Market Pricing for Data Streams

Melika Abolhassani,[†*] **Hossein Esfandiari,**[†] **MohammadTaghi Hajiaghayi,**[†*]
Brendan Lucier,[‡] **Hadi Yami**[†*]
†Univeristy of Maryland, ‡Microsoft Research
brlucier@microsoft.com, {melika, hossein, hajiagha, hadiyami}@cs.umd.edu

Abstract

Internet-enabled marketplaces such as Amazon deal with huge datasets registering transaction of merchandises between lots of buyers and sellers. It is important that algorithms become more time and space efficient as the size of datasets increase. An algorithm that runs in polynomial time may not have a reasonable running time for such large datasets. Here, we study the development of pricing algorithms that are appropriate for use with massive datasets. We especially focus on the streaming setting, the common model for big data analysis.

We present an envy-free mechanism for social welfare maximization problem in the streaming setting using $O(k^2 l)$ space, where k is the number of different goods and l is the number of available items of each good. We also provide an α-approximation mechanism for revenue maximization in this setting given an α-approximation mechanism for the corresponding offline problem exists. Moreover, we provide mechanisms to approximate the optimum social welfare (or revenue) within $1 - \epsilon$ factor, in space independent of l which would be favorable in case l is large compared to k. Finally, we present hardness results showing approximation of *optimal prices* that maximize social welfare (or revenue) in the streaming setting needs $\Omega(l)$ space.

We achieve our results by developing a powerful sampling technique for bipartite networks. The simplicity of our sampling technique empowers us to maintain the sample over the input sequence. Indeed, one can construct this sample in the distributed setting (a.k.a, MapReduce) and get the same results in two rounds of computations, or one may simply apply this sampling technique to provide faster offline algorithms.

1 Introduction

Modern, Internet-enabled marketplaces have the potential to serve an extremely large volume of transactions. Giant online markets such as eBay and Amazon work with massive datasets that record the exchange of goods between many different buyers and sellers. Such datasets present an opportunity: it is natural to analyze the history of transactions to estimate demand and solve central pricing problems. Indeed, a recent and exciting line of literature in the algorithmic game theory community has set out to understand how the availability of data, in the form of samples from a transaction history, can be employed to tune prices and design mechanisms (Balcan et al. 2008; Cole and Roughgarden 2014; Dhangwatnotai, Roughgarden, and Yan 2010; Hsu et al. 2015; Morgenstern and Roughgarden 2015). Big-data environments are a boon for such tasks. However, as datasets grow ever larger, data-analysis algorithms must become ever more efficient. An algorithm that runs in polynomial (or even linear) time may not have a reasonable running time in practice.

In this paper we study the development of pricing algorithms that are appropriate for use with massive datasets. We will adopt the model of streaming algorithms, a standard model for massive dataset analysis. In the streaming algorithm model, a stream of data arrives sequentially and must be analyzed by an algorithm with limited memory. These streams can only be read once (or a limited number of times), and hence streaming algorithms must be frugal in the amount and nature of data that they choose to store.

Streaming algorithms were first theoretically introduced in fields such as data mining and machine learning over 20 years ago in order to model problems in which the data cannot be accessed all at once. Over the past decade, there has been a significant demand for algorithms to process and handle dynamic data coming from huge and growing graphs such as social networks, webpages and their links and citations of academic work.

These algorithmic techniques are also relevant to market design problems. For instance, one might imagine that there is a set of items (e.g., goods for sale) and a set of potential buyers (e.g., individual consumers) to which they should be matched. Such markets are essentially bipartite matching problems, which have themselves been the subject of study in the context of streaming algorithms. This begs the question: to what extent can market design and pricing problems be resolved adequately in the streaming model?

We have in mind two main applications of solving this pricing problem on a massive collection of static data, especially on the data of previous sales. First, by computing optimal prices on past transactions, one can subsequently employ those prices as a guideline for setting future prices; this is a key step in many recent pricing methodologies based on statistical learning theory. Also, the optimal welfare or

*Supported in part by NSF CAREER award CCF-1053605, NSF BIGDATA grant IIS-1546108, NSF AF:Medium grant CCF-1161365, DARPA GRAPHS/AFOSR grant FA9550-12-1-0423, and another DARPA SIMPLEX grant.

revenue in hindsight is a useful benchmark for the online pricing mechanism being employed by the platform, and can therefore be used to evaluate and tune.

