Algorithms for Multi-Product Pricing

Gagan Aggarwal[1] *, Tomás Feder **, Rajeev Motwani[1] ***, and An Zhu[1] †

Computer Science Department, Stanford University, Stanford, CA 94305.
gagan,rajeev,anzhu@cs.stanford.edu. tomas@theory.stanford.edu

Abstract. In the information age, the availability of data on consumer profiles has opened new possibilities for companies to increase their revenue via data mining techniques. One approach has been to strategically set prices of various products, taking into account the profiles of consumers. We study algorithms for the multi-product pricing problem, where, given consumer preferences among products, their budgets, and the costs of production, the goal is to set prices of multiple products from a single company, so as to maximize the overall revenue of the company. We present approximation algorithms as well as negative results for several variants of the multi-product pricing problem, modeling different purchasing patterns and market assumptions.

1 Introduction

Through interaction with online consumers, e-commerce websites can gather data reflecting consumer preferences. Such data allows significant revenue increases through strategic price setting via sophisticated analytical tools. While the airline and hotel industry were the traditional users of revenue management [9, 14], corporations in other verticals in retail and manufacturing have recently started employing intelligent pricing strategies to boost their bottom lines. For instance, Dell quotes different prices to different market segments for the same product, enabling Dell to increase its market share and profitability [10]. Other documented examples include Ford [2] and ShopKo Stores [11].

Motivated by the availability of such data, Rusmevichientong, Van Roy, and Glynn formulated the non-parametric multi-product pricing problem [13]. In multi-product pricing, given demands and production costs, the problem is to determine an assignment of prices to products that maximizes overall revenue or profit [6, 7]. When faced with choice between multiple substitutable products, consumers may be indifferent between two or more products and substitute one product for another. The problem of modeling substitutability among products and determining optimal pricing in that context remains a challenging open problem in this research area. As noted in [13], in order to capture

* Supported in part by a SGF fellowship from Stanford and NSF Grant EIA-0137761.
** 268 Waverley St., Palo Alto, CA 94301.
*** Supported in part by NSF Grant IIS-0118173 and EIA-0137761, an Okawa Foundation Research Grant, and grants from Microsoft and Veritas.
† Supported in part by a GRPW fellowship from Bell Labs, Lucent Technologies, and NSF Grant EIA-0137761.

substitutability, in most models, consumer demand functions are generally assumed to take on specific parametric forms [3]. However, these parametric forms may not reflect the true demand function accurately. We adopt the non-parametric approach proposed in [13].

The non-parametric approach employs large quantities of consumer data in order to reduce reliance on rigid parametric models, where the consumer data is obtained from Internet sites. A concrete example is the General Motors' Auto Choice Advisor website [4]. This website is set up to advise potential purchasers of cars (of all brands) on products that meet their requirements. The website asks the users various questions about their preferences with respect to cars, such as model, budget, primary use, etc., and then recommends some cars that satisfy their requirements. Thus, GM has access to large quantities of data that reflect consumer preferences. This data can be used by GM to optimize the pricing of their own vehicles. Based on this scenario, Rusmevichientong et al. [12, 13] introduced the non-parametric multi-product pricing model, which we describe next.

Consider the problem of pricing n different products $\mathcal{P}$ indexed by $A = \{1,\ldots,n\}$, based on a set of data samples, $(R_1,B_1),\ldots,(R_m,B_m)$, each associated with one of m different consumers that have visited an e-commerce website. For each j, let $R_j = \pi(A)$ be the preference ordering of the j^{th} consumer over all n products, where products ranked lower are preferred over products ranked higher. Further, let b_{ij} be the *budget* of consumer j for product i, i.e. the maximum amount consumer j is willing to spend for product i. Also, let $B_j = \{b_{ij}, \forall 1 \leq i \leq n\}$ denote the sequence of budgets of the j^{th} consumer. We assume consumers are consistent, i.e., the order of their budgets for various products obeys their preference ordering. Each sample (R_j,B_j) represents all available information about the j^{th} consumer. We further assume that, given that the products are priced at $P = \{P_1,\ldots,P_n\}$, the j^{th} consumer will purchase the lowest-ranked product in their preference list which she can afford, i.e., they will buy product k, where $k = \arg\min_{B_{ij} \geq P_i} R_j(i)$. This model of consumer behavior is termed *Rank-Buying*. In addition, we assume that there is a *Price-Ladder (PL)* constraint, i.e., we know the relative ordering of the prices of various products. Such a constraint can usually be deduced from market studies/policies and manufacturing costs. Assuming each consumer buys only one product, the objective is to set the product prices so as to maximize the revenue.

