Paper Analysis Report

Zhixin Zhang

November 27, 2021

Abstract

The computing complexity of designing optimal prices under multi-parameter settings is still a problem. In this field, I've read [1] and will present the main results together with some of the proofs.

1 Introduction

Pricing mechanism design is an interesting problem. Generally there are some buyers and some items, one of the problems is how to design good allocation and pricing mechanisms to maximize the seller's expected revenue. Under the single-parameter setting, this problem is well-understood, as Myerson[3] has provided a closed-form characterization for the optimal mechanism. However, the multidimensional setting lacks solid results. Xi Chen[1] considered the simplest multi-parameter setting with only one buyer who can at most choose one item from m items. Also, there are other very strong restrictions. Under such setting, the allocation rule can be obtained trivially as the seller will always try to sell an item to the only buyer, so researchers concentrate on the pricing mechanism. They show that searching for the optimal price vector is generally a NP problem, and only under a limited condition, the pricing problem surely has polynomial algorithm, while most general cases are NPC.

I divide this report into three parts. Section 2 presents preliminary knowledge including definitions and the main results in [1]. Section 3 introduce the core proof idea of the authors. For the whole proof is quite complex and some details are boring, in section 4 I only present a few brilliant ones. The most important parts are the graph based exponential algorithm for general case and the reduction from *PARTITION* when proving NPC nature.

2 Preliminaries

In this section, I first present the basic knowledge to the pricing problem under a specific setting. And some general definitions will be presented as well. Then I show the main results on computing complexity of the pricing problem in [1].

2.1 Basic Settings and Definitions

First, I present the basic settings. There is one buyer and m items. The only buyer has valuation towards each item. Specifically, for item i, the valuation v_i is sampled from a **discrete and rational** distribution F_i, *i.e.* $v_i \sim F_i$. It's guaranteed that F_i are independent. We formally define the support set of distribution F_i as follows.

Definition 1 *The support set of distribution F_i is denoted by V_i, that $V_i = \{x | Pr\{v_i = x\} \neq 0\}$.*

Intuitively, the support set includes all possible values of a distribution. With this definition, we can represent values in a vector manner, as $\mathbf{v} \in \times_{i\in[m]} V_i = V, F = \times_{i\in[m]} F_i, \mathbf{v} \sim F$. In the following sections, we only concentrate on **finite and discrete** support set. Thus the **scale of input** can be decided by support sizes $|V_i|(i \in [m])$ and the number of itmes m.

What we're interested in is designing a good price to maximize seller's expected revenue. Denote a price as a vector $\mathbf{p} \in [0, +\infty)^m$. The buyer will consider only items satisfies $v_i \geq p_i$ and choose the one with greatest **linear utility** $v_i - p_i$. The seller's revenue is the corresponding p_i. If there is no

such item, the buyer will give up and the seller is going to gain 0. Note that there might be items with same utility, under this case, authors provide the following tie-breaking rule: the item with highest price has highest priority and if the prices are still not unique, the item with smallest index has the highest priority. Understood the buying strategy, We then formally define the expected revenue of the seller as follows naturally.

Definition 2 *The expected revenue is* $R(\mathbf{p}) = \sum_{i\in[m]} p_i Pr\{i \text{ is chosen by the buyer}\}$.

For a fixed $\mathbf{v}$, the revenue is a constant denoted by $R(\mathbf{v}, \mathbf{p})$. After defined the expected revenue, the optimal price vector is then naturally described as $\mathbf{p}^* = arg\max_{\mathbf{p}} R(\mathbf{p})$.

Then we formally define our problem in two perspectives. One is the **optimization version**, the other is the **decision version**.

Definition 3 *Given the distribution* F*, find the optimal price* $\mathbf{p}$ *that maximizes* $R(\mathbf{p})$*. Denote this problem by* P_{OPT}.

Definition 4 *Given the distribution* F *and a rational number* $t \geq 0$*, decide whether* $R(\mathbf{p}*) \geq t$*. Denote this problem by* P_{DEC}.

2.2 Results

The most important results of [1] are four theorems considering the time complexity in the some general cases. I present them as follows.

Theorem 1 P_{DEC} *is in NP.*

Theorem 2 *If* $\max_{i\in[m]} |V_i| \leq 2$*,* P_{OPT} *has polynomial algorithm.*

Theorem 3 *If* $|V_i| = 3, i \in [m]$*,* P_{DEC} *is in NPC.*

Theorem 4 *If* $F_i, i \in [m]$ *are indentical, then* P_{DEC} *is in NPC.*

Theorem 1 indcates that designing the optimal pricing mechanism has exponential algorithm and the authors provide an brilliant one. Different from some classical NPC problems such as the TSP problem where brute force algorithms are quite easy to find, providing an exponential algorithm for P_{OPT} requires careful construction. Theorem 2 lies very strong restriction on the problem and provide a polynomial algorithm. Theorem 3 later shows even a tiny extension will drag the problem into the abyss of NPC. It's a pity that [1] hasn't shown what will happen if only some of the support sizes exceeds 2. But intuitively, the problem might still be in NPC. Lastly, Theorem 4 discusses another common simplification and indicates even under such setting the problem can't jump out of the abyss of NPC.

3 Core Ideas

In this section I simply present the core ideas for proving the four main results.

To prove Theorem 1, we just need to give an exponential algorithm for the optimization version. And the $R(\mathbf{p}^*)$ provide sufficient information to make decision in P_{DEC} in $O(1)$, indicating P_{DEC} is in NP. To do the optimization, we first prove computing $R(\mathbf{p})$ can be done in polynomial time. Then we split the price space into cells of which the number is exponential. After that, we do linear programming in each cell in polynomial time, thus indicating P_{DEC} is in NP. The appealing thing is the linear programming problem in each cell is well-constructed that leads to a beautiful algorithm based on graph theorem.

We decompose theorem 2 into 2 stages. The authors extract the core of the theorem by making a non-degeneracy assumption. After proving the slightly weakened theorem, we generalize it. Under the non-degeneracy assumption, we keep obtaining stronger conditions on the optimal $\mathbf{p}^*$ and finally shows a well defined subspace of size $O(m^2)$ where the $\mathbf{p}^*$ lies in. As we can compute $R(\mathbf{p})$ in polynomial time, then a simple "enumerate and check" algorithm is polynomial.