1.1 Our Results

In this paper, we instantiate our high-level question by focusing on the *envy-free pricing problem* with big data. Our model is as follows. Suppose there is a bipartite graph G with a set of n unit-demand buyers $b_1, \dots, b_n$ on one side, and a set of k distinct types of items $v_1, \dots, v_k$, with l copies of each on the other side. The utility of buyer b_j for item v_i is denoted by u_{v_i,b_j}, and is shown by a weighted edge between the corresponding two vertices. The price assigned to item i is denoted by p_i. The goal is to assign prices to items, and then items to buyers, such that the assignment is *envy-free*[1], i.e., each buyer prefers the item assigned to her rather than an item assigned to another buyer. Subject to the envy-freeness condition, the designer wishes to maximize either the social welfare or revenue of the corresponding allocation. This is precisely the envy-free pricing problem introduced by Guruswami et al. (Guruswami et al. 2005). We ask: how well can envy-free prices be computed in the *streaming setting*?

We note that there are many possible representations of the input as a data stream. We will perform our analysis under a model in which the utility values u_{v_i,b_j} arrive in a data stream in an arbitrary order. We note that there are other potential options, such as assuming that all values associated with a certain agent arrive simultaneously, or that the values u_{v_i,b_j} are not provided directly but rather the input contains only the revealed preference of a buyer in response to prices. We leave the exploration of these alternative models as an avenue for future work.

We consider both social-welfare maximization and revenue maximization versions of the envy-free pricing problem. First, we provide streaming mechanisms that compute both allocation and prices of the items using $O(k^2 l)$ space. Later, we present streaming mechanisms that only compute the prices using space $\tilde{O}(k^3)$, approximating social welfare (or revenue) within a factor of $1-\epsilon$, where poly-logarithmic factors are hidden in the notation of $\tilde{O}$. At the end, we present lower bounds on the required space of any mechanism that computes optimum prices for either social-welfare maximization or revenue maximization.

In Theorem 1.1 we provide an envy-free streaming mechanism for the social-welfare maximization problem using $O(k^2 l)$ space.

Theorem 1.1 *There exists an envy-free mechanism for the social-welfare maximization problem in the streaming setting using $O(k^2 l)$ space. This mechanism remembers the allocation as well as the prices.*

Indeed, finding the maximum matching in a bipartite graph with $O(k)$ vertices in the streaming setting requires $\Omega(k^2)$ space (Chitnis et al. 2015). Thus, for $l = 1$, the space required by our mechanisms in Theorem 1.2 and Theorem 1.1 are tight.

[1]If we assign item v_{i_1} to buyer b_{j_1} and item v_{i_2} to buyer b_{j_2}, then we have $u_{v_{i_1},b_{j_1}} - p(i_1) \geq u_{v_{i_2},b_{j_1}} - p(i_2)$.

The following theorem extends our result to the revenue maximization problem. Even in the static (i.e., non-streaming) environment, this problem has resisted constant-factor approximation factors for simple versions, including the case of unit-demand bidders studied here. We frame our result as a reduction: given an algorithm for computing envy-free prices in the static setting, we show how to construct a streaming algorithm with the same approximation guarantee. As with the welfare maximization problem, the required space is $O(k^2 l)$.

Theorem 1.2 *Given an envy-free α-approximation mechanism for the revenue maximization problem, there exists an envy-free α-approximation mechanism for the revenue maximization problem in the streaming setting using $O(k^2 l)$ space. This mechanism remembers the allocation as well as the prices.*

Each of the above results are with respect to algorithms that return not only a profile of envy-free prices, but also the corresponding allocation. We note that the size of the allocation is $O(kl)$, and thus any mechanism that remembers the allocation requires at least $\Omega(kl)$ space. This space may be quite large when l is large, which is not desirable. What if we are only interested in determining the envy-free prices, and just being within an approximation of the maximum social welfare(or revenue)? As it turns out, this variation of the problem allows significant improvement when l is large. We provide an almost optimal streaming mechanism using $\tilde{O}(k^3)$ space that computes prices. That is, the required space here is poly-logarithmic in the number of buyers and the number of copies of each item type. The following theorem states our result for the social-welfare maximization problem.

