right hand side of Lemma 5 so they are aligned with a perfect matching in $G$, and this yields Theorem 15.

**Lemma 5.**

$$\mathbb{E}[|M|] = \sum_i \mathbb{E}[u_i] + \sum_j \mathbb{E}[r_j].$$

**Proof.** By definition of the random variables,

$$\mathbb{E}[|M|] = \mathbb{E}\left[\sum_{i=1}^n u_i + \sum_{j=1}^n r_j\right] = \sum_i \mathbb{E}[u_i] + \sum_j \mathbb{E}[r_j],$$

where the first equality follows from the fact that if $(i, j) \in M$ then $u_i + r_j = 1$ and the second follows from linearity of expectation.

\(^5\)As stated above, with probability 1 there are no ties.
Algorithm 6. (Algorithm RANKING: Economic Viewpoint)

1. **Initialization:** \( \forall j \in S: \) Pick \( w_j \) independently and uniformly from \([0, 1]\).
   Set price \( p_j \leftarrow e^{w_j} - 1 \).

2. **Online buyer arrival:** When a buyer, say \( i \), arrives, match her to the cheapest unmatched good she likes; if none, leave \( i \) unmatched.

Output the matching, \( M \), found.

While running Algorithm 6, assume that the adversary has picked the order of arrival of buyers, say \( \rho(B) \), and Step (1) has been executed. We next define several ways of executing Step (2). Let \( \mathcal{R} \) denote the run of Step (2) on the entire graph \( G \). Corresponding to each good \( j \), let \( G_j \) denote graph \( G \) with vertex \( j \) removed. Define \( \mathcal{R}_j \) to be the run of Step (2) on graph \( G_j \).

Lemma 7 and Corollary 8 establish a relationship between the sets of available goods for a buyer \( i \) in the two runs \( \mathcal{R} \) and \( \mathcal{R}_j \); the latter is crucially used in the proof of Lemma 11. For ease of notation in proving these two facts, let us renumber the buyers so their order of arrival under \( \rho(B) \) is 1, 2, \ldots, \( n \). Let \( T(i) \) and \( T_j(i) \) denote the sets of unmatched goods at the time of arrival of buyer \( i \) (i.e., just before the buyer \( i \) gets matched) in the graphs \( G \) and \( G_j \), in runs \( \mathcal{R} \) and \( \mathcal{R}_j \), respectively. Similarly, let \( S(i) \) and \( S_j(i) \) denote the set of unmatched goods that buyer \( i \) is incident to in \( G \) and \( G_j \), in runs \( \mathcal{R} \) and \( \mathcal{R}_j \), respectively.

We have assumed that Step (1) of Algorithm 6 has already been executed and a price \( p_k \) has been assigned to each good \( k \). With probability 1, the prices are all distinct. Let \( F_1 \) and \( F_2 \) be subsets of \( S \) containing goods \( k \) such that \( p_k < p_j \) and \( p_k > p_j \), respectively.

**Lemma 7.** For each \( i, 1 \leq i \leq n \), the following hold:

1. \((T_j(i) \cap F_1) = (T(i) \cap F_1)\).
2. \((T_j(i) \cap F_2) \subseteq (T(i) \cap F_2)\).

**Proof.** Clearly, in both runs, \( \mathcal{R} \) and \( \mathcal{R}_j \), any buyer having an available good in \( F_1 \) will match to the most profitable one of these, without even considering the rest of the goods. Since \( j \notin F_1 \), the two runs behave in an identical manner on the set \( F_1 \), thereby proving the first statement.

The proof of the second statement is by induction on \( i \). The base case is trivially true since \( j \notin F_2 \). Assume that the statement is true for \( i = k \) and let us prove it for \( i = k + 1 \). By the first statement, we need to consider only the case that there are no available goods for the \( k^{th} \) buyer in \( F_1 \) in the runs \( \mathcal{R} \) and \( \mathcal{R}_j \). Assume that in run \( \mathcal{R}_j \), this buyer gets matched to good \( l \); if she remains unmatched, we will take \( l \) to be null. Clearly, \( l \) is the most profitable good she is incident to in \( T_j(\hat{k}) \). Therefore, the most profitable good she is incident to in run \( \mathcal{R} \) is the best of \( l \), the most profitable good in \( T(k) - T_j(k) \), and \( j \), in case it is available. In each of these cases, the induction step holds. \( \Box \)
In the corollary below, the first two statements follow from Lemma 7 and the third statement follows from the first two.

**Corollary 8.** For each \( i, 1 \leq i \leq n \), the following hold:

1. \((S_j(i) \cap F_1) = (S(i) \cap F_1)\).
2. \((S_j(i) \cap F_2) \subseteq (S(i) \cap F_2)\).
3. \(S_j(i) \subseteq S(i)\).

Next we define a new random variable, \( u_e \), for each edge \( e = (i, j) \in E \). This is called the threshold for edge \( e \) and is given in Definition 9. It is critically used in the proofs of Lemmas 11 and 13.

**Definition 9.** Let \( e = (i, j) \in E \) be an arbitrary edge in \( G \). Define random variable, \( u_e \), called the threshold for edge \( e \), to be the utility of buyer \( i \) in run \( R_j \). Clearly, \( u_e \in [0, 1 - \frac{1}{e}] \).

**Property 10.** (No-Surpassing for OBM) Let \( p_j \) be such that the bid of \( j \), namely \( 1 - p_j \), is better than the best bid that buyer \( i \) gets in run \( R_j \). Then, in run \( R \), no bid to \( i \) will surpass \( 1 - p_j \).

**Lemma 11.** Corresponding to each edge \((i, j) \in E \), the following hold.

1. \( u_i \geq u_e \), where \( u_i \) and \( u_e \) are the utilities of buyer \( i \) in runs \( R \) and \( R_j \), respectively.
2. Let \( z \in [0, 1 - \frac{1}{e}] \). Conditioned on \( u_e = z \), if \( p_j < 1 - z \), then \( j \) will definitely be matched in run \( R \).

**Proof.** 1) By the third statement of Corollary 8, \( i \) has more options in run \( R \) as compared to run \( R_j \), and therefore \( u_i \geq u_e \).

2) In run \( R \), if \( j \) is already matched when \( i \) arrives, there is nothing to prove. Next assume that \( j \) is not matched when \( i \) arrives. The crux of the matter is to prove that the no-surpassing property holds; if so, in run \( R \), \( i \) will not have any option that is better than \( j \) and will therefore get matched to \( j \). Since \( 1 - p_j > z \), \( S_j(i) \cap F_1 = \emptyset \). Therefore by the first statement of Corollary 8, \( S(i) \cap F_1 = \emptyset \). Since \( i \) will get no bid better than \( j \) in \( R \), the no-surpassing property indeed holds and \( i \) must get matched to \( j \). \( \Box \)

**Remark 12.** The random variable \( u_e \) is called threshold because of the second statement of Lemma 11. It defines a value such that whenever \( p_j \) is smaller than this value, \( j \) is definitely matched in run \( R \).