Rusmevichientong et al. [12, 13] studied this basic model and presented various heuristics. However, they left open the problem of designing algorithms with provable quality guarantees. In this paper, we present a PTAS for this problem. In addition, we consider some interesting variants of the original model:

1. Given the prices, suppose the consumer will buy the most expensive (least expensive) product that she can afford. These variants, called the *Max-Buying* (respectively, *Min-Buying*) models, were proposed in [12, 13]. Assuming a price ladder, the Min-Buying model has been shown to be solvable in polynomial time using dynamic programming [12, 13].
2. In another variant, referred to as *No-Price-Ladder (NPL)*, we do not assume a price ladder on the products.

3. In yet another variant, we are given costs of production, and the goal is to maximize the profit rather than the revenue.
4. We also consider the model where there are a limited number of copies of each product – inventory/production constraints might lead to such a condition.

We present algorithms and negative results for various combinations of these variations on the original model. Our results are summarized in the following table. The first column lists the particular model being considered in terms of price ladder or no price ladder, the buying behavior, and limited versus unlimited copies of products. The second column gives upper and lower bounds on the approximation bounds achievable in polynomial time. The last column gives extensions of the model where the algorithm remains applicable.

Model	Upper [Lower] Bounds	Extensions
PL & Rank-Buying	PTAS	Max-Buying instead of Rank-Buying
PL & Max-Buying & Limited-Copies	4	Consumers arrive online
NPL & Max-Buying	1.59 [16/15]	Maximize Profit instead of Revenue
NPL & Min-Buying	$\log m$ $[1+\varepsilon]$	Upper bound holds for all models

The rest of the paper is organized as follows. In Sect. 2, we present the PTAS and the 4-approximation algorithm for the Rank-Buying with Price-Ladder problem and related models. In Sect. 3, we discuss the No-Price-Ladder model. We present the 1.59-approximation algorithm and the hardness result for the Max-Buying with No-Price-Ladder problem, and the hardness result for the case of Min-Buying with No-Price-Ladder. Section 4 presents the $O(\log m)$-approximation algorithm that works for all models and discusses some related issues. Finally, we conclude with some open problems in Sect. 5.

2 Rank/Max Buying with Price-Ladder

We first show that when there is a Price-Ladder constraint, the Rank-Buying model can be reduced to the Max-Buying model[1].

Lemma 1. *We can transform an instance I in the Rank-Buying with Price-Ladder model to an equivalent instance I' in the Max-Buying with Price-Ladder model.*

Proof. In the instance I, consider a consumer j and any two products i and i' such that $P_i \leq P_{i'}$ and $R_j(i) < R_j(i')$, where $R_j(i)$ denotes the position of product i in j's preference list, with the most preferred product ranked lowest. For such a pair of products, if the consumer could afford both products, Max-Buying would lead the consumer to

[1] Note that the reduction is valid only in this setting, and may not work for other extensions and variations in this paper.

buy i', while Rank-Buying would let her buy i, a product which is cheaper and more preferred. In order to reduce Rank-Buying to Max-Buying, we would like to eliminate all such product pairs without altering the solution to the instance. We note that for such product pairs i, i', since the budgets are assumed to be in the same order as the rank, the budget for i is higher than the budget for i'. Since $P_i \leq P_{i'}$, the consumer can afford i, whenever she can afford i', and since the consumer buys by rank, she would buy i rather than i'. Thus, we see that the consumer never buys i'. So we can reduce the budget $b_{i'j}$ to 0, without affecting the outcome in the Rank-Buying model. By repeating this for every product pair with conflicting rank and price orders, we can create an equivalent instance I' in which the rank order (equivalently, budget order) conforms to the price order for each consumer. Consequently, Max-Buying gives the same outcome as Rank-Buying on this new instance I', which in turn gives the same outcome as Rank-Buying on the original instance I. $\square$

We now present a PTAS for the Max-Buying model, which along with the above transformation, will give us a PTAS for the Rank-Buying model. We begin by noting that given any solution assigning prices to products, we can transform the solution to one in which the prices are taken from the set of distinct budget values, without decreasing the revenue generated by the solution. This general observation holds for all models studied in this paper.

Assume that the products are listed in the order of decreasing prices (as specified by the Price-Ladder), i.e., $P_1 \geq P_2 \geq \cdots \geq P_n$. We first relax the problem in two ways:

1. Let $\mathcal{B} = \max_{i,j} B_{ij}$. We discretize the search space of possible prices into values of the form $d_i = \mathcal{B}/s^i$, where $s > 1$ will be chosen later.
2. We relax the constraint that a consumer can purchase at most one product. Instead, we allow the consumer to buy multiple products. However, if j buys a product at price p, then she is not allowed to buy any other product with price p', where $p \leq p' < s^k p$, where the integer k is chosen later.