Theorem 3 and 4 are proved with generally similar ideas. First carefully select a NPC problem. Then construct a P_{DEC} problem and transfer the expression of $R(\mathbf{p})$ into a quadratic form which

includes terms maximized iff the answer of well selected NPC problem is "yes". Then set the t in P_{DEC} to the theoretical supremum of such quadratic problem, then $R(\mathbf{p}) \geq t$ iff the answer of selected NPC problem is "yes", which finishes the reduction. To be more detailed, theorem 3 is reduced from **Integer Partition Problem** and theorem 4 is reduced from **Integer Knapsack Problem with Repetition**. The selected NPC problem are all in the form of choosing something summed up to a given constant, so the quadratic term is highly related with such sum. In the next section, I present only some proofs of theorem 3 and ignore theorem 4.

4 Excellent Proofs

In this section I present some of the proofs. I doesn't change the skeleton of the proofs in [1]. But I provide more intuitive ideas, delete some tedious details and add more explanation at some obscure part.

Before start, authors provide several lemmas for rigor or convenience. I present them below without proofs but explanations.

Lemma 1 *The tie-breaking rule won't shift the supremum of* $R(\mathbf{p})$.

Pay attention that not any tie-breaking rule satisfies this good property so this lemma is important. In fact, you can choose any index you like if both the utility and price are equal, however, choose the highest price when utility equals might be necessary.

Lemma 2 *Denote* $a_i = arg\min_x Pr\{v_i = x\} > 0, b_i = arg\max_x Pr\{v_i = x\} > 0, P = \times_{i\in[m]}[a_i, b_i]$. *Then for any* $\mathbf{p} \in [0, +\infty)$, *there exists* $\mathbf{p}' \in P$, *such that* $R(\mathbf{p}') \geq R(\mathbf{p})$.

According to this lemma, we can discuss in the field P only rather than an infinite interval. Pay attention that in the following sections, I will regard this as a default setting and won't mention it.

Lemma 3 *There exists a price vector* $\mathbf{p}^* \in P$, *such that* $R(\mathbf{p}^*) = sup_{\mathbf{p}} R(\mathbf{p})$.

If the supremum can't be reached, the function $R(\mathbf{p})$ just has no optimal solution. One such case is $y = \arctan x$, which is bounded by 1 but never reach it and there is not maximum value. So this lemma guarantees the existence of optimal price vector.

4.1 Proofs of Theorem 1

First I present a rough proof that we can compute $R(\mathbf{p})$ in polynomial time. Review the definition of $R(\mathbf{p}) := \sum_{i\in[m]} p_i Pr\{\text{item i is chosen by the buyer}\}$, then the core problem is to compute the probability of each item to be selected. We have,

$$\begin{aligned} Pr\{\text{item i is chosen by the buyer}\} &= Pr\{v_i - p_i \text{ is the maximum}\} \\ &= Pr\{v_i - p_i \geq v_j - p_j, j \in [m]\} \\ &= \sum_{v_i \in V_i} Pr\{v_i\} Pr\{v_i - p_i \geq v_j - p_j, j \in [m]\} \\ &= \sum_{v_i \in V_i} Pr\{v_i\} Pr\{v_j \leq v_i - p_i + p_j, j \in [m]\} \\ &= \sum_{v_i \in V_i} Pr\{v_i\} \prod_{j\in[m]\setminus\{i\}} F_j(v_i - p_i + p_j) \end{aligned} \tag{1}$$

where $\mathbf{1}[c]$ is the identical function which yields 1 iff the condition c is satisfied, otherwise, it yields 0. If we can access $F_i(x)$ in $O(1)$, then Eq. 1 indicates computing probability for each i costs $O(m|V_i|)$. Then computing the $R(\mathbf{p})$ only costs $O(m\sum_{i\in[m]})|V_i|$ which is polynomial. The roughness of this proof is I ignore the tie-breaking rule and only present the core ideas, however, it's no difficult to make it rigorous.

Then we're going to cut the price space. In fact, to simplify the problem in each cell, we would like to fix the item selected in one cell when the value $\mathbf{v}$ is fiexed. Following this idea, a series of hyperplanes can be applied as follows.

$$\begin{cases} p_i = s_i, & i \in [m], s_i \in V_i \\ p_i - s_i = p_j - t_j, & i, j \in [m], i \neq j, s_i \in V_i, t_j \in V_j. \end{cases} \tag{2}$$

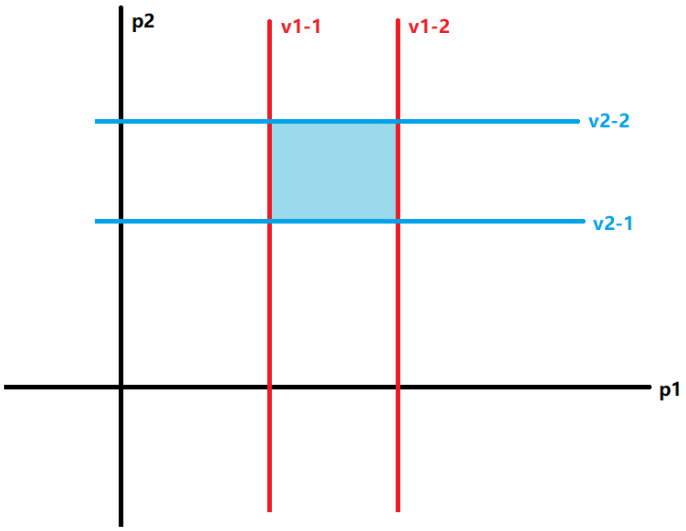

Figure 1: Example of half-planes of type 1.

The group of hyperplanes in 2 are boundaries of half-planes, which are in the following form.

$$\begin{cases} L_i \le p_i \le R_i, & i \in [m] \\ A_{ij} \le p_i - p_j \le B_{ij}, & i, j \in [m], i \ne j \end{cases}$$

where L_i, R_i are **neighbour elements** from V_i and A_{ij}, B_{ij} are **neighbour differences** from V_i, V_j. Figure 1 and figure 2 shows examples when $m = 2$. The light blue indicate the intersection of corresponding half-planes. To be more detailed, figure 1 shows the intersection of half-planes in the form of $p_i \ge c$ and figure 2 exhibits those in the form $p_i - p_j \ge c$. Pay attention that there is no other value $x \in V_1$ satisfying $v_{1-1} < x < v_{1-2}$, for v_{1-1} and v_{1-2} are neighbours. And it's similar for $v_{2-x} - v_{1-y}$ and $v_{2-z} - v_{1-w}$.