The intuitive reason for the next, and most crucial, lemma is the following. The smaller \( u_e \) is, the larger is the range of values for \( p_j \), namely \([0, 1 - u_e]\), over which \((i, j)\) will be matched and \( j \) will accrue revenue of \( p_j \). Integrating \( p_j \) over this range, and adding \( E[u_i] \) to it, gives the desired bound. Crucial to this argument is the fact that \( p_j \) is independent of \( u_e \). This follows from the fact that \( u_e \) is determined by run \( R_j \) on graph \( G_j \), which does not contain vertex \( j \).
Lemma 13. Corresponding to each edge \((i, j) \in E\),
\[ \mathbb{E}[u_i + r_j] \geq 1 - \frac{1}{e}. \]

Proof. By the first part of Lemma 11, \( \mathbb{E}[u_i] \geq \mathbb{E}[u_e] \).

Next, we will lower bound \( \mathbb{E}[r_j] \). Let \( z \in [0, 1 - \frac{1}{e}] \) and let us condition on the event \( u_e = z \). The critical observation is that \( u_e \) is determined by the run \( R_j \). This is conducted on graph \( G_j \), which does not contain vertex \( j \). Therefore \( u_e \) is independent of \( p_j \).

By the second part of Lemma 11, \( r_j = p_j \) whenever \( p_j < 1 - z \). We will ignore the contribution to \( \mathbb{E}[r_j] \) when \( p_j \geq 1 - z \). Let \( w \) be s.t. \( e^{w-1} = 1 - z \).

Now \( p_j \) is obtained by picking \( x \) uniformly at random from the interval \( [0, 1] \) and outputting \( e^{x-1} \). In particular, when \( x \in [0, w) \), \( p_j < 1 - z \). If so, by the second part of Lemma 11, \( j \) is matched and revenue is accrued in \( r_j \), see Figure 2. Therefore,
\[ \mathbb{E}[r_j \mid u_e = z] \geq \int_0^w e^{x-1} dx = e^{w-1} - \frac{1}{e} = 1 - \frac{1}{e} - z. \]

Let \( f_{u_e}(z) \) be the probability density function of \( u_e \); clearly, \( f_{u_e}(z) = 0 \) for \( z \notin [0, 1 - \frac{1}{e}] \). Therefore,
\[ \mathbb{E}[r_j] = \mathbb{E}[\mathbb{E}[r_j \mid u_e]] = \int_{z=0}^{1-1/e} \mathbb{E}[r_j \mid u_e = z] \cdot f_{u_e}(z) dz \]
\[ \geq \int_{z=0}^{1-1/e} \left( 1 - \frac{1}{e} - z \right) \cdot f_{u_e}(z) dz = 1 - \frac{1}{e} - \mathbb{E}[u_e], \]
where the first equality follows from the law of total expectation and the inequality follows from fact that we have ignored the contribution to \( \mathbb{E}[r_j \mid u_e] \) when \( p_j \geq 1 - z \). Hence we get
\[ \mathbb{E}[u_i + r_j] = \mathbb{E}[u_i] + \mathbb{E}[r_j] \geq 1 - \frac{1}{e}. \]

Remark 14. Observe that Lemma 13 is not a statement about \( i \) and \( j \) getting matched to each other, but about the utility accrued by \( i \) and the revenue accrued by \( j \) by being matched to various goods and buyers, respectively, over the randomization executed in Step (1) of Algorithm 6.

Theorem 15. The competitive ratio of RANKING is at least \( 1 - \frac{1}{e} \).

Proof. Let \( P \) denote a perfect matching in \( G \). The expected size of matching produced by RANKING is
\[ \mathbb{E}[|M|] = \sum_i \mathbb{E}[u_i] + \sum_j \mathbb{E}[r_j] = \sum_{(i,j) \in P} \mathbb{E}[u_i + r_j] \geq n \left( 1 - \frac{1}{e} \right), \]
where the first equality uses Lemma 5, the second follows from linearity of expectation and the inequality follows from Lemma 13 and the fact that \( |P| = n \). The theorem follows. \( \square \)
Figure 2: The shaded area is a lower bound on $\mathbb{E}[r_j | u_e = z]$.

Remark 16. In case $G$ does not have a perfect matching, let $P$ denote a maximum matching in $G$, of size $k$, say. Then summing $\mathbb{E}[u_i]$ and $\mathbb{E}[r_j]$ over the the vertices $i$ and $j$ matched by $P$, we get that the expected size of matching produced by RANKING is at least $k \left(1 - \frac{1}{e}\right)$.

4 Algorithm for SINGLE-VALUED

Algorithm 19, which will be denoted by $A_2$, is an online algorithm for SINGLE-VALUED. Before execution of Step (1) of $A_2$, the order of arrival of queries, say $\rho(B)$, is fixed by the adversary. We will define several random variables whose purpose will be quite similar to that in RANKING and they will be given similar names as well; however, their function is not as closely tied to these economics-motivated names as in RANKING, see also Remark 17. Three of these random variables are the price $p_j$ and total revenue $r_j$ of each bidder $j \in A$, and the utility $u_i$ of each query $i \in Q$.

We now describe how values are assigned to these random variables in a run of Algorithm 19. In Step (1), for each bidder $j$, $A_2$ picks a price $p_j \in \left[\frac{1}{e}, 1\right]$ via the specified randomized process. Furthermore, the revenue $r_j$ and degree $d_j$ of bidder $j$ are both initialized to zero, the latter represents the number of times $j$ has been matched. During the run of $A_2$, $j$ will get matched to at most $k_j$ queries; each match will add $b_j$ to the total revenue generated by the algorithm. $b_j$ is broken into a revenue and a utility component, with the former being added to $r_j$ and the latter forming $u_i$. At the end of $A_2$, $r_j$ will contain all the revenue accrued by $j$.

In Step (2), on the arrival of query $i$, we will say that bidder $j$ is available if $(i,j) \in E$ and
d_j < k_j. At this point, for each available bidder \( j \), the effective bid of \( j \) for \( i \) is defined to be 
\[
ebid(j) = b_j \cdot (1 - p_j);
\]
clearly, \( \text{ebid}(j) \in [0, b_j \cdot (1 - \frac{1}{e})] \). Query \( i \) accepts the bidder whose effective bid is the largest. If there are no bids, matching \( M \) remains unchanged. If \( i \) accepts \( j \)'s bid, then edge \((i, j)\) is added to matching \( M \) and the weight of this edge is set to \( b_j \). Furthermore, the utility of \( i \), \( u_i \), is defined to be \( \text{ebid}(j) \) and the revenue \( r_j \) of \( j \) is incremented by \( b_j \cdot p_j \). Once all queries are processed, matching \( M \) and its weight \( W \) are output.

**Remark 17.** The economics-based names of random variables used in our proof of RANKING came from [EFFS21]. Although we have used the same names for similar random variables in Sections 4 and 5.2, for SINGLE-VALUED and GENERAL, the reader should not attribute an economic interpretation to these the names as was done in RANKING.

### 4.1 Analysis of Algorithm 19

For the analysis of Algorithm \( A_2 \), we will use the random variables \( W, p_j, r_j \) and \( u_i \) defined above; their values are fixed during the execution of \( A_2 \). In addition, corresponding to each edge \( e = (i, j) \in E \), in Definition 22, we will introduce a new random variable, \( u_e \), which will play a central role.