Consider the modified instance. By the first relaxation, we lose a factor of s; by the second relaxation, we lose a factor of $1 + \frac{1}{s^k} + \frac{1}{s^{2k}} + \frac{1}{s^{3k}} + \cdots \leq \frac{1}{1-s^{-k}}$. Combining the two error factors gives a factor of $\frac{s}{1-s^{-k}}$, which is minimized at $s = (k+1)^{1/k}$, where it equals $(k+1)^{\frac{1}{k}}(1+\frac{1}{k}) = 1 + \frac{\log k}{k}(1+o(1))$. This approximation factor is 4 for $k = 1$ and can be made $1+\varepsilon$, for any constant ε, by taking a suitably large k.

We next show how to obtain the optimal solution to this modified problem by dynamic programming. Define $F(i, x_i, x_{i-1}, x_{i-2}, \ldots, x_{i-k+1})$ to be the maximum revenue generated by only considering products with prices ranging from d_0 (=$\mathcal{B}$) to d_i, with x_j being the last product with price set to d_j or higher. Note that in order to respect the price ladder, x_j must precede x_{j+1} or be equal to it in the price ladder. To compute $F(i+1, x_{i+1}, x_i, \ldots, x_{i-k+2})$, we enumerate through each choice of x_{i-k+1} (the number of choices is at most x_{i-k+2}). Let $C(x_{i+1}, x_i, \ldots, x_{i-k+2}, x_{i-k+1})$ denote the number of consumers that satisfy the following two conditions:

– For $i-k+2 \leq j \leq i$, if prices of all products $x_{j-1}+1, \ldots, x_j$ are set to d_j, the consumer will not be able to afford any of these products.

– If the price of products $x_i+1,\ldots,x_{i+1}$ was set to d_{i+1}, the consumer will be able to afford at least one of these products.

Define $G(i+1,x_{i+1},x_i,\ldots,x_{i-k+1}) = F(i,x_i,\ldots,x_{i-k+1}) + d_{i+1}C(x_{i+1},x_i,\ldots,x_{i-k+1})$. In other words, $G(i+1,x_{i+1},\ldots,x_{i-k+2},x_{i-k+1})$ is the sum of $F(i,x_i,\ldots,x_{i-k+1})$ and the payoff generated by consumers who buy products at price d_{i+1}, while ensuring that they have not bought products of price d_j, for $i-k+2 \leq j \leq i$. We obtain the following recurrence:

$$F(i+1,x_{i+1},\ldots,x_{i-k+2}) = \max_{0\leq x_{i-k+1}\leq x_{i-k+2}} G(i+1,x_{i+1},\ldots,x_{i-k+2},x_{i-k+1}).$$

This leads to the following theorem.

Theorem 1. *The Max-Buying with Price-Ladder problem has a PTAS. It can be approximated within a factor of* $(k+1)^{\frac{1}{k}}(1+\frac{1}{k}) = 1+\frac{\log k}{k}(1+o(1))$, *in time* $\binom{n+k-1}{k}n^2m^2 = O(n^{k+2}m^2)$ *and space* $O(n^{k+1}m)$. *With an additional approximation factor of* $1+\frac{1}{\mathrm{poly}(n)}$, *the complexity can be improved to time* $O(n^{k+1}\log n)$ *and space* $O(n^k \log n)$.

Proof. The optimal solution will be the best value of $F(l,\ldots)$, where d_l is the smallest price considered. The number of choices for the arguments of $F(i,x_i,x_{i-1},\ldots,x_{i-k+1})$ is nm for the value of i (since there are at most $O(nm)$ distinct B_{ij}'s, and it is easy to see that there exists an optimal solution where the set of distinct prices is a subset of the set of distinct budget values) and $\binom{n+k-1}{k}$ for the x_j's. Each value of F requires $O(nm)$ computation time and $O(1)$ space, giving the stated bounds. If we restrict the smallest allowed price to $\frac{\mathcal{B}}{n\mathrm{poly}(n)}$, we incur an additional $(1+\frac{1}{\mathrm{poly}(n)})$-approximation factor, but reduce the number of choices for i from nm to $O(\log n)$, giving the better time and space bounds. □

In Appendix A, we present another algorithm which requires only linear space, but has a worse (super-polynomial) time bound.

We now consider the Max-Buying with Price-Ladder model with the additional constraint that there are only N_i copies of the i^{th} product. Since the number of copies of the products are limited, we need to redefine the optimal revenue generated by a setting of prices. Given a price setting and an order in which the consumers arrive, we assume that when a consumer arrives, she buys the most expensive product she can afford that is still available. We define the maximum revenue for a setting of prices as the payoff generated by the best arrival order of consumers (or the best allocation of the limited copies of products to consumers in the offline setting). In the more realistic case of consumers arriving in an arbitrary order, the payoff generated is within factor 2 of the best possible allocation as shown by the following lemma.