Theorem 1.3 *Let ϵ be an arbitrary small constant. There exists a streaming mechanism for the social-welfare maximization problem which with high probability gives an envy-free $(1-\epsilon)$-approximate solution using $\tilde{O}(k^3)$ space. This mechanism only remembers the prices.*

The following theorem extends our results to the revenue maximization problem as well. Note that, again here the required space is poly-logarithmic in the number of buyers and the number of copies of each item type.

Theorem 1.4 *Given an envy-free α-approximation mechanism for the revenue maximization problem, and any small constant ϵ, there exists a streaming mechanism for the revenue maximization problem which with high probability gives an envy-free $(1-\epsilon)\alpha$-approximate solution using $\tilde{O}(k^3)$ space. This mechanism only remembers the prices.*

To show that the approximation in Theorem 1.3 is necessary, we prove there is no streaming mechanism to find the prices that maintain the optimal social-welfare using space sublinear in l.

Theorem 1.5 *There is no envy-free streaming mechanism that finds the welfare-optimal envy-free prices using space $o(l)$. This bound holds even for $k = 2$.*

As with welfare maximization, any algorithm that computes revenue-optimal envy-free prices would require space that is at least linear in l.

Theorem 1.6 *There is no envy-free streaming mechanism which finds the set of prices that maximize the revenue using a space sublinear in l. This bound holds even for $k = 2$.*

Note that due to space constrains, some of the proofs and figures are omitted in this version and included in the full version. The full version of the paper is provided on arXiv.

1.2 Related Work

In this paper we focus our attention on the problem of finding envy-free prices for unit-demand bidders in the streaming setting, a problem that has received much attention in the static setting. The revenue-maximizing envy-free pricing problem was introduced by Guruswami et al. (Guruswami et al. 2005). There has since been a significant line of work attacking variants of this problem (Cheung and Swamy 2008; Chen, Ghosh, and Vassilvitskii 2011; Chen and Deng 2014), and mounting evidence suggests that it is computationally hard to obtain better than a polylogarithmic approximation for general unit-demand bidders (Briest and Krysta 2011; Chalermsook et al. 2012). For welfare maximization, it is well-known that a Walrasian equilibrium corresponds to a set of envy-free prices that optimizes welfare, and such an equilibrium always exists for unit-demand bidders. Moreover, in the static setting such prices can be found in polynomial time (Shapley and Shubik 1971; Bikhchandani and Mamer 1997). Our focus is on developing streaming algorithms for these problems.

Our motivation of determining prices from sampled data relates to a recent line of literature on the sample complexity of pricing problems and applications of statistical learning theory. Much of this work has focused on the problem of learning an approximately revenue-optimal reserve price in a single-item auction (Dhangwatnotai, Roughgarden, and Yan 2010; Fu et al. 2015; Huang, Mansour, and Roughgarden 2015; Cole and Roughgarden 2014). More generally, statistical learning methods have been used to quantify the sampling complexity of learning approximately optimal auctions, in the prior-free context by Balcan et al. (Balcan et al. 2008) and in a prior-independent setting by Morgenstern and Roughgarden (Morgenstern and Roughgarden 2015).

Hsu et al. (Hsu et al. 2015) study the genericity of market-clearing prices learned from sampled data, and demonstrate that under some conditions on buyer preferences (including the unit-demand case studied here) prices computed from a large dataset will approximately clear a "similar" market; that is, one where buyer preferences are drawn from a the same underlying distribution.

Our technical results build upon recent work in the streaming algorithms literature on maximum matching. Chitnis et al. (Chitnis et al. 2015) consider the matching problem in the streaming setting and provide optimum solutions to both vertex cover and matching in $\tilde{O}(k^2)$ space, where k is the size of the solution. In addition, they show that any streaming algorithm for the maximum matching problem requires $\Omega(k^2)$ space. Later, they extend this result to dynamic streams in which we have both addition and deletion of edges (Chitnis et al. 2016).