**Lemma 18.**

\[
\mathbb{E}[W] = \sum_{i=1}^{n} \mathbb{E}[u_i] + \sum_{j=1}^{m} \mathbb{E}[r_j].
\]

**Proof.** For each edge \((i, j) \in M\), its contribution to \( W \) is \( b_j \). Furthermore, the sum of \( u_i \) and the contribution of \((i, j)\) to \( r_j \) is also \( b_j \). This gives the first equality below. The second equality follows from linearity of expectation.

\[
\mathbb{E}[W] = \mathbb{E} \left[ \sum_{i=1}^{n} u_i + \sum_{j=1}^{m} r_j \right] = \sum_{i=1}^{n} \mathbb{E}[u_i] + \sum_{j=1}^{m} \mathbb{E}[r_j].
\]

As in the case of RANKING, we will define several runs of Algorithm 19. In these runs, we will assume Step (1) is executed once. We next define several ways of executing Step (2). Let \( \mathcal{R} \) denote the run of Step (2) on the entire graph \( G \). Corresponding to each bidder \( j \in A \), let \( G_j \) denote graph \( G \) with bidder \( j \) removed. Define \( \mathcal{R}_j \) to be the run of Step (2) on graph \( G_j \).

Analogous to Lemma 7 and Corollary 8 proved for RANKING, we will prove Lemma 20 and Corollary 21, which establish a relationship between the available bidders for a query \( i \) in the two runs \( \mathcal{R} \) and \( \mathcal{R}_j \). One difference is that now bidders are available in multiplicity and therefore we will have to use the notion of a multiset rather than a set.

\(^{6}\)We failed to come up with more meaningful names for these random variables and therefore have stuck to the old names.
As before, let us renumber the queries so their order of arrival under $\rho$ multiplicity than $B$ities, and belongs to both multisets, moreover with multiplicity that is the minimum of the two multiplic-

For each $i$, $T_i$ and $A_i$ at least as large as that in $A$.

We have assumed that Step (1) of Algorithm 6 has already been executed and a price $p_k$ has been assigned to each bidder $k$. The effective bid of bidder $k$ is $\text{ebid}(k) = b_k \cdot (1 - p_k)$. With probability $1$, the effective bids of all bidders are distinct. Let $F_1$ be the multiset containing $k_l$ copies of $l$ for each $l \in A$ such that $b_l \cdot (1 - p_l) > b_j \cdot (1 - p_j)$. Similarly, let $F_2$ be the multiset containing $k_l$ copies of $l$ for each $l \in A$ such that $b_l \cdot (1 - p_l) < b_j \cdot (1 - p_j)$

**Lemma 20.** For each $i$, $1 \leq i \leq n$, the following hold:

1. $(T_j(i) \cap F_1) = (T(i) \cap F_1)$.
2. $(T_j(i) \cap F_2) \subseteq (T(i) \cap F_2)$.

---

**Algorithm 19. ($A_2$: Algorithm for SINGLE-VALUED)***

1. **Initialization:** $M \leftarrow \emptyset$.  
   $\forall j \in A$, do: 
   (a) Pick $w_j$ uniformly from $[0,1]$ and set price $p_j \leftarrow e^{w_j-1}$.  
   (b) $r_j \leftarrow 0$.  
   (c) $d_j \leftarrow 0$.

2. **Query arrival:** When query $i$ arrives, do:
   (a) $\forall j \in A$ s.t. $(i,j) \in E$ and $d_j < k_j$ do:  
      i. $\text{ebid}(j) \leftarrow b_j \cdot (1 - p_j)$.  
      ii. Offer effective bid of $\text{ebid}(j)$ to $i$.  
   (b) Query $i$ accepts the bidder whose effective bid is the largest.  
      (If there are no bids, matching $M$ remains unchanged.)
      If $i$ accepts $j$’s bid, then do:  
      i. Set utility: $u_i \leftarrow b_j \cdot (1 - p_j)$.  
      ii. Update revenue: $r_j \leftarrow r_j + b_j \cdot p_j$.  
      iii. Update degree: $d_j \leftarrow d_j + 1$.  
      iv. Update matching: $M \leftarrow M \cup (i,j)$. Define the weight of $(i,j)$ to be $b_j$.  
   (c) **Output:** Output matching $M$ and its total weight $W$. 

A multiset contains elements with multiplicity. Given two multisets $A$ and $B$, we will say that $A \subseteq B$ if corresponding to each element, say $j$, in $A$, $B$ also contains $j$, moreover with multiplicity at least as large as that in $A$. Similarly, $A \cap B$ is the multiset containing each element, say $j$, that belongs to both multisets, moreover with multiplicity that is the minimum of the two multiplicities, and $A - B$ is the multiset containing each element, say $j$, that belongs to $A$ with a higher multiplicity than $B$, moreover with multiplicity that is the difference of the two multiplicities.

As before, let us renumber the queries so their order of arrival under $\rho(B)$ is $1, 2, \ldots, n$. Let $T(i)$ and $T_j(i)$ denote the multisets of available bidders at the time of arrival of query $i$ (i.e., just before the query $i$ gets matched) in runs $R$ and $R_j$, respectively. In particular, $T(1)$ will contain $k_l$ copies of $l$ for each bidder $l$ and $T_j(1)$ will contain $k_l$ copies of $l$ for each bidder $l$, other than $j$. Similarly, let $S(i)$ and $S_j(i)$ denote the projections of $T(i)$ and $T_j(i)$ on the bidders available to query $i$, in runs $R$ and $R_j$, respectively.

We have assumed that Step (1) of Algorithm 6 has already been executed and a price $p_k$ has been assigned to each bidder $k$. The effective bid of bidder $k$ is $\text{ebid}(k) = b_k \cdot (1 - p_k)$. With probability $1$, the effective bids of all bidders are distinct. Let $F_1$ be the multiset containing $k_l$ copies of $l$ for each $l \in A$ such that $b_l \cdot (1 - p_l) > b_j \cdot (1 - p_j)$. Similarly, let $F_2$ be the multiset containing $k_l$ copies of $l$ for each $l \in A$ such that $b_l \cdot (1 - p_l) < b_j \cdot (1 - p_j)$
The proof of this lemma is identical to that of Lemma 7, other than the use of multisets instead of sets, and is omitted. As in OBM, it uses the fact that the effective bid of a bidder \( l \) is the same, namely \( \text{ebid}(l) = b_l(1 - p_l) \), for any query which \( l \) desires.

**Corollary 21.** For each \( i, 1 \leq i \leq n \), the following hold:

1. \( (S_j(i) \cap F_j) = (S(i) \cap F_j) \).
2. \( (S_j(i) \cap F_2) \subseteq (S(i) \cap F_2) \).
3. \( S_j(i) \subseteq S(i) \).

Next we define a new random variable, \( u_e \), for each edge \( e = (i, j) \in E \). This is called the truncated threshold for edge \( e \) and is given in Definition 22. It is critically used in the proofs of Lemmas 25 and 26.

**Definition 22.** Let \( e = (i, j) \in E \) be an arbitrary edge in \( G \). Define random variable, \( u_e \), called the truncated threshold for edge \( e \), to be \( u_e = \min\{ut_i, b_j \cdot (1 - \frac{1}{e})\} \), where \( ut_i \) is the utility of query \( i \) in run \( R_j \).

**Definition 23.** Let \( j \in A \). Henceforth, we will denote \( k_j \) by \( k \) in order to avoid triple subscripts. Let \( i_1, \ldots, i_k \) be queries such that for \( 1 \leq l \leq k, (i_l, j) \in E \). Then \( (j; i_1, \ldots, i_k) \) is called a \( j \)-star. Let \( X_j \) denote this \( j \)-star. The contribution of \( X_j \) to \( \mathbb{E}[W] \) is \( \mathbb{E}[r_j] + \sum_{l=1}^{k} \mathbb{E}[u_{i_l}] \), and it will be denote by \( \mathbb{E}[X_j] \).