Lemma 2. *Let OPT denote the revenue generated by the best allocation. If* $\mathcal{R}(\pi)$ *denotes the revenue generated by arrival order* π *of consumers, then* $\mathcal{R}(\pi) \geq \frac{1}{2}OPT \quad \forall\pi$.

Proof. Let $A(i)$ (respectively, $B(i)$) be the set of consumers in the optimal (respectively, π) allocation who buy product i at price P_i. Consider those products i for which we have $|A(i)| > |B(i)|$. Since some copies of product i are still left over at the end of the π-allocation, the consumers in $A(i) - B(i)$ must all have bought some product with price at least P_i. So we charge the extra revenue generated by product i under optimal allocation to the revenue generated by these consumers under allocation π. Since each consumer gets charged at most once, the extra revenue generated by optimal allocation is no more than the total revenue generated by π, and hence the lemma. We note that this bound of 2 is tight. □

Theorem 2. *The Limited-Copies variant can be approximated to be a factor of 4, giving an 8-approximation algorithm for the online case.*

Proof. We use the same relaxation techniques as in the PTAS above. We set $k = 1$, which leads to the relaxation where a consumer is allowed to buy one product at every price d_i. The prices differ from each other by powers of 2. We set up a recursion for $F(i, x_i)$ (we enumerate over all possible values of $x_{i-1} \leq x_i$, adding the payoff from setting the price of products $x_{i-1} + 1, \ldots, x_i$ to d_i to the optimal payoff $F(i-1, x_{i-1})$ from prices d_{i-1} and higher, and take the maximum over all these values to get $F(i, x_i)$). This gives us an approximation factor of 4. The reason higher values of k do not work lies in the difficulty of recursion in the dynamic programming. If we try to set up a recursion for $F(i, x_i, x_{i-1})$ instead of $F(i, x_i)$, the values x_i and x_{i-1} are not enough (in the limited copies scenario) to determine the products available at price d_{i-1}, and hence not enough to determine the set of consumers that buy products with price d_i. □

The Max-Rank model can be extended to take into account the competitors' products and the PTAS works in that case as well. In addition to our products $A = \{1, \ldots, n\}$, the competitors produce the remaining products $\overline{A} = \{n+1, \ldots, N\}$. Each consumer has a complete ranking list and budgets for all N products. In addition, the prices of the competitors' products $P_{n+1}, \ldots, P_N$ are specified. Again, each consumer buys the lowest-ranked product that she can afford. If a consumer buys a competitors' product, then we get no revenue from that consumer. The objective is to maximize the revenue generated by our products. We can reduce any instance that involves competitors' products to an instance limited only to our products. For each consumer j, we can find the lowest-ranked competitors' product that she can afford, say C_j. If any of our products is ranked higher than C_j in j's preference list, then j will never buy that product. On the other hand, if she can afford a product ranked lower than C_j, then she will buy it. Thus, it is sufficient and necessary to modify j's preference list to include only those products that are ranked lower than C_j. This model assumes that the competitors do not respond to our pricing decisions. The detailed proof of the following lemma can be found in [12].

Lemma 3. *Any problem instance in the Max-Rank with Price-Ladder model that includes competitors' products can be reduced to one without any competitors' products, without changing the value of the optimal solution.*

3 The No-Price-Ladder model

We now study a model where no ordering constraints are specified on the prices of products (*No-Price-Ladder*). We first study the Max-Buying with No-Price-Ladder problem and give a 1.59-approximation algorithm for the problem. We also show that it is NP-hard to approximate this problem better than $16/15$. Then, we discuss the Min-Buying with No-Price-Ladder model. The Min-Buying with Price-Ladder problem can be solved optimally by using dynamic programming [13]. However, removing the Price-Ladder constraint makes the problem hard.

3.1 An Algorithm for Max-Buying with No-Price-Ladder

The unordered nature of prices renders the previous approach ineffective. Instead, we use linear programming and rounding techniques to get a 1.59-approximation algorithm for this variant. We will also show how to derandomize this algorithm. Consider the following linear program:

$$
\begin{aligned}
\text{Maximize:} \quad & \textstyle\sum_p y_{jp} p & & \\
\text{subject to:} \quad & \textstyle\sum_p x_{ip} = 1 & \forall i & \quad (1) \\
& y_{jp} \le \textstyle\sum_{p \le B_{ij}} x_{ip} & \forall j, p & \quad (2) \\
& \textstyle\sum_p y_{jp} \le 1 & \forall j & \quad (3) \\
& 0 \le x_{ip} \le 1 & & \\
& 0 \le y_{jp} \le 1 & &
\end{aligned}
$$

Here x_{ip} indicates that product i is assigned price p and y_{jp} indicates that consumer j buys a product at price p. Clearly, if x_{ip} and y_{jp} take values in $\{0,1\}$, the objective function is exactly the revenue we are maximizing. Instead, we relax the constraints on x_{ip} and y_{jp} to allow fractional solutions. We round the fractional optimal solution to obtain an integer solution by assigning product i price p with probability x_{ip}.