Under this setting, when we fix the value vector as $\mathbf{v}$, then in on cell, the relation between p_i and v_i is fixed which decides the utility is negative or not. Similarly, the relation between $u_i = v_i - p_i$ and $u_j = v_j - p_j$ is fixed as well, since $u_i > u_j$ can be obtained from $p_i - p_j < v_i - v_j$. All of these indicates that the item selected by the buyer will be the same for price vectors lying in the same cell. With this property, the expression of $R(\mathbf{p})$ in one cell is relatively fixed as even for different price vector, the probability term is the same. More formally, for a cell C, we have $R(\mathbf{p}) = \sum_{i \in [m]} p_i \gamma_i(C)$ where $\gamma_i(C) = Pr\{\text{item i is selected by the buyer} | \mathbf{p} \in C\}$. Then an exponential algorithm is quite clear. We can enumerate all cells first. Then in each cell C, select any price vector $\mathbf{p} \in C$ and compute $\boldsymbol{\gamma}(C)$. Then we only need to solve a linear programming in the form,

$$\begin{aligned} max. \quad & \sum_{i \in [m]} p_i \gamma_i(C) \\ s.t. \quad & \begin{cases} L_i \le p_i \le R_i \\ A_{ij} \le p_i - p_j \le B_{ij}. \end{cases} \end{aligned} \tag{3}$$

Finally, compare all optimal solutions among cells and choose the one maximizing $R(\mathbf{p})$. This algorithm finishes the proof that P_{DEC} is in NP. However, I would like present finer structure of the linear programming problem which leads to a fancy algorithm.

First, we can easily find the optimization conditions are suitable for constructing a **system of difference constraints**[2]. I will briefly introduce the idea of this method. Intuitively, considering the problem of finding the shortest paths in a directed graph from a single source. One of the most famous method is Bellman-Ford algorithm. In this algorithm, the core is keeping apply the **relaxation** operation to update shortest distances toward each vertex. The relaxation operation is based on a

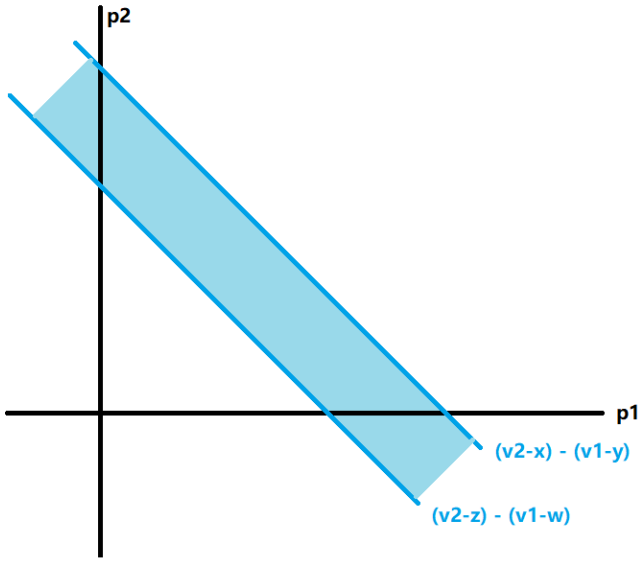

Figure 2: Example of half-planes of type 2.

distance triangle inequation between two vertices u, v. Assume there exists one directed edge $\langle u, v\rangle$ and the weight of it is $w(u, v)$, then

$$dist(v) \leq dist(u) + w(u, v) \tag{4}$$

is the **necessary condition** that $dist(u), dist(v)$ are the shortest distances from the source to u, v. We can modify inequality 4 in the following form,

$$dist(v) - dist(u) \leq w(u, v)$$

which yields the upper bound of the difference of shortest distances. Then if the constraints of a linear programming can be expressed in the form of providing lower bound or upper bound towards difference between variables, then we can construct a directed graph and apply the Bellman-Ford algorithm to find a feasible solution. Pay attention that we regard the optimization variables as distances in the graph.

Back to our linear programming problem 3. We construct the directed graph $G = \langle V, E\rangle$ as follows.

- Let $V := [m] \cup \{S\}$, where S is the super source.
- For constraints like $p_i - p_j \leq B_{ij}$, add one edge $\langle j, i\rangle$ with weight B_{ij}. For constraints like $p_i - p_j \geq A_{ij}$, *i.e.* $p_j - p_i \leq -A_{ij}$, add one edge $\langle i, j\rangle$ with weight $-A_{ij}$. For constraints like $p_i \leq R_i$, add one edge $\langle S, i\rangle$ with weight R_i. For constraints like $p_i \geq L_i$, add one edge $\langle i, S\rangle$ with weight $-L_i$.

Then the shortest distance from S to all other vertices provide a feasible solution of $\mathbf{p}$ according to the above discussion. And as the linear sum $\sum_{i\in[m]} p_i\gamma_i(C)$ has non-negative coefficients, the larger $\mathbf{p}$ is, the greater the sum is. However, inequation among a shortest path will provide an upper bound of $\mathbf{p}$. Suppose there is one shortest path $\{S, x_1, x_2, ..., x_k\}$. Existence of such path means a series of inequalities are satisfied,

$$p_{x_1} - 0 \leq_1, p_{x_2} - p_{x_1} \leq \lambda_2, ..., p_{x_k} - p_{x_{k-1}} \leq \lambda_k \Rightarrow p_{x_k} \leq \sum_{i\in[k]} \lambda_i.$$

the shortest distance allocation obviously reaches the equal condition, thus maximizing the $R(\mathbf{p})$ in the corresponding cell. If there is negative cycles, there won't be any feasible solution as the shortest paths don't exists. There is also an additional result as follows.

Corollary 1 *The optimal price vector* $\mathbf{p}$ *is a* ***linear combination*** *of elements in* $\{v_i|i \in [m]\} \cup \{v_i - v_j|i, j \in [m], i \neq j\}$.

Since the optimal $\mathbf{p}$ is the corresponding shortest distances, this corollary is obvious.