Corresponding to \( j \)-star \( X_j = (j; i_1, \ldots, i_k) \), denote by \( e_l \) the edge \( (i_l, j) \in E \), for \( 1 \leq l \leq k \). Furthermore, let \( u_{e_l} \) denote the truncated threshold random variable corresponding to \( e_l \).

**Property 24.** (No-Surpassing for SINGLE-VALUED) Let \( p_j \) be such that the effective bid of \( j \) to \( i_l \), namely \( \text{ebid}(j) = b_j \cdot (1 - p_j) \), is better than the best bid that query \( i_l \) gets in run \( R_j \). Then, in run \( R \), no bid to \( i_l \) will surpass \( b_j \cdot (1 - p_j) \).

**Lemma 25.** Corresponding to \( j \)-star \( X_j = (j; i_1, \ldots, i_k) \), the following hold.

- For \( 1 \leq l \leq k \), \( u_{i_l} \geq u_{e_l} \).

**Proof.** By the third statement of Corollary 21, \( i_l \) has more options in run \( R \) as compared to run \( R_j \). Furthermore, the truncation of the random variable only aids the inequality needed and therefore \( u_{i_l} \geq u_{e_l} \). \( \square \)

Our next goal is to lower bound the contribution of an arbitrary \( j \)-star, \( \mathbb{E}[X_j] \), which in turn involves lower bounding \( \mathbb{E}[r_j] \). The latter crucially uses the fact that \( p_j \) is independent of \( u_{e_l} \). This follows from the fact that \( u_{e_l} \) is determined by run \( R_j \) on graph \( G_j \), which does not contain vertex \( j \).

**Lemma 26.** Let \( j \in A \) and let \( X_j = (j; i_1, \ldots, i_k) \) be a \( j \)-star. Then

\[
\mathbb{E}[X_j] \geq k \cdot b_j \cdot \left(1 - \frac{1}{e}\right).
\]

16
Proof. We will first lower bound $\mathbb{E}[r_j]$. Let $f_U(b_j \cdot z_1, \ldots, b_j \cdot z_k)$ be the joint probability density function of $(u_{e_1}, \ldots, u_{e_k})$; clearly, $f_U(b_j \cdot z_1, \ldots, b_j \cdot z_k)$ can be non-zero only if $z_l \in [0, 1 - \frac{1}{e}]$, for $1 \leq l \leq k$. By the law of total expectation,

$$\mathbb{E}[r_j] = \int_{(z_1, \ldots, z_k)} \mathbb{E}[r_j | u_{e_1} = b_j \cdot z_1, \ldots, u_{e_k} = b_j \cdot z_k] \cdot f_U(b_j \cdot z_1, \ldots, b_j \cdot z_k) \, dz_1 \ldots dz_k,$$

where the integral is over $z_l \in [0, (1 - \frac{1}{e})]$, for $1 \leq l \leq k$.

For lower-bounding the conditional expectation in this integral, let $w_l \in [0, 1]$ be s.t. $e^{w_l - 1} = 1 - z_l$, for $1 \leq l \leq k$. For $x \in [0, 1]$, define the set $S(x) = \{l | 1 \leq l \leq k \text{ and } x < w_l\}$.

**Claim 27.** Conditioned on $(u_{e_1} = b_j \cdot z_1, \ldots, u_{e_k} = b_j \cdot z_k)$, if $p_j = e^{x-1}$, then the degree of $j$ at the end of Algorithm $A_2$ is at least $|S(x)|$, i.e., the contribution to $r_j$ in this run was $\geq b_j \cdot p_j \cdot |S(x)|$.

Proof. Suppose $l \in S(x)$, then $x < w_l$. In run $R_j$, the maximum effective bid that $i_l$ received has value $b_j \cdot z_l$. In run $R$, if at the arrival of query $i_l$, $j$ is already fully matched, the contribution to $r_j$ in this run was $k \cdot b_j \cdot p_j$ and the claim is obviously true. If not, then since $x < w_l$, $b_j \cdot (1 - p_j) > b_j \cdot z_l$. As in Lemma 11, the crux of the matter is to show that the no-surpassing property holds and the proof is also similar, other than the use of Lemma 20 and Corollary 21, since the available goods are now multisets. Therefore, query $i_l$ will receive its largest effective bid from $j$, $i_l$ will get matched to it, and $r_j$ will be incremented by $b_j \cdot p_j$. The claim follows. \qed

For $1 \leq l \leq k$, define indicator functions $I_l : [0, 1] \to \{0, 1\}$ as follows.

$$I_l(x) = \begin{cases} 
1 & \text{if } x < w_l, \\
0 & \text{otherwise}.
\end{cases}$$

Clearly, $|S(x)| = \sum_{l=1}^{k} I_l(x)$. By Claim 27,

$$\mathbb{E}[r_j | u_{e_1} = b_j \cdot z_1, \ldots, u_{e_k} = b_j \cdot z_k] \geq b_j \cdot \int_0^1 |S(x)| \cdot e^{x-1} \, dx$$

$$= b_j \cdot \int_0^1 \sum_{l=1}^{k} I_l(x) \cdot e^{x-1} \, dx = b_j \cdot \sum_{l=1}^{k} \int_0^1 I_l(x) \cdot e^{x-1} \, dx = b_j \cdot \sum_{l=1}^{k} \int_0^{w_l} e^{x-1} \, dx$$

$$= b_j \cdot \sum_{l=1}^{k} \left( e^{w_l - 1} - \frac{1}{e} \right) = b_j \cdot \sum_{l=1}^{k} \left( 1 - \frac{1}{e} - z_l \right).$$

Since $I_l(x) = 0$ for $x \in [w_l, 1]$, we get that $\int_0^1 I_l(x) \cdot e^{x-1} \, dx = \int_0^{w_l} e^{x-1} \, dx$; this fact has been used above. Therefore,

$$\mathbb{E}[r_j] = \int_{(z_1, \ldots, z_k)} \mathbb{E}[r_j | u_{e_1} = b_j \cdot z_1, \ldots, u_{e_k} = b_j \cdot z_k] \cdot f_U(b_j \cdot z_1, \ldots, b_j \cdot z_k) \, dz_1 \ldots dz_k$$
\[
\geq b_j \cdot \int_{(z_1, \ldots, z_k)} \sum_{l=1}^{k} \left( 1 - \frac{1}{e} - z_l \right) \cdot f_U(b_j \cdot z_1, \ldots b_j \cdot z_k) \, dz_1 \ldots dz_k
\]
\[
= k \cdot b_j \cdot \left( 1 - \frac{1}{e} \right) - \sum_{l=1}^{k} \mathbb{E}[u_l],
\]
where both integrals are over \( z_l \in [0, (1 - \frac{1}{e})] \), for \( 1 \leq l \leq k \).