Theorem 3. *The Max-Buying with No-Price-Ladder problem can be approximated in polynomial time within a factor $\frac{e}{e-1} < 1.59$.*

Proof. We introduce some notation first. Let $s_{jp} = \sum_{p \le B_{ij}} x_{ip}$ be the total amount of (fractional) products priced at p which consumer j can afford. Let $t_{jp} = \sum_{p \le p'} s_{jp'}$ be the total amount of products priced p or higher which consumer j can afford to buy. Let $z_{jp} = \sum_{p \le p'} y_{jp'}$, the total amount of products consumer j buys at a price p or higher.

We thus have the following relations: $y_{jp} \le s_{jp}$ and $z_{jp} \le t_{jp}$. Now, the probability that consumer j buys at price at least p is $q_{jp} = 1 - \prod_i (1 - \sum_{p \le p' \le B_{ij}} x_{ip'})$. Recalling the definition of t_{jp}, we have that $q_{jp} \ge 1 - (1 - \frac{t_{jp}}{n})^n > 1 - e^{-t_{jp}} \ge \min(t_{jp}, 1)(1 - e^{-1})$.

We now look at the optimal fractional solution. Consumer j buys at price at least p with fractional value exactly $r_{jp} = \min(t_{jp}, 1)$. This implies that q_{jp} majorizes $(1 - e^{-1}) r_{jp}$. Thus, we get that the expected value of our rounded solution is at least a $(1 - e^{-1})$ fraction of that of the optimal fractional solution, thus giving us an approximation factor of 1.59.

The algorithm can be derandomized by replacing any two $0 < x_{ip} < 1$ and $0 < x_{ip'} < 1$ by $x_{ip} + \varepsilon$ and $x_{ip'} - \varepsilon$, for a suitable ε. Since the expected payoff is a linear function of ε, either any positive ε makes the payoff nondecreasing, or else, any negative ε ensures the payoff does not decrease. We may select such an ε with the appropriate sign to obtain $x_{ip} = 0$ or $x_{ip'} = 0$. Repeatedly performing this transformation ensures $x_{ip} = 0$ or $x_{ip} = 1$ for all x_{ip}. $\square$

We remark that this linear programming formulation can be extended to the model where the goal is to maximize profit instead of revenue. For each product i, let $c(i)$ denote the fixed manufacturing cost. We redefine x_{ip} to indicate whether product i is assigned profit p, and y_{jp} to indicate whether consumer j buys a product with profit p. We substitute inequality (2) in our linear programming formulation with the following: $y_{jp} \leq \sum_{p:p+c(i)\leq B_{ij}} x_{ip}$. The rest of the analysis goes through as before.

3.2 Negative Result for Max-Buying with No-Price-Ladder

Consider a special case of the Max-Buying with No-Price-Ladder problem where each consumer j specifies a set S_j of products she is interested in. In addition, her budget $b_{ij} = B_j$ for $i \in S_j$, and $b_{ij} = 0$ otherwise. Also, $B_j \in \{a,b\}$, for all j. We call this the *Uniform-two-budget* problem. We show below that even this special case of the Max-Buying with No-Price-Ladder problem is MAX SNP-hard.

Theorem 4. *The Uniform-two-budget problem with Max-Buying and No-Price-Ladder cannot be approximated better than* $16/15$ *unless* $P = NP$. *There exists a polynomial time algorithm to approximate it within* $1/0.78$.

Proof. If there are only two distinct budget values $a > b$, then the only prices used in an optimal solution are a and b. A consumer with budget a will always spend b, and may or may not spend the additional $a-b$. For every product i, consider a boolean variable x_i with $x_i = 1$ if i has price a and $x_i = 0$ if i has price b. Since we are considering the Max-Buying setting, a consumer j with budget a will pay the additional $a-b$ if the disjunction of x_i for the products in S_j holds, while a consumer j with budget b will pay b if the disjunction of the $\overline{x_i}$ for the products in S_j holds. The problem is thus an instance of MAX-SAT with disjunctions of positive literals having weight $a-b$ and disjunctions of negative literals having weight b. Since the MAX-SAT problem can be approximated within $1/0.78$ [1], this gives us an algorithm for solving the Uniform-two-budget case with an approximation factor of $1/0.78$.