McGregor (McGregor 2005) considered the matching problem in the streaming setting with several passes. He provides a $(1 - \epsilon)$-approximation algorithm for unweighted graphs and a $(0.5 - \epsilon)$-approximation algorithm for weighted graphs, both with constant number of passes and using $\tilde{O}(n)$ space.

Esfandiari et al. (Esfandiari et al. 2015) consider the maximum matching problem in planer graphs and bounded arboricity graphs. They provide a constant approximation of the size of a maximum matching in these graphs using $\tilde{O}(n^{2/3})$ space in the streaming setting.

Later, simultaneously Bury et al. (Bury and Schwiegelshohn 2015) and Chitnis et al. (Chitnis et al. 2016) extend this algorithm to work for both addition and deletion of edges using a larger space of $\tilde{O}(n^{4/5})$.

2 Pricing problem: Maximizing Social Welfare

In this section, we consider the problem of assigning prices to items, and items to buyers in a streaming setting such that the assignment would be envy-free, and the *social welfare* is maximized. The social welfare would be sum of the weights (or utilities) of the assigned edges. In Subsection 2.1, we propose the optimum mechanism with $O(k^2 l)$ space, and in Subsection 2.2 we approximate the optimum mechanism with improved memory.

2.1 Envy-Free Mechanism with $O(k^2 l)$ Space

In this subsection, we propose a pricing mechanism to maximizing the social welfare in our setting. As we explained earlier, we only use $O(k^2 l)$ memory for storage of the stream of edges. Our approach is to store the $kl + 1$ edges with maximum weight for each item, and to run the optimum algorithm to find the social welfare maximizing envy-free assignment in offline setting when the stream ends. We call the optimum streaming algorithm of this subsection *SWM* to use it in Subsection 2.2.

The following theorem is the main result of this subsection.

Theorem 2.1 *Our streaming assignment algorithm which assigns prices to items and items to buyers in the aforementioned market is an envy-free social welfare maximizing assignment, and it uses $O(kl^2)$ memory.*

2.2 Improving Space Efficiency While Approximating Social Welfare

In this subsection, we try to improve space efficiency in the problem solved in Subsection 2.1, when we relax the goal of achieving maximum social welfare to obtaining an approximation of it. More specifically, suppose we have k item types, l available items of each type, and n buyers, and the utilities of buyers for item types are revealed in a streaming fashion. Recall that we can find the social welfare maximizing prices for the items and an assignment of items to buyers in $O(k^2 l)$ available memory. In this subsection, our goal is to find prices for item types when the amount of available memory is independent of l (the number of available items of each item type). We prove when each buyer picks the most profitable item based on the prices that our algorithm suggests and his own utilities, there would be no more than l requests for any

Algorithm 1:

Input: Weighted bipartite graph G with set of vertices $B \cup V$, l number of available items from each item type, and constants $\epsilon, \delta > 0$.
Output: Price vector $\vec{p}$ which yields a $(1-2\epsilon)$-approximation of opt SW and an envy-free assignment with probability $1-\delta$.

$t \leftarrow 3\ \frac{-log(\delta)+log(2k)+klog(n)}{\epsilon^2}$
$B' \leftarrow \emptyset$
for $b \in B$ **do**
  Add b to B' with probability $\frac{t}{l}$
Let G' be the subgraph of G induced by $B' \cup V$;
$l' \leftarrow t(1-\epsilon)$ be the number of available items of each item type in G'
Upon stream of edges in G, ignore any edge $e \notin G'$
Find optimal $\vec{p}$ using SWM Algorithm (Subsection 2) on edges of G'
Return $\vec{p}$;

of the item types with high probability, and the social welfare would be a good approximation of the optimum social welfare. Thus, we can conclude this self selection of items by customers is envy-free and valid with high probability, and we do not have to deal with item to buyer allocations after setting the prices. Our approach here is to collect a sample of buyers while the data is being streamed, decide the prices based on this sample, and prove that these prices would yield a good approximation of the social welfare and an envy-free assignment of items to buyers in the original graph while the assignment of items is done by the buyers themselves and not by us. This algorithm is especially favorable over previous ones when the number of different item types is relatively small compared to the total number of items. In other words, when k is small compared to l.