By Lemma 25, \( \mathbb{E}[u_{ij}] \geq \mathbb{E}[u_{ij'}] \), for \( 1 \leq l \leq k \). Hence we get
\[
\mathbb{E}[X_j] = \mathbb{E}[r_j] + \sum_{l=1}^{k} \mathbb{E}[u_{ij}] \geq k \cdot b_j \cdot \left( 1 - \frac{1}{e} \right),
\]

\[\square\]

**Theorem 28.** The competitive ratio of Algorithm \( \mathcal{A}_2 \) is at least \( 1 - \frac{1}{e} \). Furthermore, it is budget-oblivious.

**Proof.** Let \( P \) denote a maximum weight \( b \)-matching in \( G \), computed in an offline manner. By the assumption made in Remark 3, its weight is
\[
w(P) = \sum_{j=1}^{m} k_j \cdot b_j.
\]

Let \( T_j \) denote the \( j \)-star, under \( P \), corresponding to each \( j \in A \). The expected weight of matching produced by \( \mathcal{A}_2 \) is
\[
\mathbb{E}[W] = \sum_{i=1}^{n} \mathbb{E}[u_i] + \sum_{j=1}^{m} \mathbb{E}[r_j] = \sum_{j=1}^{m} \mathbb{E}[T_j] \geq \sum_{j=1}^{m} b_j \cdot k_j \left( 1 - \frac{1}{e} \right) \cdot w(P),
\]
where the first equality uses Lemma 5, the second follows from linearity of expectation and the inequality follows from Lemma 26.

Finally, Algorithm \( \mathcal{A}_2 \) is budget-oblivious because it does not need to know \( k_j \) for bidders \( j \); it only needs to know during a run whether the \( k_j \) bids available to bidder \( j \) have been exhausted. The theorem follows. \[\square\]

### 5 Algorithm for SMALL, After Assuming No-Surpassing Property

Two new difficulties arise for the problem GENERAL. The first is the inherent structural difficulty described in Section 5.1. Second, since bidders can have different bids for different queries, the no-surpassing property does not hold anymore, see Example 29.

**Example 29.** Assume that in the given instance for GENERAL, \( j, j' \) are two of the bidders, and \( 1, \ldots, k \) are \( k \) of the queries, where \( k \) is a large number. Assume bid \((l, j) = \alpha \) for \( 1 \leq l \leq k \) and bid \((l, j') = \alpha - 1 \) for \( 1 \leq l \leq k - 1 \). Further, assume that bid \((k, j') = (\alpha - 1) \cdot (k - 1) \). Let the budgets be \( B_j = \alpha \cdot k \) and \( B_{j'} = (\alpha - 1) \cdot (k - 1) \).
Now consider a run in which $p_j = p' = p$. Assume that in run $R_i$, the best effective bid to $1, \ldots, k - 1$ comes from $j'$, and in run $R_j$, the best effective bid to $1, \ldots, k - 1$ comes from $j$. In run $R_j$, the budget of $j'$ is exhausted when $k$ arrives and assume that $k$ does not get any bids, making $u_e = 0$ for $e = (k, j)$. Now in run $R_i$, ebid($k, j$) = $\alpha(1 - p)$ and ebid($k, j'$) = $(\alpha - 1) \cdot (k - 1) \cdot (1 - p)$. Thus, even though ebid($k, j$) > $u_e$, $k$ will be matched to $j'$ and not $j$. Clearly, this phenomenon will hold for all runs in which $p'_j$ is not too much larger than $p_j$.

For the rest of this section, we will make this assumption:

Assumption of No-Surpassing for GENERAL: Let $p_j$ be such that the effective bid of $j$ to $i$, namely ebid($i, j$) = bid($i, j$) $\cdot$ (1 − $p_j$), is better than the best bid that query $i$ gets in run $R_j$. Then, in run $R$, no bid to $i$ will surpass bid($i, j$) $\cdot$ (1 − $p_j$).

Section 5.2 presents an algorithm for GENERAL, using fake money; the above-stated assumption is used in its analysis, in particular in the proof of Claim 36. Section 5.4 shows that by upper bounding the fake money used in the worst case, we get an optimal algorithm for SMALL, again based on the above-stated assumption.

### 5.1 Structural Difficulties in GENERAL

To describe the structural difficulties in GENERAL, we provide three instances in Example 30. In order to obtain a completely unconditional result, we would need to adopt the following convention: assume bidder $j$ has $L_j$ money leftover and impression $i$ just arrived. Assume that $j$’s bid for $i$ is bid($i, j$). If bid($i, j$) > $L_j$, then $j$ should not be allowed to bid for $i$, since $j$ has insufficient money.

Under this convention, it is easy to see that even a randomized algorithm will accrue only $W$ expected revenue on at least one of the instances given in Example 30, provided it is greedy, i.e., if a match is possible, it does not rescind this possibility; the latter condition is a simple way of ensuring that the algorithm is “fine tuned” for a particular type of example. Note that the optimal for each instance is 2$W$.

**Example 30.** Let $W \in \mathbb{Z}_+$ be a large number. We define three instances of GENERAL, each having two bidders, $b_1$ and $b_2$, with budgets of $W$ each. Instances $I_1$ and $I_2$ have $W + 1$ queries, where for the first $W$ queries, both bidders bid $1$ each. For the last query, under $I_1$, $b_1$ bids $W$ and $b_2$ is not interested. Under $I_2$, $b_2$ bids $W$ and $b_1$ is not interested. Instance $I_3$ has $2W$ queries and both bidders bid $1$ for each of them.

Therefore, to obtain a non-trivial competitive ratio, bidder $j$ must be allowed to bid for $i$ even if $L_j < \text{bid($i, j$)}$. This amounts to the use of free disposal, since $j$ will be allowed to obtain query $i$ for less money than its value for $i$. Next, let’s consider a second convention: if $L_j < \text{bid($i, j$)}$, then $j$ will bid $L_j$ for $i$. As stated in Remark 38, this convention is not supported by our proof technique, since Claim 36 fails to hold, breaking the proof of Lemma 35 and hence Lemma 37.

This led us to a third convention: if $L_j < \text{bid($i, j$)}$, then $j$ will bid $L_j$ real money and bid($i, j$) − $L_j$ “fake” money for $i$. As a result, the total revenue of the algorithm consists of real money as well as fake money; in Algorithm 33, these are denoted by $W$ and $W_f$, respectively. The problem now
is that Lemma 37, which compares the total revenue of the algorithm, namely \( W + W_f \), with the optimal offline revenue, does not yield the competitive ratio of Algorithm 33. Remark 38 explains why our proof technique does not allow us to dispense with the use of fake money.

We note that when Algorithm 33 is run on instances of OBM, it reduces to RANKING. Therefore, it is indeed a (simple) extension of RANKING to GENERAL.

5.2 Algorithm for GENERAL

Algorithm 33, which will be denoted by \( A_3 \), is an attempt at online algorithm for GENERAL. As stated in Section 5.1, because of the use of fake money, we will not be able to give a competitive ratio for it, instead, in Lemma 37, we will compare the sum of real and fake money spent by the algorithm with the real money spent by an optimal offline algorithm.

In algorithm \( A_3 \), \( L_j \in \mathbb{Z}_+ \) will denote bidder \( j \)'s leftover budget; it is initialized to \( B_j \). At the arrival of query \( i \), bidder \( j \) will bid for \( i \) if \((i, j) \in E \) and \( L_j > 0 \). In general, \( i \) will receive a number of bids. The exact procedure used by \( i \) to accept one of these bids is given in algorithm \( A_3 \); its steps are self-explanatory. If \( i \) accepts \( j \)'s bid then \( i \) is matched to \( j \), the edge \((i, j)\) is assigned a weight of \( \text{bid}(i, j) \) and \( L_j \) is decremented by \( \min\{L_j, \text{bid}(i, j)\} \).