We can know transfer the linear programming problem into a graph problem. However, there is still a tiny problem that enumerating cells is a bit difficult. Luckily, the authors show there is no need to actually enumerate all cells. In fact, we can enumerate all the spanning trees of a graph with size $m + 1$ and for each edge on such trees, trying every value in the set $\{v_i - v_j|v_i \in V_i, v_j \in V_j, i \neq j\} \cup \{v_i|v_i \in V_i\}$. Then in a particular weighted spanning tree, regard the distance from the root to each vertex as the corresponding price and find the optimal combination among all sorts of trees. As the spanning trees of course cover all shortest paths among all possible graph structures based on all cells, this algorithm can exactly output the optimal price vector. There might be many redundant spanning trees, however, the total number of spanning trees are still exponential.

To avoid too many fuzzy words, the above discussion is not rigorous as I ignore the tie-breaking rule. In order to make the proof more rigorous, there might be both strict inequation and weak inequation, and the condition for no feasible solution will include the appearance of zero cycles where at least one strict inequation lies.

4.2 Proofs of Theorem 2

Now support sets' size won't exceed 2. The first part of authors' non-degeneracy assumption is the sizes of all support sets are exactly 2. Then we can represent V_i as $V_i := \{a_i, b_i\}$ that $0 < a_i < b_i$. The probability of $v_i = b_i$ is represented by q_i that $0 < q_i < 1$. The second part or the non-degeneracy assumption is,

- **Non-degeneracy assumption**. $b_1 < b_2 < ... < b_m$, $a_i \neq a_j$ and $t_i = b_i - a_i \neq t_j$ for all $i \neq j$.

Then the authors apply a "flow" of constraints on price vector $\mathbf{p}$. I'll present them all once as follows. Let's denote the optimal price vector space by OPT.

Lemma 4 *If* $\mathbf{p} \in OPT$ *satisfies* $p_i > a_i$, *then* $\mathbf{p} = \mathbf{b}$.

Lemma 5 *If* $\mathbf{p} \in OPT$, *then there is no more than* 1 *such* i *that* $p_i = a_i$.

Lemma 6 *If* $\mathbf{p} \in OPT$ *and there is one such* k *that* $p_k = a_k$, *then for all* $i \in [m]$, $p_i \in \{b_i, b_i - t_k\}$.

Lemma 7 *If* $\mathbf{p} \in OPT$, *then for* $l < k$ *we have* $p_l = b_l$.

Lemma 8 ***(Monotonicity)*** *If* $\mathbf{p} \in OPT$ *and there is one such* k *that* $p_k = a_k$. *Then for* $k < c < d$ *and* $t_c, t_d > t_k$, *if* $p_c = b_c - t_k$, *we have* $p_d = b_d - t_k$ *as well.*

Lemma 8 is the core lemma. I firstly present how can we construct a polynomial algorithm based on lemma 8 with assistance of other lemmas. First, as $\mathbf{b} \in OPT$, we check it. Then there is only one k that $p_k = a_k$. We can enumerate such index with m times. For each k, the $l < k$ must satisfy $p_l = b_l$ thus we don't need to consider them. For $l > k$ and $t_l < t_k$, we can directly set $p_l = b_l$, otherwise $p_l = b_l - t_k = (b_l - a_l) - t_k + a_l < a_l$ will exceed the field of price vector. For the rest $l > k$ that $t_l > t_k$, we enumerate the last index that $p_l = b_l$ with at most $m - k$ times and the rest $w > l$ satisfies $p_w = b_w - t_k$. Ignore the time for computing $R(\mathbf{p})$ after two layers of enumeration, the total rounds won't exceed $\sum_{k=1}^{m} m - k = \frac{m(m-1)}{2} \in O(m^2)$. As computing $R(\mathbf{p})$ can be done in polynomial time, we have finished the construct of a polynomial algorithm.

Then we present more detailed proofs of the five lemmas. The proofs are not difficult and all finished by reduction to absurdity. What's excellent is the process that gradually shrinks the search space for optimal price vector.

4.2.1 Proof of Lemma 4

Assume $\mathbf{p} \in OPT$ and $p_i > a_i, i \in [m]$. What's more, there is at least one k that $p_k < b_k$. Record the method of dividing the price space into cells. Similarly (but we don't need to divide OPT as it's already a cell), as we fix the $\mathbf{v}$, the item selected will be generally fixed in the whole space OPT. Though

the tie-breaking rule will bring some obstacles at the boundary, we can still obtain $R(\mathbf{v}, \mathbf{b}) \geq R(\mathbf{v}, \mathbf{p})$ since $p_i \leq b_i$ holds for every $i \in [m]$. Moreover, there is at least one $\mathbf{v}^*$ that $R(\mathbf{v}^*, \mathbf{b}) > R(\mathbf{v}^*, \mathbf{p})$. One possible $\mathbf{v}^*$ will force the smallest $k \in [m]$ that $p_k < b_k$ to be the item selected, such as,

$$v_i^* = \begin{cases} b_i, & i = k \\ a_i, & i \neq k. \end{cases}$$

Then under the tie-breaking rule, i will be selected and $R(\mathbf{v}^*, \mathbf{b}) = b_k > R(\mathbf{v}^*, \mathbf{p}) = p_k$. Finally, as $R(\mathbf{p}) = \sum_{\mathbf{v} \sim V} R(\mathbf{v}, \mathbf{p}) \cdot Pr\{\mathbf{v}\}$, we directly have $R(\mathbf{b}) > R(\mathbf{p})$, which violates our assumption and finishes the proof.

4.2.2 Proofs of Lemma 5, 6, 7

First I present the proof of lemma 5.

Assume $\mathbf{p} \in OPT$ and there are at least 2 indices satisfying $p_k = a_k$. We then construct a price vector $\mathbf{p}'$. Choose the smallest k that $p_k = a_k$. Denote $S := \{i \in [m] | b_i - p_i = t_k\}$. Of course, we have $k \in S$. Then define $\mathbf{p}'$ as follows,

$$p_i' = \begin{cases} p_i + \epsilon, & i \in S \\ p_i, & else \end{cases}$$

where ϵ is a sufficiently small positive constant. Fix the value vector $\mathbf{v}$, then we discuss the following three cases.