For the hardness result, we reduce MAX-3SAT, which is hard to approximate within $\frac{8}{7} - \varepsilon$ for any $\varepsilon > 0$ (see [5]), to our problem. Consider an instance of MAX-3SAT. Replace clauses that have both positive and negative literals by two clauses that have only positive or only negative literals, such that the original clause is satisfied if both the new clauses are satisfied. For example, the clause $x \vee y \vee \overline{z}$ is replaced by clauses $x \vee y \vee t$ and $\overline{t} \vee \overline{z}$. Since the number of clauses at most doubles, the modified problem is hard to approximate within $\frac{16}{15} - \varepsilon$. Now we reduce this modified instance to an instance of our problem. Let n be the number of clauses. We create consumers corresponding to every clause, and the literals in each clause correspond to the products of interest to the consumers (i.e. S_j). For a product, setting the corresponding variable to be *true* (respectively, *false*) corresponds to setting the price to a (respectively, b). For the positive clauses, there is one consumer with $B_j = a = n$, while for the negative clauses, we have n identical consumers, each with budget $B_j = b = 1$. A solution for the pricing instance corresponds naturally to a solution for the MAX-3SAT instance. The only difference in the objective function values arises from the fact that even if the consumers with budget a do not pay a, they pay at least b. Let n_a be the number of consumers with budget a.

A price setting where n_{ab} of the n_a consumers with budget a spend $b = 1$ has a contribution of n_{ab} from these consumers to the total payoff. Since setting the price of all products to $a = n$ leads to a payoff of $n_a n$, and $n_{ab} \leq n_a$, in the optimal solution, the fraction of the total payoff contributed by these n_{ab} consumers is at most $1/n$, which is negligible. Thus any $16/15 - \varepsilon$ approximate solution for this multi-product pricing instance leads to a $16/15 - \varepsilon$ approximate solution for the MAX-SAT instance. Thus, the Uniform-two-budget case of the Max-Buying with No-Price-Ladder problem cannot be approximated better than $16/15$ unless $P = NP$. □

We now give a hardness result for the Min-Buying with No-Price-Ladder problem.

3.3 Min-Buying with No-Price-Ladder

We consider the Uniform-two-budget case of the problem, where each consumer j specifies a set S_j of products she is interested in, and her budget $b_{ij} = B_j$ for $i \in S_j$, and $b_{ij} = 0$ otherwise. Also, $B_j \in \{a, b\}$, for all j.

Theorem 5. *The Uniform-two-budget case of the Min-Buying with No-Price-ladder problem is NP-hard to approximate within* $1 + \varepsilon$*, for some constant* $\varepsilon > 0$*.*

Proof. We do a reduction from the following MAX-CSP: Consider the MAX-SAT problem consisting of the following two types of clauses, conjunction of only positive literals and disjunction of only negative literals.

We first show that this version of MAX-CSP is MAX SNP-hard. We achieve this goal by first showing that it is NP-hard. Then, the results from Khanna, Sudan, and Williamson [8] will imply that it is MAX SNP-hard (given that it is NP-hard). Specifically, we show that the following version of MAX-CSP is NP-hard: The conjunctions contain only singleton x_i's and the disjunctions are of the form $\overline{x_i} \vee \overline{x_j}$ with two literals. We first note that there exists an optimal solution to this MAX-CSP which maximizes the number of x_i set to 1 while satisfying all $\overline{x_i} \vee \overline{x_j}$ clauses. If a solution does not satisfy all disjunctive clauses, we can convert it into a equally good or better solution as follows: Satisfy all the disjunctive clauses by negating either of the literals for any unsatisfied disjunctive clause. Now, if we have a vertex i for each x_i and view the disjunctions $\overline{x_i} \vee \overline{x_j}$ as edges (i, j), then this MAX-CSP is equivalent to the the maximum independent set problem, which is NP-hard to solve.

Given an instance of this MAX-CSP, we reduce it to an instance of the Min-Buying with No-Price-Ladder problem as follows. There are two distinct budget values $a > b$ with $a = 2b$. Corresponding to each variable x_i, we have a product i. For a product, setting the price to a (respectively, b) corresponds to setting the corresponding variable to be *true* (respectively, *false*). Corresponding to a conjunctive clause, we have a consumer with budget a interested in the products appearing in the clause. Similarly, corresponding to a disjunctive clause, we have a consumer with budget b interested in the products appearing in the clause. Since we are in the Min-Buying setting, a consumer with budget a will always pay b, and will pay the an additional b if the conjunction of the x_i for the products it is interested in holds. A consumer with budget b will pay b if the disjunction of the $\overline{x_i}$ for the products it is interested in holds. If the maximum number of satisfiable clauses is at least a constant fraction of the total number of clauses (which can be ensured by adding some dummy clauses to the instance), then the MAX-SNP-hardness of the MAX-CSP problem implies MAX-SNP-hardness of this case of the pricing problem. □

4 General Algorithms for All Models

In this section, we present an algorithm and hardness result applicable to all six models – {Rank-Buying, Max-Buying, Min-Buying} with {Price-Ladder, No-Price-Ladder}. Recall that the number of products is n while the number of consumers is m.