Let B be the set of our n buyers, and V the set of k item types. Assume we have l available items of each type. Let G be the weighted bipartite graph of buyers and item types showing utilities of buyers for items. For arbitrary constants $\delta, \epsilon > 0$, our algorithm finds prices of items in V such that the greedy item picking strategy by buyers would yield a valid envy-free assignment and achieves a social welfare that is $(1-2\epsilon)$-approximation of the maximum possible social welfare with probability $1-\delta$. We define a new parameter $t = 3\ \frac{-log(\delta)+log(2k)+klog(n)}{\epsilon^2}$, and sample every buyer in B with probability $\frac{t}{l}$. Let B' be the set of buyers chosen in our sampling, and G' be the induced subgraph of G when we remove all the vertices that are not in B'. We assume there are $(1-\epsilon)t$ available copies from each item type in graph G' which can be sold to the buyers in B'. As we discussed in previous sections, we can find the optimal prices for items in B' to achieve maximum Social welfare in graph G' in $O(k^2 t)$ available memory. After this step, we use the same prices for the general case, and prove that these prices along with the greedy item selection by buyers satisfies the aforementioned criteria.

We bound the approximation ratio of Algorithm 1 by following theorem.

Theorem 2.2 *The pricing suggested by Algorithm 1 along with the greedy selection of items by the buyers yields a valid envy-free assignment and a $(1-2\epsilon)$-approximation of the maximum possible social welfare with probability $1-\delta$. The space needed by Algorithm 1 is independent of l, the number of available items of each item type.*

3 Pricing problem: Maximizing Revenue

Just like the previous section, we try to find a pricing for items, and an envy-free assignment of items to buyers in our described market when the input is revealed in a streaming fashion. However, in this section, we aim to maximize *revenue* instead of social welfare. In Subsection 3.1, we propose the optimum mechanism with $O(k^2 l)$ space, and in Subsection 3.2 we approximate the optimum mechanism with improved memory.

3.1 Envy-Free Mechanism with $O(k^2 l)$ Space

In this subsection, we show that if we are given an envy-free α-approximation mechanism for the revenue maximization problem then we can have an envy-free α-approximation mechanism for the revenue maximization problem in the streaming setting with $O(k^2 l)$ available memory. We call this mechanism designed for the streaming setting *RM* for use in the later subsection.

The following theorem is the main result of this subsection:

Theorem 3.1 *Given an envy-free α-approximation mechanism for the offline revenue maximization problem, there exists an envy-free α-approximation mechanism for the revenue maximization problem in the streaming setting using $O(k^2 l)$ space.*

3.2 Improving Space Efficiency While Approximating Optimum Revenue

In Subsection 3.1, we introduced a simple streaming mechanism (RM) that finds the price vector and an envy-free assignment of items to buyers to α-approximate maximum *revenue* given a mechanism that α-approximate the maximum revenue in the offline case. In this subsection, we are concerned with reducing the amount of space used by our streaming algorithm. As we mentioned earlier, $O(k^2 l)$ available space is needed for any streaming algorithm that finds a revenue maximizing assignment in our setting. Just like the previous section, we are interested in a streaming algorithm for which the amount of space used is independent of l, the number of copies of each available item type. Algorithm 2 is our algorithm for this purpose. As a result of reduction in the required memory, the revenue of the assignment given by our algorithm loses another $(1-2\epsilon)$ approximation factor compared to the maximum possible revenue. When l is small compared to k, this algorithm would be beneficial since it dramatically improves the amount of space used. The algorithm and some of the proofs are similar to the ones in the previous section.

We bound the approximation ratio of Algorithm 2 by following theorem.

Theorem 3.2 *The pricing suggested by Algorithm 2 along with the greedy selection of items by the buyers yields a valid*

Algorithm 2:

Input: Weighted bipartite graph G with set of vertices $B \cup V$, l number of available items from each item type, and constants $\epsilon, \delta > 0$.

Output: Price vector $\vec{p}$ which yields a $(1-2\epsilon)$-approximation of opt revenue and an envy-free assignment with probability $1-\delta$.