- If indices reaching the highest utility appear only in S, it will slightly decrease, however, the item selected won't change since ϵ is sufficiently small. Then we have $R(\mathbf{v}, \mathbf{p})' = R(\mathbf{v}, \mathbf{p}) + \epsilon > R(\mathbf{v}, \mathbf{p})$. The condition of this case is $\max_{i \in [m]}(v_i - p_i) = t_k$. To understand the condition, consider all other items. For item $l \in S, l \neq k$, only when $u_l = v_l - p_l \in \{0, b_l - p_l = t_k\}$ it is possible to reach the highest utility, where 0 is at the case $v_l = a_l$. Similarly for $l \notin S$, the possible utility $u_l \in \{0, b_l - p_l \neq t_k\}$. According to this, if we set $\max_{i \in [m]}(v_i - p_i) = t_k$ can of course restrict all indices reaches the highest utility in S (don't forget $t_i = b_i - a_i > 0$ as assumed).
- If indices reaching the highest utility strictly intersect with S (which means some of best indices lie in S and others are not), then the only item in S that maximizes the utility can only be k since $l \in S, l \neq k \Rightarrow b_l > p_l > a_l$ (otherwise we will get $t_k = 0$ or $t_l = t_k, l \neq k$ which violates the assumptions). What's more, with the fact presented in the first case that possible highest utility satisfies $u_l \in \{0, b_l - p_l\}$. As $l \in S \Rightarrow b_l - p_l = t_k$ and $l \notin S \Rightarrow b_l - p_l \neq t_k$, the "strict" intersection condition forces $\max_{i \in [m]}(v_i - p_i) = 0$. Then interestingly, we have $v_l = p_l = b_l$ or $v_l = a_l \leq p_l$ for all $l \notin S$ and $v_l = a_l < p_l$ for all $l \in S, l \neq k$ and $p_k = v_k = a_k$. Under this setting, the highest utility will remain the same even prices in S are risen by ϵ as there are optimal elements outside of S. Considered $p_l > a_l$ for $l \in S, l \neq k$ and there is at least two l that $p_l = a_l$, there must be at least one $l \notin S$ satisfying $p_l = a_l > a_k = p_k$ as we choose the smallest a_k. This means item k won't be chosen according to the tie-breaking rule. Then directly we have $R(\mathbf{v}, \mathbf{p}') = R(\mathbf{v}, \mathbf{p})$.
- If indices reaching the highest utility doesn't appear in S, it won't change since we only modify prices of items in S and ϵ is sufficiently small. So the item selected won't change. Thus we have $R(\mathbf{v}, \mathbf{p}') = R(\mathbf{v}, \mathbf{p})$.

According to the above discussion, we have $R(\mathbf{v}, \mathbf{p}') \geq R(\mathbf{v}, \mathbf{p})$. Note that we can choose $\mathbf{v}^*$ as $v_k^* = b_k$ and $v_i^* = a_i, i \neq k$ and obtain $R(\mathbf{v}^*, \mathbf{p}') > R(\mathbf{v}^*, \mathbf{p})$. So $R(\mathbf{p}') > R(\mathbf{p})$ for sure, which ends the proof.

The proofs of other two lemmas are quite similar to the above. For lemma 6, we set $S := \{i \in [m] | b_i - p_i = b_l - p_l\}$ where l is one item that $p_l \notin \{b_l, b_l - t_k\}$. For lemma 7, we set $S := \{i < k | p_i = b_i - t_k > a_i\}$. Then the following steps share the same template with the proof of lemma 5 and require careful discussions as well.

4.2.3 Proof of Lemma 8

Though lemma 8 is the core lemma to obtain an optimal price space of size $O(m^2)$, the proof is the most abstruse one among these five lemmas. I failed to extract intuitive ideas from such huge amounts of formulas and decide to ignore it here.

4.2.4 Generalization

We've discussed the proofs under the non-degeneracy assumption, now we're going to generalize the results.

Denote the original setting as I. The idea is to slightly disturb I and obtain a setting satisfying the non-degeneracy assumption. Then there will be two optimal price vector spaces, one is under I and the other is under the disturbed setting. Let the disturb approaches 0, we can show the two spaces are equivalent in the sense of maximum value, which finishes the proof.

We reserve one assumption that $b_1 < ... < b_m$. After one round of sorting, this condition can be satisfied. So it won't affect the generalization of the main results. The new setting can be constructed as follows.

- For $V_i = \{b_i\}$, set $V_{i,\epsilon} := \{b_i + i\epsilon, b_i + 2i\epsilon\}$ and $Pr\{v_{i,\epsilon} = b_i + 2i\epsilon\} = \frac{1}{2}$.
- For $V_i = \{a_i, b_i\}$, set $V_{i,\epsilon} := \{a_i + i\epsilon, b_i + 2i\epsilon\}$ and keep $Pr\{v_{i,\epsilon} = b_i + 2i\epsilon\} = Pr\{v_i = b_i\} = q_i$.

Under the new setting denoted by I_ϵ, we directly claim the the optimal price space's size doesn't exceed $O(m^2)$. Denote the shrunk price space under I_ϵ as A_ϵ (of size $O(m^2)$) and one price vector in it is $\mathbf{p}_\epsilon$. Pay attention that A_ϵ has no trivial relation with A and we need to link them with limit operation.

We first show the limit $\lim_{\epsilon\to 0} \max_{\mathbf{p}_\epsilon \in A_\epsilon} R_\epsilon(\mathbf{p}_\epsilon)$ exists and can be computed in polynomial time. If such limit exists, we have,

$$\lim_{\epsilon\to 0} \max_{\mathbf{p}_\epsilon \in A_\epsilon} R_\epsilon(\mathbf{p}_\epsilon) = \max_{\mathbf{p}_\epsilon \in A_\epsilon} \left\{ \lim_{\epsilon\to 0} R_\epsilon(\mathbf{p}_\epsilon) \right\} \tag{5}$$

If we can prove the limit for all $p_\epsilon \in A_\epsilon$ exists, then the exchange of "max" and "lim" are naturally at this discrete case. And simultaneously, the existence of $\lim_{\epsilon\to 0} \max_{\mathbf{p}_\epsilon \in A_\epsilon} R_\epsilon(\mathbf{p}_\epsilon)$ is proved. Apply corollary 1, we directly have $\mathbf{p}_\epsilon$ are all **affine combination** of ϵ. Then as ϵ approaches 0, the limit $\lim_{\epsilon\to 0} R_\epsilon(\mathbf{p}_\epsilon)$ exists of course. Knowing the limit $\lim_{\epsilon\to 0} \max_{\mathbf{p}_\epsilon \in OPT_\epsilon} R_\epsilon(\mathbf{p}_\epsilon)$ exists, compute it in polynomial time is quite easy, as we can compute the $O(m^2)$ number of limit $\lim_{\epsilon\to 0} R_\epsilon(\mathbf{p}_\epsilon)$ in polynomial time. Pay attention that to compute the limit $\lim_{\epsilon\to 0} R_\epsilon(\mathbf{p}_\epsilon)$, we can enumerate $O(m^2)$ number of $\mathbf{p}_\epsilon$ and compute corresponding $R_\epsilon(\epsilon)$ in polynomial time as discussed in section 3, then directly set $\epsilon = 0$ in the expression.