Theorem 6. *Consider all six models: Rank/Max/Min-Buying with Price-Ladder/No-Price-Ladder. An algorithm which is allowed to assign only k distinct prices to the products cannot have an approximation ratio better than* $\frac{H_m}{k}$ *with respect to an unrestricted optimal solution.*

Proof. To show this lower bound, we create an instance with $n = m$. Consider a situation where a consumer j is interested only in product j and has budget m/j for it and 0 for all other products. In the optimal solution, product i is priced m/i. Thus, in the optimal solution, consumer j spends his budget m/j, and the total payoff is mH_m. Now consider a solution which assigns only k distinct prices to the products. Products priced at any single price m/j can be afforded by at most the first j consumers, thus giving a payoff of $\leq m$ for that price. Thus the total payoff with k distinct prices is at most km. This gives the H_m/k lower bound. □

When $k = 1$, the above bound is tight. Let $B_j^{max} = \max_i B_{ij}$. Assume that the consumers are ordered by their maximum budgets, B_j^{max}, in decreasing order. If we set the prices of all products to B_j^{max}, then the first j consumers will be able to afford some product they are interested in, and pay price B_j^{max} for it (irrespective of the policy governing consumer behavior in case of ties). Thus, the payoff generated will be jB_j^{max}. If $j' = \arg\max_j jB_j^{max}$, then we set the price of all products to be $B_{j'}^{max}$. For this single-price algorithm, we get the following theorem.

Theorem 7. *The single price algorithm provides an* H_m*-approximation for revenue maximization.*

Proof. We note that the revenue generated by an optimal solution $OPT \leq \sum_j B_j^{max}$, since each consumer j spends at most B_j^{max}, her maximum budget over all products. Let $\mathcal{R}$ be the revenue generated by the single price algorithm. Then, $\mathcal{R} = j'B_{j'}^{max} \geq jB_j^{max}$ for all j. Thus, $OPT \leq \sum_j B_j^{max} \leq \sum_j \mathcal{R}/j = \mathcal{R}H_m$. □

5 Conclusion and Open Problems

In this paper, we studied the non-parametric multi-product pricing problem. We presented a PTAS for the realistic Rank-Buying with Price-Ladder model, thus providing a useful pricing tool to companies with access to customer profiles. We also present approximation algorithms and complexity analysis for various extensions and variants of the model.

Many problems still remain open. The complexity of the Rank-Buying with Price-Ladder problem is unresolved. It will be interesting to extend the results for profit maximization to the Rank-Buying with Price-Ladder problem. One can also study other extensions of the multi-product pricing problem. A possible extension might be to consider a *budget range* for each consumer – each consumer has a lower as well as upper bound on the amount they are willing to spend to buy a product. Another model of (especially) game theoretic interest is Max-Gain-Buying, where each consumer buys the product that gives it the maximum *gain*, which is defined to be the difference between the utility (budget) and price of a product.

References

1. T. Asano, and D. Williamson. "Improved approximation algorithms for MAX SAT." In *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 96–115, 2000.
2. P. Coy. "The Power of Smart Pricing." *Business Week*, April 10, 2000
3. G. Dobson and S. Kalish. "Heuristics for Pricing and Positioning a Product-line Using Conjoint and Cost Data." *Management Science*, vol. 39, no. 2, pp. 160–175, 1993.
4. General Motors. *The Auto Choice Advisor web site.* `http://www.autochoiceadvisor.com`.
5. J. Håastad. "Some Optimal Inapproximability results." In *Proceedings of the 28th ACM Symposium on the Theory of Computing*, pp. 1–10, 1997.
6. W. Hansom and K. Martin. "Optimal Bundle Pricing." *Management Science*, vol. 36, pp. 155–174, 1990.
7. J.R. Hauser and P. Simmie. "Profit Maximizing Perceptual Positions: An Integrated Theory for the Selection of Product Features and Price." *Management Science*, vol. 27, no. 1, pp. 33–56, 1981.
8. S. Khanna, M. Sudan, and D. Williamson. "A Complete Classification of the Approximability of Maximization Problems Derived from Boolean Constraint Satisfaction." In *Proceedings of the 29th ACM Symposium on the Theory of Computing*, pp. 11–20, 1997.
9. J. I. McGill, and G. Van Ryzin. "Revenue Management: Research Overview and Prospects." *Transportation Science*, vol. 33, no. 2, pp. 233–256, 1999.
10. G. McWilliams. "Lean Machine: How Dell Fine Tunes Its PC Pricing to Gain Edge in a Slow Market." *Wall Street Journal*, June 8, 2001.
11. A. Merrick. "Priced to Move: Retailers Try to Get Leg Up on Markdowns With New Software." *Wall Street Journal*, August 7, 2001
12. P. Rusmevichientong. "A Non-Parametric Approach to Multi-Product Pricing: Theory and Application." PhD Thesis, 2003.
13. P. Rusmevichientong, B. Van Roy, and P. Glynn. "A Non-Parametric Approach to Multi-Product Pricing." Submitted to *Operations Research*, 2002.
14. L. R. Weatherford, and S. E. Bodily. "A Taxonomy and Research Overview of Perishable-Asset Revenue Management: Yield Management, Overbooking, and Pricing." *Management Science*, vol. 40, no. 5, pp. 831-844, 1992.