$t \leftarrow 3\,\frac{-log(\delta)+log(2k)+klog(n)}{\epsilon^2}$

$B' \leftarrow \emptyset$

for $b \in B$ **do**

  Add b to B' with probability $\frac{t}{l}$

Let G' be the subgraph of G induced by $B' \cup V$;

$l' \leftarrow t(1-\epsilon)$ be the number of available items of each item type in G'

Upon stream of edges in G, ignore any edge $e \notin G'$

Find optimal $\vec{p}$ using RM Algorithm (Subsection 3.1) on edges of G'

Return $\vec{p}$;

envy-free assignment and a $(1-2\epsilon)\alpha$-approximation of the maximum possible revenue with probability at least $1-2\delta$ given an envy-free mechanism that α-approximates maximum revenue in the offline case. The space needed by Algorithm 2 is independent of l, the number of available items of each item type.

4 Hardness Results

In Sections 2 and 3, we provided optimum as well as approximation algorithms for envy-free pricing problems to maximize social welfare or revenue in the streaming setting. In this section, we provide lower bounds on the required spaces to solve these problems optimally. To provide these hardness results we use the communication hardness of set disjointness problem.

4.1 Hardness of Social Welfare Maximization

In Section 2, we presented a streaming algorithm which finds an envy-free social welfare maximizing assignment of prices to items and items to buyers using $O(k^2 l)$ memory, where k is the number of item types and l is the number of available items of each type. This result raises the following interesting question. Is it necessary to have $\Omega(l)$ available memory for solving this problem? In other words, if the number of item types is small compared to the total number of items (or k is small compared to l, can we solve the problem in space independent of l? In this section we prove for any constant $\epsilon > 0$, no streaming algorithm can ϵ-approximate the envy-free social welfare maximizing *prices* in $o(l)$ space. The proof is done via a reduction from *Disjointness*, a well-known communication complexity problem .

Definition 4.1 Disjointness Problem *is a communication complexity problem in which Alice is given a string $x \in \{0,1\}^n$ and Bob is given a string $y \in \{0,1\}^n$. Their goal is to decide whether there is an index i, such that $x_i = y_i = 1$. Index i in this case is called an* intersection*. It is known that the minimum number of bits required to be exchanged between Alice and Bob to find an intersection is $\Omega(n)$ bits even with multi-passes allowed.*

Theorem 4.2 *For any arbitrary small constant $\epsilon > 0$, there is no streaming algorithm which uses $o(l)$ space and ϵ-approximates all the item prices of the social welfare maximizing price vector.*

Proof. For an arbitrary ϵ, assume for the sake of contradiction there exists an algorithm A which can find an ϵ-approximation of an optimal pricing in $o(l)$ space. We show a reduction from any instance of Disjointness problem to an instance of our market design problem such that if Algorithm A exists, Disjointness problem can be solved using $o(l)$ space.

Let $\mathcal{I}_1$ be an instance of Disjointness problem with $x \in \{0,1\}^l$ as Alice's string and $y \in \{0,1\}^l$ as Bob's string. A corresponding instance of our market design problem $\mathcal{I}_2$ can be built as follows. Consider two item types in $\mathcal{I}_2$, one corresponding to Alice and one corresponding to Bob. Suppose each of these two item types have $2l$ copies available. Let $G = (V_1, V_2, E)$ be the bipartite graph of item types and buyers in instance $\mathcal{I}_2$, with $V_1 = \{v_{Alice}, v_{Bob}\}$ as the item type vertices. We start with $2l$ buyer vertices $V_a = \{a_1, a_2, \ldots, a_l\}$ and $V_b = \{b_1, b_2, \ldots, b_l\}$ in V_2. For any index i, if $x_i = 1$, we connect v_{Alice} to both vertices a_i and b_i. Similarly, for any index i such that $y_i = 1$, we connect v_{Bob} to both vertices a_i and b_i. Let I_x be the set of all indices j such that $x_j = 1$ in string x. We add a set U_{Alice} with $2l - |I_x|$ buyer vertices to V_2, and connect v_{Alice} to all vertices in U_{Alice} so that the degree of v_{Alice} is exactly $2l$. Similarly, we add a set of vertices U_{Bob} with $2l - |I_y|$ buyer vertices to V_2 and connect v_{Bob} to all vertices in U_{Bob}. Finally, we add two buyer vertices u_1, u_2 to V_2 and connect v_{Bob} to both of them. Note that the set of buyer vertices V_2 are $\{u1, u2\} \cup V_a \cup V_b \cup U_{Alice} \cup U_{Bob}$. The utility of buyers u_1 and u_2 for Bob's item are ϵ and ϵ^3 respectively. The utility of any other buyer for any other item connected to it (v_{Alice} or v_{Bob}) is 1. This means, the weight of the edge (v_{Bob}, u_1) is ϵ, the weight of the edge (v_{Bob}, u_2) is ϵ^3, and the weight of all the other edges in E is 1.