Then author then shows the fact that,

$$\max_{\mathbf{p}} R(\mathbf{p}) = \lim_{\epsilon\to 0} \max_{\mathbf{p}_\epsilon \in A_\epsilon} R_\epsilon(\mathbf{p}_\epsilon).$$

proved by two small lemmas $\max_{\mathbf{p}} R(\mathbf{p}) \geq \lim_{\epsilon\to 0} \max_{\mathbf{p}_\epsilon \in A_\epsilon} R_\epsilon(\mathbf{p}_\epsilon)$ and $\max_{\mathbf{p}} R(\mathbf{p}) \leq \lim_{\epsilon\to 0} \max_{\mathbf{p}_\epsilon \in A_\epsilon} R_\epsilon(\mathbf{p}_\epsilon)$. I won't present the detailed proofs here but I would like to present the imperfection of the generalization. The existed discussion of generalization can only provide us a polynomial algorithm to find the optimal expected revenue, but ignore how to find the optimal price vector, which is required by P_{OPT}. Intuitively, as $\epsilon \to 0$, the setting I_ϵ is approaching the original setting I. Thus, we believe the optimal price vector does as well. Then, we can enumerate the $O(m^2)$ number of price vectors and compute their limit. Then there must be at least one $\lim_{\epsilon\to 0} \mathbf{p}_\epsilon$ lies in OPT. Even if we can't find all vectors in OPT, we have found one.

4.3 Proofs of Theorem 3

The NPC problem the authors selected is quite easy to understand. The formal definition of it is presented in the following.

Definition 5 *PARTITION*

- ***Input**: A set $C = \{c_1, ..., c_m\}$ of m **positive** integers.*
- ***Problem**: Does there exists a partition of C into two subsets with equal sum?*

Given the input of a partition problem, we construct P_{DEC} as follows.

- Select $0 < a < b$.

- Set $V_i = \{0, a, b\}, Pr\{v_i = b\} = q_i, Pr\{v_i = a\} = r_i$ that $q_i = \frac{c_i}{M}, M = 2^m c_1^3, r_i = \frac{b-a}{a(1-t_i)} \cdot q_i, t_i = \frac{b}{2a} \cdot \sum_{j \neq i, j \in [m]} q_j$.

As all support sets are of the same values, we can obtain a very strong constraints on OPT. Specifically, we will prove

Lemma 9 $OPT = \{a, b\}^m$.

Directly, according to lemma 2, we have $OPT = [0, b]^m$. According to corollary 1, since $0, a, b$ are all integers, the optimal price vectors must be integer vectors as well. Then we have $OPT = [0 : b]^m$, where $[0 : b]$ denotes integers in $[0, b]$.

Suppose lemma 9 is not correct. Then there is at least on j satisfying $p_j \notin \{a, b\}$. Denote a large number $N := 2^m c_1^2$, then obviously $q_i = O(\frac{1}{N})$. According to the definition of t_i, we have $t_i = O(\frac{m}{N})$. According to the definition of r_i, we have,

$$bq_i = a(q_i + r_i) - ar_i t_i$$

then we obtain $r_i = O(\frac{1}{N})$. Denote $|T_1 - T_2| \leq \epsilon$ as $T_1 = T_2 \pm \epsilon$. Then we have,

$$r_i = \frac{b-a}{a(1-t_i)} \cdot q_i = \frac{b-a}{a} \cdot q_i \pm 2\frac{b-a}{a} q_i t_i = \frac{b-a}{a} \cdot q_i \pm O(\frac{m}{N^2}).$$

The second equation holds when $t_i \leq \frac{1}{2}$. Since $t_i = O(\frac{m}{N}) = O(\frac{m}{2^m c_1^2})$, we can claim that for sufficiently large m, t_i is sufficiently small, then the condition $t_i \leq \frac{1}{2}$ can be satisfied.

We then bound the probability that item i is selected. Denote that probability by γ_i. As $v_i \geq p_i$ is one necessary condition, we have,

$$\gamma_i \leq Pr\{v_i \geq p_i\}.$$

Apply this to $R(\mathbf{p})$, we have,

$$R(\mathbf{p}) = \sum_{i \in [m]} p_i \gamma_i \leq \sum_{i \in [m]} p_i Pr\{v_i \geq p_i\}. \tag{6}$$

There is also a sufficient condition that i will be selected:

$$v_i \geq p_i, v_j = 0, i \neq j.$$

Based on this condition, we have,

$$\gamma_i \geq Pr\{v_i \geq p_i\} \prod_{j \in [m], j \neq i} (1 - q_j - r_j) \geq Pr\{v_i \geq p_i\} \left(1 - O(\frac{m}{N})\right).$$

The last inequation can be explained by,

$$\prod_{j \in [m], j \neq i} (1 - q_j - r_j) = \left(1 - O(\frac{1}{N})\right)^{m-1} \geq \left(1 - O(\frac{1}{N})\right)^m = 1 - O(\frac{m}{N}) + Res \geq 1 - O(\frac{m}{N}).$$

Apply this to $R(\mathbf{b})$, we have,

$$R(\mathbf{b}) \geq \sum_{i \in [m]} bPr\{v_i \geq p_i\} \left(1 - O(\frac{m}{N})\right) = \sum_{i \in [m]} bq_i \left(1 - O(\frac{m}{N})\right) = \left(\sum_{i \in [m]} bq_i\right) - O(\frac{m^2}{N^2}). \tag{7}$$

In fact, the "sufficient" condition here is not so sufficient with tie-breaking rule, however, for price vector with fixed components like $\mathbf{b}$, this won't affect inequality .

Then we carefully discuss Eq. 6.