A Alternate Algorithm for Price-Ladder with Rank/Max-Pricing

Theorem 8. *For the Max-Buying with Price-Ladder model, an approximation ratio of $1+\varepsilon(1+o(1))$ can be achieved in time $O(n^{\frac{1}{\varepsilon}\log\log\log n})$ and space $O(n)$ for any $\varepsilon > 0$ by choosing $\frac{\log n}{\varepsilon}(1+o(1))$ distinct prices.*

Proof. For this, we guess the number of consumers buying at a certain price, instead of guessing the products priced at a certain price as in Theorem 1.

We first modify the instance so that every product is of interest to at most one consumer. In the original instance, if m_i consumers are interested in the some product i, we create m_i new products with each of the m_i consumers interested in one of these products in the new instance. We specify the price order of these new products as follows: the m_i products are constrained to have prices in an order that is the reverse of the budget order of the corresponding consumers for product i. Thus, an optimal solution to the new instance will assign the same price to all these m_i products, and hence, an optimal solution to the new instance can easily be converted to a solution for the original one.

We may assume that the prices used in the solution are of the form $\mathcal{B}/(1+\mu)^i$, where $\mathcal{B}$ is the highest budget, and no smaller than $\delta\mathcal{B}/n$, by incurring a loss of $((1+\mu)(1+\delta))$ in the approximation factor. This restriction results in at most $k = \frac{\log n}{\mu}(1+o(1))$ possible prices. Let $\mathcal{S}$ be the maximum revenue generated using only a single price as in Theorem 7. We further restrict the solution space such that every chosen price gives a revenue of at least $\delta\mathcal{S}/k$, by losing an approximation factor of $1+\delta$. In addition, we only consider solutions where the number of consumers that buy at a price p to be of the form $(1+\delta)^i$, with a further loss of $1+\delta$ in the approximation factor. This leaves at most $\frac{\log k}{\delta}(1+o(1))$ possible choices for the number of consumers that buy at a given price. Overall, the approximation factor is $(1+\mu)(1+\delta)^3$.

The total number of choices of how many consumers will pay which price is thus $L = (\frac{\log k}{\delta})^k = (\frac{\log\log n - \log\mu}{\delta})^{\frac{\log n}{\mu}} = n^{\frac{1}{\mu}(\log(\log\log n - \log\mu) - \log\delta)}$. If we choose $\mu + 3\delta = \mu(1+o(1))$ and $\mu = \varepsilon(1+o(1))$, then the approximation factor is $1+\varepsilon(1+o(1))$ and $L = n^{\frac{1}{\varepsilon}\log\log\log n}$.

Consider any of these L choices, which specifies the number of consumers N_p that buy at any (rounded) price p. This gives us a projected revenue of $\mathcal{R} = \sum_p N_p p$. We next try to find a solution that generates a revenue of at least $\mathcal{R}$. We use a greedy strategy: consider the products one by one in decreasing order of price. Start with the highest price p, set the price of the next product to p, until there are N_p consumers buying at price p. We repeat for the next lower (rounded) price and so on. If there exists an unrounded solution with at least N_p consumers buying at (rounded) price p, then this procedure will always be able to find N_p consumers buying at price p. We can see this as follows: the unrounded solution has more than N_p consumers buying at price p. Since our greedy solution has to pick fewer consumers at each price, the set of products available to be priced at p is always a superset of the set of products priced p in the unrounded solution. Thus, the greedy algorithm would never run out of products or consumers. As argued earlier, we can easily modify this solution for the modified instance, where each product is of interest to only one consumer, to get a solution for the original instance without any loss in revenue. □