Now suppose both Alice and Bob know about algorithm A. Let E_{Alice} be the set of edges connected to v_{Alice}, and E_{Bob} the set of edges connected to v_{Bob} in graph G. Note that Alice only knows about E_{Alice} and buyer vertices $V_2 \backslash (\{u_1, u_2\} \cup U_{Bob})$. Similarly, Bob only knows about E_{Bob} and buyer vertices $V_2 \backslash U_{Alice}$. Alice starts streaming her edges and running algorithm A on it. She sends the information that algorithm A stores in $o(l)$ available space to Bob. Bob at the other end receives all the information stored by algorithm A and sent by Alice, and continues running algorithm A by streaming his own edges. Algorithm A can find social welfare maximizing prices for both Alice and Bob items in $o(l)$ space. The algorithm finishes at Bob's end after he streams all of his edges. At this point we claim that Bob can decide whether the strings have intersection or not based on the following two case. If the price suggested by algorithm A for his item type is less than ϵ, Bob should declare no intersections exist and if the price is above ϵ^2 he should declare existence of at least one intersection. Furthermore, algorithm A never set a price between ϵ and ϵ^2 for v_{Bob}. Next we prove why this claim is valid.

Suppose Alice and Bob's strings have no intersections. Then in graph G, no buyer is connected to both Alice and Bob. That is v_{Bob} is connected to $2l$ buyer vertices with utility 1 for his item and none of his buyers want Alice's item. Alice's item is also connected to $2l$ buyer vertices that do not want Bob's item. The optimal prices for both Alice and Bob's items to maximize social welfare is 1 in this case, and no item is sold to buyers u_1 and u_2.

On the other hand, if the strings have at least one intersection, say at index i, both v_{Alice} and v_{Bob} are connected to a_i and b_i. To maximize social welfare in this case, Alice sell all of her items to the $2l$ buyers who want her item at price 1, and Bob can sell at most $2l-2$ items to those who want his item at price 1 and has to sell two items to buyers u_1 and u_2 who want to pay ϵ and ϵ^3 for his item respectively. Therefore, the price for Bob's item should be ϵ^3. Since the goal is to maximize social welfare, Bob cannot decide to leave out u_1 and u_2 and sell $2l-2$ items at price 1 to the buyers whose utility for his item is 1.

By assumption, algorithm A can ϵ-approximate all optimal prices for social welfare maximization while Alice and Bob stream the edges using only $o(l)$ available space. Specifically, if the optimal price for v_{Bob} is 1, i.e, there is no intersection in the two strings, algorithm A sets a price higher than ϵ for v_{Bob}. Otherwise, in case the optimal price for v_{Bob} is ϵ^3, algorithm A sets a price lower than ϵ^2 for Bob's item. These two prices are the only optimal prices for Bob's item and thus, only these two cases exist. Hence, Bob can distinguish between these two cases by checking the price set for his item once the algorithm ends. Any price less than ϵ^2 corresponds to an intersection, and any price higher than ϵ signals no intersection between the strings. □

4.2 Hardness of Revenue Maximization

In Subsection 4.1, we presented a hardness proof to show no streaming algorithm exists to approximate the optimal prices using $o(l)$ available space in our market design problem with the goal of *social welfare maximization*. In this subsection, we establish a hardness result for the case that our goal is to maximize the revenue, however, the result of this subsection does not involve any approximation. That is we only guarantee there exists no algorithm which finds the exact optimal prices for revenue maximization market design problem in $o(l)$ space. Just like previous subsection, our approach is based on a reduction from *Disjointness problem*.