- For $p_i = b$, we have $Pr\{v_i \geq p_i\} = q_i$, and the term's contribution to the sum is bq_i.
- For $p_i = a$, we have $Pr\{v_i \geq p_i\} = q_i + r_i$, and the term's contribution to the sum is $a_i(q_i + r_i)$. And we have, $a(q_i + r_i) \leq a\left(q_i + \frac{b-a}{a} q_i + O(\frac{m}{N^2})\right) = bq_i + O(\frac{m}{N^2})$.

- For $a < p_i < b$, since p_i is an integer, the contribution is $p_i q_i \leq (b-1)q_i = bq_i - \frac{c_i}{M} \leq bq_i - \frac{1}{M}$.
- For $0 < p_i < a$, since p_i is an integer, the contribution is $p_i(q_i + r_i) \leq (a-1)(q_i + r_i) = bq_i - q_i - r_i(1 - at_i) \leq bq_i - q_i \leq bq_i - \frac{1}{M}$.

Note that $\frac{1}{M} = \frac{1}{2^m c1^3} \gg \frac{m^2}{2^{2m} c1^4} = \frac{m^2}{N^2}$ in the last two cases. Thus according to all the four cases, we have,

$$\begin{aligned} R(\mathbf{p}) &= \sum_{i\in[m]} bq_i + C_1 O(\frac{m}{N^2}) - C_2 \frac{1}{M} \\ &\leq \sum_{i\in[m]} bq_i + C_1 O(\frac{m^2}{N^2}) - C_2 \frac{1}{M} \\ &< \sum_{i\in[m]} bq_i - O(\frac{m^2}{N^2}) \quad (*) \\ &\leq R(\mathbf{b}). \end{aligned}$$

The inequation marked $(*)$ is based on the fact that $\frac{1}{M}$ is quite large and $C_1 \neq 0$ since there is at least one j satisfying $p_j \notin \{a, b\}$. Now we finish the proof of lemma 9.

Based on lemma 9, there only two kinds of positive contribution to the expected revenue: a and b. Denote $S := \{i | p_i = a_i\}, T := \{i | p_i = b_i\}$. Now we have obtained a **partition** of $[m]$ and is going to link with the selected NPC problem. With simple discussion, we obtain,

$$\begin{aligned} Pr\{R(\mathbf{v}, \mathbf{p}) = a\} &= 1 - \prod_{i\in S}(1-q_i) + \prod_{j\in T}(1-q_j)\left[\prod_{i\in S}(1-q_i) - \prod_{i\in S}(1-q_i-r_i)\right] \\ Pr\{R(\mathbf{v}, \mathbf{p}) = b\} &= \prod_{i\in S}(1-q_i)\left[1 - \prod_{j\in T}(1-q_j)\right]. \end{aligned} \tag{8}$$

The amazing method is to approximate above probabilities with terms of which the order is less than 3. Formally, denoted $\epsilon = \frac{m^3}{N^3}$, we're going to modify the probabilities in the form $x + O(\epsilon)$. Basic approximations are,

$$\begin{aligned} \prod_{i\in D}(1-q_i) &= 1 - \sum_{i\in D} q_i + \sum_{i\neq j\in D} q_i q_j \pm O(\epsilon) \\ \prod_{i\in D}(1-q_i-r_i) &= 1 - \sum_{i\in D}(q_i + r_i) + \sum_{i\neq j\in D}(q_i + r_i)(q_j + r_j) \pm O(\epsilon) \end{aligned}$$

where $D \subseteq [m]$. You can unfold the product and easily verify these basics. Based on these results, we

can modify Eq. 8 with following statements,

$$
\begin{aligned}
&1-\prod_{i\in S}(1-q_i)=\sum_{i\in S}q_i-\sum_{i\neq j\in S}q_iq_j\pm O(\epsilon)\\
&\prod_{j\in T}(1-q_j)\left[\prod_{i\in S}(1-q_i)-\prod_{i\in S}(1-q_i-r_i)\right]\\
&=\left(1-\sum_{i\in T}q_i+\sum_{i\neq j\in T}q_iq_j\pm O(\epsilon)\right)\left[\sum_{i\neq j\in S}q_iq_j+\sum_{i\in S}r_i-\sum_{i\neq j\in S}(q_i+r_i)(q_j+r_j)\pm O(\epsilon)\right]\\
&=\sum_{i\in S}r_i-\sum_{i\in S}r_i\sum_{j\in T}q_j+\sum_{i\neq j\in S}q_iq_j-\sum_{i\neq j\in S}(q_i+r_i)(q_j+r_j)\pm O(\epsilon)\\
&\prod_{i\in S}(1-q_i)\left[1-\prod_{j\in T}(1-q_j)\right]\\
&=\left(1-\sum_{i\in S}q_i+\sum_{i\neq j\in S}q_iq_j\pm O(\epsilon)\right)\left(\sum_{i\in T}q_i-\sum_{i\neq j\in T}q_iq_j\pm O(\epsilon)\right)\\
&=\sum_{j\in T}q_j-\sum_{i\neq j\in T}q_iq_j-\sum_{i\in S}q_i\sum_{j\in T}q_j\pm O(\epsilon).
\end{aligned}
$$

All above are long but easy to understand. Then we have approximates,

$$
\begin{aligned}
Pr\{R(\mathbf{v},\mathbf{p})=a\}&=\sum_{i\in S}(q_i+r_i)-\sum_{i\in S}r_i\sum_{j\in T}q_j-\sum_{i\neq j\in S}(q_i+r_i)(q_j+r_j)\pm O(\epsilon)\\
Pr\{R(\mathbf{v},\mathbf{p})=b\}&=\sum_{j\in T}q_j-\sum_{i\neq j\in T}q_iq_j-\sum_{i\in S}q_i\sum_{j\in T}q_j\pm O(\epsilon).
\end{aligned}
\tag{9}
$$

Apply Eq. 9 to the definition of $R(\mathbf{p})$, we obtain,

$$
\begin{aligned}
R(\mathbf{p})=a\cdot&\left(\sum_{i\in S}(q_i+r_i)-\sum_{i\in S}r_i\sum_{j\in T}q_j-\sum_{i\neq j\in S}(q_i+r_i)(q_j+r_j)\right)\\
+b\cdot&\left(\sum_{j\in T}q_j-\sum_{i\neq j\in T}q_iq_j-\sum_{i\in S}q_i\sum_{j\in T}q_j\right)\pm O(\epsilon).
\end{aligned}
\tag{10}
$$