Note that we do not require that there is sufficient left-over money, i.e., \( L_j \geq \text{bid}(i, j) \), for \( j \) to bid for \( i \). In case \( L_j < \text{bid}(i, j) \) and \( i \) accepts \( j \)'s bid, then \( \text{bid}(i, j) - L_j \) of the money paid by \( j \) for \( i \) is fake money; this will be accounted for by incrementing \( W_f \) by \( \text{bid}(i, j) - L_j \). The rest, namely \( L_j \), is real money and is added to \( W \). If \( \text{bid}(i, j) \geq L_j \) and \( i \) accepts \( j \)'s bid, then \( L_j \) becomes zero and \( j \) does not bid for any future queries. At the end of the algorithm, random variable \( W \) denotes the total real money spent and \( W_f \) denotes the total fake money spent.

The offline optimal solution to this problem is defined to be a matching of queries to advertisers that maximizes the weight of the matching; this is done with full knowledge of graph \( G \). As stated in Remark 3, we will assume that under such a matching, \( P \), the budget \( B_j \) of each bidder \( j \) is fully spent, i.e., \( w(P) = \sum_{j=1}^{m} B_j \).

5.3 Analysis of Algorithm 33

Lemma 31.

\[
\mathbb{E}[W + W_f] = \sum_{i=1}^{n} \mathbb{E}[u_i] + \sum_{j=1}^{m} \mathbb{E}[r_j].
\]

Proof. For each edge \((i, j) \in M\), its contribution to \( W + W_f \) is \( \text{bid}(i, j) \). Furthermore, the sum of \( u_i \) and the contribution of \((i, j)\) to \( r_j \) is also \( \text{bid}(i, j) \). This gives the first equality below. The second equality follows from linearity of expectation.

\[
\mathbb{E}[W + W_f] = \mathbb{E} \left[ \sum_{i=1}^{n} u_i + \sum_{j=1}^{m} r_j \right] = \sum_{i=1}^{n} \mathbb{E}[u_i] + \sum_{j=1}^{m} \mathbb{E}[r_j].
\]

\[\square\]
Recall that for SINGLE-VALUED, we gave Lemma 20 and Corollary 21, which established a relationship between the available bidders for a query \( i \) in the two runs \( R \) and \( R_j \). These facts dealt with multisets rather than sets; the latter sufficed for Lemma 7 and Corollary 8, which were used in the analysis of RANKING. In Section 4, we also defined operations on multisets.

We will need Lemma 20 and Corollary 21 for analyzing Algorithm 33 as well, though the definitions of the multisets will be guided by the following: If bidder \( k \in A \) has leftover money of \( L_k \), as determined by Algorithm 33, then we will say that \( i \) has \( L_k \) copies of \( k \) available to it. Furthermore, if \( i \)'s bid for \( k \) is bid \((i,k)\) and this bid is successful, then \( L_k \) will be decremented by \( \min\{L_k, \text{bid}(i,k)\} \), as stated in Step 2(b)(v) of the algorithm, and the available copies of \( k \) for the next bidder will decrease accordingly.

As before, let us renumber the queries so their order of arrival under \( \rho(B) \) is \( 1, 2, \ldots n \). Let \( T(i) \) and \( T_j(i) \) denote the multisets of available copies of each bidder at the time of arrival of query \( i \) (i.e., just before the query \( i \) gets matched), in runs \( R \) and \( R_j \), respectively. Similarly, let \( S(i) \) and \( S_j(i) \) denote the multisets obtained by restricting \( T(i) \) and \( T_j(i) \) to the bidders that have edges to query \( i \) in graphs \( G \) and \( G_j \), respectively.

We have assumed that Step (1) of Algorithm 6 has already been executed and a price \( p_k \) has been assigned to each good \( k \). With probability 1, the prices are all distinct. Let \( F_1 \) be the multiset containing \( B_l \) copies of \( l \) for each \( l \in A \) such that \( p_l < p_j \). Similarly, let \( F_2 \) be the multiset containing \( B_l \) copies of \( l \) for each \( l \in A \) such that \( p_l > p_j \).

Under the definitions and operations stated above, it is easy to check that Lemma 20 and Corollary 21 hold for Algorithm 33 as well. Therefore, Lemma 25 also carries over. Definition 22 needs to be modified to the following.

**Definition 32.** Let \( e = (i,j) \in E \) be an arbitrary edge in \( G \). Define random variable, \( u_e \), called the truncated threshold for edge \( e \), to be \( u_e = \min\{u_i, \text{bid}(i,j) \cdot (1 - \frac{1}{e})\} \), where \( u_i \) is the utility of query \( i \) in run \( R_j \).

Definition 23 needs to be changed to the following.

**Definition 34.** Let \( j \in A \). Let \( i_1, \ldots, i_k \) be queries such that for \( 1 \leq l \leq k \), \((i_l,j) \in E \) and \( \sum_{l=1}^k \text{bid}(i_l,j) = B_j \). Then \((j; i_1, \ldots, i_k)\) is called a \( B_j \)-star. Let \( X_j \) denote this \( B_j \)-star. The contribution of \( X_j \) to \( \mathbb{E}[W] \) is \( \mathbb{E}[r_j] + \sum_{l=1}^k \mathbb{E}[u_{i_l}] \), and it will be denote by \( \mathbb{E}[X_j] \).

Corresponding to \( B_j \)-star \( X_j = (j; i_1, \ldots, i_k) \), denote by \( e_l \) the edge \((i_l,j) \in E \), for \( 1 \leq l \leq k \). Furthermore, let \( u_{e_l} \) denote the truncated threshold random variable corresponding to \( e_l \). The next lemma crucially uses the fact that \( p_j \) is independent of \( u_{e_l} \); the reason for this fact is the same as in SINGLE-VALUED.

**Lemma 35.** Let \( j \in A \) and let \( X_j = (j; i_1, \ldots, i_k) \) be a \( B_j \)-star. Then

\[
\mathbb{E}[X_j] \geq B_j \cdot \left(1 - \frac{1}{e}\right).
\]
Algorithm 33. ($A_3$: Algorithm for GENERAL)

1. **Initialization:** $M \leftarrow \emptyset$, $W \leftarrow 0$ and $W_f \leftarrow 0$
   - For all $j \in A$, do:
     a. Pick $w_j$ uniformly from $[0, 1]$ and set price $p_j \leftarrow e^{w_j - 1}$.
     b. $r_j \leftarrow 0$.
     c. $L_j \leftarrow B_j$.

2. **Query arrival:** When query $i$ arrives, do:
   - For all $j \in A$ s.t. $(i, j) \in E$ and $L_j > 0$ do:
     i. ebid$(i, j) \leftarrow$ bid$(i, j) \cdot (1 - p_j)$.
     ii. Offer effective bid of ebid$(i, j)$ to $i$.
   - Query $i$ accepts the bidder whose effective bid is the largest.
     (If there are no bids, matching $M$ remains unchanged.)
     If $i$ accepts $j$’s bid, then do:
     i. Set utility: $u_i \leftarrow$ bid$(i, j) \cdot (1 - p_j)$.
     ii. Update revenue: $r_j \leftarrow r_j +$ bid$(i, j) \cdot p_j$.
     iii. Update matching: $M \leftarrow M \cup (i, j)$.
     iv. Update weight: $W \leftarrow \min\{L_i, \text{bid}(i, j)\}$ and $W_f \leftarrow \max\{0, \text{bid}(i, j) - L_i\}$.
     v. Update $L_j$: $L_j \leftarrow L_j - \min\{L_i, \text{bid}(i, j)\}$.