Theorem 4.3 *There is no streaming algorithm which uses $o(l)$ memory, and finds the revenue maximizing price vector for an envy free assignment.*

References

Balcan, M.-F.; Blum, A.; Hartline, J. D.; and Mansour, Y. 2008. Reducing mechanism design to algorithm design via machine learning. *Journal of Computer and System Sciences* 74(8):1245 – 1270. Learning Theory 2005.

Bikhchandani, S., and Mamer, J. W. 1997. Competitive Equilibrium in an Exchange Economy with Indivisibilities. *Journal of Economic Theory* 74(2):385–413.

Briest, P., and Krysta, P. 2011. Buying cheap is expensive: Approximability of combinatorial pricing problems. *SIAM J. Comput.* 40(6):1554–1586.

Bury, M., and Schwiegelshohn, C. 2015. Sublinear estimation of weighted matchings in dynamic data streams. *arXiv preprint arXiv:1505.02019*.

Chalermsook, P.; Chuzhoy, J.; Kannan, S.; and Khanna, S. 2012. Improved hardness results for profit maximizations pricing problems with unlimited supply. In *Proceedings of APPROX*.

Chen, N., and Deng, X. 2014. Envy-free pricing in multi-item markets. *ACM Trans. Algorithms* 10(2):7:1–7:15.

Chen, N.; Ghosh, A.; and Vassilvitskii, S. 2011. Optimal envy-free pricing with metric substitutability. *SIAM Journal on Computing* 40(3):623–645.

Cheung, M., and Swamy, C. 2008. Approximation algorithms for single-minded envy-free profit-maximization problems with limited supply. In *Foundations of Computer Science, 2008. FOCS '08. IEEE 49th Annual IEEE Symposium on*, 35–44.

Chitnis, R.; Cormode, G.; Hajiaghayi, M.; and Monemizadeh, M. 2015. Parameterized streaming: maximal matching and vertex cover. In *Proceedings of the 26th Annual ACM-SIAM Symposium on Discrete Algorithms*, 1234–1251. SIAM.

Chitnis, R.; Cormode, G.; Esfandiari, H.; Hajiaghayi, M.; McGregor, A.; Monemizadeh, M.; and Vorotnikova, S. 2016. Kernelization via sampling with applications to dynamic graph streams. 1326–1344.

Cole, R., and Roughgarden, T. 2014. The sample complexity of revenue maximization. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*, STOC '14, 243–252. New York, NY, USA: ACM.

Dhangwatnotai, P.; Roughgarden, T.; and Yan, Q. 2010. Revenue maximization with a single sample. In *Proceedings of the 11th ACM Conference on Electronic Commerce*, EC '10, 129–138. New York, NY, USA: ACM.

Esfandiari, H.; Hajiaghayi, M. T.; Liaghat, V.; Monemizadeh, M.; and Onak, K. 2015. Streaming algorithms for estimating the matching size in planar graphs and beyond. In *Proceedings of the 26th Annual ACM-SIAM Symposium on Discrete Algorithms*, 1217–1233. SIAM.

Fu, H.; Immorlica, N.; Lucier, B.; and Strack, P. 2015. Randomization beats second price as a prior-independent auction. In *Proceedings of the Sixteenth ACM Conference on Economics and Computation*, EC '15, 323–323. New York, NY, USA: ACM.

Guruswami, V.; Hartline, J. D.; Karlin, A. R.; Kempe, D.; Kenyon, C.; and McSherry, F. 2005. On profit-maximizing envy-free pricing. In *Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms*, 1164–1173. SIAM.

Hsu, J.; Morgenstern, J.; Rogers, R. M.; Roth, A.; and

Vohra, R. 2015. Do prices coordinate markets? *CoRR* abs/1511.00925.

Huang, Z.; Mansour, Y.; and Roughgarden, T. 2015. Making the most of your samples. In *Proceedings of the Sixteenth ACM Conference on Economics and Computation*, EC '15, 45–60. New York, NY, USA: ACM.

McGregor, A. 2005. Finding graph matchings in data streams. In *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques*. Springer. 170–181.

Morgenstern, J., and Roughgarden, T. 2015. The pseudo-dimension of nearly-optimal auctions. In *NIPS*, Forthcoming.

Shapley, L. S., and Shubik, M. 1971. The assignment game i: The core. *International Journal of Game Theory* 1:111–130. 10.1007/BF01753437.