Then we're going to extract some constant regardless of price. Consider Eq. 10, the first order term is,

$$
\begin{aligned}
&a\cdot\sum_{i\in S}(q_i+r_i)+b\cdot\sum_{j\in T}q_j\\
&=b\cdot\sum_{j\in[m]}q_j+\sum_{i\in S}(a(q_i+r_i)-bq_i)\\
&=b\cdot\sum_{j\in[m]}q_j+\sum_{i\in S}(ar_it_i)\\
&=C_1+\sum_{i\in S}(ar_it_i).
\end{aligned}
$$

For the second order term, we deal with $a\sum_{i\neq j\in S}(q_i+r_i)(q_j+r_j)$ first:

$$\begin{aligned}
&a\sum_{i\neq j\in S}(q_i+r_i)(q_j+r_j)\\
&=\frac{1}{2}\sum_{i\in S}(q_i+r_i)\sum_{j\in S,j\neq i}a(q_j+r_j)\\
&=\frac{1}{2}\sum_{i\in S}(q_i+r_i)\sum_{j\in S,j\neq i}(bq_j+ar_jt_j)\\
&=\frac{b}{2}\sum_{i\in S}q_i\sum_{j\in S,j\neq i}q_j+\frac{b}{2}\sum_{i\in S}r_i\sum_{j\in S,j\neq i}q_j\pm O(\epsilon).
\end{aligned}$$

The second order term is,

$$\begin{aligned}
&\left(-a\cdot\sum_{i\in S}r_i\sum_{j\in T}q_j-a\cdot\sum_{i\neq j\in S}(q_i+r_i)(q_j+r_j)\right)+\left(-b\cdot\sum_{i\neq j\in T}q_iq_j-b\cdot\sum_{i\in S}q_i\sum_{j\in T}q_j\right)\\
&=-b\sum_{i\neq j\in S}q_iq_j-\frac{b}{2}\sum_{i\in S}r_i\sum_{j\in S,j\neq i}q_j-a\sum_{i\in S}r_i\sum_{j\in T}q_j-b\sum_{i\neq j\in T}q_iq_j-b\sum_{i\in S}q_i\sum_{j\in T}q_j\pm O(\epsilon)\\
&=-b\sum_{i\neq j\in[m]}q_iq_j-\frac{b}{2}\sum_{i\in S}r_i\sum_{j\in S,j\neq i}q_j-a\sum_{i\in S}r_i\sum_{j\in T}q_j\pm O(\epsilon)\\
&=C_2-\frac{b}{2}\sum_{i\in S}r_i\sum_{j\in S,j\neq i}q_j-a\sum_{i\in S}r_i\sum_{j\in T}q_j\pm O(\epsilon)\\
&=C_2-\sum_{i\in S}r_i\left(\frac{b}{2}\sum_{j\in S,j\neq i}q_j-a\sum_{j\in T}q_j\right).
\end{aligned}$$

Finally we get,

$$\begin{aligned}
R(\mathbf{p})&=(C_1+C_2)+\sum_{i\in S}r_i\left(at_i-\frac{b}{2}\sum_{j\in S,j\neq i}q_j-a\sum_{j\in T}q_j\right)\pm O(\epsilon)\\
&=C+\sum_{i\in S}r_i\left(\frac{b}{2}\sum_{j\neq i}q_j-\frac{b}{2}\sum_{j\in S,j\neq i}q_j-a\sum_{j\in T}q_j\right)\pm O(\epsilon)\\
&=C+\sum_{i\in S}r_i\left(\frac{b}{2}-a\right)\sum_{j\in T}q_j\pm O(\epsilon)\qquad(11)\\
&=C+\frac{b-a}{a}\cdot\left(\frac{b}{2}-a\right)\cdot\frac{1}{M^2}\cdot\left(\sum_{i\in S}c_i\right)\cdot\left(\sum_{j\in T}c_j\right)\pm O(\epsilon)\\
&=C+C_3\cdot\frac{1}{M^2}\cdot\left(\sum_{i\in S}c_i\right)\cdot\left(\sum_{j\in T}c_j\right)\pm O(\epsilon).
\end{aligned}$$

Denote $H=\frac{1}{2}\sum_{i\in[m]}c_i$, then it's obvious that Eq. 11 reaches the greatest value when $\sum_{i\in S}c_i=\sum_{j\in T}c_j$, and that value is,

$$C+C_3\cdot\frac{1}{M^2}H^2+O(\epsilon).$$

Otherwise, the second greatest value is,

$$C+C_3\cdot\frac{1}{M^2}(H-1)(H+1)+O(\epsilon)=C+C_3\cdot\frac{1}{M^2}H^2+O(\epsilon)-\frac{1}{M^2}.$$

As we know, $\epsilon = \frac{m^3}{N^3} = c_1^2 \cdot \frac{m^3}{M^3} = o(\frac{1}{M^2})$, then the second greatest value is much smaller than the greatest value. Thus we can choose $t^* = C + C_3 \cdot \frac{1}{M^2}(H^2 - 0.1)$. Consider the condition,

$$R(\mathbf{p}) \geq t^* = C + C_3 \cdot \frac{1}{M^2}(H^2 - 0.1) \tag{12}$$

If 12 is satisfied, the answer to P_{DEC} is "yes" and simultaneously, the answer to *PARTITION* is "yes" and vise versa. This finishes the reduction.

5 Speech

Limited by the length and my ability, I can only show some of the proofs in [1]. However, all the major results and proof ideas are carefully presented in this report. I try my best to exhibit the ideas in a more clear way and sacrifice rigor sometimes. For the proofs of theorem 1, I modify many proof details to make it clearer as I'm familiar with linear space and graph theorem. I'm sorry that in the later two major proofs there are more seemingly "copy -paste" part for my poor algebraic analysis skill. I still present them here for the consideration of a relatively complete proof.

References

[1] Xi Chen, Ilias Diakonikolas, Dimitris Paparas, et al. The complexity of optimal multidimensional pricing for a unit-demand buyer. *Games and Economic Behavior*, 110:139–164, 2018.

[2] Thomas H. Cormen, Charles E. Leiserson, Ronald L.Rivest, and Clifford Stein. Introduction to Algorithms, Third Edition. pages 387–394, 2009.

[3] R.B. Myerson. Optimal auction design. *Mathematical Methods of Operations Research*, pages 58–73, 1981.