3. **Output:** Output matching $M$, real money spent $W$, and fake money spent $W_f$.

**Proof.** We will first lower bound $E[r_j]$. Let $f_U(\text{bid}(i_1, j) \cdot z_1, \ldots, \text{bid}(i_k, j) \cdot z_k)$ be the joint probability density function of $(u_{c_1}, \ldots, u_{c_k})$; clearly, $f_U(\text{bid}(i_1, j) \cdot z_1, \ldots, \text{bid}(i_k, j) \cdot z_k)$ can be non-zero only if $z_l \in [0, 1 - \frac{1}{k}]$, for $1 \leq l \leq k$.

By the law of total expectation,

$$E[r_j] = \int_{(z_1, \ldots, z_k)} E[r_j \mid u_{c_1} = \text{bid}(i_1, j) \cdot z_1, \ldots, u_{c_k} = \text{bid}(i_k, j) \cdot z_k] \cdot f_U(\text{bid}(i_1, j) \cdot z_1, \ldots, \text{bid}(i_k, j) \cdot z_k) \, dz_1 \ldots dz_k,$$

where the integral is over $z_l \in [0, (1 - \frac{1}{k})]$, for $1 \leq l \leq k$.

For lower-bounding the conditional expectation in this integral, let $w_l \in [0, 1]$ be s.t. $e^{w_l - 1} = 1 - z_l$, for $1 \leq l \leq k$. Let $x \in [0, 1]$. For $1 \leq l \leq k$, define indicator functions $I_l : [0, 1] \rightarrow \{0, 1\}$ as follows.

$$I_l(x) = \begin{cases} 1 & \text{if } x < w_l, \\ 0 & \text{otherwise}. \end{cases}$$

Furthermore, define

$$V(x) = \sum_{l=1}^{k} I_l(x) \cdot \text{bid}(i_l, j).$$

**Claim 36.** Conditioned on $(u_{c_1} = \text{bid}(i_1, j) \cdot z_1, \ldots, u_{c_k} = \text{bid}(i_k, j) \cdot z_k)$, if $p_j = e^{x - 1}$, where $x \in [0, 1]$, then the contribution to $r_j$ in this run of algorithm $A_3$ was $\geq p_j \cdot V(x)$.
Proof. Suppose $I_l(x) = 1$, then $x < w_l$. In run $R_l$, the maximum effective bid that $i_l$ received has value $\text{bid}(i_l, j) \cdot z_l$. In run $R$, if on the arrival of query $i_l$, $L_j = 0$, i.e., $j$ is already fully matched, then the contribution to $r_j$ in this run was $B_j \cdot p_j$ and the claim is obviously true. If $L_j > 0$, then since $x < w_l$, $1 - p_j > z_l$. Therefore, by Corollary 21, query $i_l$ will receive its largest effective bid from $j$. Hence, $i_l$ will get matched to $j$ and $r_j$ will be incremented by $\text{bid}(i_l, j) \cdot p_j$. The claim follows.

By Claim 36,

$$
\mathbb{E}[r_j \mid u_{e_1} = \text{bid}(i_1, j) \cdot z_1, \ldots, u_{e_k} = \text{bid}(i_k, j) \cdot z_k] \geq \int_0^1 V(x) \cdot e^{x-1} dx
$$

$$
= \sum_{l=1}^k \text{bid}(i_l, j) \cdot \int_0^1 I_l(x) \cdot e^{x-1} dx = \sum_{l=1}^k \text{bid}(i_l, j) \cdot \int_0^{w_l} e^{x-1} dx
$$

$$
= B_j \cdot \sum_{l=1}^k \left( e^{w_l - 1} - \frac{1}{e} \right) = B_j \cdot \sum_{l=1}^k \left( 1 - \frac{1}{e} - z_l \right).
$$

Therefore, $\mathbb{E}[r_j] =$

$$
\int_{(z_1, \ldots, z_k)} \mathbb{E}[r_j \mid u_{e_1} = \text{bid}(i_1, j) \cdot z_1, \ldots, u_{e_k} = \text{bid}(i_k, j) \cdot z_k] \cdot f_U(\text{bid}(i_1, j) \cdot z_1, \ldots, \text{bid}(i_k, j) \cdot z_k) \, dz_1 \ldots dz_k
$$

$$
\geq B_j \cdot \int_{(z_1, \ldots, z_k)} \sum_{l=1}^k \left( 1 - \frac{1}{e} - z_l \right) \cdot f_U(\text{bid}(i_1, j) \cdot z_1, \ldots, \text{bid}(i_k, j) \cdot z_k) \, dz_1 \ldots dz_k
$$

$$
= B_j \cdot \left( 1 - \frac{1}{e} \right) - \sum_{l=1}^k \mathbb{E}[u_{e_l}].
$$

By Lemma 25, $\mathbb{E}[u_{e_l}] \geq \mathbb{E}[u_{e_1}]$, for $1 \leq l \leq k$. Hence we get

$$
\mathbb{E}[X_j] = \mathbb{E}[r_j] + \sum_{l=1}^k \mathbb{E}[u_{e_l}] \geq B_j \cdot \left( 1 - \frac{1}{e} \right),
$$

Lemma 37. Algorithm $A_3$ satisfies

$$
\mathbb{E}[W + W_f] \geq \left( 1 - \frac{1}{e} \right) \cdot w(P).
$$

Furthermore, it is budget-oblivious.

23
Proof. Let $P$ denote a maximum weight $b$-matching in $G$. By the assumption made in Remark 3, its weight is

$$w(P) = \sum_{j=1}^{m} B_j.$$ 

Let $T_j$ denote the $j$-star, under $P$, corresponding to each $j \in A$. The expected weight of matching produced by $A_3$ is

$$E[W + W_f] = \sum_{i=1}^{n} E[u_i] + \sum_{j=1}^{m} E[r_j] \geq \sum_{j=1}^{m} B_j \cdot \left(1 - \frac{1}{e}\right) = \left(1 - \frac{1}{e}\right) \cdot w(P),$$

where the first equality uses Lemma 5, the second follows from linearity of expectation and the inequality follows by using Lemma 35.

Finally, Algorithm $A_3$ is budget-oblivious because it does not need to know the budgets $B_j$ for bidders $j$; it only needs to know during a run whether $B_j$ has been exhausted. The lemma follows. 

Remark 38. Let us consider the following two avenues for dispensing with the use of fake money altogether; we will show places where our proof technique breaks down for each one. Assume $L_j < \text{bid}(i,j)$.

1. Why not modify Step 2 of Algorithm 33 so that $j$’s bid for $i$ is taken to be $L_j$ instead of $\text{bid}(i,j)$?

2. Why not modify Step 2(b)(i) so it sets $u_i$ to $L_j \cdot (1 - p_j)$ rather than $B_j \cdot (1 - p_j)$

Under the first avenue, we cannot ensure $u_i \geq u_e$, since it may happen that $u_e > L_j \cdot (1 - p_j) = u_i$. The condition $u_i \geq u_e$ is used for deriving $E[u_i] \geq E[u_e]$, which is essential in the proof of Lemma 35.

To make the second avenue work, the proof of Claim 36 would need to be changed as follows: the last case, $L_j > 0$, will need to be split into the two cases given above. However, under Case 2, which applies if $L_j < \text{bid}(i,j)$, even though $p_j < p$, the largest effective bid that query $i_l$ receives may not be the one from $j$, since the effective bid of $j$ has value $L_j \cdot (1 - p_j) < \text{bid}(i_l,j) \cdot (1 - p_j)$. Therefore, $i_l$ may not get matched to $j$, thereby invalidating Claim 36.

5.4 Algorithm for SMALL

We will use Lemma 37 to show that Algorithm 33 yields algorithms for SMALL by upper bounding the fake money used in the worst case. Their budget-obliviousness follows from that of Algorithm 33.

Conditional Theorem 39. Algorithm $A_3$ is an optimal online algorithm for SMALL; furthermore, it is budget-oblivious.
Proof. Let $I$ be an instance of SMALL.

$$W_f \leq \sum_{j \in A} \max_{(i, j) \in E} \{\text{bid}(i, j) - 1\}$$

Therefore,

$$\mu(I) = \max_{j \in A} \left\{ \frac{\max_{(i, j) \in E} \{\text{bid}(i, j) - 1\}}{B_j} \right\} \geq \frac{\sum_{j \in A} \max_{(i, j) \in E} \{\text{bid}(i, j) - 1\}}{\sum_{j \in A} B_j} \geq \frac{W_f}{w(P)},$$

where $\mu(I)$ is defined in Section 2. Now, by definition of SMALL,

$$\lim_{n(I) \to \infty} \mu(I) = 0,$$

where $n(I)$ denotes the number of queries in instance $I$.

Therefore

$$\lim_{n(I) \to \infty} \frac{W_f}{w(P)} = 0.$$

The theorem follows from Lemma 37. \hfill \square

6 Discussion

The open question mentioned in the Introduction, of removing the assumption of no-surpassing property from our proof of Algorithm 33 for SMALL, deserves special attention because of its potential impact in the ad auctions marketplace.

The main open question of the area is to make a substantial dent on GENERAL, or even better give an optimal online algorithm for it, without making any assumptions on bids vs budgets. As a result of work presented in this paper, in the spectrum from OBM to GENERAL, SINGLE-VALUED is the most general problem that yields to an optimal online algorithm\(^7\).

Clearly, new ideas and approaches are needed to attack GENERAL. This paper has identified a valuable test bed for such ideas, namely SINGLE-VALUED, without reducing it to other problems.

7 Acknowledgements

I wish to thank Asaf Ferber, Alon Orlitsky, Will Ma and Thorben Trobst for valuable comments and discussions.

\(^7\)For reasons given in Section 1.3, SINGLE-VALUED needs to be solved without reduction to other problems
References

[ABM19] Gagan Aggarwal, Ashwinkumar Badanidiyuru, and Aranyak Mehta. Autobidding with constraints. In International Conference on Web and Internet Economics, pages 17–30. Springer, 2019.

[AGKM11] Gagan Aggarwal, Gagan Goel, Chinmay Karande, and Aranyak Mehta. Online vertex-weighted bipartite matching and single-bid budgeted allocations. In Proceedings of the twenty-second annual ACM-SIAM symposium on Discrete Algorithms, pages 1253–1264, 2011.

[BC21] Guy Blanc and Moses Charikar. Multiway online correlated selection. arXiv preprint arXiv:2106.05579, 2021.

[BJN07] Niv Buchbinder, Kamal Jain, and Joseph Seffi Naor. Online primal-dual algorithms for maximizing ad-auctions revenue. In European Symposium on Algorithms, pages 253–264, 2007.

[BM08] Benjamin Birnbaum and Claire Mathieu. On-line bipartite matching made simple. ACM Sigact News, 39(1):80–87, 2008.

[DJK13] Nikhil R Devanur, Kamal Jain, and Robert D Kleinberg. Randomized primal-dual analysis of ranking for online bipartite matching. In Proceedings of the twenty-fourth annual ACM-SIAM symposium on Discrete algorithms, pages 101–107. SIAM, 2013.

[DM22] Nikhil Devanur and Aranyak Mehta. Online matching in advertisement auctions. In Federico Echenique, Nicole Immorlica, and Vijay V. Vazirani, editors, Online and Matching-Based Market Design. Cambridge University Press, 2022. [To appear] https://www.ics.uci.edu/~vazirani/AdAuctions.pdf.

[EFFS21] Alon Eden, Michal Feldman, Amos Fiat, and Kineret Segal. An economic-based analysis of ranking for online bipartite matching. In SIAM Symposium on Simplicity in Algorithms, 2021.

[EIV22] Federico Echenique, Nicole Immorlica, and Vijay V. Vazirani. One-sided matching markets. In Federico Echenique, Nicole Immorlica, and Vijay V. Vazirani, editors, Online and Matching-Based Market Design. Cambridge University Press, 2022. [To appear] https://www.ics.uci.edu/~vazirani/Chapter2.pdf.

[FHTZ20] Matthew Fahrbach, Zhiyi Huang, Runzhou Tao, and Morteza Zadimoghaddam. Edge-weighted online bipartite matching. In 2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS), pages 412–423, 2020.

[GM08] Gagan Goel and Aranyak Mehta. Online budgeted matching in random input models with applications to adwords. In SODA, volume 8, pages 982–991, 2008.

[GS62] David Gale and Lloyd S Shapley. College admissions and the stability of marriage. The American Mathematical Monthly, 69(1):9–15, 1962.

[HT22] Zhiyi Huang and Thorben Trobst. Online matching. In Federico Echenique, Nicole Immorlica, and Vijay V. Vazirani, editors, Online and Matching-Based Market Des-
[HZZ20] Zhiyi Huang, Qiankun Zhang, and Yuhao Zhang. Adwords in a panorama. In 2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS), pages 1416–1426. IEEE, 2020.

[Ins19] Simons Institute. Online and matching-based market design, 2019. https://simons.berkeley.edu/programs/market2019.

[JMM+03] Kamal Jain, Mohammad Mahdian, Evangelos Markakis, Amin Saberi, and Vijay V Vazirani. Greedy facility location algorithms analyzed using dual fitting with factor-revealing LP. Journal of the ACM (JACM), 50(6):795–824, 2003.

[KP00] Bala Kalyanasundaram and Kirk R Pruhs. An optimal deterministic algorithm for online b-matching. Theoretical Computer Science, 233(1-2):319–325, 2000.

[KVV90] Richard M Karp, Umesh V Vazirani, and Vijay V Vazirani. An optimal algorithm for on-line bipartite matching. In Proceedings of the twenty-second annual ACM symposium on Theory of computing, pages 352–358, 1990.

[Meh13] Aranyak Mehta. Online matching and ad allocation. 2013.

[MSVV07] Aranyak Mehta, Amin Saberi, Umesh Vazirani, and Vijay Vazirani. Adwords and generalized online matching. Journal of the ACM (JACM), 54(5), 2007.

[Udw21] Rajan Udwani. Adwords with unknown budgets. arXiv preprint arXiv:2110.00504, 2021.